

RESEARCH LIBRARY

Cubic Divisor Transport and Reconstruction

Hilbert 12

Lean and evidence guide

Komorebi P.

Standardized publication edition / 8 October 2026

This companion makes the boundary between checked formal statements and ordinary analytic arguments explicit. The current v0.14 block formalizes integer coefficients of the principal correction and explicit zero/pole families. The continuation sources also contain algebraic, finite, and conditional analytic lemmas. Their variables and hypotheses are part of their conclusions' scope.

No theorem in this project identifies the elliptic-gamma value with the arithmetic candidate. Kernel acceptance of a conditional comparison does not supply its hypotheses. The complex divisor interpretation, cubic counting in disks, canonical-product convergence, and reconstruction of the principal correction remain ordinary proofs in the traditional manuscript.

Fresh formal verification: passed at module level. The selected source contains 42 mathematical modules and 453 distinct theorem declarations. Every selected theorem has a fresh compiler type report and a complete transitive axiom audit. The current v0.14 subset is separately checked: 25 theorems, comprising 11 new and 14 unchanged inherited declarations. Only `propext`, `Classical.choice`, and `Quot.sound` occur; no `sorryAx` or project axiom occurs. Positive arithmetic controls pass and deliberately false arithmetic controls are rejected. The single all-import/combined process exceeded the tested memory cap; its recorded resource failure is distinct from the complete module-scoped checks.

The canonical source tree contains one selected copy of each mathematical module. Repeated files from the continuation chain and foundations are byte-identical. Historical counts are not added across versions. Exact source origins, hashes, declaration types, run commands, resource bounds, diagnostics, and axiom dependencies are retained alongside this guide.

Contents

1 The exact integer model in the current block 3
2 Explicit families and the cardinality boundary 3
3 Complete current declaration inventory 5
4 Deduplicated continuation-source catalogue 6
5 Conditional results that must retain their hypotheses 7
6 Mapping to the traditional manuscript 8
7 Fresh replay, resource bounds and provenance 9
8 Portable verification 10
9 Remaining formal obligations 10

1 The exact integer model in the current block

`H12PrincipalDivisor.V` is $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$. Its coordinates are the coefficients of the lattice basis $(\tau, s, 1/5)$ before the ordinary complex embedding. The formal sources do not define elliptic gamma, meromorphic divisors, or the embedding λ.

For integer a, b, c, j, the supplied definitions are

$$p(b,c) = -\big((2b - 3c) \operatorname{div} 7\big),$$

$$d(b,c) = 7p(b,c) + 2b - 3c,$$

$$H(a) = \begin{cases} 1, & a > 0, \\ 0, & a \le 0, \end{cases}$$

$$w(c) = \begin{cases} 4, & c \bmod 5 = 0, \\ -1, & c \bmod 5 \neq 0, \end{cases}$$

$$B(a,b,j) = H(j) - H(2j - a) - H(a) + H(b),$$

$$m(a,b,c) = w(c)B\big(a, b, p(b,c)\big).$$

The Lean integer division is Euclidean division. The checked identity `digit_is_remainder` states $d(b,c) = (2b - 3c) \bmod 7$, with $0 \le d(b,c) < 7$. Thus the pivot agrees with the ceiling used in the ordinary proof:

$$p(b,c) = \left\lceil \frac{3c - 2b}{7} \right\rceil.$$

The source proves the Euclidean remainder identity; this displayed translation into ceiling notation explains the convention.

The formal lattice endomorphism and lift are

$$P(a,b,c) = (2a - b,\ 4a - 3c,\ 5a - 2c),$$

$$L(a,b,c) = \big(p(b,c),\ 2p(b,c) - a,\ 20p(b,c) + 5b - 8c\big).$$

`principal5_cube` proves $P^3(v) = 7v$. `lift_identity` proves

$$P(L(a,b,c)) = (a,\ b - 8d(b,c),\ c - 5d(b,c)).$$

The weight is preserved by the lift. The wall is in $\{-1, 0, 1\}$, so the correction coefficient is in $\{-4, -1, 0, 1, 4\}$. These are universal integer identities.

