Carnegie Mellon University

A Lean Formalisation of Sphere Packing in Dimension 8

Swiss Mathematical Society Spring Meeting: Formalisation and Proof Assistants

Sidharth Hariharan
based on joint work with Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta and Maryna Viazovska

27 March, 2026

sidharthhariharan@cmu.edu

Contents

What did we do?

Project Blueprint

Project Blueprint

Why write a blueprint?

Why write a blueprint?

- A blueprint is… a blueprint!

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

- Initial version written by Maryna based on original paper [6]

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

- Initial version written by Maryna based on original paper [6]
- First major modifications: Seewoo's proofs of the modular form inequalities [5]

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

- Initial version written by Maryna based on original paper [6]
- First major modifications: Seewoo's proofs of the modular form inequalities [5]
- Second major modifications: Library building for sphere packing theory (mostly by Gareth Ma)

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

- Initial version written by Maryna based on original paper [6]
- First major modifications: Seewoo's proofs of the modular form inequalities [5]
- Second major modifications: Library building for sphere packing theory (mostly by Gareth Ma)
- Third major modifications: Upstreaming parts of my MSc thesis [4]

Why write a blueprint?

- A blueprint is… a blueprint!
- Guide the effort by fleshing out some of the finer details of the proof that are anticipated to come up during formalisation.
- A blueprint is a **working document**!

Blueprint Timeline

- Initial version written by Maryna based on original paper [6]
- First major modifications: Seewoo's proofs of the modular form inequalities [5]
- Second major modifications: Library building for sphere packing theory (mostly by Gareth Ma)
- Third major modifications: Upstreaming parts of my MSc thesis [4]

`https://thefundamentaltheor3m.github.io/Sphere-Packing-Lean/blueprint`

Defining Sphere Packings in Lean

Defining Sphere Packings in Lean

The important data of a sphere packing can be captured by the set of centres of the balls making up the packing. We thus define a sphere packing to just be the set of centres.

Defining Sphere Packings in Lean

The important data of a sphere packing can be captured by the set of centres of the balls making up the packing. We thus define a sphere packing to just be the set of centres.

```
structure SpherePacking (d : ℕ) where
  centers : Set (EuclideanSpace ℝ (Fin d))
  separation : ℝ
  separation_pos : 0 < separation := by positivity
  centers_dist : Pairwise (separation ≤ dist · · : centers → centers → Prop)
```

Defining Sphere Packings in Lean

The important data of a sphere packing can be captured by the set of centres of the balls making up the packing. We thus define a sphere packing to just be the set of centres.

```
structure SpherePacking (d : ℕ) where
  centers : Set (EuclideanSpace ℝ (Fin d))
  separation : ℝ
  separation_pos : 0 < separation := by positivity
  centers_dist : Pairwise (separation ≤ dist · · : centers → centers → Prop)
```

```
structure PeriodicSpherePacking (d : ℕ) extends SpherePacking d where
  lattice : Submodule ℤ (EuclideanSpace ℝ (Fin d))
  lattice_action : ∀ ⦃ x y ⦄, x ∈ lattice → y ∈ centers → x + y ∈ centers
  lattice_discrete : DiscreteTopology lattice := by infer_instance
  lattice_isZLattice : IsZLattice ℝ lattice := by infer_instance
```

Defining Sphere Packings in Lean

The important data of a sphere packing can be captured by the set of centres of the balls making up the packing. We thus define a sphere packing to just be the set of centres.

```
structure SpherePacking (d : ℕ) where
  centers : Set (EuclideanSpace ℝ (Fin d))
  separation : ℝ
  separation_pos : 0 < separation := by positivity
  centers_dist : Pairwise (separation ≤ dist · · : centers → centers → Prop)
```

```
structure PeriodicSpherePacking (d : ℕ) extends SpherePacking d where
  lattice : Submodule ℤ (EuclideanSpace ℝ (Fin d))
  lattice_action : ∀ ⦃ x y ⦄, x ∈ lattice → y ∈ centers → x + y ∈ centers
  lattice_discrete : DiscreteTopology lattice := by infer_instance
  lattice_isZLattice : IsZLattice ℝ lattice := by infer_instance
```

```
abbrev SpherePacking.balls (S : SpherePacking d) : Set (EuclideanSpace ℝ (Fin d)) :=
  ⋃ x : S.centers, ball (x : EuclideanSpace ℝ (Fin d)) (S.separation / 2)
```

Stating the Main Theorem

Stating the Main Theorem

```
noncomputable def SpherePacking.density (S : SpherePacking d) : ℝ ≥ 0 ∞ :=
  limsup S.finiteDensity atTop
```

Stating the Main Theorem

```
noncomputable def SpherePacking.density (S : SpherePacking d) : ℝ ≥ 0 ∞ :=
  limsup S.finiteDensity atTop
```

```
def SpherePackingConstant (d : ℕ) : ℝ ≥ 0 ∞ := ⨆ S : SpherePacking d, S.density
```

Stating the Main Theorem

```
noncomputable def SpherePacking.density (S : SpherePacking d) : ℝ ≥ 0 ∞ :=
  limsup S.finiteDensity atTop
```

```
def SpherePackingConstant (d : ℕ) : ℝ ≥ 0 ∞ := ⨆ S : SpherePacking d, S.density
```

```
noncomputable def E8Packing : PeriodicSpherePacking 8 where
  separation := √2
  lattice := E8Lattice
  centers := E8Lattice
  centers_dist := by ⋯
```

Stating the Main Theorem

```
noncomputable def SpherePacking.density (S : SpherePacking d) : ℝ ≥ 0 ∞ :=
  limsup S.finiteDensity atTop
```

```
def SpherePackingConstant (d : ℕ) : ℝ ≥ 0 ∞ := ⨆ S : SpherePacking d, S.density
```

```
noncomputable def E8Packing : PeriodicSpherePacking 8 where
  separation := √2
  lattice := E8Lattice
  centers := E8Lattice
  centers_dist := by ⋯
```

```
theorem SpherePacking.MainTheorem : SpherePackingConstant 8 = E8Packing.density :=
  sorry
```

Mathematical Outcomes (A Very Proper Subset)

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

- Fixing the definition of E_2
- q-expansions of the Eisenstein Series
- H-functions (4th powers of Jacobi Θ-functions) as modular forms
- Definition/API for Δ, relation to η
- Dimension formula for level one

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

- Fixing the definition of E_2
- q-expansions of the Eisenstein Series
- H-functions (4th powers of Jacobi Θ-functions) as modular forms
- Definition/API for Δ, relation to η
- Dimension formula for level one

Construction of the Magic Function

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

- Fixing the definition of E_2
- q-expansions of the Eisenstein Series
- H-functions (4th powers of Jacobi Θ-functions) as modular forms
- Definition/API for Δ, relation to η
- Dimension formula for level one

Construction of the Magic Function

- Definitions and API for a, b, g, their integrands and parametrisations
- Proving `PolyFourierCoeffBound`, a tool to analyse asymptotic behaviour
- Systematically bounding the integrands of a
- Proving slash action relations for ψ_I, ψ_S, ψ_T
- Cauchy-Goursat for unbounded rectangles

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

- Fixing the definition of E_2
- q-expansions of the Eisenstein Series
- H-functions (4th powers of Jacobi Θ-functions) as modular forms
- Definition/API for Δ, relation to η
- Dimension formula for level one

Construction of the Magic Function

- Definitions and API for a, b, g, their integrands and parametrisations
- Proving `PolyFourierCoeffBound`, a tool to analyse asymptotic behaviour
- Systematically bounding the integrands of a
- Proving slash action relations for ψ_I, ψ_S, ψ_T
- Cauchy-Goursat for unbounded rectangles

