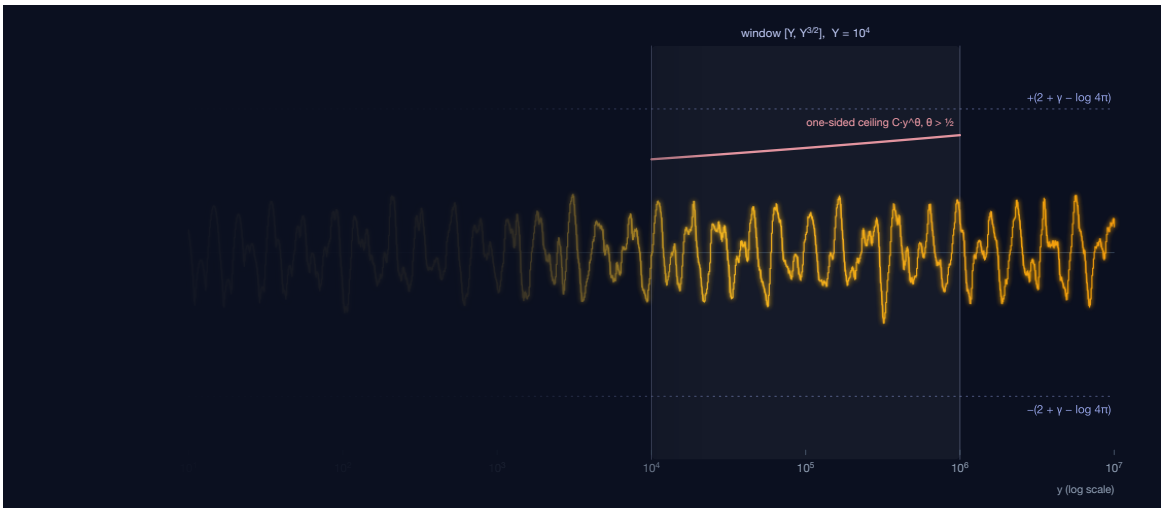
AN ILLUSTRATED BLUEPRINT

The architecture of the announced proof of the Riemann hypothesis

Levent Alpöge's announcement of 20 November 2026 and its Lean archive of 175,187 modules: 14 components in four strata, 14 interfaces, two machine-checked implications and 107 pinned declarations

Companion: the interactive atlas `atlas/index.html` in this repository (`riemann-rev`). Data: `data/model.json`, built from the reading project at 348ec487 (imported 21 November 2026, 00:17 UTC). Lean sources pinned to Deicyde/riemann-hypothesis at 40e56bab (archive toolchain leanprover/lean4:v4.33.1, Mathlib 0df444a3). All times UTC.



The centered prime sum, normalized. Computed: $(G(y) - \log 2\pi)/\sqrt{y}$ for $10 \le y \le 10^7$, where $G(y) = \sum_{n \le y} \Lambda(n)(y/n-1) - y \log y + (1+\gamma)\, y$. Dotted lines: $\pm(2+\gamma-\log 4\pi)$, the band that RH forces. Shaded column: one power window $[Y, Y^{3/2}]$, $Y = 10^4$, with an illustrative one-sided ceiling $C\, y^{\theta}$ ($\theta = 0.55$) in the same normalization.

STATUS as of 20 November 2026, 23:51 UTC (headline of the reading project's status record)

"Two independent kernels accepted the complete proof, and all five theorems matched the canonical Mathlib challenge."

Announced proof

Of RH and of GRH for Dirichlet characters, in a four-page announcement with a Lean archive. This document maps its architecture; it does not assert that the Riemann hypothesis is proved.

✓✓ Checked by two kernels
✓ Kernel-checked at the archive pin

Machine-checked implications, with their remaining inputs as explicit premises (Section 7). Cut I: One-sided prime windows imply RH (checked by two kernels). Cut II: `TopG` and the cohomological windows imply RH and Dirichlet GRH (kernel-checked at the archive pin).

↗ Full replay reported

Externally reported full replay. As recorded by the reading project: "Andrew Sutherland and Simon `DeDeo` each report independently building and replaying all 175,187 original modules with Lean 4.33.1. `DeDeo` reports zero errors and crashes. Sutherland additionally records a con-leche comparison of the five targets and a lean4lean check of every original module. Five modules required higher reduction limits. The individual logs identify the names, flags and successful reruns." These are an external builder's results, not checks made for this document. The reading project adds: ""

◇ Source-read

Source-read. The producers of the arithmetic inputs were located at pinned source lines and read, not separately replayed.

Contents

1 How to read this blueprint 2

1.1 Verification tiers . . . 2

1.2 The four strata . . . 3

1.3 Ways into the blueprint . . . 3

2 The proof in one picture 4

3 The destination 5

1 The five endpoints . . . 5

2 Reflection . . . 7

3 Conductor descent . . . 8

4 The analytic engine 9

4 The centered prime sum . . . 9

5 Mellin transform and positive residues . . . 10

6 The window criterion . . . 11

7 The order of choices . . . 13

5 The estimate 15

8 The finite character comparison . . . 15

9 The geometric family and the theta chain . . . 16

10 Hosts for every character . . . 18

6 Arithmetic supply 19

11 The cohomological window . . . 19

12 Auxiliary primes and Behrend towers . . . 21

13 The Galois row . . . 22

14 Seven inputs at the decisive call . . . 23

7 The verified cuts and the Lean skeleton 25

7.1 The architecture as a checked skeleton . . . 26

8 The blueprint: 107 declarations 32

9 The announcement, entry by entry 40

10 Sources, pins and reproduction 42

10.1 Pins . . . 42

10.2 The status record . . . 42

10.3 Provenance of the imported data . . . 43

10.4 Reproduction . . . 43

1 How to read this blueprint

This is a map of an *announced* proof. Levent Alpöge's four-page announcement of 20 November 2026 claims proofs of the Riemann hypothesis and of the generalized Riemann hypothesis for Dirichlet characters, accompanied by a Lean archive of 175,187 modules. The map divides the argument into 14 mathematical components in four strata, joined by 14 *interfaces*: the propositions through which the output of one component becomes the input of another. For each component it gives a plain-language gist, a precise statement in standard notation, an explanation of the idea, the pinned Lean declarations, a verification tier with its status, and what remains to be checked.

The statements are paraphrases, in standard notation, of the reading project's chapters and of the archive's source lines they cite; nothing here is a new proof, and this repository does not re-check the archive. Two implications have been machine-checked with their remaining inputs as explicit hypotheses (Section 7). A full replay of the archive has been reported by an external builder; the reading project records its own checks separately, and its status record of 20 November 2026, 23:51 UTC is quoted in Section 10.2. Nothing in this document asserts that the Riemann hypothesis is proved.

Reading order. The components are numbered as in the model, from the destination backwards: the five exported statements and the classical reductions to the half-plane theorem (Section 3), the analytic engine that turns one-sided prime-sum bounds into zero-free half-planes (Section 4), the estimate that produces those bounds (Section 5), and the arithmetic supply on which the estimate rests (Section 6). The picture of Section 2 runs the other way, from the arithmetic on the left to the endpoints on the right.

Links. A monospace name such as `Deduction.poleData` links to its source lines in the public mirror Deicyde/riemann-hypothesis, pinned at commit 40e56bab, whose tree is identical to the archive; circled numbers and node titles link inside this document.

1.1 Verification tiers

Every component and every blueprint node carries one tier. The meanings are those of the model; node tiers are computed from the receipts of the reading project.

✓✓ Checked by two kernels	Accepted by two independent proof checkers in runs of the reading project: a Lean replay or verified-mode comparison, and the Rust kernel NanoDA. The ledger names the runs and their receipts.
✓ Kernel-checked at the archive pin	Named declarations passed type, transitive-axiom (only `propext`, `Classical.choice`, `Quot.sound`) and kernel-replay checks at Lean 4.33.1.
◐ Partly checked	Some of the component's named declarations are kernel-checked; the producers of its remaining inputs are source-read.
⊂ Inside a checked closure	Called (directly or transitively) by a kernel-checked declaration, so it was elaborated and kernel-checked as part of that closure and its axioms are within that declaration's report. For a definition this only means that it typechecks: a proposition it defines (for instance `VWinAt`) can still be a hypothesis.
◇ Source-read	Statement and proof term located at pinned source lines and read; not replayed as a named declaration by the reading project's audit. It is part of the complete proof, whose replays are listed in the ledger.
↗ Full replay reported	Checked in a full replay of the archive reported by an external builder; the reading project has not (yet) recorded a run of its own for it.
○ Motivation	A heuristic or historical idea from the announcement; no theorem is attached to it.

1.2 The four strata

- **Arithmetic supply.** Galois representations, theta lifts and Iwasawa-type cohomology bounds that are meant to produce the families on which a prime sum is small.
  11 The cohomological window, 12 Auxiliary primes and Behrend towers, 13 The Galois row, 14 Seven inputs at the decisive call
- **The estimate.** A finite character comparison turns geometric control of a chain of theta lifts of length about $\sqrt{x}$ into an upper bound for a weighted prime sum.
  8 The finite character comparison, 9 The geometric family and the theta chain, 10 Hosts for every character
- **The analytic engine.** A localized Landau oscillation argument converts one-sided bounds on whole power windows into a zero-free half-plane.
  4 The centered prime sum, 5 Mellin transform and positive residues, 6 The window criterion, 7 The order of choices
- **The destination.** Euler factors, conductors and functional equations turn the half-plane theorem into the five exported statements, including Mathlib's `RiemannHypothesis`.
  1 The five endpoints, 2 Reflection, 3 Conductor descent

1.3 Ways into the blueprint

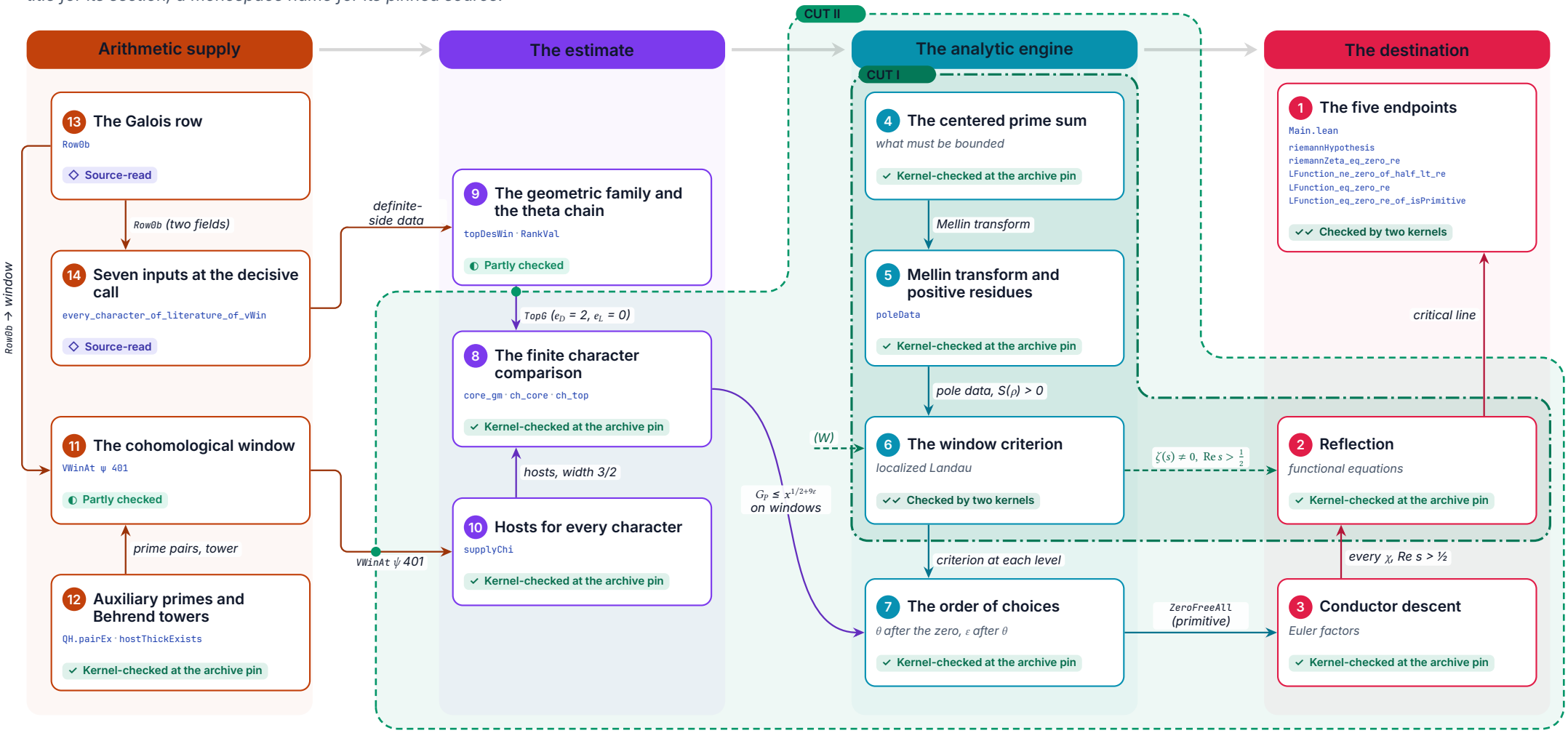
The reading project proposes five itineraries through the blueprint graph; each step links to the declaration table of Section 8.

1. **Read the last page of the proof first.** *How does a right-half-plane estimate imply ordinary RH?*
   ✓✓ Mathlib RH → ◇ The two zeta exports share a proof → ◇ Specialize at modulus one → ✓ Reflect a zeta zero
2. **Find the mathematical estimate.** *How wide must the intervals be, and where does the loss in the exponent come from?*
   ✓ Produce primitive zero-free levels → ⊂ Choose epsilon after a candidate zero → ⊂ The prime-sum inequality on tied families → ✓ From the height discrepancy to the prime sum → ◇ Choose a common geometric family → ◇ Combine divisibility, slopes and windows
3. **Find who supplies the arithmetic.** *Which assertion has to produce the compatible Galois, theta and cohomology data?*
   ◇ Closed half-plane assembly → ◇ The half-plane theorem from one row → ◇ Producer of the Galois row → ◇ Assemble the automorphic inputs → ◇ Seven inputs for every character → ◇ Cohomological window from the row → ◇ Produce the four counting bounds
4. **Follow a singular twist and an arbitrary finite prime sum.** *Which bound controls a twist of unbounded abstract length, and how can one estimate work over every allowed finite set of coefficient primes?*
   ◇ Produce the four counting bounds → ◇ Supply the absolute bound for the reference layer → ◇ Combine the cohomology counts → ✓ Assemble the arithmetic tower window → ✓ From a finitely generated tower to two length bounds → ⊂ Control exceptional twists by the fixed layer → ✓ Choose one set of tower constants → ◇ Case I gives one bound across coefficient primes → ◇ A nonzero integer controls one character orbit → ◇ Convert the joint window into `VWin` → ◇ Reuse the tower interval for the positive part → ◇ Combine the two prime sums at a point → ◇ Add contributions inside the exceptional set → ◇ Bound the sum over arbitrary finite sets of other primes
5. **Recover the centering constant in the ordinary zeta window.** *Why is the linear correction $(1+\gamma)x$?*
   ⊂ The first moment of the Chebyshev error → ✓ Evaluate the first moment → ⊂ Match the two limits at one → ⊂ Dominate the centered character sum → ⊂ Pass the Mellin integral to $\sigma = 1$ → ⊂ Read Euler's constant from the logarithmic derivative

The proof in one picture

status as of 20 November 2026, 23:51 UTC · Lean names link to the mirror at 40e56bab

From the arithmetic supply (left) to the exported statements (right). Arrows: the 14 interfaces, labelled by the proposition they carry. Dashed frontiers: the two machine-checked implications. Click a title for its section, a monospace name for its pinned source.



CUT I ✓✓ Checked by two kernels details →
One-sided prime windows imply RH
premise $\forall \theta \in (\tfrac{1}{2}, 1)\ \exists C\ \forall Y_0\ \exists Y \geq Y_0\ \forall y \in [Y, Y^{3/2}]:\ G(y) \leq C y^{\theta}$
⟹ Mathlib's `RiemannHypothesis`

CUT II ✓ Kernel-checked at the archive pin details →
`TopG` and the cohomological windows imply RH and Dirichlet GRH
premise $\mathrm{TopG} \wedge \forall \psi$ primitive : $\mathrm{VWinAt}\ \psi\ 401$
⟹ RH ∧ half-plane nonvanishing ∧ critical-strip zeros, for every positive modulus

Verification tiers
- ✓✓ Checked by two kernels
- ✓ Kernel-checked at the archive pin
- ◐ Partly checked
- ⊂ Inside a checked closure
- ◇ Source-read
- ↗ Full replay reported
- ○ Motivation

Numbers follow the reading order of Sections 3–6 (destination first). Green dots: where the premises of Cut II cross its frontier. (W): where the premise of Cut I enters the window criterion; its zeta route bypasses the order of choices and the conductor descent. Tiers as of 2026-11-20 23:51 UTC.

3 The destination

Euler factors, conductors and functional equations turn the half-plane theorem into the five exported statements, including Mathlib's `RiemannHypothesis`.