2 Explicit families and the cardinality boundary

The new module defines

$$Z_n(i,j,k) = (-i,\ j,\ 5(n + k)),$$

$$P_n(i,j,k,r) = (-i,\ j,\ 5(n + k) - r).$$

Here j is a free family index, distinct from the pivot $p(b,c)$. The order lemmas have the exact hypotheses

$$n > 0, \quad i > 0, \quad j > 0, \quad j \le n, \quad k > 0,$$

and the pole lemma additionally assumes $1 \le r \le 4$. They conclude

$$m(Z_n(i,j,k)) = 4, \qquad m(P_n(i,j,k,r)) = -1.$$

The order statements do not require $i \le n$ or $k \le n$. Those additional bounds enter the separate box lemmas.

For the zero-family injection theorem, equality of two points with the same n implies equality of i, j, k, without positivity assumptions. The pole-family injection theorem also recovers r, assuming both residue indices lie in $[1,4]$. Disjointness of the two families only requires the pole residue bound $1 \le r \le 4$.

If $n \ge 1$ and $1 \le i, j, k \le n$, the checked boxes give

$$\begin{aligned} &-n \le a < 0, \qquad 0 < b \le n, \\ &5n < c \le 10n && \text{for } Z_n, \\ &5n < c < 10n && \text{for } P_n. \end{aligned}$$

A positive i excludes the origin in both families. No origin exclusion is silently inferred from an analytic convention.

The traditional manuscript applies these ingredients to

$$\lambda(a,b,c) = \frac{a\tau + bs + c/5}{\beta}.$$

The injectivity of that complex embedding, the finite-cardinality application giving n^3 and $4n^3$ points, the triangle inequality in the complex plane, and the identification of m with actual meromorphic orders are outside this formal block. Accordingly, the resulting cubic disk-counting theorem is an ordinary theorem supported by checked arithmetic ingredients. It is not itself a Lean theorem here.

3 Complete current declaration inventory

There are 14 inherited theorem declarations in `H12PrincipalDivisor` and 11 new v0.14 declarations in `H12CubicDivisorGrowth`. All are freshly replayed in the current-block run. The following table covers all 25, without counting definitions as theorems.

Declaration	Exact scope
`H12PrincipalDivisor`	
`principal5_cube`	The lattice endomorphism cubed is multiplication by seven.
`lift_identity`	Exact lifted image with both residue carries.
`digit_is_remainder`	Digit equals the Euclidean remainder modulo seven.
`digit_range`	Digit lies in the interval from zero to six.
`unique_lift_digit`	Uniqueness under the residue range and lift equation.
`lift_preserves_fifth_weight`	The lifted fifth-residue weight is unchanged.
`wall_bounds`	Wall lies between minus one and one.
`wall_values`	Wall is minus one, zero, or one.
`order_difference`	Coefficient is the difference of lifted/base cone weights.
`order_values`	Coefficient is one of the five possible values.
`tau_ray_order`	Positive tau-ray coefficients equal four.
`s_ray_order`	Nonzero s-ray coefficients equal minus four.
`negative_integer_order`	Positive-index negative-integer coefficients equal four.
`zero_order`	Coefficient at the origin equals zero.
`H12CubicDivisorGrowth`	
`positive_pivot`	The strict chamber inequality makes the pivot positive.
`chamber_wall`	The specified sign chamber has wall one.
`chamber_order`	Chamber coefficient equals its fifth-residue weight.
`zero_family_order`	Positive-family coefficient equals four.
`pole_family_order`	Pole-family coefficient equals minus one.
`zero_family_injective`	Equal zero points recover the three free indices.
`pole_family_injective`	Equal pole points recover all four indices.
`families_disjoint`	No zero point equals a pole point with residue one to four.
`zero_box_bounds`	Closed upper coordinate bound for the zero box.
`pole_box_bounds`	Strict upper coordinate bound for the pole box.
`origin_not_in_families`	Positive first index excludes the origin in both families.

The full machine-readable inventory also records every definition and structure, source file and line, exact source signature, and compiler-expanded theorem type. Definition entries do not add to the theorem count.

4 Deduplicated continuation-source catalogue

The selected continuation tree contains 42 mathematical modules and 453 distinct theorem names. It includes the current 25 as a subset. Selection uses the newest retained copy of each module; every repeated occurrence has the same file hash. Foundation archives v0.1–v0.3 add provenance occurrences, not extra copies or declarations. The last column distinguishes modules with transitive Mathlib dependencies from those using bundled Std.

