Formalizing sphere eversion using Lean's mathematical library

Floris van Doorn

University of Paris-Saclay in Orsay

21 June 2023

Talk Topics

In this talk:

1. Lean and mathlib
2. The sphere eversion project

Lean

Lean is an interactive theorem prover mainly developed by Leonardo de Moura at Microsoft Research.

It is open source, and under active development for 10 years.

Let's see it in action.

mathlib

mathlib is the mathematical library of Lean.

It is a general-purpose: it contains algebra, topology, analysis, differential geometry, probability theory, category theory, combinatorics, logic, ...

The organization is decentralized: contributors come with their own plans and goals.

It is large: mathlib has over 1 million lines of code, written by over 270 contributors.

It is active: There are more than 100 contributions every week and every contribution is reviewed by one of the 23 mathlib maintainers.

I have worked with Lean since 2014 and am a mathlib maintainer since 2019.

Selected Lean formalizations

- Spectral sequences (2017; van Doorn et al.)
- Definition of perfectoid spaces (2019; Buzzard, Commelin, Massot)
- Independence of the continuum hypothesis (2019; van Doorn, Han)
- Witt vectors (2020; Commelin, Lewis)
- Finiteness of the class group of a global field (2021; Baanen, Dahmen, Narayanan, Nuccio)
- Liquid tensor experiment (2022; Commelin et al.)
- Sphere eversion project (2022; van Doorn, Massot, Nash)

Lean 3 vs Lean 4

- 2018: The Lean core developers started working on Lean 4
- 2021: First milestone release of Lean 4
- 2022: Lean 4 has all essential features of Lean 3
- Oct 2022 - present: port mathlib to Lean 4
  - Partially automated, with many manual fixes
  - Various problems arose during the port
  - Now almost done: over 90% of mathlib has been ported

Why Lean 4?

- Turn Lean into a fully-fledged programming language, compiled into C
- Lean 4 is mostly written in Lean
- Lean 4 is faster.
- Flexible macro expansion

```
syntax "{ " ident (" : " term)? " // " term " }" : term
macro_rules
  | `({ $x : $type // $p }) => ``(Subtype (fun ($x:ident : $type) => $p))
#check { x : ℕ // x < 10 }
```

- Convenient do notation

```
def List.findM? (p : α → m Bool)
  (xs : List α) : m (Option α) := do
  for x in xs do
    let b ← p x
    if b then
      return some x
  return none
```

- More info in *Functional programming in Lean*

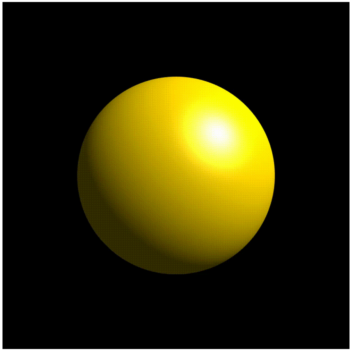
Talk Topics

In this talk:

1. Lean and mathlib
2. The sphere eversion project

Sphere eversion

We can turn a sphere inside out without tearing or creasing it, but allowing self-intersections.



To precisely state this,
we use the following definition:
f is an immersion $\iff$
f is locally an embedding $\iff$
the total derivative of f is injective.

Sphere eversion

Theorem (Smale, 1957)

There is a smooth transformation of immersions

$$\mathbb{S}^2 \to \mathbb{R}^3$$

starting with the inclusion map and ending with the antipodal map.

We don't give an explicit construction, instead we use a technique called convex integration.

Convex integration (1)

Suppose we want to turn $f : \mathbb{R}^2 \to \mathbb{R}^3$ into an immersion.

Convex integration (1)

Suppose we want to turn $f : \mathbb{R}^2 \to \mathbb{R}^3$ into an immersion.