COMPONENTS

1. The five endpoints
2. Reflection
3. Conductor descent

ACROSS THE STRATUM BOUNDARY

7 The order of choices → 3 Conductor descent
`ZeroFreeAll` *(primitive)*

1 The five endpoints

✓✓ Checked by two kernels

`Main.lean`

Ordinary RH in Mathlib's own formulation, and Dirichlet GRH for every character of every positive modulus.

STATEMENT

$$\zeta(s) = 0,\ s \notin \{-2, -4, \dots\},\ s \neq 1 \implies \operatorname{Re} s = \frac{1}{2};$$
$$L(s, \chi) = 0,\ 0 < \operatorname{Re} s < 1 \implies \operatorname{Re} s = \frac{1}{2}.$$

The first export has exactly Mathlib's type `RiemannHypothesis`. The other four state the zeta zeros in expanded form, nonvanishing of $L(s, \chi)$ for $\operatorname{Re} s > \frac{1}{2}$ away from the principal pole, and the critical-line conclusion in the open strip for all characters and for primitive ones.

All five pass through one assertion: the half-plane theorem for every Dirichlet character. The extension to further L-functions mentioned in the announcement has no endpoint in the archive.

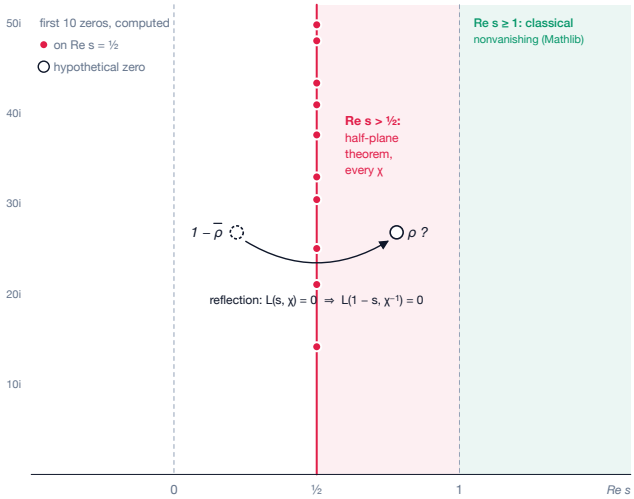


Figure 1. Critical strip, reflection and the half-plane theorem. Schematic plane with the computed first zeta zeros on the critical line; a hypothetical off-line zero ρ and its mirror image $1 - \overline{\rho}$ under the functional equation.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`riemannHypothesis` — type: `RiemannHypothesis` (Mathlib)	Main.lean, L4
`riemannZeta_eq_zero_re` — the same statement, quantifiers written out	Main.lean, L6
`LFunction_ne_zero_of_half_lt_re` — half-plane nonvanishing, all characters	Main.lean, L8
`LFunction_eq_zero_re` — critical-strip zeros, all characters	Main.lean, L10
`LFunction_eq_zero_re_of_isPrimitive` — critical-strip zeros, primitive characters	Main.lean, L12
Statements (target types)	Statements.lean, L3–16
Mathlib `RiemannHypothesis` — Mathlib 0df444a3	RiemannZeta.lean, L182–186 (Mathlib)

USES ⟵

2 Reflection · *critical line*
`Claim.zeta` · `Claim.strip`

SUPPLIES ⟶

— the exported statements

BLUEPRINT NODES 13 declarations: 5✓✓, 8◇ (Section 8)

✓✓ Mathlib RH ✓✓ Zeta in expanded form ✓✓ Nonvanishing right of one half ✓✓ Dirichlet GRH in the open strip
✓✓ Primitive critical-strip zeros ◇ Zeta target type ◇ Half-plane target type ◇ Dirichlet strip target type
◇ Primitive strip target type ◇ The two zeta exports share a proof ◇ All-character half-plane theorem
◇ Primitive critical-line wrapper ◇ Three endpoint propositions

STATUS as of 2026-11-20 23:51 UTC ✓✓ Checked by two kernels

Two independent kernels accepted the complete proof, and all five theorems matched the canonical Mathlib challenge. Andrew Sutherland and Simon `DeDeo` each report independently building and replaying all 175,187 original modules with Lean 4.33.1. `DeDeo` reports zero errors and crashes. Sutherland additionally records a con-leche comparison of the five targets and a lean4lean check of every original module. Five modules required higher reduction limits. The individual logs identify the names, flags and successful reruns. (Status recorded by the reading project at 2026-11-20 23:51 UTC.)

WHAT REMAINS TO CHECK

What machine checking cannot settle: that the five Lean statements say what is meant (they are compared with an independently written Mathlib challenge), the soundness of the checkers and of Mathlib's definitions, and the mathematical community's own reading of the argument.

Audit criterion (✓✓ Mathlib RH): *Exact endpoint replay.* Compile the five unmodified exports at the archive pins, verify their exact types, inspect their transitive axioms, and replay the selected kernel checker.

IN THE ANNOUNCEMENT

Entry 20 (11/16/26): Dirichlet GRH · *Endpoint statement matched*

2 Reflection

✓ Kernel-checked at the archive pin

functional equations

A zero left of the critical line reflects to a zero right of it, so the right half-plane is all one has to clear.

STATEMENT

$$\zeta(s) = 2(2\pi)^{-w}\Gamma(w)\cos\left(\tfrac{\pi w}{2}\right)\zeta(w), \quad w = 1 - s;$$
$$\Lambda(s,\chi) = \varepsilon(\chi)\,\Lambda(1-s,\chi^{-1}) \ \ (\chi \text{ primitive}).$$

For zeta, put $w = 1 - s$ with $\operatorname{Re} w > \frac{1}{2}$. The power and Gamma factors do not vanish, and a zero of the cosine would make s an even integer, i.e. 0 or a trivial zero. Hence a nontrivial zero on the left gives a zero on the right.

For a primitive character the completed functional equation sends a strip zero at s to a zero of $L(\cdot, \chi^{-1})$ at $1 - s$; the Gamma factors are regular and nonzero for $0 < \operatorname{Re} s < 1$. Taking χ^{-1} matters when χ is complex.

Figure 1 (*Critical strip, reflection and the half-plane theorem*) illustrates this component as well.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`Formalism.VW.rh_of_right` — reflect a zeta zero	RhOfRight.lean, L5–62
`Claim.prim_reflect` — transfer a primitive zero to 1 − s	ClaimStrip.lean, L42–57
`Claim.strip` — half-plane ⟹ critical-strip zeros on the line	ClaimStrip.lean, L59–73
`Claim.zeta` — half-plane ⟹ the zeta statement	ClaimStrip.lean, L75–100
`Formalism.VW.gammaFactor_ne_zero` — Gamma factors do not vanish in the strip	CritLine.lean, L6–9

USES ←

3 Conductor descent · *every χ, Re s > ½*
`UpperTheory.Top.imprimitive`

SUPPLIES →

1 The five endpoints · *critical line*
`Claim.zeta` · `Claim.strip`

BLUEPRINT NODES 8 declarations: 3 ✓, 1 ⊂, 4 ◇ (Section 8)

◇ Specialize at modulus one ✓ Reflect a zeta zero ◇ Reflect a primitive character ◇ Gamma factors in the strip
✓ Reflect an arbitrary character ⊂ Transfer a primitive zero to 1−s ✓ Zeta component of the conjunction
◇ A second zeta deduction

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`Claim.strip`, `Claim.zeta` and `Formalism.VW.rh_of_right` passed type, axiom and kernel checks at Lean 4.33.1 (Helios job 26639692).

WHAT REMAINS TO CHECK

None for the deduction itself; it is classical and machine-checked.

3 Conductor descent

✓ Kernel-checked at the archive pin

Euler factors

An imprimitive character differs from its primitive inducing character by finitely many Euler factors, none of which vanishes for $\operatorname{Re} s > 0$.

STATEMENT

$$L(s,\chi) = L(s,\chi^*)\prod_{p\mid N}\bigl(1-\chi^*(p)p^{-s}\bigr),$$

$$\bigl|\chi^*(p)p^{-s}\bigr| \le p^{-\sigma} < 1 \quad (\sigma > 0).$$

Primes dividing the conductor have $\chi^*(p) = 0$; for the others the second term of the factor has modulus less than one. So nonvanishing passes from primitive characters to all characters, with the single exception of the principal character's pole at $s = 1$.

The strict left boundary of the strip matters: $1 - 2^{-s}$ vanishes on $\operatorname{Re} s = 0$, so $L(s, \mathbf{1}_2) = (1-2^{-s})\zeta(s)$ has extra zeros on the imaginary axis.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.Top.imprimitive` — primitive ⟹ all characters	Imprimitive.lean, L5–35
`Claim.prim_zero` — remove the imprimitive Euler factors	ClaimStrip.lean, L23–33
`Claim.factor_ne_zero` — a removed Euler factor is nonzero	ClaimStrip.lean, L7–21

USES ⟵

7 The order of choices · *`ZeroFreeAll` (primitive)*
`zeroFreeAll_of_topG`

SUPPLIES ⟶

2 Reflection · *every χ, Re s > ½*
`UpperTheory.Top.imprimitive`

BLUEPRINT NODES 4 declarations: 1 ✓, 2 ⊂, 1 ◇ (Section 8)

⊂ Remove imprimitive Euler factors ⊂ Nonzero removed Euler factor ◇ Pass from primitive to all characters
✓ Restore all characters

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`UpperTheory.Top.imprimitive` passed its type, axiom and defining-module checks in the 1,421-module Dirichlet closure (Helios job 26647685).

WHAT REMAINS TO CHECK

None for the deduction; classical and machine-checked.

4 The analytic engine

A localized Landau oscillation argument converts one-sided bounds on whole power windows into a zero-free half-plane.

COMPONENTS

- 4 The centered prime sum
- 5 Mellin transform and positive residues
- 6 The window criterion
- 7 The order of choices

ACROSS THE STRATUM BOUNDARY

8 The finite character comparison → 7 The order of choices
$G_P \le x^{1/2+9\varepsilon}$ *on windows*

7 The order of choices → 3 Conductor descent
`ZeroFreeAll` *(primitive)*

4 The centered prime sum

✓ Kernel-checked at the archive pin

what must be bounded

A ramp-weighted von Mangoldt sum, centered so that the main terms cancel exactly, and its positive combinations over a cyclic family of characters.

STATEMENT

$$G(y) = \sum_{n \le y} \Lambda(n)\Big(\frac{y}{n} - 1\Big) - y\log y + (1+\gamma)\,y,$$

$$G_P(x) = G(x) + \frac{1}{3o+1} \sum_{\chi \in P \setminus \{1\}} \operatorname{Re} g_\chi(x).$$

Here $g_\chi(y) = \sum_{n \le y} \Lambda(n)\chi(n)(y/n - 1) + y(L'/L)(1, \chi)$ for nonprincipal primitive χ, and P consists of the primitive characters inducing the cyclic group generated by a fixed primitive ψ (order o). At the trivial level $G_P = G$.

By the explicit formula $G(y) = \log 2\pi - \sum_\rho y^\rho/(\rho(\rho-1)) + O(y^{-2})$, so under RH $|G(y) - \log 2\pi| \le (2 + \gamma - \log 4\pi)\sqrt{y}$. The proof needs much less: an *upper* bound $G_P(x) \le x^{1/2+O(\varepsilon)}$ on whole windows.

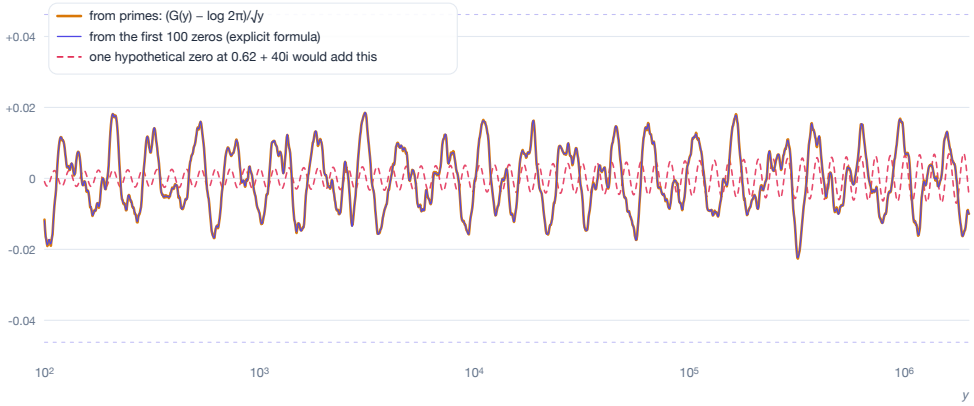


Figure 2. Prime sum versus the sum over zeros. Computed: normalized G(y) from primes (amber) and the explicit formula with the first 100 zeta zeros (indigo), 100 ≤ y ≤ 2e6; dashed: the term a hypothetical zero at 0.62+40i would add (it grows like $y^{0}.12$).

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`Formalism.Gz`	StatementOC.lean, L13
`Deduction.G` / Gchi (`CharFamily`)	CharFamily.lean, L10–45
`Formalism.HC.Level.Gm`	ExplicitFormulaErrorOfHostConditions.lean, L38
`UpperTheory.Comparison₂.Gz_eq`	GzEq.lean, L6–47

USES ⟵

— no incoming interface

SUPPLIES ⟶

5 Mellin transform and positive residues · *Mellin transform*
`MellinRamp`

BLUEPRINT NODES 8 declarations: 1 ✓, 7 ⊂ (Section 8)

⊂ The weighted prime sum ⊂ Uniform families on power windows ⊂ The first moment of the Chebyshev error
✓ Evaluate the first moment ⊂ Match the two limits at one ⊂ Read Euler's constant from the logarithmic derivative
⊂ Dominate the centered character sum ⊂ Pass the Mellin integral to $\sigma = 1$

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`Gz_eq` (`Formalism.Gz` = `An.G` = `Deduction.G` pointwise) passed its checks in a 130-module union; the definitions are inside every checked closure below.

WHAT REMAINS TO CHECK

None: a definition and a checked normalization identity.

IN THE ANNOUNCEMENT

Entry 5 (10/20/26): The smoothed von Mangoldt sum · *Formula matched*

5 Mellin transform and positive residues

✓ Kernel-checked at the archive pin `poleData`

The Mellin transform of a ramp is $n^{-s}/(s(s-1))$; a zero of any family member becomes a pole whose residue cannot be cancelled, because the weights are positive.

STATEMENT

$$\int_1^\infty \Big(\frac{y}{n} - 1\Big)_+ y^{-s-1}\,dy = \frac{n^{-s}}{s(s-1)},$$

$$\operatorname*{Res}_{s=\rho} = -\frac{S(\rho)}{\rho(\rho-1)}, \qquad S(\rho) = \frac{1}{2}\sum_i a_i\big(m_{\chi_i}(\rho) + m_{\chi_i^{-1}}(\rho)\big) > 0.$$

Summing ramps against $\Lambda(n)\chi(n)$ produces $-L'/L$; the compensating linear term removes the pole at $s = 1$, so $\int_1^\infty G_\chi(y)y^{-s-1}dy = -[(L'/L)(s,\chi) - s(L'/L)(1,\chi)]/(s(s-1))$.

Taking real parts averages χ and χ^{-1}, so every member vanishing at ρ contributes a positive multiple of the same complex number. For a real zero $0 < \beta < 1$ the residue is even positive, which lets the one-sided bound exclude exceptional real zeros too.

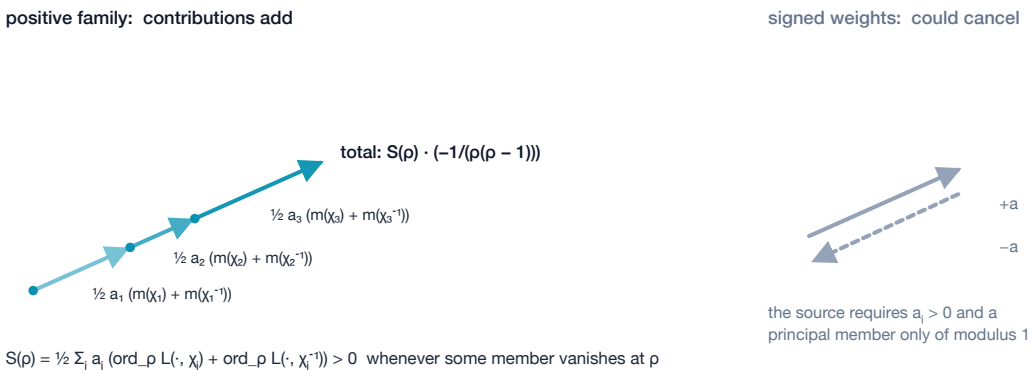


Figure 3. Positive weights keep the residue alive. Schematic, exact in direction: at a common zero ρ every member of a positive family contributes a positive multiple of the same complex number $-1/(\rho(\rho-1))$, so the contributions add; with signed weights two contributions could cancel.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`MellinRamp`	MellinRamp.lean, L10–32
`mellinGchiOfNeOne`	mellinGchiOfNeOne.lean, L116–125
`Deduction.poleData`	poleData.lean, L248–259
positivity of $S(\rho)$	poleData.lean, L336–383

USES ←

4 The centered prime sum · *Mellin transform*
`MellinRamp`

SUPPLIES →

6 The window criterion · *pole data, $S(\rho) > 0$*
`Deduction.poleData`

BLUEPRINT NODES 1 declarations: 1 ✓ (Section 8)

✓ Poles of the finite-family transform

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`Deduction.poleData` passed type, axiom and kernel checks (38-module landmark closure, Helios job 26639692).