Module	Theorems	Source	Dependencies
H12AnchorObstruction	9	v0.10	Std
H12CMNormCertificate	8	v0.6	Mathlib
H12CMThreefold	11	v0.7	Std
H12CanonicalTheta	10	v0.6	Mathlib
H12ConjugationOrder	25	v0.7	Std
H12ConjugationRelations	8	v0.7	Std
H12ContractingRigidity	3	v0.6	Mathlib
H12CoprimeNormalization	8	v0.6	Mathlib
H12CubicBranch	4	v0.6	Mathlib
H12CubicDivisorGrowth	11	v0.14	Std
H12DescentPacket	21	v0.9	Std
H12DivisorCarries	6	v0.13	Std
H12DualFourier	9	v0.11	Std
H12ExplicitDescent	20	v0.9	Std
H12FifthIsogeny	8	v0.6	Mathlib
H12FifthModuli	14	v0.6	Mathlib
H12FifthTorsionAlgebra	8	v0.6	Mathlib
H12FlatGerbeAction	9	v0.8	Std
H12FourierExtension	5	v0.11	Std
H12GerbeDescent	29	v0.8	Std
H12LineNorms	4	v0.6	Mathlib
H12MixedGauge	13	v0.10	Std
H12NormAlgebra	10	v0.6	Mathlib
H12NormProjector	10	v0.6	Mathlib
H12NormSemigroup	6	v0.6	Mathlib
H12NormTransfer	13	v0.12	Std
H12OrbitAntidifference	14	v0.11	Std
H12OrbitObstruction	11	v0.6	Mathlib
H12PacketCarries	5	v0.6	Mathlib
H12PhaseRegulator	7	v0.6	Mathlib
H12PolarizedReality	6	v0.6	Mathlib
H12PrincipalDivisor	14	v0.14	Std
H12PrincipalModel	4	v0.12	Std
H12PrincipalSeven	17	v0.12	Std
H12SevenClassDetector	9	v0.10	Std
H12SignedAlgebra	11	v0.6	Mathlib
H12SignedGeometry	18	v0.6	Mathlib
H12Taps	8	v0.6	Mathlib
H12TateMarking	9	v0.6	Mathlib
H12ThreeStepTransport	4	v0.6	Mathlib
H12UnitCoinvariants	21	v0.10	Std
H12WeilCharacter	13	v0.8	Std
Total (distinct names)	453		

5 Conditional results that must retain their hypotheses

The continuation lemmas can be useful without supplying the final arithmetic comparison. Four examples show exactly how the distinction is represented in the types.

The arithmetic candidate cube

`H12.CMNormCertificate` uses an explicitly defined integer polynomial `u21`. Over any commutative ring, its cleared relation assumes

$$b^3 = 7, \qquad x^3 = 2 + 3w, \qquad w^2 + w + 1 = 0.$$

Write $U = u_{21}(b, x, w)$, $A = \mathrm{unitA}(b, x, w)$, and $A_7 = \mathrm{unitAI7}(b, x, w)$, using the Lean definitions `u21`, `unitA`, and `unitAI7`. The exact conclusion is

$$U^3(2 - wb)A_7 = 64827\, w(2 + (1 + w)b)A.$$

Over a field of characteristic zero, `normalized_cube_relation` proves

$$\left(\frac{U}{21}\right)^3 (2 - wb)\frac{A_7}{7} = w(2 + (1 + w)b)A.$$

The quotient conclusion `cube_as_relative_unit` additionally requires $2 - wb \neq 0$ and $A_7/7 \neq 0$. No gamma quantity appears in these theorems. Conversion of the original arithmetic formula into the polynomial normal form is checked separately by exact computations; it is not inferred from these polynomial identities.

Reality and branch separation

For complex z, u, c, `normalized_reality_implies_cube` assumes

$$c \neq 0, \qquad \|z\| = \|u\|,$$

$$\mathrm{Im}(cz^3) = \mathrm{Im}(cu^3) = 0,$$

$$\mathrm{Re}(cz^3) > 0, \qquad \mathrm{Re}(cu^3) > 0.$$

It concludes $z^3 = u^3$. The full comparison `normalized_reality_with_branch` additionally assumes

$$u \neq 0, \qquad \|z/u - 1\| < 1,$$

and concludes $z = u$. Independently, `H12.CubicBranch.cube_comparison_with_enclosure` assumes only $u \neq 0$, $z^3 = u^3$, and that distance bound. An enclosure supporting the distance bound does not establish exact reality or the cube equality. In particular, the unresolved exact reality of the normalized gamma cube remains an unresolved premise.