Quasimodular Forms and Inequalities

Mathematical Outcomes (A Very Proper Subset)

Sphere Packing Theory

- Definitions of sphere packings, periodic packings, density, sphere packing constant, periodic sphere packing constant
- Density formula for periodic packings
- The theory of the E_8 lattice and the associated sphere packing
- Progress on Cohn-Elkies [2, Theorem 3.1]

Modular Forms

- Fixing the definition of E_2
- q-expansions of the Eisenstein Series
- H-functions (4th powers of Jacobi Θ-functions) as modular forms
- Definition/API for Δ, relation to η
- Dimension formula for level one

Construction of the Magic Function

- Definitions and API for a, b, g, their integrands and parametrisations
- Proving `PolyFourierCoeffBound`, a tool to analyse asymptotic behaviour
- Systematically bounding the integrands of a
- Proving slash action relations for ψ_I, ψ_S, ψ_T
- Cauchy-Goursat for unbounded rectangles

Quasimodular Forms and Inequalities

- Serre derivatives: definitions and API, equivariance and invariance
- Serre derivatives of E_2, E_4, E_6
- Defining F and G, restriction to the imaginary axis, positivity of restrictions
- Modular linear differential equations

Metaprogramming Outcomes

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

```
example : (1 + 3.5 + I) * (1 + I) = 7 / 2 +
    11 / 2 * I := by norm_num1
example : (3 + 4 * I)⁻¹ * (3 + 4 * I) = 1 :=
    by norm_num1
example : 1 + I ≠ 0 := by norm_num1
example : 1 + I ≠ 1 + 2 * I := by norm_num1
example : im ((1 + 3 * I)⁻¹) = -0.3 := by
    norm_num1
example : 10 * re ((1 + 3 * I)⁻¹) = 1 := by
    norm_num1
example : (conj (3 + 4 * I) : ℂ) * (3 + 4 *
    I) = 25 := by norm_num1
```

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

```
example : (1 + 3.5 + I) * (1 + I) = 7 / 2 +
    11 / 2 * I := by norm_num1
example : (3 + 4 * I)⁻¹ * (3 + 4 * I) = 1 :=
    by norm_num1
example : 1 + I ≠ 0 := by norm_num1
example : 1 + I ≠ 1 + 2 * I := by norm_num1
example : im ((1 + 3 * I)⁻¹) = -0.3 := by
    norm_num1
example : 10 * re ((1 + 3 * I)⁻¹) = 1 := by
    norm_num1
example : (conj (3 + 4 * I) : ℂ) * (3 + 4 *
    I) = 25 := by norm_num1
```

Proving `Tendsto` given atomic limits

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

```
example : (1 + 3.5 + I) * (1 + I) = 7 / 2 +
    11 / 2 * I := by norm_num1
example : (3 + 4 * I)⁻¹ * (3 + 4 * I) = 1 :=
    by norm_num1
example : 1 + I ≠ 0 := by norm_num1
example : 1 + I ≠ 1 + 2 * I := by norm_num1
example : im ((1 + 3 * I)⁻¹) = -0.3 := by
    norm_num1
example : 10 * re ((1 + 3 * I)⁻¹) = 1 := by
    norm_num1
example : (conj (3 + 4 * I) : ℂ) * (3 + 4 *
    I) = 25 := by norm_num1
```

Proving `Tendsto` given atomic limits

- Developed mainly by Cameron Freer with inputs from Seewoo Lee and Bhavik Mehta

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

```
example : (1 + 3.5 + I) * (1 + I) = 7 / 2 +
    11 / 2 * I := by norm_num1
example : (3 + 4 * I)⁻¹ * (3 + 4 * I) = 1 :=
    by norm_num1
example : 1 + I ≠ 0 := by norm_num1
example : 1 + I ≠ 1 + 2 * I := by norm_num1
example : im ((1 + 3 * I)⁻¹) = -0.3 := by
    norm_num1
example : 10 * re ((1 + 3 * I)⁻¹) = 1 := by
    norm_num1
example : (conj (3 + 4 * I) : ℂ) * (3 + 4 *
    I) = 25 := by norm_num1
```

Proving `Tendsto` given atomic limits

- Developed mainly by Cameron Freer with inputs from Seewoo Lee and Bhavik Mehta
- Proves `Tendsto (fun z => expr(f₁ z, ..., f ₙ z)) l (nhds c)` where `Tendsto f ᵢ l (nhds a ᵢ)` are known and the expression is continuous

Metaprogramming Outcomes

A `norm_num` extension for $\mathbb{C}$

- Developed mainly by Edison Xie with inputs from Heather Macbeth, Bhavik Mehta, SH
- Initially a `conv`-tactic, `norm_numI`
- Simplifies expressions in $\mathbb{C}$ by splitting to real and imaginary parts, running `norm_num` on each part, and recombining them

```
example : (1 + 3.5 + I) * (1 + I) = 7 / 2 +
    11 / 2 * I := by norm_num1
example : (3 + 4 * I)⁻¹ * (3 + 4 * I) = 1 :=
    by norm_num1
example : 1 + I ≠ 0 := by norm_num1
example : 1 + I ≠ 1 + 2 * I := by norm_num1
example : im ((1 + 3 * I)⁻¹) = -0.3 := by
    norm_num1
example : 10 * re ((1 + 3 * I)⁻¹) = 1 := by
    norm_num1
example : (conj (3 + 4 * I) : ℂ) * (3 + 4 *
    I) = 25 := by norm_num1
```

Proving `Tendsto` given atomic limits

- Developed mainly by Cameron Freer with inputs from Seewoo Lee and Bhavik Mehta
- Proves `Tendsto (fun z => expr(f₁ z, ..., f ₙ z)) l (nhds c)` where `Tendsto f ᵢ l (nhds a ᵢ)` are known and the expression is continuous

```
-- Two atoms, polynomial
example (h₁ : Tendsto f atTop (nhds 0)) (h₂ :
    Tendsto g atTop (nhds 1)) :
    Tendsto (fun z => f z ^ 2 + f z * g z + g
    z ^ 2) atTop (nhds 1) := by tendsto_cont

-- Unused hypotheses don't interfere
example (h₁ : Tendsto f atTop (nhds 0)) (h₂ :
    Tendsto g atTop (nhds 1))
    (unrelated : Tendsto f atBot (nhds 5)) :
    Tendsto (fun z => f z * g z) atTop (nhds
    0) := by tendsto_cont
```

Active Goals (as of our Lean Together talk on 21 January)

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

- Periodic Sphere Packing Constant = Sphere Packing Constant
- Poisson Summation Formula

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

- Periodic Sphere Packing Constant = Sphere Packing Constant
- Poisson Summation Formula

Magic Function Construction

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

- Periodic Sphere Packing Constant = Sphere Packing Constant
- Poisson Summation Formula

Magic Function Construction

- Schwartzness: one-dimensional and eight-dimensional
- Proving that $\Phi_1 \cdots \Phi_6$ are integrable
- Proving that a has double zeroes at lattice points
- Bounding the integrals that make up b

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

- Periodic Sphere Packing Constant = Sphere Packing Constant
- Poisson Summation Formula

Magic Function Construction

- Schwartzness: one-dimensional and eight-dimensional
- Proving that $\Phi_1 \cdots \Phi_6$ are integrable
- Proving that a has double zeroes at lattice points
- Bounding the integrals that make up b

Modular Forms, Quasimodular Forms, and Inequalities

Active Goals (as of our Lean Together talk on 21 January)

Sphere Packing Theory

- Periodic Sphere Packing Constant = Sphere Packing Constant
- Poisson Summation Formula

Magic Function Construction

- Schwartzness: one-dimensional and eight-dimensional
- Proving that $\mathbf{\Phi_1 \cdots \Phi_6}$ are integrable
- Proving that a has double zeroes at lattice points
- Bounding the integrals that make up b

Modular Forms, Quasimodular Forms, and Inequalities

- More modular form identities and positivity results
- Limit computation $\lim_{t \to 0^+} F(it)/G(it) = 18/\pi^2$, and finish proof of the inequalities
- Dimension formula in level 1
- Upstreaming the definition/facts about the Serre Derivative, the proof of the Jacobi identity $\Theta_2^4 + \Theta_4^4 = \Theta_3^4$, and facts about E_2 and Δ to `mathlib`.