We can do this by first ensuring that $\partial_1 f(x) := \frac{\partial f(x)}{\partial x_1} \neq 0$ and next that $\partial_2 f(x)$ is not collinear with $\partial_1 f(x)$

In both steps we want that $\partial_j f(x)$ lives in some open subset $\Omega_x \subseteq \mathbb{R}^3$.

Convex integration (1)

Suppose we want to turn $f : \mathbb{R}^2 \to \mathbb{R}^3$ into an immersion.

We can do this by first ensuring that $\partial_1 f(x) := \frac{\partial f(x)}{\partial x_1} \neq 0$ and next that $\partial_2 f(x)$ is not collinear with $\partial_1 f(x)$

In both steps we want that $\partial_j f(x)$ lives in some open subset $\Omega_x \subseteq \mathbb{R}^3$.

Suppose there exists a family of loops $\gamma : \mathbb{R}^2 \times \mathbb{S}^1 \to \mathbb{R}^3$ such that γ_x takes values in Ω_x and has average $\partial_j f(x)$.

Note: Such loops only exist if $\partial_j f(x)$ is in the convex hull of Ω_x.

Convex integration (2)

We want that $\partial_j f(x)$ lives in some open subset $\Omega_x \subseteq \mathbb{R}^3$.
We have a family of loops $\gamma : \mathbb{R}^2 \times \mathbb{S}^1 \to \mathbb{R}^3$ such that γ_x takes values in Ω_x and has average $\partial_j f(x)$.

Convex integration (2)

We want that $\partial_j f(x)$ lives in some open subset $\Omega_x \subseteq \mathbb{R}^3$.
We have a family of loops $\gamma : \mathbb{R}^2 \times \mathbb{S}^1 \to \mathbb{R}^3$ such that γ_x takes values in Ω_x and has average $\partial_j f(x)$.

Now let $N \gg 0$ and replace f by

$$g : x \mapsto f(x) + \frac{1}{N} \int_0^{Nx_j} [\gamma_x(s) - \partial_j f(x)] ds.$$

By a simple computation of partial derivatives, we see that

- $\partial_j g(x) \approx \gamma_x(Nx_j) \in \Omega_x$;
- $\partial_i g(x) \approx \partial_i f(x)$ for $i \neq j$;
- $g(x) \approx f(x)$.

The h-principle

We use this in much greater generality to formalize a seminal result in differential topology, Gromov's original h-principle (or homotopy principle).[1]

The homotopy principle provides a very general technique to construct solutions to partial differential relations.

Originally proven by Mikhael Gromov in 1973; we followed a proof by Mélanie Theillière from 2018.

[1] More precisely: the relative, parametric C^0-dense h-principle for open and ample first order differential relations on functions between smooth manifolds.

Formalization of the h-principle

We wanted to show that we can formalize deep geometric arguments in an interactive theorem prover.

We wrote a blueprint with a detailed LaTeX proof.

Formalization of the h-principle

We wanted to show that we can formalize deep geometric arguments in an interactive theorem prover.

We wrote a blueprint with a detailed LaTeX proof.

We had to formalize convex integration and jet spaces as part of the formalization.

We made 140 contributions to mathlib in the process, about convexity of sets, parametric integrals, differential geometry and various other topics.

This project took us about a year (part-time).

Final Result

```
theorem Smale :
  ∃ f : ℝ → 𝕊² → ℝ³,
    smooth (↿f : ℝ × 𝕊² → ℝ³) ∧
    f 0 = (coe : 𝕊² → ℝ³) ∧
    f 1 = (-coe : 𝕊² → ℝ³) ∧
    ∀ t, immersion (f t)
```

Thank You

The h-principle

Theorem (Gromov, 1973)

If $\mathcal{R}$ is an open and ample[2] *partial differential relation for functions between manifolds then $\mathcal{R}$ satisfies the h-principle, i.e. any formal solution can be smoothly deformed into a holonomic one inside $\mathcal{R}$.*

[2]Ampleness is a geometric condition that ensures that certain convex hulls are large enough for the convex integration argument to work.