A contraction lemma

`H12.ContractingRigidity.continuous_contraction_invariant_constant` accepts $d : \mathbb{C} \to \mathbb{C}$, $\rho \in \mathbb{C}$, and assumes

$$\|\rho\| < 1, \qquad d \text{ is continuous at } 0, \qquad \forall z,\ d(\rho z) = d(z).$$

It proves $d(z) = d(0)$ for every z. Adding $d(0) = 1$ gives a trivial phase. The theorem does not construct a gamma comparison multiplier or prove its continuity, contraction invariance, periodic descent, or normalization. Those inputs cannot be replaced by sampled numerical returns.

A regulator endpoint identity

For $g : \mathbb{R} \to \mathbb{C}$, `H12.PhaseRegulator.regulator3` defines

$$\mathcal{R}(g) = -2\int_0^{1/3} xg(x)\,dx + \int_{1/3}^1 (x-1)g(x)\,dx.$$

The endpoint theorem assumes, for every real x, that f has derivative $f_1(x)$, that f_1 has derivative $f_2(x)$, and that f_2 is continuous. It proves

$$\mathcal{R}(f_2) = 3f(1/3) - 2f(0) - f(1).$$

If $e^{f(1)} = e^{f(0)}$ is additionally supplied, the periodic-cube theorem concludes

$$e^{\mathcal{R}(f_2)}(e^{f(0)})^3 = (e^{f(1/3)})^3.$$

The comparison lemmas require the corresponding derivative and endpoint conditions for two supplied functions, and either a shared continuous second derivative or equality of the exponentiated regulator values. They do not identify those functions with the gamma correction and the arithmetic model.

6 Mapping to the traditional manuscript

Traditional claim	Formal coverage and remaining input
Principal lattice transport and coefficient formula	The current 14 inherited integer lemmas prove the lattice cube, lift/remainder identities, wall bounds and coefficient values. Their interpretation as complex divisor orders is ordinary mathematics.
Explicit cubic-growth families	The current 11 new lemmas prove coefficient signs and values, injections, disjointness, boxes, and origin exclusion. Complex embedding, finite cardinalities, and disk counting remain ordinary arguments.
Theta-section obstruction	No formal theta growth or Jensen zero-counting theorem is instantiated for the principal correction.
Canonical genus-three product	Convergence, local divisor equality, minimal ungrouped absolute genus, tail bounds, and logarithmic differentiation remain ordinary analysis.
Reflection and special-point cube formula	The exact meromorphic identities and normalization coefficients are not formalized by the coefficient modules.
Arithmetic candidate cube	The inherited polynomial block proves cleared/normalized identities under visible algebraic relations. No gamma value is substituted.
Original gamma/arithmetic equality	Conditional branch, reality and rigidity lemmas are checked implications. The missing comparison hypotheses remain open.

The machine-readable `CLAIM_COVERAGE.json` identifies exact declarations for each covered component. It marks the analytic and arithmetic-comparison obligations explicitly. A source theorem count measures the selected formal source, not completion of the research target.

7 Fresh replay, resource bounds and provenance

The final ledger combines disjoint theorem-name sets: 438 complete module-scoped audits, eight focused `H12CMNormCertificate` audits, and seven direct-source `H12PhaseRegulator` audits. Their union is exactly the 453 selected names, with no repeats or missing entries. All canonical and preserved archive-source hashes match.

The first 1536 MiB attempt compiled 38 of 42 modules but failed at the interpreter memory threshold for three analysis modules; a fourth module was dependency-blocked. The aggregate auditor and combined import also failed. At 3072 MiB, all 42 modules compiled, while the two aggregate processes still exceeded that cap. Those failures are retained with their exact diagnostics and exit codes.

The module-scoped pass then had a 120-second wall timeout in `H12CMNormCertificate`; its unchanged focused replay and eight audits passed under a 300-second budget. The import-based `H12PhaseRegulator` auditor still exceeded 3072 MiB. Its replacement auditor consists of the *exact original source-byte prefix*, followed only by declared `#check` and `#printaxioms` commands. The recorded prefix hash equals the original source hash. This re-elaborates the same proofs and prints their types and axioms without importing its own serialized module. That source-prefix replay and all seven audits passed at the same 3072 MiB cap and a 300-second budget.

The current Std-only run uses one worker, a 1536 MiB Lean heap threshold, and 120-second per-process wall/CPU budgets. The complete inherited evidence uses one worker and 3072 MiB, with the two focused retries budgeted at 300 seconds. Actual kernel-reported per-process peak resident memory reached 784.7 MiB for the current-block run and 2435.0 MiB across successful final module/audit processes. Exact per-process wall and CPU times and RSS are in the JSON audits. Initial attempts without per-process RSS measurement say so; no missing historical peak is reconstructed.