WHAT REMAINS TO CHECK

None: an analytic lemma, machine-checked.

6 The window criterion

✓✓ Checked by two kernels *localized Landau*

If a positive family's prime sum stays below Cy^θ throughout arbitrarily late windows $[Y, Y^{1+\delta}]$, none of its L-functions vanishes to the right of θ.

STATEMENT

$$\exists C\, \forall Y_0\, \exists Y \geq Y_0\, \forall y \in [Y, Y^{1+\delta}] :\ G_F(y) \leq C\, y^\theta \implies L(s, \chi_i) \neq 0 \ \text{ for } \operatorname{Re} s > \theta.$$

With $u = \log y$ and $g(u) = G_F(e^u)$, the moments $A_k(z) = \int g(u)\, \frac{u^k e^{-\sigma u}}{k!} e^{-itu}\, du$ are $(-1)^k \Phi^{(k)}(z)/k!$ for the Laplace transform Φ. The kernel peaks at $u = k/\sigma$, so $k = \lceil (1+\delta/2)\sigma U \rceil$ aims it at the window $[U, (1+\delta)U]$; the tails are exponentially small.

The one-sided trick: $|g| + g = 2\max(g, 0) \leq 2R$ for any majorant $R \geq g$, which controls $|A_k(\sigma + it)| +$

Re $A_k(\sigma)$. A pole at distance d contributes about d^{-k} while the window bound gives $e^{\eta U} r^{-k}$ with $r > d$; since $k \geq U$ and η is arbitrary, the pole cannot exist.

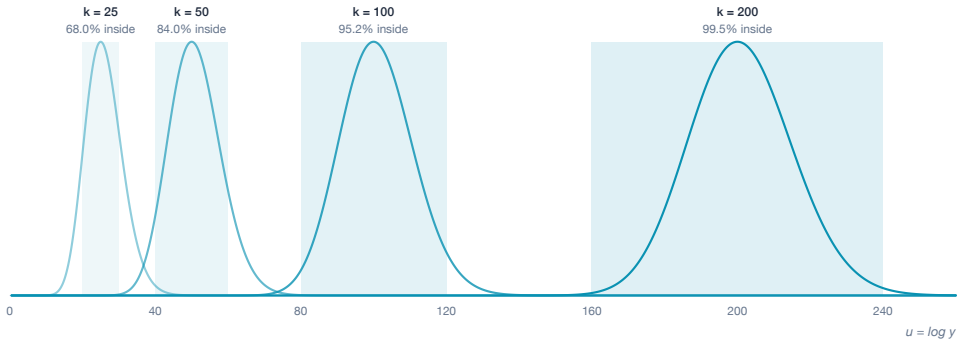


Figure 4. The spotlight of high derivatives. Computed: kernels $u^k\, e^{-u}/\mathrm{k}!$ (normalized to peak 1) with k = ceil(1.25 U) for U = 20, 40, 80, 160 and the windows [U, 1.5 U] in u = log y; labels give the fraction of each kernel's mass inside its window.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`Deduction.zeroFree_of_rpow_window` — the criterion (statement)	ZeroFreeOfRpowWindow.lean, L11–14
`Deduction.landau_window` — rule out a pole right of θ	landauWindow.lean, L83–95
isolation of an offending pole (proof)	landauWindow.lean, L83–116
one-sided moment inequality	landauWindowIneq.lean, L57–99
tail estimate	landauTails.lean, L80–102

USES ←

5 Mellin transform and positive residues · *pole data, $S(\rho) > 0$*
`Deduction.poleData`

SUPPLIES →

7 The order of choices · *criterion at each level*
`zeroFree_of_rpow_window`

BLUEPRINT NODES 4 declarations: 1 ✓, 3 ⊂ (Section 8)

⊂ The analytic criterion for a level ✓ The character-family power-window theorem ⊂ Apply the pole obstruction
⊂ Rule out a pole right of θ

STATUS as of 2026-11-20 23:51 UTC ✓✓ Checked by two kernels

`zeroFree_of_rpow_window` passed at the archive pin; its specialization to zeta (`riemannHypothesis_of_prime_windows`) passed a fresh Lean replay over 10,530 modules and the Rust kernel NanoDA on 61,697 exported declarations.

WHAT REMAINS TO CHECK

None: machine-checked twice.

IN THE ANNOUNCEMENT

Entry 6 (10/8/26): Two-support positivity test · *Historical calculation*

7 The order of choices

✓ Kernel-checked at the archive pin

θ after the zero, ε after θ

The loss exponents are fixed first; given a zero with $\beta > \frac{1}{2}$ choose $\theta \in (\frac{1}{2}, \beta)$, then ε with $\frac{1}{2} + 9\varepsilon \le \theta$. No bound at $\varepsilon = 0$ is ever needed.

STATEMENT

$$\theta = \frac{1}{2} + \frac{1}{2}\min(\beta - \tfrac{1}{2}, \tfrac{1}{2}),$$
$$\tfrac{1}{2} + (e_D + e_L + 7)\,\varepsilon \le \theta \quad (e_D = 2,\ e_L = 0).$$

This is where the arithmetic and the analysis meet. The arithmetic must deliver, for every small ε, families with $G_P(x) \le x^{1/2+(e_D+e_L+7)\varepsilon}$ on arbitrarily late windows $[Y, Y^{3/2}]$, with e_D, e_L independent of ε. The families themselves may depend on ε.

Applying the criterion at each primitive character's cyclic level gives `ZeroFreeAll`: $L(s,\psi) \neq 0$ for $\mathrm{Re}\, s > \frac{1}{2}$ and every primitive ψ.

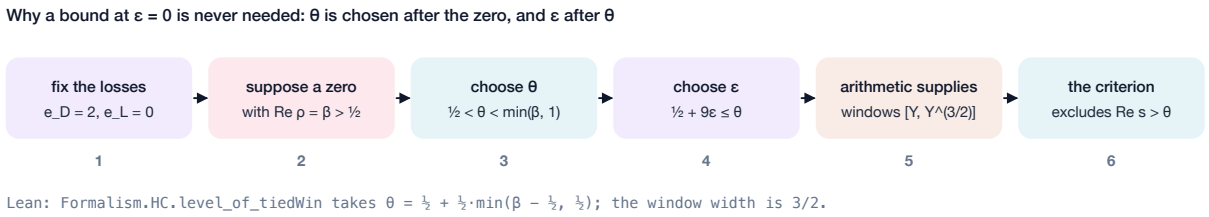


Figure 5. The order of choices (schematic). Schematic: the loss exponents are fixed first; a candidate zero $\beta > 1/2$ is given; θ is chosen between $1/2$ and β; ε is chosen so that $1/2 + (e_D+e_L+7)\,\varepsilon \le \theta$; the arithmetic then supplies windows for that ε; the analytic criterion excludes the zero.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`Formalism.HC.level_of_tiedWin`	— choice of θ and ε after a candidate zero	TopWin.lean, L68–81
`UpperTheory.Top.zeroFreeAll_of_topG`	— `TopG` and hosts ⟹ primitive zero-free half-plane	ZeroFreeAllOfTopG.lean, L26–34
`UpperTheory.Top.zero_free_all_of_green_v_win`	— selects the window width 3/2	TopPlug.lean, L12–15

USES ←

8 The finite character comparison · $G_P \le x^{1/2+9\varepsilon}$ *on windows*
`level_gm_le_on_family_scales`

6 The window criterion · *criterion at each level*
`zeroFree_of_rpow_window`

SUPPLIES →

3 Conductor descent · *`ZeroFreeAll` (primitive)*
`zeroFreeAll_of_topG`

BLUEPRINT NODES 5 declarations: 1 ✓, 3 ⊂, 1 ◇ (Section 8)

◇ Select the window width 3/2 ✓ Produce primitive zero-free levels ⊂ Assemble the cyclic levels
⊂ Pass from levels to characters ⊂ Choose epsilon after a candidate zero

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`zeroFreeAll_of_topG` passed its checks in the 1,421-module Dirichlet closure.

WHAT REMAINS TO CHECK

None for the logic; the premise `TopG` is the arithmetic input.

Audit criterion (⊂ Choose epsilon after a candidate zero): *Separate the analytic implication.* Trace the finite positive-weight character family through `poleData`, `landau_window` and the uniform window bound. Check the pole treatment and the order of quantifiers.

5 The estimate

A finite character comparison turns geometric control of a chain of theta lifts of length about $\sqrt{x}$ into an upper bound for a weighted prime sum.

COMPONENTS

- 8 The finite character comparison
- 9 The geometric family and the theta chain
- 10 Hosts for every character

ACROSS THE STRATUM BOUNDARY

14 Seven inputs at the decisive call → 9 The geometric family and the theta chain
definite-side data

11 The cohomological window → 10 Hosts for every character
VWinAt ψ 401

8 The finite character comparison → 7 The order of choices
$G_P \le x^{1/2+9\varepsilon}$ on windows

8 The finite character comparison

✓ Kernel-checked at the archive pin

`core_gm · ch_core · ch_top`

Finite Fourier orthogonality on a subgroup and an exact main-term identity convert a geometric discrepancy bound into the prime-sum bound, with exponent $\frac{1}{2} + (e_D + e_L + 7)\varepsilon$.

STATEMENT

$$Q\,\frac{\delta}{2}\Big(G_\Lambda(x) - \Sigma(x) - 4x^{1/2+\varepsilon}\Big) \le E + \frac{2Q}{x} \quad\Longrightarrow\quad G_\Lambda(x) \le x^{1/2+(e_D+e_L+7)\varepsilon}.$$

The geometric discrepancy contains a cost proportional to x/m and another proportional to m. With $m = x^a$ the larger exponent is $\max(1-a, a)$, smallest at $a = \frac{1}{2}$: this is the square root. The worst of the four budget terms is $4x^{1/2+13\varepsilon/8}$.

Summing a character over the subgroup annihilates it unless it is trivial there; the survivors that are not retained ($\Sigma \le 4o\,x^{1/2+\varepsilon}$) need a separate square-root input (`EAtH.bdG`). The losses add up to exactly $(e_D + e_L + 7)\varepsilon$.

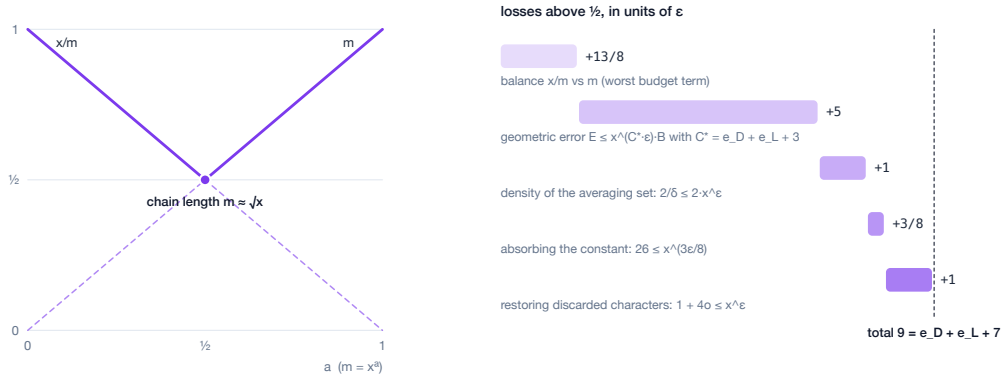


Figure 6. Where the square root and each ε are paid. Left: exponents 1-a of x/m and a of m when m = x^a; the larger is smallest at a = 1/2. Right: exact staircase of the losses in units of ε, from the finite comparison to 1/2 + (e_D+e_L+7) ε = 1/2 + 9 ε for the source's choice e_D = 2, e_L = 0.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.Comparison₂.core_gm` — from the height discrepancy to the prime sum	CoreGm.lean, L12–74
`Deduction.ch_core` — average over characters	ChCore.lean, L343–351
subgroup identity (in `ChCore.lean`)	ChCore.lean, L16–54
`Deduction.ch_top` — bound the discrepancy per index	ChTop.lean, L37–72
`UpperTheory.Comparison₂.windowInequality_of_artinInputs` — the estimate on a whole window	WindowInequalityOfArtinInputs.lean, L17–25

USES ←

9 The geometric family and the theta chain · *`TopG` ($e_D = 2, e_L = 0$)*
`UpperTheory.Top.TopG`

10 Hosts for every character · *hosts, width 3/2*
`UpperTheory.Top.supplyChi`

SUPPLIES →

7 The order of choices · *$G_P \le x^{1/2+9\varepsilon}$ on windows*
`level_gm_le_on_family_scales`

BLUEPRINT NODES 5 declarations: 4 ✓, 1 ⊂ (Section 8)

⊂ The prime-sum inequality on tied families ✓ Estimate the weighted level on a window
✓ From the height discrepancy to the prime sum ✓ Bound the discrepancy per index ✓ Average over characters

STATUS as of 2026-11-20 23:51 UTC — ✓ Kernel-checked at the archive pin

`ch_top`, `ch_core`, `core_gm` and `windowInequality_of_artinInputs` passed at the archive pin (316 modules), with the comparison hypotheses `EAtH` and `CoreAtH` explicit.

WHAT REMAINS TO CHECK

The hypotheses: existence of the hosts, frames and witnesses at every scale of the window, and the square-root bound for discarded characters.

IN THE ANNOUNCEMENT

Entry 14 (11/15/26): The provisional exponent two-thirds · *Intermediate strategy*

Entry 16 (10/14/26): Period pairings and approximate orthogonality · *Supporting comparison*

9 The geometric family and the theta chain

◐ Partly checked — `topDesWin` · `RankVal`

A chain of about $\sqrt{x}$ theta lifts, with divisibility, slope and covolume control, is chosen so that both loss exponents are fixed numbers: $e_D = 2$, $e_L = 0$.

STATEMENT

$$\sqrt{x\delta_1} \le m \le 2\sqrt{x\delta_1}, \quad \delta_1 = x^{-\varepsilon/8} \;\Rightarrow\; m \approx x^{1/2-\varepsilon/16};$$
$$\mu(A) \le \mu_{\max}(B) + \log C;$$
$$w^2 = p \;\Rightarrow\; w \mid W_c V_0 - W_0 V_c.$$

`TopShape` calls `topDesWin` with $C_{\mathrm{abs}} = 2$, $C_0 = 3$, $e_D = 2$, $e_L = 0$, which gives the exponent $\frac{1}{2} + 9\varepsilon$. `RankVal` fixes the chain length; the announcement's "length of chain of theta lifts $\sqrt{X}$" is this parameter.

Three local tools are visible in isolation: the denominators of an iterated kernel clear by $\prod_j D_j$ (`ChainInt`); a 2×2 minor is divisible by w when $w^2 = p$ (`key_minor_dvd`); and Bost's slope inequality bounds the slope of an injectively mapped lattice. The global gain requires these to hold simultaneously on one family.

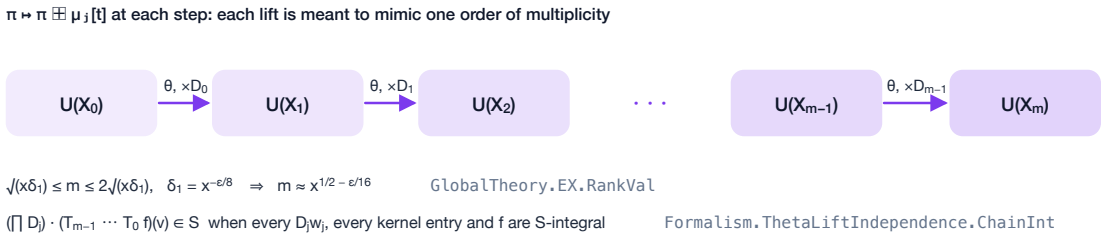


Figure 7. The chain of theta lifts (schematic). Schematic: unitary groups U(X0) ⟶ U(X1) ⟶ ... ⟶ U(Xm) linked by theta lifts; each lift carries a denominator D_j; the archive's `RankVal` puts m between $\sqrt{x\delta_1}$ and $2\sqrt{x\delta_1}$, $\delta_1 = x^{-\varepsilon/8}$.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.Top.topDesWin` — combine divisibility, slopes and windows	DesWin.lean, L11–31
`UpperTheory.Top.shape_closed_link_n_of_green` — calls `topDesWin` with $e_D = 2$, $e_L = 0$	TopShape.lean, L33–55
`GlobalTheory.EX.RankVal` — the chain length and the other ranks	PairRowIffQdRows.lean, L28–32
`Formalism.MainNotions.ColumnSystems.Pf.key_minor_dvd` — a local source of divisibility	KeyMinorDvd.lean, L132–155
`Formalism.F.slopeInequality_of` — the lattice slope inequality	SlopeInequality.lean, L111–125

USES ⟵

14 Seven inputs at the decisive call · *definite-side data*
`every_character_of_theta_integrals_local_matching`

SUPPLIES ⟶

8 The finite character comparison · *`TopG` ($e_D = 2$, $e_L = 0$)*
`UpperTheory.Top.TopG`

BLUEPRINT NODES 11 declarations: 1 ✓, 9 ◇, 1 ○ (Section 8)

◇ Apply the shape to the same chosen data ◇ Supply the actual fields of the shape ◇ Choose a common geometric family
◇ Combine divisibility, slopes and windows ◇ Control the local divisibility loss ◇ Sum the local losses
◇ The geometric ranks ✓ A local source of divisibility ◇ An existence check on a period identity
◇ The lattice slope inequality ○ The square-root multiplicity analogy

STATUS as of 2026-11-20 23:51 UTC — ◐ Partly checked

`key_minor_dvd` passed in its 509-module closure. `topDesWin` and the family construction are source-read here and included in the externally reported full build.