Contour Integration

Defining the Magic Function in Lean

Defining the Magic Function in Lean

```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

Defining the Magic Function in Lean

```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

```
def Φ₁ : ℝ → ℂ := fun t ↦ I * Φ₁' r (z₁' t)
                  ⋮
def Φ₆ : ℝ → ℂ := fun t ↦ I * Φ₆' r (z₆' t)
```

Defining the Magic Function in Lean

```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

```
def Φ₁ : ℝ → ℂ := fun t ↦ I * Φ₁' r (z₁' t)
                  ⋮
def Φ₆ : ℝ → ℂ := fun t ↦ I * Φ₆' r (z₆' t)
```

```
def I₁' : ℝ → ℂ := fun x ↦ ∫ t in (0 : ℝ)..1, Φ₁ x t
                   ⋮
def I₆' : ℝ → ℂ := fun x ↦ 2 * ∫ t in Ici (1 : ℝ), Φ₆ x t
```

Defining the Magic Function in Lean

```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

```
def Φ₁ : ℝ → ℂ := fun t ↦ I * Φ₁' r (z₁' t)
                  ⋮
def Φ₆ : ℝ → ℂ := fun t ↦ I * Φ₆' r (z₆' t)
```

```
def I₁' : ℝ → ℂ := fun x ↦ ∫ t in (0 : ℝ)..1, Φ₁ x t
                   ⋮
def I₆' : ℝ → ℂ := fun x ↦ 2 * ∫ t in Ici (1 : ℝ), Φ₆ x t
```

```
def I₁ : V → ℂ := fun x ↦ I₁' ( ‖ x ‖ ^ 2)
                  ⋮
def I₆ : V → ℂ := fun x ↦ I₆' ( ‖ x ‖ ^ 2)
```

Defining the Magic Function in Lean

```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

```
def Φ₁ : ℝ → ℂ := fun t ↦ I * Φ₁' r (z₁' t)
                  ⋮
def Φ₆ : ℝ → ℂ := fun t ↦ I * Φ₆' r (z₆' t)
```

```
def I₁' : ℝ → ℂ := fun x ↦ ∫ t in (0 : ℝ)..1, Φ₁ x t
                   ⋮
def I₆' : ℝ → ℂ := fun x ↦ 2 * ∫ t in Ici (1 : ℝ), Φ₆ x t
```

```
def I₁ : V → ℂ := fun x ↦ I₁' ( ‖ x ‖ ^ 2)
                  ⋮
def I₆ : V → ℂ := fun x ↦ I₆' ( ‖ x ‖ ^ 2)
```

```
def a' : ℝ → ℂ := fun x ↦ I₁' x + I₂' x + I₃' x + I₄' x +
    I₅' x + I₆' x
def a : V → ℂ := fun x ↦ a' ( ‖ x ‖ ^ 2)
```

Defining the Magic Function in Lean

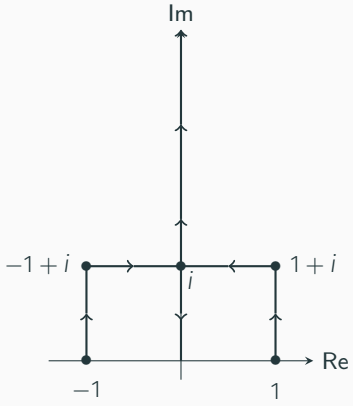
```
def Φ₁' : ℂ → ℂ := fun z ↦ φ₀'' (-1 / (z + 1)) * (z + 1) ^ 2 * cexp ( π * I * r * (z : ℂ))
                   ⋮
def Φ₆' : ℂ → ℂ := fun z ↦ φ₀'' (z) * cexp ( π * I * r * (z : ℂ))
```

```
def Φ₁ : ℝ → ℂ := fun t ↦ I * Φ₁' r (z₁' t)
                  ⋮
def Φ₆ : ℝ → ℂ := fun t ↦ I * Φ₆' r (z₆' t)
```

```
def I₁' : ℝ → ℂ := fun x ↦ ∫ t in (0 : ℝ)..1, Φ₁ x t
                   ⋮
def I₆' : ℝ → ℂ := fun x ↦ 2 * ∫ t in Ici (1 : ℝ), Φ₆ x t
```

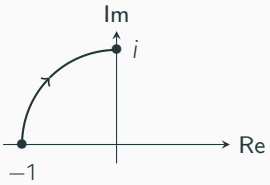
```
def I₁ : V → ℂ := fun x ↦ I₁' ( ‖ x ‖ ^ 2)
                  ⋮
def I₆ : V → ℂ := fun x ↦ I₆' ( ‖ x ‖ ^ 2)
```

```
def a' : ℝ → ℂ := fun x ↦ I₁' x + I₂' x + I₃' x + I₄' x +
     I₅' x + I₆' x
def a : V → ℂ := fun x ↦ a' ( ‖ x ‖ ^ 2)
```

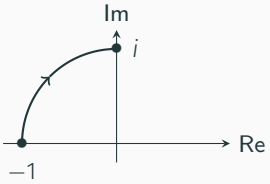


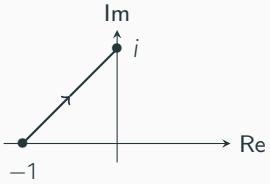
Why use rectangular contours?

Why use rectangular contours?

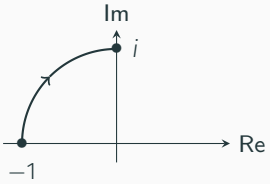


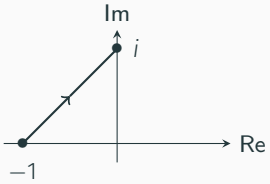
Why use rectangular contours?

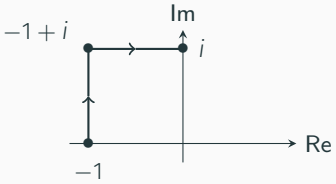




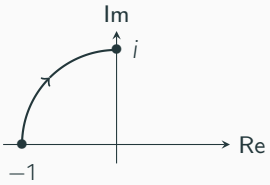
Why use rectangular contours?

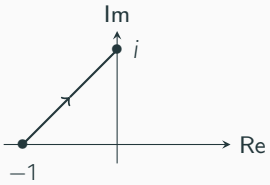


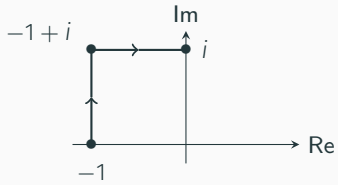




Why use rectangular contours?