The verifier scans the accepted mathematical sources for admitted proofs, project axioms, `unsafe`, and `native_decide`. Each selected theorem receives both a compiler type report and a transitive axiom report. The permitted dependencies are `propext`, `Classical.choice`, and `Quot.sound`; these are ordinary foundational dependencies in Lean. An empty text scan alone would not establish kernel acceptance.

A valid arithmetic control is accepted. The deliberate proposition $(0 : \mathbb{Z}) = 1$ with a `decide` proof is rejected. This control supplements the module builds and axiom audits; it is not a replacement for them.

Four output payloads named in the foundation v0.2 historical replay receipt were not preserved. Those old output claims remain unverified; the available source and archive bytes are separately authenticated. The fresh compiler evidence here is independent of that receipt.

The release source files are immutable copies of the preserved archive modules. The replay builds into a new temporary directory, hashes its inputs before and after, and does not reuse prior project object files. Pinned Mathlib object files provide the dependency cache; no mathematical module from this project is taken from an old compiled cache.

The runtime is Lean 4.34.0-rc2, commit `6a10ac8c22beadecabdbb0919c2b50214762f91d`. Mathlib is pinned to `aeb72f1e9981b6d35b426b16dca7759be3e7ba06`. The producing environment uses the same narrow executable-self-location adapter documented in the runtime evidence. It redirects the current process's unavailable executable link to its equivalent self link; the compiler and proof kernel are unchanged. Ordinary Lean installations do not need this adapter.

8 Portable verification

Install the pinned Lean toolchain. The current block uses only bundled Std and can be checked with

```
python3 verify.py --lean /path/to/pinned/bin/lean --scope current
```

The full selected-source replay uses separate module-scoped axiom auditors, with the exact-source-prefix method for `H12PhaseRegulator`. The single all-import process exceeded the recorded memory caps; its aggregate failure remains visible. The replay needs the pinned Mathlib checkout and its cache. The provided `lakefile.toml` records the exact git revision. Obtain that dependency and its matching cache with the normal Lake commands before the full replay:

```
lake update
lake exe cache get
python3 verify.py --lean /path/to/pinned/bin/lean \
  --mathlib .lake/packages/mathlib --scope all \
  --memory-mib 3072 --timeout 300 --audit-strategy auto
```

The script processes modules sequentially with `-j1`, treats warnings as errors, imposes the stated Lean heap and per-process wall/CPU budgets, and records actual child-process peak RSS using POSIX `wait4`. It checks the compiler commit and Mathlib revision. The `auto` strategy selects the recorded source-prefix method for `H12PhaseRegulator`; other modules use import-based auditors. The memory option is a Lean heap threshold, not an operating-system address-space limit. A resource-bound failure is recorded as incomplete evidence; it does not silently reduce the claim count. A focused retry can use `--only` with the exact module name and a separately named log directory.

The verifier's current default is 1536 MiB, appropriate for the Std-only block. The recorded full replay uses 3072 MiB because the lower cap failed for some analysis imports. A machine without enough available memory can still reproduce the current block; it must label a truncated inherited replay accordingly. Runtime/compiler archives and Mathlib caches are omitted from the portable publication package.

9 Remaining formal obligations

An end-to-end formal version of the traditional manuscript would still require the following bridges:

1. Define the retained period embedding and actual gamma/theta functions, prove meromorphic continuation and the full divisor formula, and link integer coefficients to analytic orders.
2. Formalize complex disk cardinalities and the upper/lower cubic bounds, then instantiate the theta-growth and Jensen arguments with their exact restrictions.
3. Construct the genus-three regularizer, prove its convergence and normalization, and justify the fourth logarithmic derivative and its rearrangements.
4. Prove the reflection and period-one identities, coefficient formulas, and branch-independent cube representation for the actual correction.
5. Supply an independent arithmetic interpretation or the missing exact comparison premises. In particular, prove exact normalized reality, periodic descent, and the required principal-norm relation where used.
6. Connect finite exact computations and interval enclosures to the hypotheses of the formal comparison lemmas through a verified input and arithmetic interface.

These obligations are not supplied as project axioms. The formal companion is complete only for the exact types recorded in its inventory and successful compiler evidence.