WHAT REMAINS TO CHECK

Nonempty compatible data: follow the witnesses of one geometric family through the test-vector grids, linked bundles and definite-side hypotheses.

IN THE ANNOUNCEMENT

Entry 1 (10/6/26): Stepanov determinant · *Historical motivation*

Entry 2 (10/9/26): Invariant differentials on a CM curve · *Historical calculation*

Entry 8 (11/15/26): A chain with square-root length · *Quantitative comparison*

Entry 9 (11/15/26): Norms, levels and arithmetic periods · *Supporting comparison*

Entry 10 (11/15/26): Divisibility of repeated lifts · *Supporting comparison*

Entry 13 (10/6/26): The slope inequality · *Mathematical statement matched*

Entry 15 (11/15/26): A common integral structure across characters · *Supporting comparison*

10 Hosts for every character

✓ Kernel-checked at the archive pin

supplyChi

For each primitive character ψ, the cohomological window produces arithmetic hosts of width $3/2$ at all large scales of a system of scales.

STATEMENT

$$\mathrm{VWinAt}\ \psi\ 401 \implies \text{hosts of width } 3/2 \text{ satisfying } (G_\chi\ \psi).\mathrm{req}.$$

The supplier is applied separately to each character; a common host for all characters would be a stronger quantifier order and is not needed. The character requirement combines the cyclotomic-level condition `EAtH` with the cohomological inequality for the host field.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.Top.supplyChi` — `VWinAt` ψ 401 ⟹ hosts of width 3/2 — SupplyChi.lean, L8–16

USES ←

11 The cohomological window · *`VWinAt` ψ 401*
`UpperTheory.Top.VWinAt`

SUPPLIES →

8 The finite character comparison · *hosts, width 3/2*
`UpperTheory.Top.supplyChi`

BLUEPRINT NODES 1 declarations: 1 ✓ (Section 8)

✓ Produce cyclic character hosts

STATUS as of 2026-11-20 23:51 UTC — ✓ Kernel-checked at the archive pin

`supplyChi` passed in the Dirichlet closure, with `VWinAt` ψ 401 as its hypothesis.

WHAT REMAINS TO CHECK

Its premise `VWinAt` ψ 401, for every primitive ψ.

6 Arithmetic supply

Galois representations, theta lifts and Iwasawa-type cohomology bounds that are meant to produce the families on which a prime sum is small.

COMPONENTS

- 11 The cohomological window
- 12 Auxiliary primes and Behrend towers
- 13 The Galois row
- 14 Seven inputs at the decisive call

ACROSS THE STRATUM BOUNDARY

14 Seven inputs at the decisive call → 9 The geometric family and the theta chain
definite-side data

11 The cohomological window → 10 Hosts for every character
VWinAt ψ 401

11 The cohomological window

◐ Partly checked

VWinAt ψ 401

On a whole interval $[Y, Y^2]$, weighted lengths of locally trivial Galois cohomology of twisted modules are at most $\sqrt{\exp(x^{\varepsilon})}$, uniformly over the relevant fields.

STATEMENT

$$\sum_{\ell \in S} \log \ell \sum_{1 \le h < \lfloor x^{401} \rfloor} s_{\ell,h}(E) \le \sqrt{\exp(x^{\varepsilon})}$$

for all $x \in [Y, Y^2]$, all finite S, all admissible E.

An ℓ-adic determinant estimate gives at most R centers; each center accounts for at most one exceptional twist $h \le Z$ (ultrametric separation). Finitely many exceptional twists forbid at most R logarithmic intervals $[a, Ba)$.

A real-variable induction (`window1`) finds $[y, 2y]$ avoiding all of them with $y \ge U/(2^{R+2}B^R)$; exponentiating gives $[Y, Y^2]$ with $X^{1/J} \le Y$, $J = 4(2B)^R$. The bound on the sums comes from four counting cases, a Ferrero–Washington input and descent.

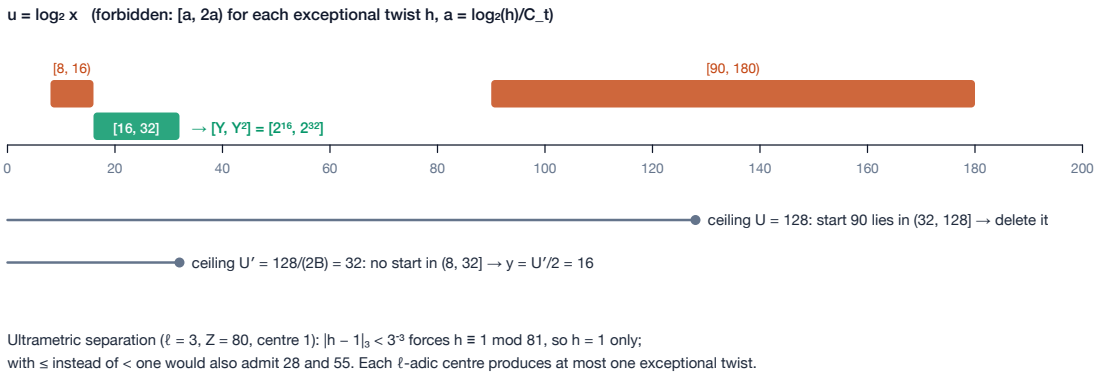


Figure 8. One interval survives all exceptional twists. Computed with the induction of `UpperTheory.V.Win.window1` on the book's example: forbidden logarithmic intervals [a, B a) for a = 8, 90 with B = 2 inside [0, 128]; the recursion lowers the ceiling 128 → 32 and returns [16, 32], i.e. $[Y, Y^2] = [2^{16}, 2^{32}]$.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.Top.VWinAt` — the cohomological window (definition)	ExistsPageN.lean, L44–47
`UpperTheory.Top.VVle` — weighted lengths of local–global kernels (definition)	ExistsPageN.lean, L12–17
`UpperTheory.V.Win.window1`	Window1.lean, L6–58
`UpperTheory.V.Win.window`	Window.lean, L7–43
`UpperTheory.One.vwin_of_row` — the window from the Galois row	VwinOfRow.lean, L4–5
`UpperTheory.V.HV.Core.of_four_alt` — combine the cohomology counts	HVCore.lean, L18–22
`UpperTheory.V.Win.thmBab_of` — Choose one set of tower constants	ThmBabOf.lean, L11–103
`UpperTheory.V.PropA.aBound_of_isTower` — From a finitely generated tower to two length bounds	ABoundOfIsTower.lean, L7–11
`UpperTheory.V.Win.thmBab_of_lemmaK` — Assemble the arithmetic tower window	ThmBabOfLemmaK.lean, L9–12

USES ←

13 The Galois row · *`Row0b` ⟶ window*
`vwin_of_row`

12 Auxiliary primes and Behrend towers · *prime pairs, tower*
`WinPair` · `hostThickExists`

SUPPLIES →

10 Hosts for every character · *`VWinAt` ψ 401*
`UpperTheory.Top.VWinAt`

BLUEPRINT NODES 25 declarations: 3 ✓, 2 ⊂, 20 ◇ (Section 8)

◇ Weighted lengths of local-global kernels ⊂ The cohomological window ◇ Cohomological window from the row
◇ Supply the automorphic window inputs ◇ Build the rank-N package ◇ Window from rank-N automorphy
◇ Pass through the descent hypotheses ◇ Produce the four counting bounds ◇ Combine the cohomology counts
✓ Assemble the arithmetic tower window ✓ Choose one set of tower constants
✓ From a finitely generated tower to two length bounds ⊂ Control exceptional twists by the fixed layer
◇ Uniform ℓ-rank of class groups in the tower ◇ Supply the absolute bound for the reference layer
◇ Case I gives one bound across coefficient primes ◇ Cases II and III from the rank-four symplectic argument
◇ Case V at twist one ◇ A nonzero integer controls one character orbit ◇ Convert the joint window into `VWin`
◇ Reuse the tower interval for the positive part ◇ Evaluate the pointwise estimate on admissible windows
◇ Combine the two prime sums at a point ◇ Add contributions inside the exceptional set
◇ Bound the sum over arbitrary finite sets of other primes

STATUS as of 2026-11-20 23:51 UTC — ◐ Partly checked

`window1` and window (the interval selection) passed at the archive pin, and so did the tower deductions `thmBab_of` (one record of window constants for every cutoff Ct ≥ 1 from `LemmaK`, `FixedIn` and `TwoIn`), `aBound_of_isTower` (a tower cohomology bound from duality and finiteness premises) and `thmBab_of_lemmaK`, each with its premises explicit. Those premises (Poitou–Tate and local Tate duality, local finiteness, Ferrero–Washington, `LemmaK`) and the counting cases are source-read here.

WHAT REMAINS TO CHECK

Uniform cohomological bound: track finiteness and absolute constants from the four counting cases to the finite prime sums on the claimed windows.

Audit criterion (◇ Combine the cohomology counts): *Uniform cohomological bound.* Track finiteness and the absolute constants from the four counting cases to the finite prime sums on the claimed windows.

IN THE ANNOUNCEMENT

Entry 11 (10/6/26): The tower of fields · *Supporting comparison*

Entry 17 (11/15/26): Cohomology and a weakening of Vandiver · *Quantitative comparison*

12 Auxiliary primes and Behrend towers

✓ Kernel-checked at the archive pin

`QH.pairEx` · `hostThickExists`

Two auxiliary primes with prescribed congruences and square-root prime-sum bounds for their quadratic twists, and an additive-combinatorial (Behrend) choice of the tower.

STATEMENT

$$q \equiv 3\ (8),\quad d \equiv 7\ (16),\quad q \mid d+1,$$
$$|g_\chi(y)| \le y^{1/2+\varepsilon/2} \text{ on } [Y, Y^A];$$
$$|S| \ge \frac{n}{L}\, e^{-C\sqrt{\log n \log L}}.$$

With $z = Y^{\varepsilon/5}$ the primes satisfy $z/2 < \log q \le z$ and $z(\log Y)^2/2 < \log d \le z(\log Y)^2$; $d+1$ is divisible by q and by every odd prime up to z. They are chosen sequentially from positive shares of candidates, and may change with Y.

Behrend: integer vectors of fixed norm, encoded in a large base, avoid short affine relations because a line meets a sphere at most twice.

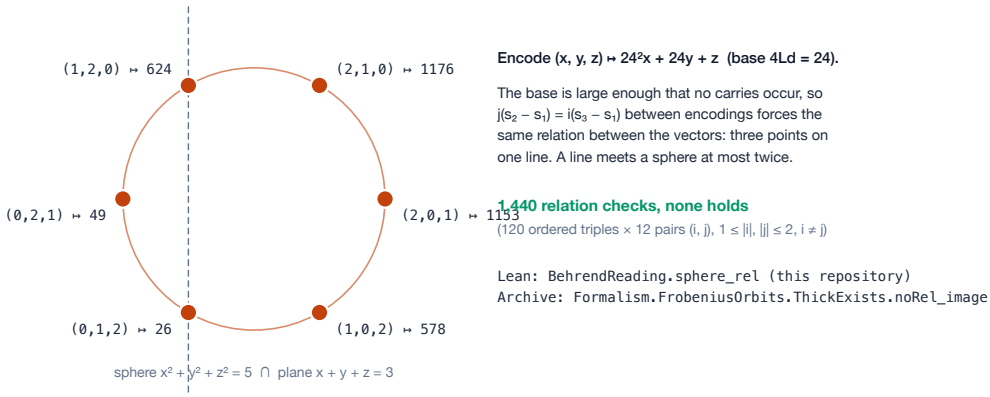


Figure 9. A Behrend digit sphere. Computed: the vectors of $\{0,1,2\}^3$ with squared norm 5 are the six permutations of (0,1,2); they lie on the circle where the sphere meets the plane x+y+z = 3. Their base-24 encodings avoid every relation j(s2-s1) = i(s3-s1) with $1 \le |i|,|j| \le 2$, $i \ne j$: 1,440 checks.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`QH.pairEx`	pairEx.lean, L112–169
two successive choices	PairExists.lean, L9–48
`Formalism.FrobeniusOrbits.ThickExists.noRel_image` — digit spheres avoid short relations	ThickExists.lean, L85–95
`hostThickExists`	ThickExists.lean, L181–187

USES ⟵

— no incoming interface

SUPPLIES ⟶

11 The cohomological window · *prime pairs, tower*
`WinPair` · `hostThickExists`

BLUEPRINT NODES 2 declarations: 1 ✓, 1 ○ (Section 8)

✓ Behrend's collision-free indices ○ The tower selection in the timeline

STATUS as of 2026-11-20 23:51 UTC ✓ Kernel-checked at the archive pin

`QH.pairEx` passed in its 293-module closure; the Behrend landmarks (`sphere_rel`, `noRel_image`, `hostThickExists`) passed in the 38-module landmark closure.

WHAT REMAINS TO CHECK

How the chosen primes feed the window for a fixed target character.

IN THE ANNOUNCEMENT

Entry 19 (10/19/26): A Behrend set in the tower construction · *Supporting comparison*

13 The Galois row

◇ Source-read `Row0b`

For regular algebraic polarized cuspidal π on GL_a over a CM field, a continuous semisimple ℓ-adic representation whose Frobenius polynomials match the Satake parameters of π.

STATEMENT

$$\iota\big(\text{charpoly } \rho(\sigma^{-1})\big) = \prod_{\alpha \in B_v} \big(X - q^{(a-1)/2}\alpha\big) \quad \text{for all but finitely many } v,$$

$$\rho : G_E \to \mathrm{GL}_a(\overline{\mathbb{Q}}_\ell).$$

The construction starts from a geometric seed and a high-rank theta lift (`tLiftK`), treats a restricted case, removes the prescribed polarization by twisting, and descends from infinitely many quadratic CM extensions $K = E(\sqrt{-p})$ by index-two patching of trace identities.

The exported row also records local–global compatibility away from ℓ and crystalline/de Rham comparisons above ℓ. A rank-one example shows that the hypotheses are inhabited.

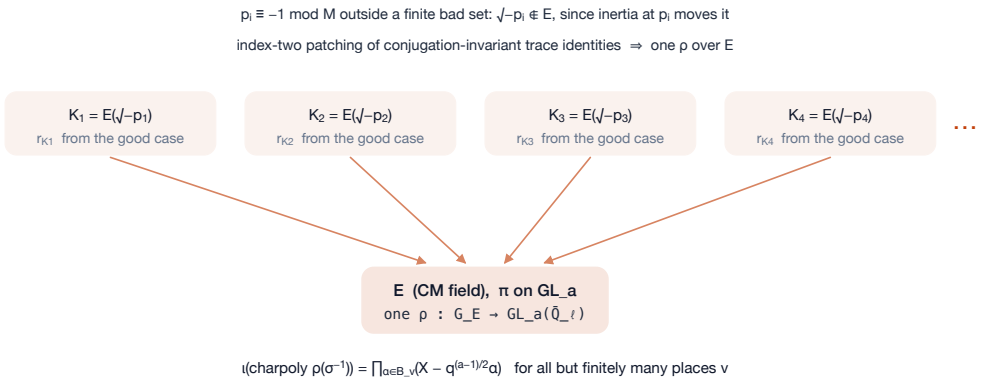


Figure 10. The Galois row by quadratic descent (schematic). Schematic: representations r_K over infinitely many quadratic CM extensions K = E($\sqrt{-p}$) are patched to one ρ over E whose Frobenius polynomials match the Satake parameters of π.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.GalStar.Row0b` — the local–global Galois row (definition)	WeightsConjBis.lean, L69–73
`UpperTheory.Gal.Int.galoisLocalGlobalStrong_recParameter` — producer of the row	RowSrc.lean, L6
`UpperTheory.Gal.Int.galoisRepresentations_of_thetaLift` — Galois representations from theta lifting	GaloisRepresentationsOfThetaLift.lean, L6
`UpperTheory.TP.K.tLiftK` — theta-lift existence at high rank (theorem)	TLiftK.lean, L4–5
`UpperTheory.TP.TLiftK` — quantifiers of the theta lift (definition)	TLiftK.lean, L12–23
`UpperTheory.Gal.Int.Geo.geoOne_holds` — weight-one geometric realization	WeightOneGeometricRealization.lean, L11

USES ←

— no incoming interface

SUPPLIES →

14 Seven inputs at the decisive call · *`Row0b` (two fields)*
`everyCharacter_of_row`

11 The cohomological window · *`Row0b` ⟶ window*
`vwin_of_row`

BLUEPRINT NODES 6 declarations: 6 ◇ (Section 8)

◇ The local–global Galois row ◇ Producer of the Galois row ◇ Galois representations from theta lifting
◇ Theta-lift existence at high rank ◇ Weight-one geometric realization ◇ Quantifiers of the theta lift

STATUS as of 2026-11-20 23:51 UTC ◇ Source-read

Source-read against 62 passages in 36 modules; included in the externally reported full build; not separately replayed by this audit.

