Lean for PDEs – ICARM & SLMath

Rémy Degenne (Université de Lille, Inria, CNRS, Centrale Lille, CRIStAL)
Michael Rothgang (Rheinische Friedrich-Wilhelms-Universität Bonn)

Section 1

Lean and Mathlib

What is Lean?

Lean is

- a functional programming language
- a theorem prover

What is Lean?

Lean is

- a functional programming language
- a theorem prover

In Lean you can

- write programs/definitions
- state theorems
- write proofs
- write programs that write proofs $\rightarrow$ tactics

Program verification, formal mathematics

What is Lean?

Is it like Sage or Mathematica?

- No: those programs perform computations, you don't build proofs in them

Is it AI that proves theorems automatically?

- No, but such AI can be built on top of proof assistants

Is it a program that automatically proves theorems in first order logic?

- No, but it can use such programs in tactics

How does it work?

You want to prove a theorem in Lean.

- You import other definitions and theorems you need from a library
- You write a statement in the Lean language
- You write a proof, using tactics to help you
- The Lean engine checks that your proof is correct

Now your theorem is part of the library, and can be used in other proofs.

How does it work?

You want to prove a theorem in Lean.

- You import other definitions and theorems you need from a library
- You write a statement in the Lean language
- You write a proof, using tactics to help you
- The Lean engine checks that your proof is correct

Now your theorem is part of the library, and can be used in other proofs.

Some other interactive theorem provers:
Rocq, Isabelle/HOL, Agda, Metamath, Mizar...

Lean and Mathlib

Lean is an interactive theorem prover.

Mathlib is its mathematical library

- 230 000 theorems about 116 000 definitions (many automatically generated)
- 650 contributors
- 53 reviewers, among which 26 maintainers who accept contributions
- All kinds of math: a monolithic library, because any part of math could be useful in combination with any other, and different parts of the library should be compatible.

→ Let's look at the documentation

What has been formalised already: let's guess

- Banach–Schauder open mapping theorem
- Birkhoff Ergodic Theorem
- Mandelbrot set is connected
- Cauchy-Kovalevskaya Theorem on existence of an analytical solution of an analytical PDE.
- Denjoy's theorem: a C^2 orientation-preserving diffeomorphism of the circle with an irrational rotation number is conjugate to a rotation.
- Sphere eversion
- Existence of Haar measure
- Existence of a smooth partition of unity
- Feit–Thompson theorem/odd order theorem
- Fermat's Last Theorem
- Four colour theorem
- Galois correspondence
- Herman-Yoccoz theorem on linearization of a circle diffeomorphism
- Jordan curve theorem
- Liouville theorem: an entire holomorphic function is a constant
- Hilbert's Nullstellensatz
- Picard-Lindelöf theorem (existence and uniqueness of solutions of ODEs)
- Poincaré-Bendixson Theorem
- Poincaré recurrence theorem
- Sard's Theorem
- The continuum hypothesis is independent of ZFC.

Let's guess: the answer

Only 5 are not formalised yet (AFAIK)

- Cauchy-Kovalevskaya Theorem on existence of an analytic solution of an analytic PDE
- Denjoy's theorem on rotation number
- Herman–Yoccoz theorem on linearization of a circle diffeomorphism
- Fermat's Last Theorem
- Sard's Theorem

Let's guess: the answer

Only 5 are not formalised yet (AFAIK)

- Cauchy-Kovalevskaya Theorem on existence of an analytic solution of an analytic PDE
- Denjoy's theorem on rotation number
- Herman–Yoccoz theorem on linearization of a circle diffeomorphism
- Fermat's Last Theorem (in progress)
- Sard's Theorem (in progress)

Notable formalisation projects

2005 Four colour theorem

2012 Odd Order Theorem

Notable formalisation projects

2005 Four colour theorem

2012 Odd Order Theorem

2014 Kepler's conjecture (Hales et al)

2019 Ellenberg-Gijswijt's result on the cap set conjecture

2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics

Notable formalisation projects

- 2005 Four colour theorem
- 2012 Odd Order Theorem
- 2014 Kepler's conjecture (Hales et al)
- 2019 Ellenberg-Gijswijt's result on the cap set conjecture
- 2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics
- 2022 unit fractions project
- 2023 upper bound on diagonal Ramsey numbers

Notable formalisation projects

- 2005 Four colour theorem
- 2012 Odd Order Theorem
- 2014 Kepler's conjecture (Hales et al)
- 2019 Ellenberg-Gijswijt's result on the cap set conjecture
- 2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics
- 2022 unit fractions project
- 2023 upper bound on diagonal Ramsey numbers