```
theorem Complex.integral_boundary_rect_eq_zero_of_differentiable_on_off_countable
        {E : Type u} [NormedAddCommGroup E] [NormedSpace ℂ E] (f : ℂ → E) (z w : ℂ) (s : Set ℂ)
        (hs : s.Countable) (Hc : ContinuousOn f (Set.uIcc z.re w.re ×ℂ Set.uIcc z.im w.im))
        (Hd :

         ∀ x ∈ Set.Ioo (min z.re w.re) (max z.re w.re) ×ℂ Set.Ioo (min z.im w.im) (max z.im w.im) \ s,
          DifferentiableAt ℂ f x
         )
        :
    (((∫ (x : ℝ) in z.re..w.re, f (↑x + ↑z.im * I)) - ∫ (x : ℝ) in z.re..w.re, f (↑x + ↑w.im * I)) + I •
          ∫ (y : ℝ) in z.im..w.im, f (↑w.re + ↑y * I)) - I • ∫ (y : ℝ) in z.im..w.im, f (↑z.re + ↑y * I)
          = 0
```

source

Cauchy-Goursat theorem for a rectangle: the integral of a complex differentiable function over the boundary of a rectangle equals zero. More precisely, if `f` is continuous on a closed rectangle and is complex differentiable at all but countably many points of the corresponding open rectangle, then its integral over the boundary of the rectangle equals zero.

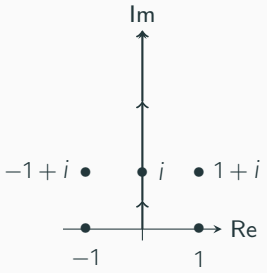
Double Zeroes at Lattice Points

Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz$$

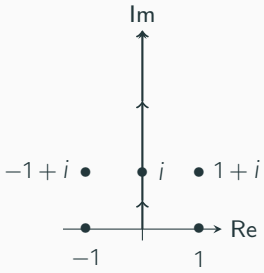
Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz$$



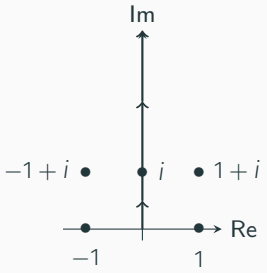
Double Zeroes at Lattice Points

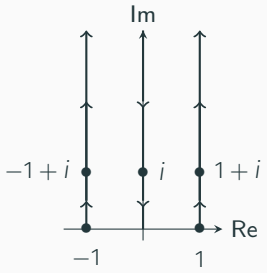
$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}$$



Double Zeroes at Lattice Points

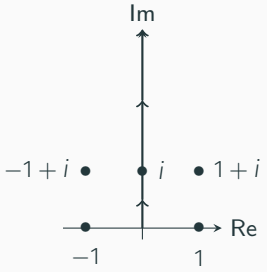
$$-4\sin^2\left(\frac{\pi\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}$$

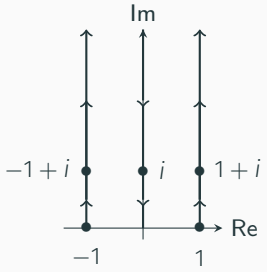




Double Zeroes at Lattice Points

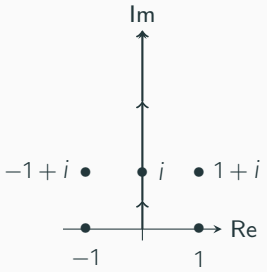
$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}$$

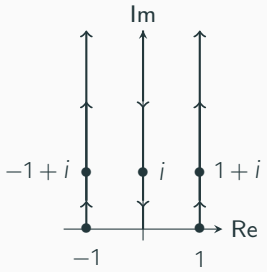


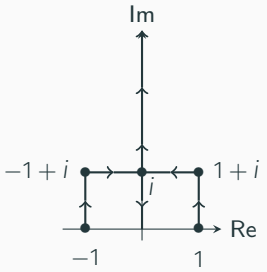


Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}$$

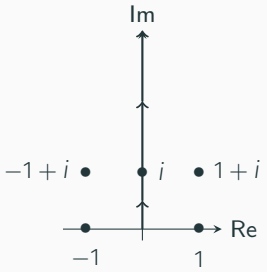


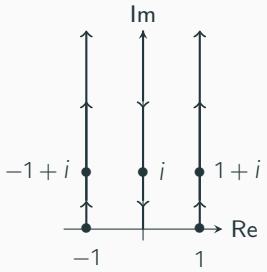


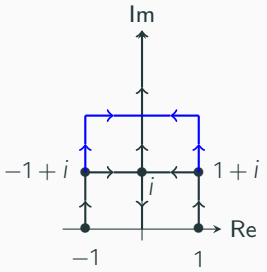


Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}$$

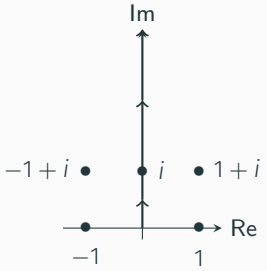


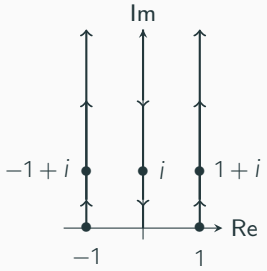


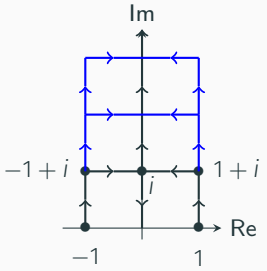


Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}$$

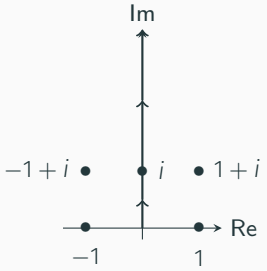


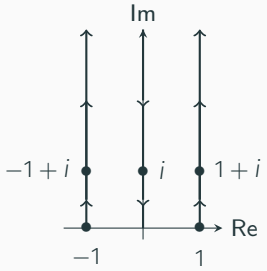


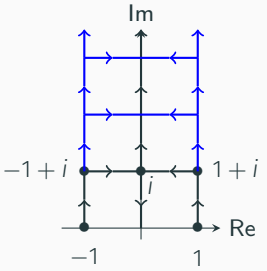


Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}$$

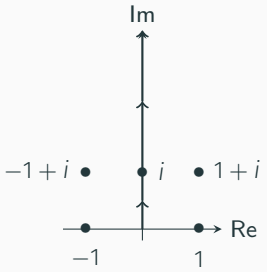


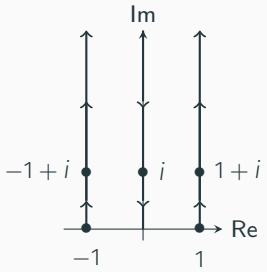


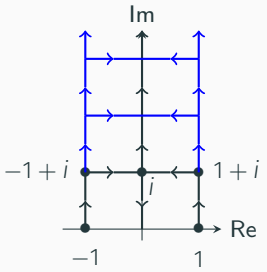


Double Zeroes at Lattice Points

$$-4\sin^2\left(\frac{\pi\,\|x\|^2}{2}\right)\int_0^{i\infty}\phi_0\left(\frac{-1}{z}\right)z^2\,e^{\pi i\|x\|^2 z}\,dz=\int_{-1}^{-1+i\infty}+\int_1^{1+i\infty}-2\int_0^{i\infty}=\int_{-1}^{i}+\int_1^{i}-2\int_0^{i}+2\int_i^{i\infty}=a(x)$$



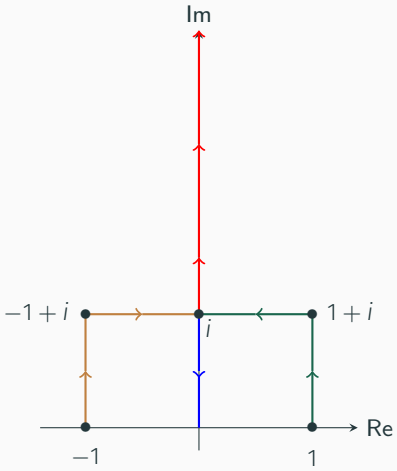




The Fourier Eigenfunction Property

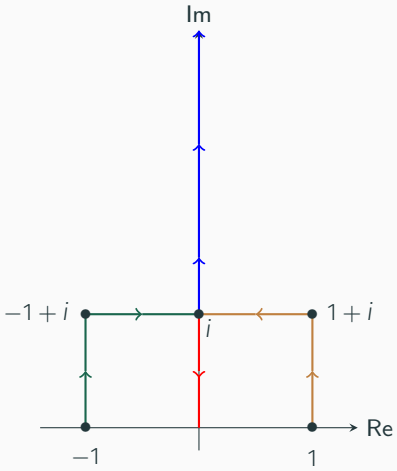
The Fourier Eigenfunction Property

Before taking the Fourier transform:



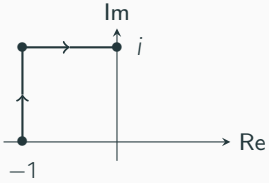
The Fourier Eigenfunction Property

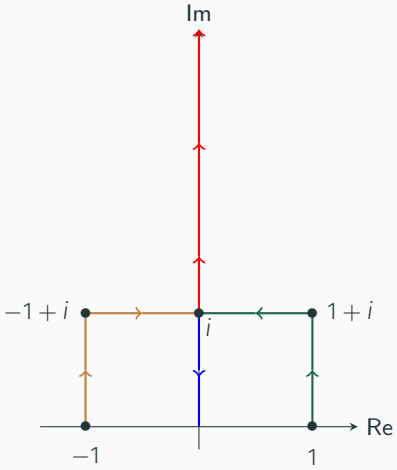
After taking the Fourier transform:



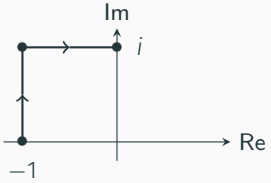
The Fourier Eigenfunction Property

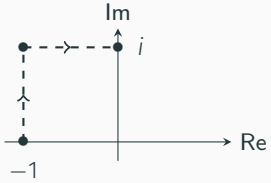
Before taking the Fourier transform:

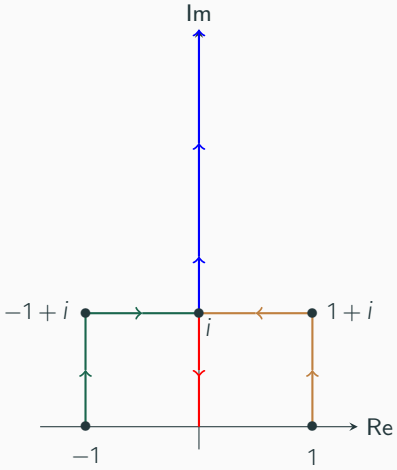




After taking the Fourier transform:

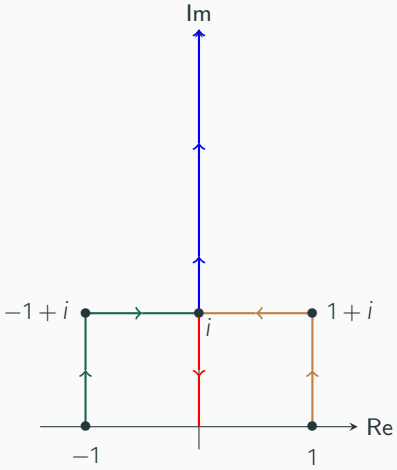


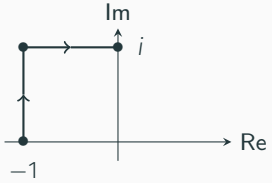


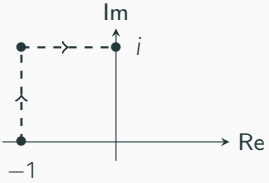


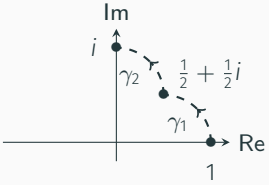
The Fourier Eigenfunction Property

After taking the Fourier transform:

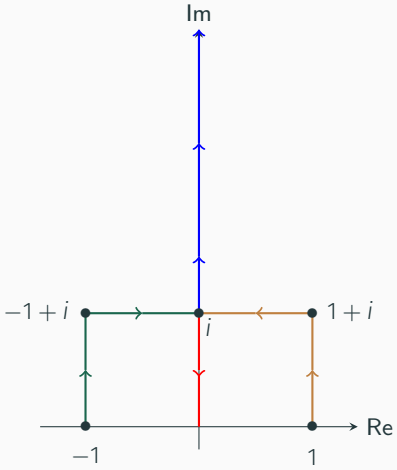


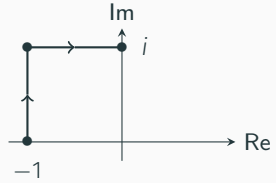


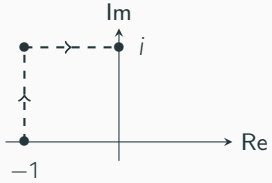


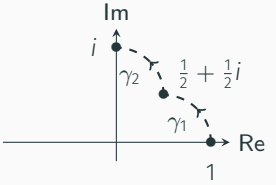


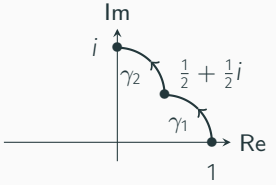
After taking the Fourier transform:



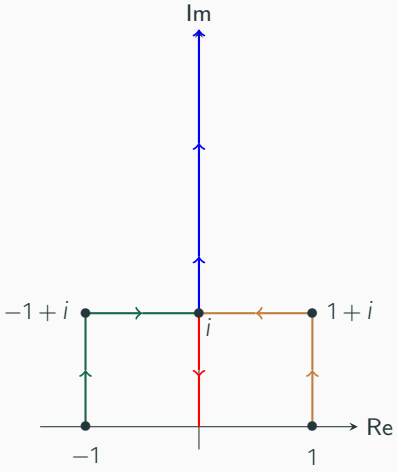


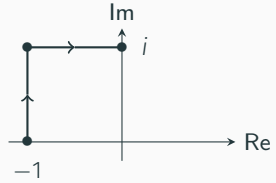


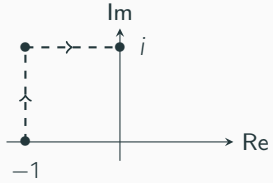


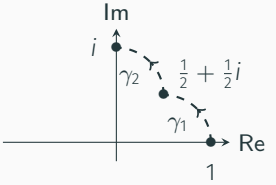


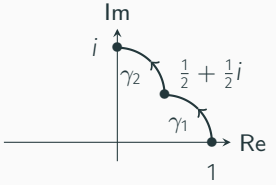
After taking the Fourier transform:

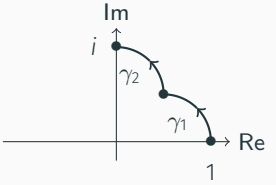




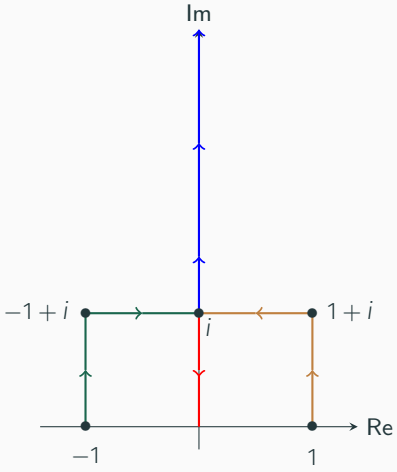