WHAT REMAINS TO CHECK

Unfold the row, the local normalization convention and the theta-lift producer together; check all quantified places, twists and weight restrictions.

Audit criterion (◇ Producer of the Galois row): *Source comparison of the Galois input.* Unfold the row, the local-normalization convention and the theta-lift producer together. Check all quantified places, twists and weight restrictions.

14 Seven inputs at the decisive call

◇ Source-read `every_character_of_literature_of_vWin`

Unitary-group and local theta packages, agreement of theta integrals, local matching, a covolume bound, local Langlands parameters and the window: the seven arguments of the arithmetic theorem.

STATEMENT

$$P,\ F,\ h_A,\ h_M,\ h_X,\ h_L,\ h_V \Longrightarrow \text{nonempty definite-side data} + \text{Green input} \Longrightarrow \text{TopG}.$$

`everyCharacter_of_row` fills two fields of the literature record with the same Galois row, adds the existence of local Langlands parameters, and obtains the cohomological window from the same row. The next line supplies the row by name, so the half-plane theorem has no explicit premise.

A name such as `published_results_eigen` is where to start verifying what the term establishes, not an explanation of it.

LEAN DECLARATIONS pinned to Deicyde/riemann-hypothesis at 40e56bab

`UpperTheory.FinalTheorems.everyCharacter_of_row` — the half-plane theorem from one row	MainConclusion.lean, L7–8
`UpperTheory.FinalTheorems.everyCharacter` — closed half-plane assembly	MainConclusion.lean, L10–11
`UpperTheory.LiteratureTheorems.every_character_of_literature_of_vWin` — assemble the automorphic inputs	EveryCharacterOfLiterature-OfVWin.lean, L810–813
`UpperTheory.Lit.every_character_of_theta_integrals_local_matching` — seven inputs for every character	SiegelEisensteinSecondTermHypotheses.lean, L69–73

USES ←

13 The Galois row · *`RowOb` (two fields)*
`everyCharacter_of_row`

SUPPLIES →

9 The geometric family and the theta chain · *definite-side data*
`every_character_of_theta_integrals_local_matching`

BLUEPRINT NODES 16 declarations: 16 ◇ (Section 8)

◇ Closed half-plane assembly ◇ The half-plane theorem from one row ◇ Existence of local Langlands parameters
◇ Assemble the automorphic inputs ◇ Three fields in the literature input ◇ The unitary and eigenvalue package
◇ The local theta package ◇ The Hodge-bundle package ◇ The local parameter input at the call site
◇ Read the parameter field ◇ Regular characters yield the Green input ◇ Bind the estimates at 2 and 3
◇ Match global integrals and local data ◇ Produce the covolume bound ◇ Seven inputs for every character
◇ Build a nonempty definite-side package

STATUS as of 2026-11-20 23:51 UTC ◇ Source-read

Source-read; included in the externally reported full build; not separately replayed by this audit.

WHAT REMAINS TO CHECK

Source comparison of each named input with the mathematics it is meant to encode.

Audit criterion (◇ Build a nonempty definite-side package): *Nonempty compatible data.* Follow the witnesses for the same geometric family through the test-vector grids, linked bundles and nonempty definite-side hypotheses.

IN THE ANNOUNCEMENT

Entry 7 (10/14/26): Interlacing weights and periods · *Supporting comparison*

Entry 12 (11/15/26): Seesaw and Bessel periods · *Supporting comparison*

Entry 18 (11/15/26): Covolumes, lattice indices and congruences · *Supporting comparison*

7 The verified cuts and the Lean skeleton

Two implications have been machine-checked with the remaining inputs as explicit hypotheses. Each is a Lean theorem, written by the reading project against the archive, whose only open inputs are the premise displayed; the declarations it calls, directly or transitively, were elaborated and kernel-checked with it (tier ⊂ in Section 8). Together they cut the proof into the part downstream of the premises, which is machine-checked, and the arithmetic supply, which is partly checked and partly source-read. In the picture of Section 2 they are the dashed frontiers.

Cut I ✓✓ Checked by two kernels · One-sided prime windows imply RH

PREMISE

$$\forall \theta \in (\tfrac{1}{2}, 1)\ \exists C\ \forall Y_0\ \exists Y \geq Y_0\ \forall y \in [Y, Y^{3/2}]\ :\ G(y) \leq C\, y^{\theta}$$

CONCLUSION

$$\Longrightarrow \text{ Mathlib's } \texttt{RiemannHypothesis}$$

Lean. `Reading.WindowBridge.riemannHypothesis_of_prime_windows`, in `references/archive-bridges/WindowBridge.lean` (this repository; it compiles only against the archive, at Lean 4.33.1).
Evidence. Fresh Lean replay over 10,530 modules (766,938 constants) and the independent Rust kernel NanoDA on 61,697 exported declarations; 31 original modules; axioms `propext`, `Classical.choice`, `Quot.sound`.
Components inside the cut. 2 Reflection, 4 The centered prime sum, 5 Mellin transform and positive residues, 6 The window criterion

Quoted verbatim from `references/archive-bridges/WindowBridge.lean`: `OneSidedWindows`, `zetaFamily`, `riemannHypothesis_of_prime_windows`.

`references/archive-bridges/WindowBridge.lean` lines 8–23

```
/-- The constant may depend on the exponent. A chosen window must work at every real point. -/
def OneSidedWindows (g : ℝ → ℝ) : Prop :=
  ∀ θ : ℝ, 1 / 2 < θ → θ < 1 → ∃ C : ℝ,
    ∀ Y₀ : ℝ, ∃ Y : ℝ, Y₀ ≤ Y ∧
      ∀ y ∈ Set.Icc Y (Y ^ (3 / 2 : ℝ)), g y ≤ C * y ^ θ

/-- The family has one index, the character modulo one and a positive coefficient of one. -/
def zetaFamily : Deduction.CharFamily where
  ι := Unit
  fin := inferInstance
  N := fun _ => 1
  nz := fun _ => inferInstance
  χ := fun _ => (1 : DirichletCharacter ℂ 1)
  a := fun _ => 1
  a_pos := fun _ => zero_lt_one
  adm := fun _ => Or.inl rfl
```

`references/archive-bridges/WindowBridge.lean` lines 37–53

```
/-- A one-sided estimate on arbitrarily late three-halves windows implies Mathlib's RH. -/
theorem riemannHypothesis_of_prime_windows (h : OneSidedWindows Deduction.G) : RiemannHypothesis := by
  apply Formalism.VW.rh_of_right
  intro s hs _
  have hhalf : (1 / 2 : ℝ) < min s.re 1 := lt_min hs (by norm_num)
  obtain ⟨θ, hθhalf, hθmin⟩ := exists_between hhalf
  have hθs : θ < s.re := lt_of_lt_of_le hθmin (min_le_left _ _)
  have hθ1 : θ < 1 := lt_of_lt_of_le hθmin (min_le_right _ _)
  obtain ⟨C, hC⟩ := h θ hθhalf hθ1
  have hscale : (1 : ℝ) + 1 / 2 = 3 / 2 := by norm_num
  have hfree : Deduction.ZeroFree zetaFamily θ :=
```

```
    Deduction.zeroFree_of_rpow_window zetaFamily (θ := θ) (δ := 1 / 2) (C := C)
      (lt_trans (by norm_num : (0 : ℝ) < 1 / 2) hθhalf) hθ1 (by norm_num)
      (by simpa only [hscale, gm_zetaFamily] using hC)
  have hz : DirichletCharacter.LFunction (1 : DirichletCharacter ℂ 1) s ≠ 0 :=
    hfree () s hθs
  rwa [DirichletCharacter.LFunction_modOne_eq] at hz
```

Cut II ✓ Kernel-checked at the archive pin · `TopG` and the cohomological windows imply RH and Dirichlet GRH

PREMISE

$$\mathrm{TopG} \;\wedge\; \forall \psi\,\text{primitive} :\; \mathrm{VWinAt}\;\psi\;401$$

CONCLUSION

$$\Longrightarrow\; \mathrm{RH} \wedge \text{half-plane nonvanishing} \wedge \text{critical-strip zeros, for every positive modulus}$$

Lean. `Reading.DirichletBridge.endpoints_of_topG_vWin401`, in `references/archive-bridges/DirichletBridge.lean` (this repository; it compiles only against the archive, at Lean 4.33.1).
Evidence. 1,421 original modules at Lean 4.33.1; seven type/axiom reports; six defining modules replayed (Helios job 26647685).
Components inside the cut. 2 Reflection, 3 Conductor descent, 4 The centered prime sum, 5 Mellin transform and positive residues, 6 The window criterion, 7 The order of choices, 8 The finite character comparison, 10 Hosts for every character

Quoted verbatim from `references/archive-bridges/DirichletBridge.lean`: `primitiveZeroFree_of_topG_vWin401`, `endpoints_of_topG_vWin401`.

`references/archive-bridges/DirichletBridge.lean` lines 11–26

```
/-- The cohomology cutoff is 401 for each primitive character. The supplied
host width is the same real number 3/2 for all these characters. -/
theorem primitiveZeroFree_of_topG_vWin401 (hG : UpperTheory.Top.TopG)
    (hV : ∀ ψ : PChar, VWinAt ψ 401) : ZeroFreeAll := by
  apply UpperTheory.Top.zeroFreeAll_of_topG (la := (3 / 2 : ℝ)) (by norm_num) hG
  intro ψ
  exact UpperTheory.Top.supplyChi ψ (hV ψ)

/-- Pass to arbitrary positive moduli, then use the archived critical-strip
and zeta reductions. The right-half-plane predicate retains its pole exception. -/
theorem endpoints_of_topG_vWin401 (hG : UpperTheory.Top.TopG)
    (hV : ∀ ψ : PChar, VWinAt ψ 401) :
    RiemannHypothesis ∧ DirichletLNonvanishingRightOfHalf ∧ DirichletLZerosOnLine := by
  have hx : DirichletLNonvanishingRightOfHalf :=
    UpperTheory.Top.imprimitive (primitiveZeroFree_of_topG_vWin401 hG hV)
  exact ⟨Claim.zeta hx, hx, Claim.strip hx⟩
```

7.1 The architecture as a checked skeleton

This repository's own Lean library (Lean 4.35.0, no imports) restates the architecture over abstract types: each component becomes a hypothesis, a field of a structure, and the theorems prove only how the components compose. It is a map, not a proof of the Riemann hypothesis. A toy model at the end of the file (`toyArchitecture`, not quoted) satisfies every hypothesis with zeros present, all on the line, so the skeleton is not vacuous; it says nothing about which model is the real one. The build passed with 0 warnings; the axioms of all 18 audited declarations are listed at the end of this section.

The objects. Points, characters and zeros, with the predicates the endpoints talk about.

lean/Atlas/Architecture.lean lines 31–60

```lean
/-- The objects the endpoints talk about. `Point` stands for complex numbers `s`, `Char` for
Dirichlet characters of all positive moduli, `Zero x s` for `L(s, x) = 0`. Lean's `L`-functions
are total functions, so a value exists even at the principal pole. -/
structure Plane where
  Point : Type
  Char : Type
  Zero : Char → Point → Prop
  /-- `Re s > 1/2`. -/
  Right : Point → Prop
  /-- `Re s = 1/2`. -/
  Line : Point → Prop
  /-- `Re s < 1/2`. -/
  Left : Point → Prop
  /-- `0 < Re s < 1`. -/
  Strip : Point → Prop
  /-- `(x, s) = (1_N, 1)`: the principal character at the pole. -/
  Pole : Char → Point → Prop
  Primitive : Char → Prop
  /-- The primitive character inducing `x`. -/
  prim : Char → Char
  /-- `x ↦ x⁻¹`. -/
  inv : Char → Char
  /-- `s ↦ 1 - s`. -/
  refl : Point → Point
  /-- The character of modulus one, whose `L`-function is `ζ`. -/
  zeta : Char
  /-- `s ∈ {-2, -4, -6, …}`. -/
  Trivial : Point → Prop
  /-- `s = 1`. -/
  One : Point → Prop
```

The target propositions. RH as in Mathlib, the half-plane theorem, its primitive form and Dirichlet GRH in the strip.

lean/Atlas/Architecture.lean lines 64–82

```lean
/-- Mathlib's `RiemannHypothesis`, transported to the abstract plane: every zero of `ζ` that is
neither trivial nor at `s = 1` lies on the critical line. -/
def RH : Prop :=
  ∀ s, P.Zero P.zeta s → ¬ P.Trivial s → ¬ P.One s → P.Line s

/-- The half-plane theorem (archive: `DirichletLNonvanishingRightOfHalf`): no `L(s, x)` vanishes
for `Re s > 1/2`, the principal pole excepted. -/
def HalfPlane : Prop :=
  ∀ x s, P.Right s → ¬ P.Pole x s → ¬ P.Zero x s

/-- The primitive half-plane statement (archive: `ZeroFreeAll`). It quantifies over primitive
characters only and has no pole exception, because Lean's totalized `ζ` takes a nonzero value
at `s = 1`. -/
def PrimitiveHalfPlane : Prop :=
  ∀ ψ s, P.Primitive ψ → P.Right s → ¬ P.Zero ψ s

/-- Dirichlet GRH in the open strip (archive: `DirichletLZerosOnLine`). -/
def StripGRH : Prop :=
  ∀ x s, P.Strip s → P.Zero x s → P.Line s
```

The destination as hypotheses. Reflection (functional equations) and conductor descent (Euler factors), components 2 Reflection and 3 Conductor descent.

lean/Atlas/Architecture.lean lines 86–118

```lean
/-- Classical facts used by the closing deductions. All of them are proved in the archive and
were kernel-checked at Lean 4.33.1 by the reading project's audit (`rh_of_right`, `Claim.strip`,
`Claim.zeta`). -/
```

```
structure Reflection where
  /-- Every point is left of, on, or right of the critical line. -/
  trichotomy : ∀ s, P.Left s ∨ P.Line s ∨ P.Right s
  /-- `s ↦ 1 - s` maps the open strip to itself. -/
  refl_strip : ∀ s, P.Strip s → P.Strip (P.refl s)
  /-- `s ↦ 1 - s` sends the left half of the strip to the right half. -/
  refl_right : ∀ s, P.Strip s → P.Left s → P.Right (P.refl s)
  /-- The completed functional equation of a primitive character reflects a strip zero of `x`
  to a zero of `x⁻¹` at `1 - s`; the Gamma factors do not vanish in the strip. -/
  functional_equation :
    ∀ ψ s, P.Primitive ψ → P.Strip s → P.Zero ψ s → P.Zero (P.inv ψ) (P.refl s)
  /-- The open strip contains no pole. -/
  strip_no_pole : ∀ x s, P.Strip s → ¬ P.Pole x s
  /-- `ζ` is the `L`-function of the primitive character of modulus one, which is its own inverse. -/
  zeta_primitive : P.Primitive P.zeta
  zeta_inv : P.inv P.zeta = P.zeta
  /-- Outside the open strip `ζ` has only the trivial zeros (functional equation) and no zeros
  for `Re s ≥ 1` (in Mathlib); `s = 1` is the pole. Stated constructively: a zeta zero that is
  neither trivial nor at `1` lies in the strip. -/
  zeta_strip : ∀ s, P.Zero P.zeta s → ¬ P.Trivial s → ¬ P.One s → P.Strip s

/-- Conductor descent (archive: `Claim.prim_zero`, `UpperTheory.Top.imprimitive`):
`L(s,x) = L(s,x*) ∏_{p ∣ N} (1 - x*(p) p^{-s})`, and no removed factor vanishes for `Re s > 0`
because `|x*(p) p^{-s}| ≤ p^{-Re s} < 1`. -/
structure Descent where
  prim_primitive : ∀ x, P.Primitive (P.prim x)
  /-- A zero of `L(s, x)` with `Re s > 1/2`, away from the pole, is a zero of `L(s, x*)`. -/
  euler_right : ∀ x s, P.Right s → ¬ P.Pole x s → P.Zero x s → P.Zero (P.prim x) s
  /-- The same in the open strip. -/
  euler_strip : ∀ x s, P.Strip s → P.Zero x s → P.Zero (P.prim x) s
```

The closing deductions. Primitive half-plane ⇒ half-plane ⇒ strip and RH.

lean/Atlas/Architecture.lean — lines 122–143

```
/-- Primitive half-plane ⇒ half-plane for every character (mirrors `UpperTheory.Top.imprimitive`). -/
theorem halfPlane_of_primitive (D : Descent P) (h : PrimitiveHalfPlane P) : HalfPlane P :=
  fun x s hs hpole hz => h (P.prim x) s (D.prim_primitive x) hs (D.euler_right x s hs hpole hz)

/-- Half-plane ⇒ Dirichlet GRH in the strip (mirrors `Claim.strip`). A zero left of the line
descends to the primitive character and reflects to a zero of its inverse on the right; a zero
on the right contradicts the hypothesis directly. -/
theorem strip_of_halfPlane (R : Reflection P) (D : Descent P) (h : HalfPlane P) : StripGRH P := by
  intro x s hstrip hz
  rcases R.trichotomy s with hl | hline | hr
  · have hp : P.Zero (P.prim x) s := D.euler_strip x s hstrip hz
    have hzr : P.Zero (P.inv (P.prim x)) (P.refl s) :=
      R.functional_equation _ s (D.prim_primitive x) hstrip hp
    exact absurd hzr
      (h _ _ (R.refl_right s hstrip hl) (R.strip_no_pole _ _ (R.refl_strip s hstrip)))
  · exact hline
  · exact absurd hz (h x s hr (R.strip_no_pole x s hstrip))

/-- Half-plane ⇒ RH (mirrors `Claim.zeta`): a nontrivial zeta zero lies in the strip, where the
previous theorem applies at modulus one. -/
theorem rh_of_halfPlane (R : Reflection P) (D : Descent P) (h : HalfPlane P) : RH P :=
  fun s hz htriv hone => strip_of_halfPlane R D h P.zeta s (R.zeta_strip s hz htriv hone) hz
```

The engine and the supply. The analytic criterion as a field, and the arithmetic supply with the order of choices (θ after the zero, ε after θ).

lean/Atlas/Architecture.lean — lines 147–184

```
/-- The analytic engine over an abstract scale of exponents.