Notable formalisation projects

- 2005 Four colour theorem
- 2012 Odd Order Theorem
- 2014 Kepler's conjecture (Hales et al)
- 2019 Ellenberg-Gijswijt's result on the cap set conjecture
- 2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics
- 2022 unit fractions project
- 2023 upper bound on diagonal Ramsey numbers
- 2023 polynomial Freiman-Rusza conjecture (Tao et al)

Notable formalisation projects

- 2005 Four colour theorem
- 2012 Odd Order Theorem
- 2014 Kepler's conjecture (Hales et al)
- 2019 Ellenberg-Gijswijt's result on the cap set conjecture
- 2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics
- 2022 unit fractions project before referee report
- 2023 upper bound on diagonal Ramsey numbers before referee report
- 2023 polynomial Freiman-Rusza conjecture (Tao et al)

- 2025 disproof of the Aharoni–Korman conjecture (Mehta)
- 2025 Carleson's theorem

Notable formalisation projects

- 2005 Four colour theorem
- 2012 Odd Order Theorem
- 2014 Kepler's conjecture (Hales et al)
- 2019 Ellenberg-Gijswijt's result on the cap set conjecture
- 2022 Liquid Tensor Experiment (Commelin et al): fundamental lemma about condensed mathematics
- 2022 unit fractions project before referee report
- 2023 upper bound on diagonal Ramsey numbers before referee report
- 2023 polynomial Freiman-Rusza conjecture (Tao et al) took 3 weeks; complete before paper submitted
- 2025 disproof of the Aharoni–Korman conjecture (Mehta)
- 2025 Carleson's theorem

Lean and Mathlib, and other projects

Lean is an interactive theorem prover.

Mathlib is its mathematical library

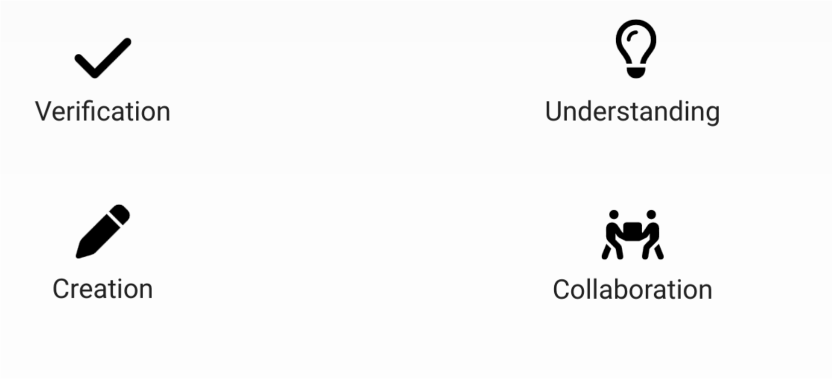
Many projects use Lean and Mathlib:

- Fermat's last theorem
- Brownian motion and stochastic integrals
- Toric varieties
- Carleson theorem
- Prime number theorem and more
- Equational theories of magmas
- Polynomial Freiman-Ruzsa conjecture
- ...

Section 2

Why formalize mathematics?

Why formalise?



Correctness/Verification

Lean checks the proof down to its axioms → certainty of correctness.

There are always folklore results, or literature gaps known only to experts.
Yet, the foundations of our various domains are solid. That's not where the biggest gain is.

New papers may contain mistakes,

- but we detect them when we try to reuse the results,
- requiring formal proofs would put a big burden on the authors
- but still, wouldn't it be great to review a paper with certified proofs?
    - You would still need to review the definitions!

Understanding and creation

- understanding: reader chooses amount of detail

  Demo by Patrick Massot and Kyle Miller:
  https://kmill.github.io/informalization/ContinuousFrom.html

Understanding and creation

- understanding: reader chooses amount of detail

  Demo by Patrick Massot and Kyle Miller:
  `https://kmill.github.io/informalization/ContinuousFrom.html`

- creation: can this lemma be generalised? unused assumptions?
- database of theorems: searching known and related results

  only requires *statements* of main results

Collaboration

Lean checks the proof correctness → enables large scale collaboration.

Example: PFR project

- PFR Conjecture attributed to Marton by Ruzsa, 1999.
- Proved by Gowers, Green, Manners and Tao, 13 November 2023 on arXiv.
- Dec 5: fully completed Lean proof.

- 25 contributors
- a website, a github repository, a *blueprint*, a dependency graph, a zulip channel
- a detailed LaTeX proof
- many Lean definitions and lemmas
- several contributions to Mathlib

Automation and AI

A typical measurability or continuity proof in Mathlib:
`by fun_prop`

- The computer can prove things for you.
- Today, Lean proofs are still more verbose than paper proofs.
- New tactics are being developed all the time.