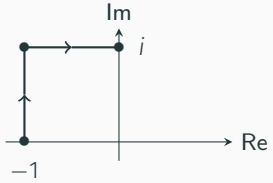


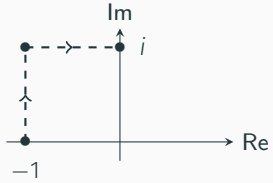


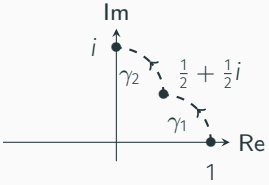


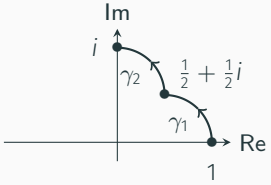
After taking the Fourier transform:

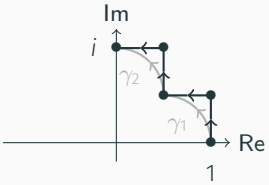




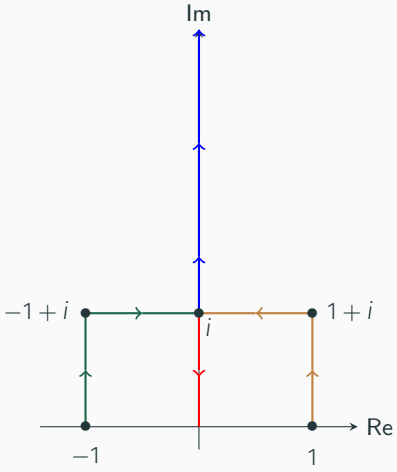


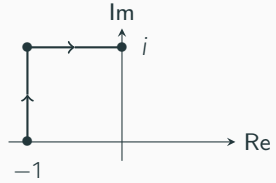


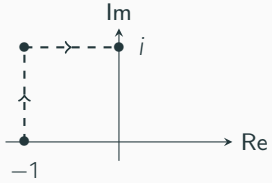


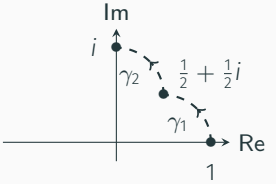


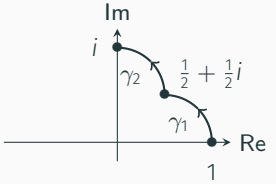
After taking the Fourier transform:

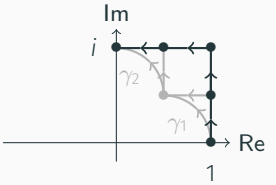




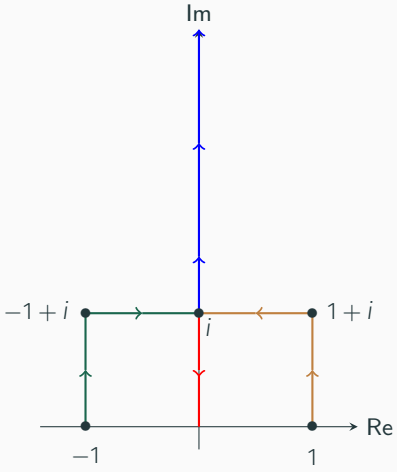


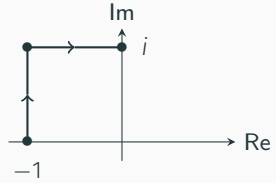


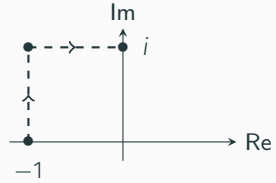


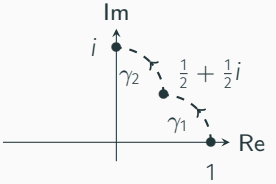


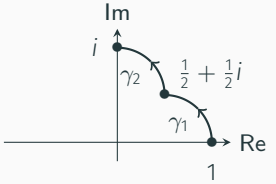
After taking the Fourier transform:

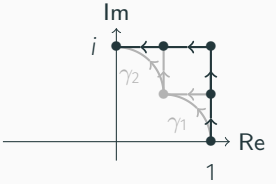


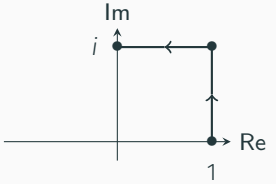












What did Gauss do?

The Old Version of Gauss

The Old Version of Gauss

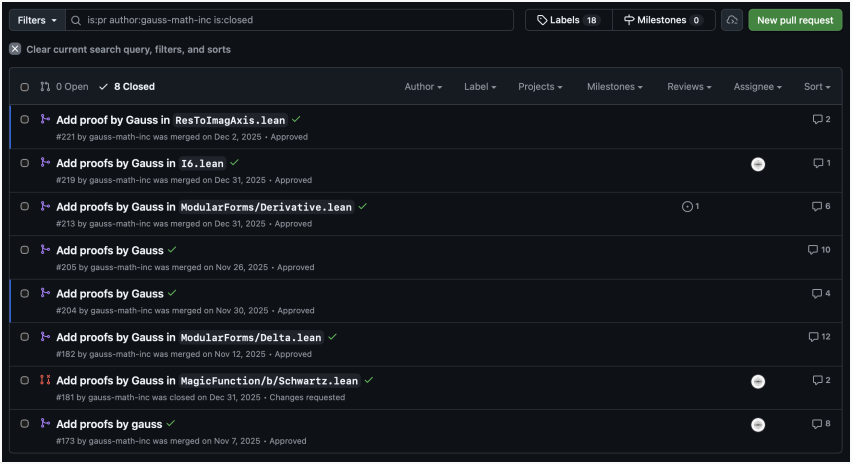
Math, Inc contacted us in late October 2025 and told us they'd filled in 30 `sorry` s from our repository, including a version of the tricky contour integral we needed.

The Old Version of Gauss

Math, Inc contacted us in late October 2025 and told us they'd filled in 30 `sorry` s from our repository, including a version of the tricky contour integral we needed. Upon our request, they PRed some—but **not all**—of their results.

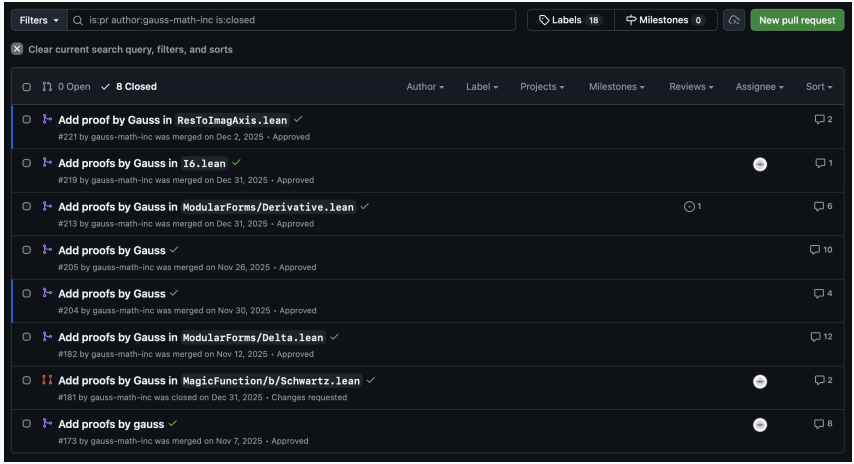
The Old Version of Gauss

Math, Inc contacted us in late October 2025 and told us they'd filled in 30 `sorry` s from our repository, including a version of the tricky contour integral we needed. Upon our request, they PRed some—but **not all**—of their results.



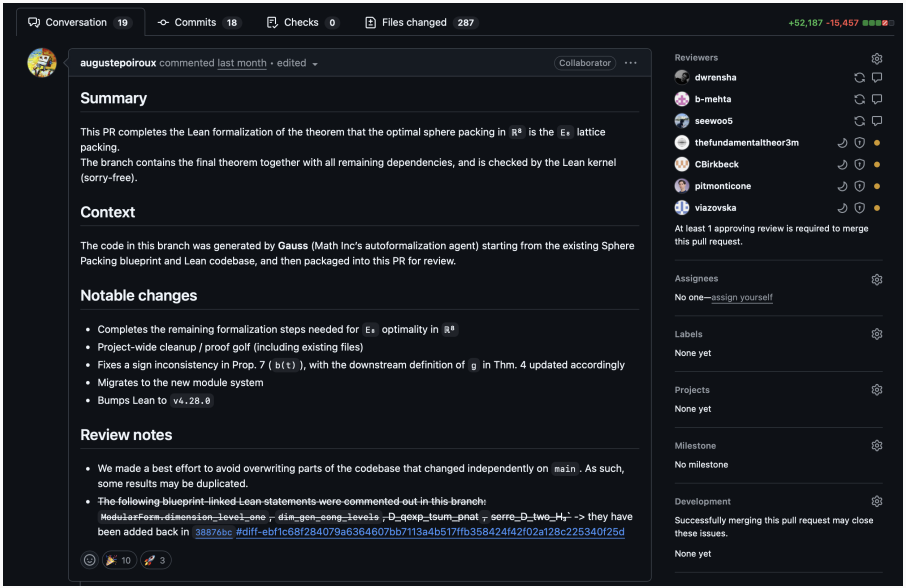
The Old Version of Gauss

Math, Inc contacted us in late October 2025 and told us they'd filled in 30 `sorry` s from our repository, including a version of the tricky contour integral we needed. Upon our request, they PRed some—but **not all**—of their results. After mid-November, the PRs stopped coming in.



The Big Autoformalisation

The Big Autoformalisation



Conversation 19 · Commits 18 · Checks 0 · Files changed 287 · +52,187 -15,457

augustepoiroux commented last month • edited — Collaborator

Summary

This PR completes the Lean formalization of the theorem that the optimal sphere packing in $\mathbb{R}^8$ is the E_8 lattice packing.
The branch contains the final theorem together with all remaining dependencies, and is checked by the Lean kernel (sorry-free).

Context

The code in this branch was generated by **Gauss** (Math Inc's autoformalization agent) starting from the existing Sphere Packing blueprint and Lean codebase, and then packaged into this PR for review.

Notable changes

- Completes the remaining formalization steps needed for E_8 optimality in $\mathbb{R}^8$
- Project-wide cleanup / proof golf (including existing files)
- Fixes a sign inconsistency in Prop. 7 (`b(t)`), with the downstream definition of `g` in Thm. 4 updated accordingly
- Migrates to the new module system
- Bumps Lean to `v4.28.0`

Review notes

- We made a best effort to avoid overwriting parts of the codebase that changed independently on `main` . As such, some results may be duplicated.
- ~~The following blueprint-linked Lean statements were commented out in this branch:~~
  ~~`ModularForm.dimension_level_one`~~ , ~~`dim_gen_cong_levels`~~ , ~~`D_qexp_tsum_pnat`~~ , ~~`serre_D_two_H`~~ -> they have been added back in `3887ebc` #diff-ebf1c68f284079a6364607bb7113a4b517ffb358424f42f02a128c225340f25d

🎉 10 · 🚀 3

Reviewers
- dwrensha
- b-mehta
- seewoo5
- thefundamentaltheor3m
- CBirkbeck
- pitmonticone
- viazovska

At least 1 approving review is required to merge this pull request.

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.
None yet

Some Comments on the Code

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)
- Gauss followed basic style requirements (no non-terminal `simp`s/finishing tactics, some documentation for files and key results), though more work needs to be done here (eg. deeply nested `have`s).

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)
- Gauss followed basic style requirements (no non-terminal `simp`s/finishing tactics, some documentation for files and key results), though more work needs to be done here (eg. deeply nested `have`s).