```

```
* `Exp` carries the exponents `θ`, with `Le θ θ'` for `θ ≤ θ'`;
* `RightOf s θ` is `Re s > θ`;
* `Family` carries the finite positive character families and `Mem x F` is membership;
* `Windows F θ` is the one-sided estimate: one constant `C` and arbitrarily late windows
  `[Y, Y^(3/2)]` on which the family's centered prime sum is at most `C·y^θ` at every point. -/
structure Engine (P : Plane) where
  Exp : Type
  Le : Exp → Exp → Prop
  RightOf : P.Point → Exp → Prop
  Family : Type
  Mem : P.Char → Family → Prop
  Windows : Family → Exp → Prop
  /-- A weaker exponent is easier to satisfy. -/
  windows_mono : ∀ F θ θ', Windows F θ → Le θ θ' → Windows F θ'
  /-- `Deduction.zeroFree_of_rpow_window`: a one-sided bound on arbitrarily late power windows
  forbids zeros of every member to the right of the exponent. Proof in the archive: the Mellin
  transform of the ramp, positive residues (`poleData`), and high derivatives aimed at the
  windows (`landau_window`). -/
  criterion : ∀ F θ, Windows F θ → ∀ x, Mem x F → ∀ s, RightOf s θ → ¬ P.Zero x s

/-- The arithmetic supply, in the form consumed by `zeroFreeAll_of_topG`. For each primitive `ψ`
and each admissible small `ε` there is a family containing `ψ` whose prime sum satisfies the
window bound with exponent `loss ε = 1/2 + (e_D + e_L + 7)·ε`, where `e_D = 2`, `e_L = 0` were
fixed **before** `ε`. In the archive this is produced from `TopG` and `VWinAt ψ 401` (through
`supplyChi`, `core_gm` and `topDesWin`). It is the one non-classical input. -/
structure Supply (P : Plane) (E : Engine P) where
  Eps : Type
  Small : Eps → Prop
  loss : Eps → E.Exp
  family : P.Char → Eps → E.Family
  mem_family : ∀ ψ ε, E.Mem ψ (family ψ ε)
  windows : ∀ ψ ε, P.Primitive ψ → Small ε → E.Windows (family ψ ε) (loss ε)
  /-- θ after the zero, ε after θ (`level_of_tiedWin`): for `Re s > 1/2` take
  `θ = 1/2 + min(Re s - 1/2, 1/2)/2`, so that `Re s > θ`, then a small `ε` with `loss ε ≤ θ`.
  This is possible because `loss ε → 1/2` as `ε → 0` with the losses fixed. -/
  choose : ∀ s, P.Right s → ∃ θ, E.RightOf s θ ∧ ∃ ε, Small ε ∧ E.Le (loss ε) θ
```

The cuts in the skeleton.

lean/Atlas/Architecture.lean — lines 186–218

```
/-- The supply and the engine give the primitive half-plane (mirrors `zeroFreeAll_of_topG`). -/
theorem primitiveHalfPlane_of_supply (E : Engine P) (S : Supply P E) : PrimitiveHalfPlane P := by
  intro ψ s hψ hs hz
  obtain ⟨θ, hθ, ε, hε, hle⟩ := S.choose s hs
  exact E.criterion (S.family ψ ε) θ (E.windows_mono _ _ _ (S.windows ψ ε hψ hε) hle)
    ψ (S.mem_family ψ ε) s hθ hz

/-- **The second verified cut** (mirrors `Reading.DirichletBridge.endpoints_of_topG_vWin401`,
kernel-checked against the archive in a 1,421-module closure): the arithmetic supply implies
RH, the half-plane theorem for every character and Dirichlet GRH in the strip. -/
theorem endpoints_of_supply (R : Reflection P) (D : Descent P) (E : Engine P) (S : Supply P E) :
    RH P ∧ HalfPlane P ∧ StripGRH P :=
  have hhalf := halfPlane_of_primitive D (primitiveHalfPlane_of_supply E S)
  ⟨rh_of_halfPlane R D hhalf, hhalf, strip_of_halfPlane R D hhalf⟩

/-- **The first verified cut** (mirrors `Reading.WindowBridge.riemannHypothesis_of_prime_windows`,
checked by the Lean kernel and by the Rust kernel NanoDA): one family `{ζ}`, and the window bound
at every exponent `θ ∈ (1/2, 1)`, give RH. No other character and no descent is involved. -/
theorem rh_of_prime_windows (R : Reflection P) (E : Engine P) (Fz : E.Family)
    (hmem : E.Mem P.zeta Fz) (Above : E.Exp → Prop)
    (choose : ∀ s, P.Right s → ∃ θ, Above θ ∧ E.RightOf s θ)
    (h : ∀ θ, Above θ → E.Windows Fz θ) : RH P := by
  intro s hz htriv hone
  have hstrip : P.Strip s := R.zeta_strip s hz htriv hone
  rcases R.trichotomy s with hl | hline | hr
  · have hz' : P.Zero (P.inv P.zeta) (P.refl s) :=
```

```
      R.functional_equation _ s R.zeta_primitive hstrip hz
    rw [R.zeta_inv] at hz'
    obtain ⟨θ, hθ, hright⟩ := choose (P.refl s) (R.refl_right s hstrip hl)
    exact absurd hz' (E.criterion Fz θ (h θ hθ) P.zeta hmem _ hright)
  · exact hline
  · obtain ⟨θ, hθ, hright⟩ := choose s hr
    exact absurd hz (E.criterion Fz θ (h θ hθ) P.zeta hmem s hright)
```

The whole architecture in one record.

lean/Atlas/Architecture.lean — lines 222–234

```
/-- The four layers of the argument bundled together. The first three are classical or
machine-checked; the fourth, `supply`, is where the arithmetic lives. -/
structure Architecture where
  plane : Plane
  reflection : Reflection plane
  descent : Descent plane
  engine : Engine plane
  supply : Supply plane engine

/-- The five endpoints, as conjunctions on the abstract plane. -/
theorem Architecture.endpoints (A : Architecture) :
    RH A.plane ∧ HalfPlane A.plane ∧ StripGRH A.plane :=
  endpoints_of_supply A.reflection A.descent A.engine A.supply
```

A checked lemma from the cohomological window. The integer core of the ultrametric separation used in component 11 The cohomological window: each centre accounts for at most one exceptional twist.

lean/Atlas/Separation.lean — lines 31–42

```
/-- Two twists in `[1, Z]` that are both `m`-adically close to the same centre `s`, with `m > Z`,
coincide: each centre accounts for at most one exceptional twist. -/
theorem at_most_one_per_centre {a b s m Z : Int} (hm : Z < m)
    (ha1 : 1 ≤ a) (haZ : a ≤ Z) (hb1 : 1 ≤ b) (hbZ : b ≤ Z)
    (hsa : m ∣ a - s) (hsb : m ∣ b - s) : a = b := by
  obtain ⟨c₁, h₁⟩ := hsa
  obtain ⟨c₂, h₂⟩ := hsb
  refine eq_of_dvd_sub hm ha1 haZ hb1 hbZ ⟨c₁ - c₂, ?_⟩
  rw [Int.mul_sub]
  generalize m * c₁ = t₁ at h₁ ⊢
  generalize m * c₂ = t₂ at h₂ ⊢
  omega
```

Axiom report. Generated 2026-11-21 00:21:04 UTC from the build log of `lean/Atlas/Audit.lean` (`verification/logs/axioms.json`); 10 of 18 declarations use no axiom at all, the others only Lean's standard axioms.

DECLARATION	AXIOMS (TRANSITIVE)	FREE
`Atlas.halfPlane_of_primitive`	none	✓
`Atlas.strip_of_halfPlane`	none	✓
`Atlas.rh_of_halfPlane`	none	✓
`Atlas.primitiveHalfPlane_of_supply`	none	✓
`Atlas.endpoints_of_supply`	none	✓
`Atlas.rh_of_prime_windows`	none	✓
`Atlas.Architecture.endpoints`	none	✓
`Atlas.toyArchitecture`	none	✓
`Atlas.Separation.eq_of_dvd_sub`	`propext`, `Quot.sound`	
`Atlas.Separation.at_most_one_per_centre`	`propext`, `Quot.sound`	
`Reading.excludeZeroByWindows`	none	✓
`Reading.criticalLineByReflection`	none	✓
`BehrendReading.normSq_nonneg`	`propext`	
`BehrendReading.coordinates_zero_of_normSq_zero`	`propext`, `Quot.sound`	
`BehrendReading.distance_pos_of_ne`	`propext`, `Quot.sound`	
`BehrendReading.chord_identity`	`propext`, `Classical.choice`, `Quot.sound`	
`BehrendReading.sphere_rel`	`propext`, `Classical.choice`, `Quot.sound`	
`BehrendReading.sphere_midpoint`	`propext`, `Classical.choice`, `Quot.sound`	

8 The blueprint: 107 declarations

The reading project's blueprint graph has 109 nodes, 107 Lean declarations and 2 ideas from the announcement without a theorem, joined by 113 edges; the model assigns every node to exactly one component. The table lists them component by component, in the reading order, with the declaration linked to its pinned source lines and its contract in one sentence. Node tiers are computed from receipts: ✓ a named declaration with a kernel-check receipt, ⊂ called by one, ↗ the five exports, ◇ located and read, ○ motivation only. Figure 11 draws the whole graph.

	NODE	DECLARATION (PINNED)	CONTRACT
1	**The five endpoints** 13 nodes		
✓✓	Mathlib RH main-rh	`riemannHypothesis` Main.lean, L4	Every nontrivial zeta zero distinct from the pole has real part one half.
✓✓	Zeta in expanded form main-zeta	`riemannZeta_eq_zero_re` Main.lean, L6	Every zero other than the negative even integers and the point one has real part one half.
✓✓	Nonvanishing right of one half main-right	`LFunction_ne_zero_of_half_lt_re` Main.lean, L8	For all positive moduli and all characters, $L(s,\chi)$ is nonzero for $\mathrm{Re}(s)>1/2$, excluding the principal pole.
✓✓	Dirichlet GRH in the open strip main-strip	`LFunction_eq_zero_re` Main.lean, L10	For every character of positive modulus, any zero with $0<\mathrm{Re}(s)<1$ satisfies $\mathrm{Re}(s)=1/2$.
✓✓	Primitive critical-strip zeros main-prim	`LFunction_eq_zero_re_of_isPrimitive` Main.lean, L12	For every primitive character of positive modulus, all zeros in the open critical strip have real part one half.
◇	Zeta target type stmt-zeta	`RiemannZetaZerosOnLine` Statements.lean, L3–4	$\forall\, s\in\mathbb{C}$, $\zeta(s)=0$, s outside $\{-2(n+1)\}$ for $n\in\mathbb{N}$ and $s\neq1$ imply $\mathrm{Re}(s)=1/2$.
◇	Half-plane target type stmt-right	`DirichletLNonvanishingRightOfHalf` Statements.lean, L6–8	$\forall\, N\neq0$, χ mod N, $s\in\mathbb{C}$, $1/2<\mathrm{Re}(s)$ and ($\chi\neq1$ or $s\neq1$) imply $L(s,\chi)\neq0$.
◇	Dirichlet strip target type stmt-strip	`DirichletLZerosOnLine` Statements.lean, L10–12	$\forall\, N\neq0$, χ mod N, $s\in\mathbb{C}$, $0<\mathrm{Re}(s)<1$ and $L(s,\chi)=0$ imply $\mathrm{Re}(s)=1/2$.
◇	Primitive strip target type stmt-prim	`PrimitiveDirichletLZerosOnLine` Statements.lean, L14–16	The preceding strip implication for each χ satisfying `IsPrimitive`.
◇	The two zeta exports share a proof rh-all	`rhOfAll` MainTheorems.lean, L9	`RiemannHypothesis`.
◇	All-character half-plane theorem all-chi	`allChi` MainTheorems.lean, L7	`UpperTheory.EveryCharacter`.
◇	Primitive critical-line wrapper crit-all	`critAll` MainTheorems.lean, L11–13	For primitive χ, every open-strip zero has real part one half.
◇	Three endpoint propositions claim-all	`claimAll` MainTheorems.lean, L15–16	`HalfPlane` ∧ Strip ∧ Zeta.
2	**Reflection** 8 nodes		
◇	Specialize at modulus one rh-every	`Formalism.VW.rh_of_every` CritLine.lean, L29–32	`EveryCharacter` implies `RiemannHypothesis`.
✓	Reflect a zeta zero rh-right	`Formalism.VW.rh_of_right` RhOfRight.lean, L5–62	Right-half-plane zeta nonvanishing away from one implies Mathlib RH.
◇	Reflect a primitive character vw-crit	`Formalism.VW.critLine` CritLine.lean, L11–27	`EveryCharacter` implies the primitive open-strip conclusion.
◇	Gamma factors in the strip gamma	`Formalism.VW.gammaFactor_ne_zero` CritLine.lean, L6–9	For every character χ and $\mathrm{Re}(s)>0$ its gamma factor at s is nonzero.

	NODE	DECLARATION (PINNED)	CONTRACT
✓	Reflect an arbitrary character strip	`Claim.strip` ClaimStrip.lean, L59–73	`Claim.HalfPlane` implies `Claim.Strip`.
⊂	Transfer a primitive zero to 1−s prim-reflect	`Claim.prim_reflect` ClaimStrip.lean, L42–57	At an open-strip zero of a primitive ψ, its inverse has a zero at 1−s.
✓	Zeta component of the conjunction bundle-zeta	`Claim.zeta` ClaimStrip.lean, L75–100	`Claim.HalfPlane` implies `Claim.Zeta`.
◇	A second zeta deduction alternate-rh	`UpperTheory.riemannHypothesis_of_everyCharacter` RiemannHypothesisOfEveryCharacter.lean, L215–231	`EveryCharacter` implies Mathlib RH.
	3 Conductor descent 4 nodes		
⊂	Remove imprimitive Euler factors prim-zero	`Claim.prim_zero` ClaimStrip.lean, L23–33	An imprimitive character zero in 0<Re(s)<1 gives a zero of its primitive inducing character.
⊂	Nonzero removed Euler factor factor	`Claim.factor_ne_zero` ClaimStrip.lean, L7–21	For prime p and Re(s)>0, $1-\psi(p)p^{-s}\neq 0$.
◇	Pass from primitive to all characters every-green	`UpperTheory.Top.every_character_of_green_v_win` TopPlug.lean, L16–18	Green ∧ `VWin` ∧ nonempty definite-side hypotheses imply `EveryCharacter`, written as three binders.
✓	Restore all characters imprimitive	`UpperTheory.Top.imprimitive` Imprimitive.lean, L5–35	`ZeroFreeAll` for primitive characters implies the all-moduli right-half-plane assertion with the principal pole exception.
	4 The centered prime sum 8 nodes		
⊂	The weighted prime sum g-m	`Formalism.HC.Level.Gm` ExplicitFormulaErrorOfHostConditions.lean, L38	$G_P(x)=G\zeta(x)$ plus a finite weighted sum of real parts of twisted smoothed errors.
⊂	Uniform families on power windows window-def	`Formalism.HC.ChebyshevInequalityAtWindowScale` TopWin.lean, L30–32	There exist `eD`,`eL` and $\varepsilon_1>0$ such that every $0<\varepsilon<\varepsilon_1$ admits a tied family with whole windows of width λ.
⊂	The first moment of the Chebyshev error moment-target	`Formalism.HermitianLattices.GL2a2.PsiZeroConst` PrimeCountAsymptoticInProgressions.lean, L18–20	The truncated integral of $(\psi(t)-t)/t^2$ from 1 to U tends to $-1-\gamma$ as U tends to infinity.
✓	Evaluate the first moment moment-theorem	`Formalism.HermitianLattices.GL2a2.psi_zero_integral_constant` PsiZeroIntegral.lean, L530–531	The archive supplies its first-moment limit without an RH assumption.
⊂	Match the two limits at one moment-calculation	`psiZeroConst_main` PsiZeroIntegral.lean, L423–521	For the trivial character modulo one the Chebyshev remainder has first moment $-1-\gamma$.
⊂	Read Euler's constant from the logarithmic derivative moment-laurent	`zeta_logderiv_limit` PsiZeroIntegral.lean, L216–277	As complex s tends puncturedly to 1, $\zeta'(s)/\zeta(s)+1/(s-1)$ tends to γ.
⊂	Dominate the centered character sum moment-majorant	`sw_bound` PsiZeroIntegral.lean, L279–301	For each fixed character, the centered von Mangoldt sum is eventually bounded in norm by $Kt/\log^2 t$.
⊂	Pass the Mellin integral to $\sigma = 1$ moment-dominated	`core_tendsto` PsiZeroIntegral.lean, L157–176	A logarithmic remainder bound lets σ times the weighted integral tend to its value at $\sigma=1$ from the right.
	5 Mellin transform and positive residues 1 nodes		
✓	Poles of the finite-family transform pole	`Deduction.poleData` poleData.lean, L248–259	For each admissible finite family there exists a transform with `PoleData` for its zero set.
	6 The window criterion 4 nodes		