AI / LLMs tools are coming. They hallucinate, but Lean can catch the mistakes!

But they won't help you if the library doesn't contain the right definitions and lemmas!

Section 3

PDEs

PDEs

$$F(D^k u(x), D^{k-1} u(x), \dots, u(x), x) = 0$$

Does Mathlib know about technique X to prove property Y of PDEs of type Z?

PDEs

$$F(D^k u(x), D^{k-1} u(x), \dots, u(x), x) = 0$$

Does Mathlib know about technique X to prove property Y of PDEs of type Z?

No.

PDEs

$$F(D^k u(x), D^{k-1} u(x), \dots, u(x), x) = 0$$

Does Mathlib know about technique X to prove property Y of PDEs of type Z?

No.

But it could, and we are possibly not far from it.

What we have

No PDE file in Mathlib, but...

- Differential calculus: derivatives, Hessian, Taylor series, C^n, etc.
- Probably all you need to state a PDE
- Integrals
- Sobolev inequality
- In pull requests, not yet in Mathlib: distributions

No Sobolev spaces (it's being worked on).

Probably easily doable: verify the fundamental solution of the Heat equation.

Stochastic PDEs

Bad news first: no stochastic integrals, no Brownian motion in Mathlib.

$$\int f \, dB$$

But: this is the current focus of the Brownian motion project.

Some probability in Mathlib

- Measures, Lebesgue integral, L^p spaces, ...
- Moments of random variables, covariance (in Banach spaces)
- Characteristic functions
- Stochastic processes and stopping times, martingales (discrete time)
- Doob's convergence theorems
- Definition of Gaussians in Banach spaces
- Almost there: Fernique's theorem, Gaussians have finite moments
- In a PR: Cameron-Martin theorem
- In the Brownian motion project: facts about multivariate Gaussians, Kolmogorov continuity theorem, Brownian motion
- In the CLT project: Prokhorov theorem, central limit theorem

A first glance

Compact.lean

Mathlib > Topology > UniformSpace > Compact.lean

```lean
    intro y hxy
    simp [comap_const_of_not_mem (compl_singleton_mem_nhds hxy) (Classical.not_not.2 rfl)]
#align uniform_space_of_compact_t2 uniformSpaceOfCompactT2

/-!
### Heine-Cantor theorem
-/


/-- Heine-Cantor: a continuous function on a compact uniform space is uniformly
continuous. -/
theorem CompactSpace.uniformContinuous_of_continuous [CompactSpace α] {f : α → β}
    (h : Continuous f) : UniformContinuous f :=
calc map (Prod.map f f) (𝓤 α)
   = map (Prod.map f f) (𝓝ˢ (diagonal α)) := by rw [nhdsSet_diagonal_eq_uniformity]
 _ ≤ 𝓝ˢ (diagonal β)                      := (h.prod_map h).tendsto_nhdsSet mapsTo_prod_map_diagonal
 _ ≤ 𝓤 β                                  := nhdsSet_diagonal_le_uniformity
#align compact_space.uniform_continuous_of_continuous CompactSpace.uniformContinuous_of_continuous

/-- Heine-Cantor: a continuous function on a compact set of a uniform space is uniformly
continuous. -/
theorem IsCompact.uniformContinuousOn_of_continuous {s : Set α} {f : α → β} (hs : IsCompact s)
    (hf : ContinuousOn f s) : UniformContinuousOn f s := by
  rw [uniformContinuousOn_iff_restrict]
  rw [isCompact_iff_compactSpace] at hs
  rw [continuousOn_iff_continuous_restrict] at hf
  exact CompactSpace.uniformContinuous_of_continuous hf
#align is_compact.uniform_continuous_on_of_continuous IsCompact.uniformContinuousOn_of_continuous
```

Lean Infoview

```
▼ Compact.lean:168:0
 ▼ Expected type
   α : Type u_1
   β : Type u_2
   γ : Type u_3
   inst✝² : UniformSpace α
   inst✝¹ : UniformSpace β
   inst✝ : CompactSpace α
   f : α → β
   h : Continuous f
   ⊢ UniformContinuous f
▶ All Messages (0)
```

What is formalisation like?

- fussy; steep learning curve
- it's fun — like a video game or programming
- improves your understanding

Let's try Lean together

Join the zulip channel: `leanprover.zulipchat.com/#narrow/channel/534314-Lean-for-PDEs-2025/topic/Links`

Start the tutorial: `github.com/RemyDegenne/GlimpseOfLean`