- Gauss made several custom definitions that aren't supported by appropriate API and whose benefit isn't completely clear (eg. `PermJ12ContourH1Hyp`).

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)
- Gauss followed basic style requirements (no non-terminal `simp`s/finishing tactics, some documentation for files and key results), though more work needs to be done here (eg. deeply nested `have`s).
- Gauss made several custom definitions that aren't supported by appropriate API and whose benefit isn't completely clear (eg. `PermJ12ContourH1Hyp`).
- The file organisation is not the best, and definitions (especially unusual ones) often don't appear where they make the most sense.

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)
- Gauss followed basic style requirements (no non-terminal `simp`s/finishing tactics, some documentation for files and key results), though more work needs to be done here (eg. deeply nested `have`s).
- Gauss made several custom definitions that aren't supported by appropriate API and whose benefit isn't completely clear (eg. `PermJ12ContourH1Hyp`).
- The file organisation is not the best, and definitions (especially unusual ones) often don't appear where they make the most sense.
- Some human-written code was also modified, and it is not yet completely clear that these modifications were good.

Some Comments on the Code

- The code was deemed legitimate by Lean4Checker and Comparator, and we do, indeed, believe to be correct. (No `native_decide`, weird metaprogramming, changing the definitions in the main theorem statement, changing the definition of `theorem`…)
- Gauss followed basic style requirements (no non-terminal `simp`s/finishing tactics, some documentation for files and key results), though more work needs to be done here (eg. deeply nested `have`s).
- Gauss made several custom definitions that aren't supported by appropriate API and whose benefit isn't completely clear (eg. `PermJ12ContourH1Hyp`).
- The file organisation is not the best, and definitions (especially unusual ones) often don't appear where they make the most sense.
- Some human-written code was also modified, and it is not yet completely clear that these modifications were good.
- **We are still reviewing, understanding, and refactoring the code. This will probably take a few months. In particular, none of the code has been merged yet.**

What *didn't* Gauss do?

What *didn't* Gauss do?

Finish the project!

Takeaways for Future Human-AI Collaboration

Why do we formalise mathematics?

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs.

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs. DeDeo and Duede write [3, pp.1-2]:

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs. DeDeo and Duede write [3, pp.1-2]:

> *[P]roofs plainly do many things: communicate mathematical ideas, transmit understanding, and unify or extend previously distinct bodies of knowledge. ... The issue ... is not whether ... a formal proof exists, but, rather, what it could mean for such a proof to correspond to a particular human proof-idea at all. That is,* ***successfully deriving a formal proof which corresponds not merely to a proof of some particular theorem, but, rather to some particular proof of that theorem is itself a kind of epistemic achievement****.*

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs. DeDeo and Duede write [3, pp.1-2]:

> *[P]roofs plainly do many things: communicate mathematical ideas, transmit understanding, and unify or extend previously distinct bodies of knowledge. … The issue … is not whether … a formal proof exists, but, rather, what it could mean for such a proof to correspond to a particular human proof-idea at all. That is,* **successfully deriving a formal proof which corresponds not merely to a proof of some particular theorem, but, rather to some particular proof of that theorem is itself a kind of epistemic achievement**.

My personal motivation for formalising Viazovska's solution was not to verify it, but to understand it: **it was really about the journey, not the destination**.

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs. DeDeo and Duede write [3, pp.1-2]:

> *[P]roofs plainly do many things: communicate mathematical ideas, transmit understanding, and unify or extend previously distinct bodies of knowledge. … The issue … is not whether … a formal proof exists, but, rather, what it could mean for such a proof to correspond to a particular human proof-idea at all. That is,* **successfully deriving a formal proof which corresponds not merely to a proof of some particular theorem, but, rather to some particular proof of that theorem is itself a kind of epistemic achievement**.

My personal motivation for formalising Viazovska's solution was not to verify it, but to understand it: **it was really about the journey, not the destination**. On recent events, Avigad writes [1, p.3]:

Why do we formalise mathematics?

Formalising complex proofs helps us understand them better, exposes hidden connections to other areas, and may even lead to the discovery of *new* proofs. DeDeo and Duede write [3, pp.1-2]:

> *[P]roofs plainly do many things: communicate mathematical ideas, transmit understanding, and unify or extend previously distinct bodies of knowledge. … The issue … is not whether … a formal proof exists, but, rather, what it could mean for such a proof to correspond to a particular human proof-idea at all. That is,* ***successfully deriving a formal proof which corresponds not merely to a proof of some particular theorem, but, rather to some particular proof of that theorem is itself a kind of epistemic achievement****.*

My personal motivation for formalising Viazovska's solution was not to verify it, but to understand it: **it was really about the journey, not the destination**. On recent events, Avigad writes [1, p.3]:

> *The formalization, on its own, is close to worthless, since* ***the correctness of Viazovska's result was never in doubt****. The participants embraced the project, rather, as a way of revisiting those results and* ***better understanding them****, and of* ***building libraries and infrastructure to support future work****. … [I]f AI can alleviate some of the tedium in formalization and help us enjoy and understand the mathematics, we will be better off, but* ***if it gets in the way of that enjoyment and understanding, we will have lost something important****.*

Finding Compatibility between Distinct Objectives

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem:

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**.

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics.

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

How can such incompatibilities be avoided?

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

How can such incompatibilities be avoided?

- I strongly believe parties wishing to collaborate should communicate clearly with each other to understand each other's objectives and find common ground.

Finding Compatibility between Distinct Objectives

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

How can such incompatibilities be avoided?

- I strongly believe parties wishing to collaborate should communicate clearly with each other to understand each other's objectives and find common ground.
- I also believe whoever is directing a human-led formalisation should also be able to direct AI tools being used to assist in the effort.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

How can such incompatibilities be avoided?

- I strongly believe parties wishing to collaborate should communicate clearly with each other to understand each other's objectives and find common ground.
- I also believe whoever is directing a human-led formalisation should also be able to direct AI tools being used to assist in the effort.

To be fair: we never explicitly said our objective was to really unpack and understand the proof.

Before we can find a solution, we must acknowledge the problem: **the sphere packers and Math Inc wanted different things**. We wanted to improve our understanding of a uniquely wonderful result in modern mathematics. Math Inc wanted to test the capabilities of a new model.

The two objectives need not be incompatible.

How are we realigning our objectives?

- We are actively collaborating with Math Inc to golf, understand, refactor and improve the Gauss code.
- We are also resolving conflicts between Gauss code an open PRs, prioritising the open PRs but encouraging the authors to incorporate anything from the Gauss code that's relevant.