	NODE	DECLARATION (PINNED)	CONTRACT
⊂	The analytic criterion for a level form-b	`UpperTheory.Comparison₂.formB` Imports.lean, L8–19	For every level, every $\lambda>1$ and $0<\theta<1$, arbitrarily late windows bounded by x^{θ} imply no zero right of θ.
✓	The character-family power-window theorem landau	`Deduction.zeroFree_of_rpow_window` ZeroFreeOfRpowWindow.lean, L11–14	For a positive-weight family, a one-sided x^{θ} bound on arbitrarily late $[Y,Y^{1+\delta}]$ implies `ZeroFree` F θ.
⊂	Apply the pole obstruction landau-lemma	`Deduction.lemmaL` lemmaL.lean, L13–22	`WindowHyp` on a finite character family implies `ZeroFree` F θ in the given parameter range.
⊂	Rule out a pole right of θ oscillation	`Deduction.landau_window` landauWindow.lean, L83–95	For the supplied `PoleData` and `WindowHypU`, any ρ in the specified pole set satisfies $\mathrm{Re}(\rho)\le\theta$.
7	**The order of choices** 5 nodes		
◇	Select the window width 3/2 zero-green	`UpperTheory.Top.zero_free_all_of_green_v_win` TopPlug.lean, L12–15	Green, `VWin` and a nonempty definite-side package imply `ZeroFreeAll`.
✓	Produce primitive zero-free levels zero-top	`UpperTheory.Top.zeroFreeAll_of_topG` ZeroFreeAllOfTopG.lean, L26–34	For $\lambda>1$, `TopG` and hosts for every primitive character imply `ZeroFreeAll`.
⊂	Assemble the cyclic levels tied-cyc	`Formalism.HC.zeroFreeOn_of_tiedWin_cyc` TopWin.lean, L90–93	Uniform window bounds, `FormB`, and family existence for cyclic levels imply `ZeroFreeOn` Q.
⊂	Pass from levels to characters tied	`Formalism.HC.zeroFreeOn_of_tiedWin` TopWin.lean, L83–88	Appropriate level data for each ψ satisfying Q imply `ZeroFreeOn` Q.
⊂	Choose epsilon after a candidate zero window-consumer	`Formalism.HC.level_of_tiedWin` TopWin.lean, L68–81	At any level, the chosen uniform arithmetic window and the analytic criterion imply zero-freeness right of one half.
8	**The finite character comparison** 5 nodes		
⊂	The prime-sum inequality on tied families prime-bound	`UpperTheory.Comparison₂.level_gm_le_on_family_scales` LevelGmLeOnFamilyScales.lean, L28–30	For every level, tied compatible families yield the windowed upper bound on the weighted prime sum.
✓	Estimate the weighted level on a window artin-window	`UpperTheory.Comparison₂.windowInequality_of_artinInputs` WindowInequalityOfArtinInputs.lean, L17–25	Identification of tied families with controlled host data gives `ChebyshevInequalityOnWindows` at any level.
✓	From the height discrepancy to the prime sum core-gm	`UpperTheory.Comparison₂.core_gm` CoreGm.lean, L12–74	For large x, `EAtH` and `CoreAtH` imply $G_P(x)\le x^{1/2+(\mathrm{eD}+\mathrm{eL}+7)\varepsilon}$.
✓	Bound the discrepancy per index ch-top	`Deduction.ch_top` ChTop.lean, L37–72	The named identifications and size, error, and norm windows give an upper bound for `gapA`.
✓	Average over characters ch-core	`Deduction.ch_core` ChCore.lean, L343–351	Under the size and character hypotheses, per-index discrepancy bounds imply a power bound for `GmA` up to `SigA`.
9	**The geometric family and the theta chain** 11 nodes		
◇	Apply the shape to the same chosen data goal-shape	`GlobalTheory.Fc.forall_goal_of_shape_of_hyps` ShapeOf.lean, L31–32	For propositions V, NR, G and a class predicate N, take `TopShapeAlt` V NR N G, V, NR and `HypsAlt` N `DefiniteSideCondition` to G.
◇	Supply the actual fields of the shape shape-fields	`GlobalTheory.Fc.goal_of_shape_of_hyps` ShapeOf.lean, L24–30	`TopShapeAlt` V NR N G with V, NR and `HypsAlt` N `DefiniteSideCondition` yields G.
◇	Choose a common geometric family shape	`UpperTheory.Top.shape_closed_link_n_of_green` TopShape.lean, L33–55	Green implies the specified `TopShapeAlt` ending in `TopG`.
◇	Combine divisibility, slopes and windows family-design	`UpperTheory.Top.topDesWin` DesWin.lean, L11–31	The listed existence, local, rank, divisibility, global and slope controls produce `FamilyClassConnected`.

	NODE	DECLARATION (PINNED)	CONTRACT
◇	Control the local divisibility loss divin	`UpperTheory.Top.divinN_of_hypV` DivinNOfHypV.lean, L117–130	For $\varepsilon>0$, `HypV`, `LinkN` and `SmallLevelRankEq` imply `DIVinN` on the family.
◇	Sum the local losses vv-sum	`UpperTheory.Top.vvRecord` DivinNOfHypV.lean, L17–57	`VVle` gives finite cohomology lengths, summability and the bound on the infinite sum of `vvTerm`.
◇	The geometric ranks rank	`GlobalTheory.EX.RankVal` PairRowIffQdRows.lean, L28–32	In particular, $\sqrt{x\delta_1}\le m\le 2\sqrt{x\delta_1}$, with $\delta_1=x^{-\varepsilon/8}$.
✓	A local source of divisibility minor	`Formalism.MainNotions.ColumnSystems.Pf.key_minor_dvd` KeyMinorDvd.lean, L132–155	For $w^2=p$ and the displayed congruences, w divides a two-by-two minor.
◇	An existence check on a period identity empty-data	`GlobalTheory.Fam.volLev_mul_wt_mid_alt` VolLevMulWtMid.lean, L24–51	At a suitable intermediate stage there exists $v\neq 0$ such that level volume times character weight equals v at every admissible datum.
◇	The lattice slope inequality slope	`Formalism.F.slopeInequality_of` SlopeInequality.lean, L111–125	The sublattice bound `SlopeLeMuMax` implies `SlopeInequality` for injective norm-bounded lattice maps.
○	The square-root multiplicity analogy paper-chain	no declaration (motivation)	The author describes repeated theta lifts as a number-field analogue of multiplicity in the function-field argument.
10	**Hosts for every character** 1 nodes		
✓	Produce cyclic character hosts supply	`UpperTheory.Top.supplyChi` SupplyChi.lean, L8–16	For every primitive ψ, `VWinAt` ψ 401 implies host existence at width 3/2.
11	**The cohomological window** 25 nodes		
◇	Weighted lengths of local-global kernels vv	`UpperTheory.Top.VVle` ExistsPageN.lean, L12–17	For all admissible local data, lengths are finite for $1\le e<t$ and every finite prime sum of $\log(\ell)$ times the sum of these lengths is $\le B$.
⊂	The cohomological window vwin-def	`UpperTheory.Top.VWinAt` ExistsPageN.lean, L44–47	For every $\varepsilon\in(0,1/52)$ and admissible layer choice e(x), sufficiently large X and each prescribed pair q,d admit a window $[Y,Y^2]$ where `VAt` holds.
◇	Cohomological window from the row vwin-row	`UpperTheory.One.vwin_of_row` VwinOfRow.lean, L4–5	`Row0b` in the stated convention implies `VWin`.
◇	Supply the automorphic window inputs window-inputs	`UpperTheory.One.Join.automorphic_window_of_inputs` TopMainZj.lean, L71–74	`OpenImageCondition`, `BaseChangeIsobaricN` and `RowT` imply `VWin`.
◇	Build the rank-N package depth-window	`UpperTheory.One.Join.v_window_of_depth_zero_residual_bc_cm` TopMainZj.lean, L55–58	The open-image, base-change, Galois and CM packages imply `VWin`.
◇	Window from rank-N automorphy lit-window	`UpperTheory.V.v_window_of_literal_automorphy_n` WindowBoundOfLiteratureLevel.lean, L33–34	`LiteratureRankNHypothesis` implies `VWin`.
◇	Pass through the descent hypotheses patch-window	`UpperTheory.V.HV.DescentVanishingReductions.v_window_of_automorphy_csd_ss_mono_odd` WindowBoundOfLiteratureLevel.lean, L30–31	`LiteratureRankNHypothesis` implies `VWin`.
◇	Produce the four counting bounds four-produce	`UpperTheory.V.HV.DescentVanishingReductions.vwin_of_hypotheses_H_E_alt` WindowBoundOfLiteratureLevel.lean, L25–29	`HypothesesH′` imply `VWin`.
◇	Combine the cohomology counts four-use	`UpperTheory.V.HV.Core.of_four_alt` HVCore.lean, L18–22	Rank-N input, an existential K bound and `CountI`, `CountII`, `CountIII`, `CountV` imply `VWin`.
✓	Assemble the arithmetic tower window tower-window-from-lemma-k	`UpperTheory.V.Win.thmBab_of_lemmaK` ThmBabOfLemmaK.lean, L9–12	One package of constants works at every cutoff exponent $Ct\ge 1$.

	NODE	DECLARATION (PINNED)	CONTRACT
✓	Choose one set of tower constants tower-window-constants	`UpperTheory.V.Win.thmBab_of` ThmBabOf.lean, L11–103	There exists a single `BConst` K such that every Ct ≥ 1 satisfies `ThmBab` K Ct.
✓	From a finitely generated tower to two length bounds tower-bound-from-rank	`UpperTheory.V.PropA.aBound_of_isTower` ABoundOfIsTower.lean, L7–11	For each prime and eligible tower, `ABound` consists of a bound away from finitely many centres and a bound in terms of a fixed layer.
⊂	Control exceptional twists by the fixed layer tower-operator-bound	`UpperTheory.V.PropA.aBound_of_descends` ABoundOfDescends.lean, L36–55	The lengths are controlled by at most R ℓ-adic centres. For all twists they are also bounded by the fixed-layer H^2 length plus R times the excess layer index.
◇	Uniform ℓ-rank of class groups in the tower tower-class-rank	`UpperTheory.V.FW.ferreroWashington` FerreroWashington.lean, L104–119	For each eligible abelian base K and prime ℓ there is an exponent r bounding the order of the quotient by ℓ-th powers by ℓ^r on every cyclotomic layer.
◇	Supply the absolute bound for the reference layer tower-fixed-growth	`UpperTheory.V.ZK.lemmaK_litN_alt` KFinal.lean, L11	One real exponent bounds the logarithm of the order of $H^2(G_{L,S}, \mathbb{Z}_\ell(e+1))$ over all eligible fields, coefficient primes and positive twists. Each such group is finite.
◇	Case I gives one bound across coefficient primes counts-i	`UpperTheory.V.Z.J.count_i_of_printed_bound_k_kato_automorphy` LinkedUnIE.lean, L49–54	Away from the exceptional set, when $\ell-1$ does not divide twice the twist, the sum over every allowed finite set of coefficient primes obeys one polynomial bound.
◇	Cases II and III from the rank-four symplectic argument counts-ii-iii	`UpperTheory.V.HV.DescentVanishingReductions.countII_and_countIII_of_printsH_of_sp4In` AdjointLiftingOfCohomology.lean, L33–39	Cases II and III have uniform polynomial bounds for each normalized character length, with finiteness. The hypotheses distinguish parity of character orders and divisibility of twice the twist.
◇	Case V at twist one counts-v	`UpperTheory.V.Z.J.count_v_of_printed_bound_k_reduced_unitary_lift` LinkedUnIE.lean, L55–61	For exceptional odd primes at twist one, Case V bounds the normalized length when the two-adic conductor crosses the specified threshold, including equality.
◇	A nonzero integer controls one character orbit counts-common-integer	`UpperTheory.V.Z.J.propGK_of_numberU3U_of_oneCharU` FPropGKU.lean, L110–172	For a fixed imaginary quadratic field, twist, character kernel and admissible auxiliary prime, one nonzero integer gives a logarithmic size bound and simultaneous valuation bounds at the other eligible primes.
◇	Convert the joint window into `VWin` counts-window	`UpperTheory.V.HV.vwin_of_counts` JointWinOfCounts.lean, L30–33	The tower window, sign bounds and five counting inputs yield `VWin`, the cohomology estimate for each primitive character at cutoff exponent 401.
◇	Reuse the tower interval for the positive part counts-joint	`UpperTheory.V.HV.jointWin_of_counts` JointWinOfCounts.lean, L25–28	For each eligible choice of character and parameters, one interval satisfies both the tower and positive-part bounds at every point.
◇	Evaluate the pointwise estimate on admissible windows counts-height	`UpperTheory.V.HV.thmHV_of_counts` JointWinOfCounts.lean, L20–23	Every package K and positive cutoff exponent Ct satisfy the positive-part height theorem under the stated counting and cohomology inputs.
◇	Combine the two prime sums at a point counts-pointwise	`UpperTheory.V.HV.pointHV_of_counts` JointWinOfCounts.lean, L16–18	After the character and layer indices are fixed, one polynomial bound controls the positive cohomological sum on eligible hosts, with an additive multiple of the layer budget.
◇	Add contributions inside the exceptional set counts-inside	`UpperTheory.V.HV.sumIn` FinalLe.lean, L86–249	For fixed character, exceptional coefficient prime and layer index, the positive-part sum at that prime is at most $(Nt)^C + C\cdot B$ when the fixed layer satisfies its budget B.
◇	Bound the sum over arbitrary finite sets of other primes counts-outside	`UpperTheory.V.HV.sumOut` SumOut.lean, L50–231	Choose a common exponent and two auxiliary primes before the host. The positive-part sum over every allowed finite set of nonexceptional odd primes is at most $(Nt)^C$.

12 Auxiliary primes and Behrend towers 2 nodes

	NODE	DECLARATION (PINNED)	CONTRACT
✓	Behrend's collision-free indices behrend	`Formalism.FrobeniusOrbits.ThickExists.noRel_image` ThickExists.lean, L85–95	For L≥1, encoding an integer sphere in base 4Ld gives a set with `NoRel` for coefficients bounded by L.
○	The tower selection in the timeline paper-behrend	no declaration (motivation)	Entry 19 links the choice of a tower to Behrend-type sets.
13	**The Galois row** 6 nodes		
◇	The local–global Galois row row-def	`UpperTheory.GalStar.Row0b` WeightsConjBis.lean, L69–73	For a regular algebraic polarized cuspidal representation over a CM field and a prime ℓ, there exist a Galois representation, weight and Hodge–Tate multiset satisfying `PropsB`.
◇	Producer of the Galois row row-prod	`UpperTheory.Gal.Int.galoisLocalGlobalStrong_recParameter` RowSrc.lean, L6	`Row0b recT` with no explicit global premise.
◇	Galois representations from theta lifting row-geo	`UpperTheory.Gal.Int.galoisRepresentations_of_thetaLift` GaloisRepresentationsOfThetaLift.lean, L6	`TLiftK` implies `Row0b recT`.
◇	Theta-lift existence at high rank tlift	`UpperTheory.TP.K.tLiftK` TLiftK.lean, L4–5	For the admissible CM data of `TLiftK`, for all sufficiently high dimensions and allowed weights a lift U′ exists.
◇	Weight-one geometric realization geo-one	`UpperTheory.Gal.Int.Geo.geoOne_holds` WeightOneGeometricRealization.lean, L11	`GeoOne` with no explicit premise.
◇	Quantifiers of the theta lift tlift-def	`UpperTheory.TP.TLiftK` TLiftK.lean, L12–23	For each admissible CM setup, n≥1 and X, there exists d_0; for each $d \geq d_0$, permissible W, Hecke characters, weights and U, an `IsLift` witness U′ exists.
14	**Seven inputs at the decisive call** 16 nodes		
◇	Closed half-plane assembly every	`UpperTheory.FinalTheorems.everyCharacter` MainConclusion.lean, L10–11	`EveryCharacter` with no explicit arithmetic premise.
◇	The half-plane theorem from one row from-row	`UpperTheory.FinalTheorems.everyCharacter_of_row` MainConclusion.lean, L7–8	`Row0b` with the specified local parameter convention implies `EveryCharacter`.
◇	Existence of local Langlands parameters rec	`UpperTheory.Zero.LLC.recTExists` RecTExists.lean, L20–31	`UpperTheory.Zero.LLC.RecTExists`.
◇	Assemble the automorphic inputs lit	`UpperTheory.LiteratureTheorems.every_character_of_literature_of_vWin` EveryCharacterOfLiteratureOfVWin.lean, L810–813	`OpenLiteratureStatements` and `VWin` imply `EveryCharacter`.
◇	Three fields in the literature input open-record	`UpperTheory.LiteratureTheorems.OpenLiteratureStatements` EveryCharacterOfLiteratureOfVWin.lean, L215–218	The record has fields Row, `RecTExists` and Row, in that order.
◇	The unitary and eigenvalue package eigen-package	`UpperTheory.LiteratureTheorems.published_results_eigen` EveryCharacterOfLiteratureOfVWin.lean, L666–672	`OpenLiteratureStatements` imply `UnitaryGroupThirdPlaceHypotheses`.
◇	The local theta package theta-package	`UpperTheory.LiteratureTheorems.published_results_first` EveryCharacterOfLiteratureOfVWin.lean, L727–735	`OpenLiteratureStatements` imply `ThetaLocalInputs`.
◇	The Hodge-bundle package hodge-package	`UpperTheory.LiteratureTheorems.published_results_arithmetic` EveryCharacterOfLiteratureOfVWin.lean, L781–786	`OpenLiteratureStatements` imply `PublishedHodgeBundleTheorems`.
◇	The local parameter input at the call site local-rec	`UpperTheory.LiteratureTheorems.local_langlands_gl_recT_exists_of_literature` EveryCharacterOfLiteratureOfVWin.lean, L313–314	`OpenLiteratureStatements` imply `RecTExists` in the GL convention.

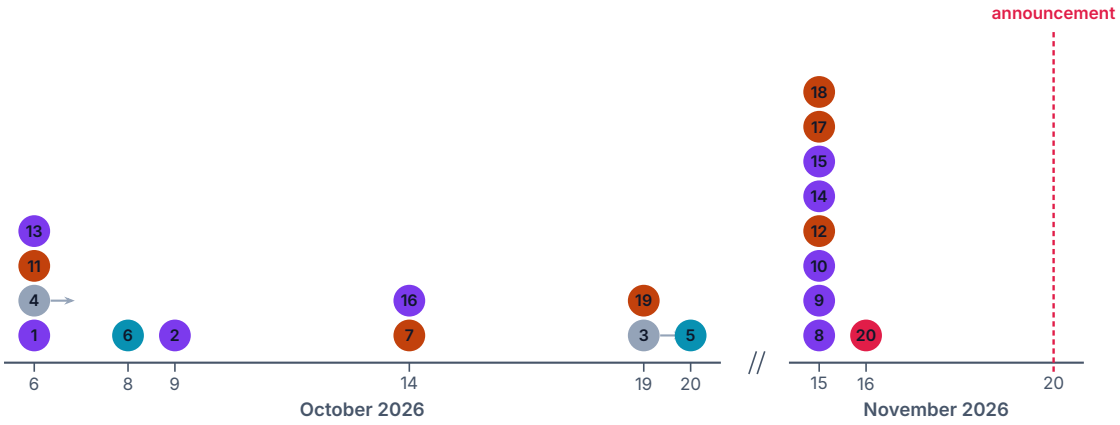
	NODE	DECLARATION (PINNED)	CONTRACT
◇	Read the parameter field local-rec-proj	`UpperTheory.LiteratureTheorems.local_langlands_recT_exists_of_literature` EveryCharacterOfLiteratureOfVWin.lean, L283–284	`OpenLiteratureStatements` imply `RecTExists`.
◇	Regular characters yield the Green input green	`GlobalTheory.GreenBridge.green_of_regular` SumRangeEqRoots.lean, L225–233	`GreenRegularCharacters` imply `GLq.Green`.
◇	Bind the estimates at 2 and 3 linked-two-three	`UpperTheory.N.boundAtTwoThree_linked_of_recT` At23OfAll.lean, L24–26	For local-global inputs, `RecTExists` and a compatible dependence parameter, the bounds at two and three hold on linked data.
◇	Match global integrals and local data theta-match	`UpperTheory.LiteratureTheorems.theta_integrals_agree_and_local_matching_of_literature` EveryCharacterOfLiteratureOfVWin.lean, L565–566	`OpenLiteratureStatements` imply `ThetaIntegralsAgree` ∧ `LocalMatching`.
◇	Produce the covolume bound covol	`SelfIntersectionVolumeEstimates.topCovolSlim2_use_alt_alt` TopCovolSlim2Use.lean, L16–17	The named geometric, arithmetic Hilbert–Samuel and Hodge-bundle hypotheses imply `CovolumeBoundOfX`.
◇	Seven inputs for every character seven	`UpperTheory.Lit.every_character_of_theta_integrals_local_matching` SiegelEisensteinSecondTermHypotheses.lean, L69–73	The seven groups of hypotheses imply `EveryCharacter`.
◇	Build a nonempty definite-side package definite	`UpperTheory.Lit.nonempty_hyps_definite_side_of_closed_bound` SiegelEisensteinSecondTermHypotheses.lean, L44–68	Unitary, theta, matching, covolume, local-parameter and linked-bound data yield Nonempty (`HypsAlt` … `DefiniteSideCondition`).



Figure 11. Blueprint graph: 109 nodes, in two halves (left: the suppliers; right: continued towards the endpoints). Layered dependency graph of the reading project's blueprint: suppliers on the left, the five endpoints on the right; bar colour = stratum, dot colour = verification tier. The node titles are those of the table above.

9 The announcement, entry by entry

The announcement lists 20 dated entries on its pages 3 and 4. The dates are as written in the announcement (a final "+" means *from that date on*, drawn as a short arrow); the relation and the assessment of each entry are the reading project's, and the component is the one of this map to which the entry belongs. Dots are coloured by the stratum of that component.



	DATE	ENTRY, RELATION AND ASSESSMENT	COMPONENT
1	10/6/26 6 October 2026	**Stepanov determinant** (page 3) *Historical motivation.* The single-line entry records an origin of the argument. Its underlying function-field determinant is unspecified.	9 The geometric family and the theta chain
2	10/9/26 9 October 2026	**Invariant differentials on a CM curve** (page 3) *Historical calculation.* The entry describes inert-prime divisibility of iterated derivatives. The argument for this displayed exponent was not identified in the selected source path.	9 The geometric family and the theta chain
3	10/19/26+ 19 October 2026 onwards	**The finite-field sign argument** (page 3) *Historical analogy.* The footnote compares a zero mode with an error bounded by iterated Parseval. Its objects and hypotheses differ from the Dirichlet L-function endpoint.	historical; no component
4	10/6/26+ 6 October 2026 onwards	**Determinantal positivity with multiplicity one** (page 3) *Historical motivation.* The short entry leaves W and S implicit. An exact theorem-to-type identification would require definitions of both.	historical; no component
5	10/20/26 20 October 2026	**The smoothed von Mangoldt sum** (page 3) *Formula matched.* The displayed zeta error has the exact ramp and the correction (1 + Euler's constant)x. Character twists enter the level average. The sentence continues onto page 4.	4 The centered prime sum
6	10/8/26 8 October 2026	**Two-support positivity test** (page 4) *Historical calculation.* The two indicator functions are a proposed model for a zero-free exponent. The final power-window criterion has a different statement.	6 The window criterion
7	10/14/26 14 October 2026	**Interlacing weights and periods** (page 4) *Supporting comparison.* A local Bessel model is constructed from two interlacing conditions. The announcement's claim about determinant values 0 and 1 has further content.	14 Seven inputs at the decisive call
8	11/15/26 15 November 2026	**A chain with square-root length** (page 4) *Quantitative comparison.* The arithmetic rank specification puts m between one and two times x raised to 1/2 minus epsilon/16. The finite kernel recursion explains iteration.	9 The geometric family and the theta chain

	DATE	ENTRY, RELATION AND ASSESSMENT	COMPONENT
9	11/15/26 15 November 2026	**Norms, levels and arithmetic periods** (page 4) *Supporting comparison.* One result relates a CM period to a logarithmic derivative. Another cancels a level volume against an index. The complete order-m cancellation requires more data.	9 The geometric family and the theta chain
10	11/15/26 15 November 2026	**Divisibility of repeated lifts** (page 4) *Supporting comparison.* A specific minor vanishes modulo a divisor and a lifted column has a higher-order remainder. Turning these local facts into the required global gain is an additional step.	9 The geometric family and the theta chain
11	10/6/26 6 October 2026	**The tower of fields** (page 4) *Supporting comparison.* The formal window ranges over fields specified by a fixing subgroup in a cyclotomic extension and by layer parameter e(x).	11 The cohomological window
12	11/15/26 15 November 2026	**Seesaw and Bessel periods** (page 4) *Supporting comparison.* Conditional unfolding identifies a theta-lift period with an integral. A middle-step volume identity is independent of the admissible character. A full iterated comparison is a stronger assertion.	14 Seven inputs at the decisive call
13	10/6/26 6 October 2026	**The slope inequality** (page 4) *Mathematical statement matched.* The Euclidean lattice model bounds the slope of an injectively mapped source by the maximal target slope plus the logarithm of the map bound. Identification of the particular period lattice is separate.	9 The geometric family and the theta chain
14	11/15/26 15 November 2026	**The provisional exponent two-thirds** (page 4) *Intermediate strategy.* The author describes a barrier for an earlier estimate. The endpoint reads a one-half-plus-loss exponent. The earlier barrier is not derived in this comparison.	8 The finite character comparison
15	11/15/26 15 November 2026	**A common integral structure across characters** (page 4) *Supporting comparison.* The final deduction imposes local and global controls on the same family. A precise comparison with the gcd in the historical description remains open.	9 The geometric family and the theta chain
16	10/14/26 14 October 2026	**Period pairings and approximate orthogonality** (page 4) *Supporting comparison.* Unfolding relates a Poincare-series pairing to an integral of periods. The later host comparison turns suitable bounds into an estimate for the weighted prime sum.	8 The finite character comparison
17	11/15/26 15 November 2026	**Cohomology and a weakening of Vandiver** (page 4) *Quantitative comparison.* The window requires finite lengths and an aggregate weighted bound. Further implications for individual K-groups are conjectural comments in the announcement.	11 The cohomological window
18	11/15/26 15 November 2026	**Covolumes, lattice indices and congruences** (page 4) *Supporting comparison.* A covolume bound is an explicit premise of the automorphic assembly. The local congruence lemma gives a representative higher-order equality.	14 Seven inputs at the decisive call
19	10/19/26 19 October 2026	**A Behrend set in the tower construction** (page 4) *Supporting comparison.* Encoding a populated integer sphere in a large base excludes short affine relations and controls intersections with short orbits.	12 Auxiliary primes and Behrend towers
20	11/16/26 16 November 2026	**Dirichlet GRH** (page 4) *Endpoint statement matched.* The source declares that zeros of every Dirichlet L-function in the open critical strip are on the critical line. This entry records source-statement correspondence.	1 The five endpoints

10 Sources, pins and reproduction

10.1 Pins

Announcement PDF (SHA-256)	`3fc97784b71ddd44ff8aa0a177e99622a1c61b13cfa843a0989ee4a7a0cfeb8d`
Lean archive (SHA-256 of the gzip stream, 14 parts)	`9631b4f020199e580ba86178cb56c6e9b09f690c2369a80dc4334aa93f49c7fc`
Lean files in the archive	175,187
Archive toolchain	`leanprover/lean4:v4.33.1`
Mathlib commit	`0df444a360eaa60ab8c11dca51a86af692955474`
Public mirror (all source links)	`Deicyde/riemann-hypothesis` at `40e56bab026552e089d7afcfeff08d2defb9c005`
Reading project commit imported	`348ec48748e0ea536aba7d64351b2ee2f7d116d7`
Import time	2026-11-21 00:17:58 UTC
Status record as of	2026-11-20 23:51:25 UTC, Helios job `26647685`
Prime-window proof export	61,697 declarations; 89,615,680 bytes compressed (SHA-256 `125bdd7041764c898f4d5b31653f3794abb61be09e484c1b791150f86cc084d0`)

10.2 The status record

The reading project's status record as of 20 November 2026, 23:51 UTC, quoted verbatim ("we" is the reading project). It reports on the archive; this repository did not replay it.

Headline. "Two independent kernels accepted the complete proof, and all five theorems matched the canonical Mathlib challenge."

Checks made by the reading project. "At Lean 4.33.1 we checked 24 named original results and 21 reading deductions across 2,244 distinct original modules. We also checked five endpoint equivalences and an Euler-factor boundary example. The one-sided prime-window implication passed fresh Lean replay with Mathlib and a second kernel on a portable proof of 61,697 declarations. The two-input deduction of ordinary RH and Dirichlet GRH, four steps from finite character comparison to the prime-sum bound and the auxiliary-prime theorem for three changing quadratic twists all passed. `LemmaK`, `FixedIn` and `TwoIn` yield one record of cohomological window constants for every cutoff exponent $C_t \geq 1$. From Poitou–Tate duality, local finiteness and local Tate duality, a tower gives its cohomological bound. Adding Ferrero–Washington and `LemmaK` yields the joint window under those same hypotheses. Eight further teaching declarations use Lean 4.35.0. At each fixed primitive character, `TopG` and just its own cohomological estimate yield a cyclic-family bound with coefficient one. For the principal character modulo one this is the exact centered ordinary prime sum. The archive supplies the ordinary first moment unconditionally. Fresh kernel replay establishes that Schoenfeld's RH-to-Chebyshev estimate suffices for the converse from RH to the window bound."

External report and authentication. "Andrew Sutherland and Simon `DeDeo` each report independently building and replaying all 175,187 original modules with Lean 4.33.1. `DeDeo` reports zero errors and crashes. Sutherland additionally records a con-leche comparison of the five targets and a lean4lean check of every original module. Five modules required higher reduction limits. The individual logs identify the names, flags and successful reruns."

The implication at the centre. "The first theorem has Mathlib's type `RiemannHypothesis`. The other endpoints give the zeta zero statement and the half-plane and critical-strip results for Dirichlet characters at every positive modulus. The arithmetic proof constructs the windows it needs for a prescribed character. Each exported proof uses precisely the three standard explicit axioms."

Build job (Helios 26647685). “Building the original library targets and Main with 48 Lake workers and persistent group objects.

External sources, as recorded by the reading project:

- Sutherland’s five named con-leche results
- Sutherland’s five raised-fuel lean4lean reruns
- Full solution and dated operator report
- Second complete original-toolchain replay
- Five individual axiom lists
- Rust process log
- Download checksums
- Earlier challenge comparison
- Independent five-target comparison
- Independent rebuild with a separately generated proof export

10.3 Provenance of the imported data

The data were imported from the reading project at commit `348ec48748e0ea536aba7d64351b2ee2f7d116d7` on 21 November 2026, 00:17 UTC; the files and their SHA-256 prefixes are:

IN THIS REPOSITORY	IMPORTED FROM THE READING PROJECT	SHA-256 (PREFIX)
`data/source/map.json`	`blueprint/map.json`	`26ad5dd085d155ba`
`data/source/status.json`	`evidence/status.json`	`d4afbf5694ca44ca`
`data/source/sources.json`	`evidence/sources.json`	`19f64f8f967813cd`
`data/source/timeline.json`	`research/agents/arithmetic/timeline.json`	`8cc2d4c4dedc8f7c`
`data/source/native-scopes.json`	`dashboard/data/native-scopes.json`	`13b55c110571a997`
`data/source/helios-progress.json`	`evidence/helios/progress.json`	`44d9fd51367c85aa`
`lean/Reading/WindowLogic.lean`	`formal/standalone/WindowLogic.lean`	`cd648265b6aa58d8`
`lean/Reading/BehrendSphere.lean`	`formal/behrend/BehrendSphere.lean`	`a026ef4bc8c5eee7`
`references/archive-bridges/WindowBridge.lean`	`formal/window-bridge/WindowBridge.lean`	`e77eb05549e7622b`
`references/archive-bridges/GeometricTransport.lean`	`formal/window-bridge/GeometricTransport.lean`	`f6213d4c45e70a1a`
`references/archive-bridges/DirichletBridge.lean`	`formal/dirichlet-bridge/DirichletBridge.lean`	`59d543a5550f953b`
`references/archive-bridges/EndpointAudit.lean`	`formal/upstream/EndpointAudit.lean`	`7946558ed7e5f2da`

10.4 Reproduction

Everything in this PDF is regenerated from the repository by the commands below (run from its root). The generator is deterministic; the PDF date is fixed to the status time through `SOURCE_DATE_EPOCH`.

```
python3 -B scripts/model.py                # data/model.json from the model and data/source
python3 -B scripts/build_figures.py        # assets/figures/*.svg and facts.json
python3 -B scripts/build_blueprint_tex.py  # blueprint/generated/*.tex
make -C blueprint                          # SVG -> PDF (rsvg-convert), LuaLaTeX via latexmk
(cd lean && lake build)                    # the skeleton and its axiom audit (lean/lean-toolchain)
```

The LaTeX build needs TeX Live (LuaLaTeX with the fonts Libertinus, Inter and JetBrains Mono, all part of TeX Live) and `rsvg-convert`; intermediates go to `blueprint/build/`.