How can such incompatibilities be avoided?

- I strongly believe parties wishing to collaborate should communicate clearly with each other to understand each other's objectives and find common ground.
- I also believe whoever is directing a human-led formalisation should also be able to direct AI tools being used to assist in the effort.

To be fair: we never explicitly said our objective was to really unpack and understand the proof. I think it just never really occurred to us that it needed to be said.

What are we doing now?

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`.

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.

What are we doing now?

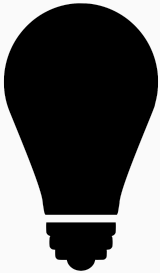
Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.

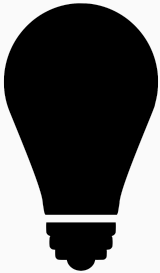
What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.



What are we doing now?

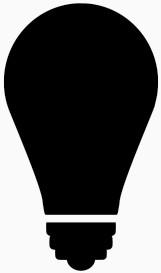
Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.



We are still trying to figure out how to approach this systematically.

What are we doing now?

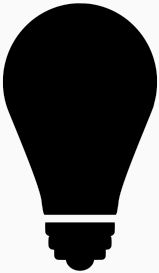
Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.



We are still trying to figure out how to approach this systematically. But we'll get there.

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.

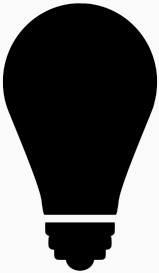


We are still trying to figure out how to approach this systematically. But we'll get there.

WE NEED YOU!

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.



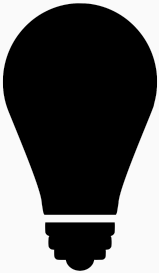
We are still trying to figure out how to approach this systematically. But we'll get there.

WE NEED YOU!

Come to packathons:

What are we doing now?

Current setup: Aayush-Auguste-David-maintained `gauss2` branch on the repo, rebased weekly(ish) on updates to `main`. Idea: chip off bits of code, work on them, and merge them to `gauss2` or `main`.



We are still trying to figure out how to approach this systematically. But we'll get there.

WE NEED YOU!

Come to packathons: Tuesdays 10:30 Eastern Time at `cmu.zoom.us/my/sidharth.hariharan`.

Acknowledgements

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song
- All our contributors! (Human and non-human alike)

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song
- All our contributors! (Human and non-human alike)
- Jeremy Avigad, Kevin Buzzard, Simon DeDeo, Leo de Moura, Emily Riehl, the Mathlib maintainers

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song
- All our contributors! (Human and non-human alike)
- Jeremy Avigad, Kevin Buzzard, Simon DeDeo, Leo de Moura, Emily Riehl, the Mathlib maintainers
- Hoskinson Center for Formal Mathematics

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song
- All our contributors! (Human and non-human alike)
- Jeremy Avigad, Kevin Buzzard, Simon DeDeo, Leo de Moura, Emily Riehl, the Mathlib maintainers
- Hoskinson Center for Formal Mathematics
- Institute for Computer-Aided Reasoning in Mathematics

Acknowledgements

- Chris Birkbeck, Seewoo Lee, Gareth Ma, Bhavik Mehta, Maryna Viazovska
- Matthew Cushman, Cameron Freer, Aayush Rajasekaran, David Renshaw, Tito Sacchi
- David Loeffler, Pietro Monticone, Utensil Song
- All our contributors! (Human and non-human alike)
- Jeremy Avigad, Kevin Buzzard, Simon DeDeo, Leo de Moura, Emily Riehl, the Mathlib maintainers
- Hoskinson Center for Formal Mathematics
- Institute for Computer-Aided Reasoning in Mathematics

Thank you very much! Merci beaucoup ! Merci vielmal! Grazie mille!

Questions?

References i

[1] J. Avigad.
Mathematicians in the age of AI.
arXiv preprint arXiv:2603.03684, 2026.

[2] H. Cohn and N. Elkies.
New Upper Bounds on Sphere Packings I.
Annals of Mathematics, 157(2):689–714, 2003.

[3] S. DeDeo and E. Duede.
A correspondence problem for mathematical proof.
arXiv preprint arXiv:2603.13680, 2026.

[4] S. Hariharan.
Viazovska's magic function in dimension 8: An attempt at formalisation.
MSci Thesis, Imperial College London, 2025.
Supervised by Bhavik Mehta.

References ii

[5] S. Lee.
Algebraic proof of modular form inequalities for optimal sphere packings.
arXiv preprint arXiv:2406.14659, 2024.

[6] M. S. Viazovska.
The sphere packing problem in dimension 8.
Annals of Mathematics, 185(3):991–1015, 2017.