# BerLean (2)

## An introduction to Lean

Yves Jäckle

TU Berlin

July 2025

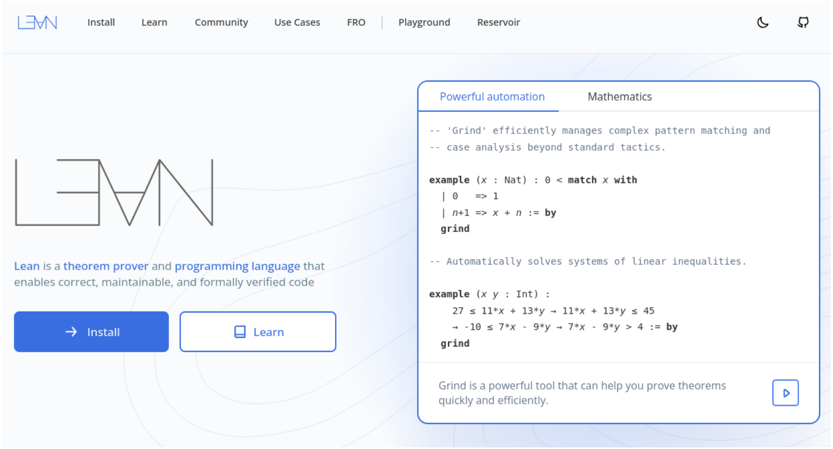
## 1st Talk

## Programs and proofs

Let's see it in action !

## How to get started

- Installation and docs: lean-lang.org



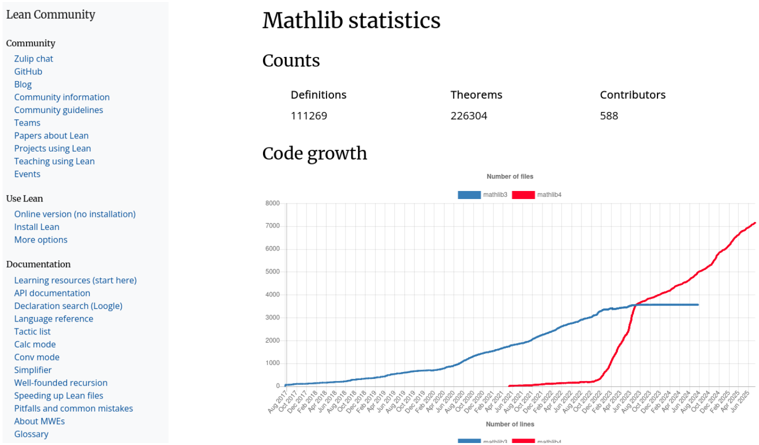
Lean is a theorem prover and programming language that enables correct, maintainable, and formally verified code

```
-- 'Grind' efficiently manages complex pattern matching and
-- case analysis beyond standard tactics.

example (x : Nat) : 0 < match x with
  | 0   => 1
  | n+1 => x + n := by
  grind

-- Automatically solves systems of linear inequalities.

example (x y : Int) :
    27 ≤ 11*x + 13*y → 11*x + 13*y ≤ 45
    → -10 ≤ 7*x - 9*y → 7*x - 9*y > 4 := by
  grind
```

Grind is a powerful tool that can help you prove theorems quickly and efficiently.

## How to get started

- More docs and guidelines: leanprover-community.github.io



Lean Community

Community

- Zulip chat
- GitHub
- Blog
- Community information
- Community guidelines
- Teams
- Papers about Lean
- Projects using Lean
- Teaching using Lean
- Events

Use Lean

- Online version (no installation)
- Install Lean
- More options

Documentation

- Learning resources (start here)
- API documentation
- Declaration search (Loogle)
- Language reference
- Tactic list
- Calc mode
- Conv mode
- Simplifier
- Well-founded recursion
- Speeding up Lean files
- Pitfalls and common mistakes
- About MWEs
- Glossary

Mathlib statistics

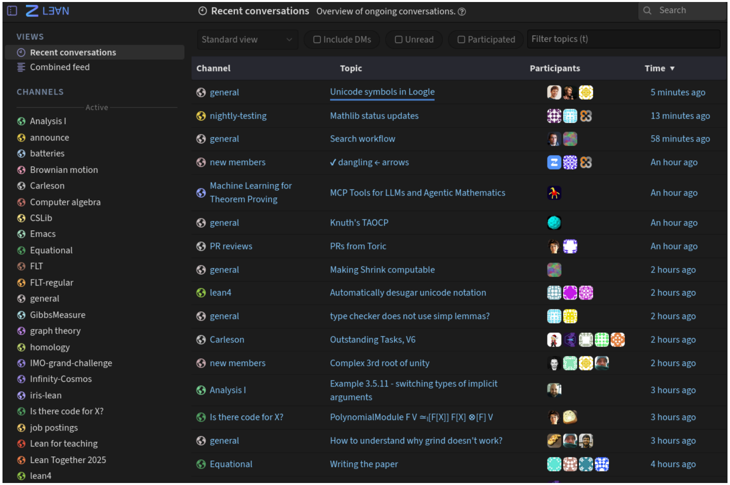
Counts

| Definitions | Theorems | Contributors |
|---|---|---|
| 111269 | 226304 | 588 |

Code growth

## How to get started

- Questions and discussions: leanprover.zulipchat.com


L∃∀N

Recent conversations Overview of ongoing conversations.

Search

VIEWS

- Recent conversations
- Combined feed

CHANNELS

Active

- Analysis I
- announce
- batteries
- Brownian motion
- Carleson
- Computer algebra
- CSLib
- Emacs
- Equational
- FLT
- FLT-regular
- general
- GibbsMeasure
- graph theory
- homology
- IMO-grand-challenge
- Infinity-Cosmos
- iris-lean
- Is there code for X?
- job postings
- Lean for teaching
- Lean Together 2025
- lean4

Standard view | ☐ Include DMs | ☐ Unread | ☐ Participated | Filter topics (t)

| Channel | Topic | Participants | Time ▼ |
|---|---|---|---|
| general | Unicode symbols in Loogle | | 5 minutes ago |
| nightly-testing | Mathlib status updates | | 13 minutes ago |
| general | Search workflow | | 58 minutes ago |
| new members | ✓ dangling ← arrows | | An hour ago |
| Machine Learning for Theorem Proving | MCP Tools for LLMs and Agentic Mathematics | | An hour ago |
| general | Knuth's TAOCP | | An hour ago |
| PR reviews | PRs from Toric | | An hour ago |
| general | Making Shrink computable | | 2 hours ago |
| lean4 | Automatically desugar unicode notation | | 2 hours ago |
| general | type checker does not use simp lemmas? | | 2 hours ago |
| Carleson | Outstanding Tasks, V6 | | 2 hours ago |
| new members | Complex 3rd root of unity | | 2 hours ago |
| Analysis I | Example 3.5.11 - switching types of implicit arguments | | 3 hours ago |
| Is there code for X? | PolynomialModule F V $\simeq_l$[F[X]] F[X] ⊗[F] V | | 3 hours ago |
| general | How to understand why grind doesn't work? | | 3 hours ago |
| Equational | Writing the paper | | 4 hours ago |

Thank you for your attention.

Do you have any questions ?

## 2nd Talk

## Subgradient descent

## First, informally

adapted from chapter 3.2 of
"Convex Optimization Algorithms"
(2015) by Dimitri P. Bertsekas

## What is it about ?

The task :

- Objective function $f : \mathbb{R} \to \mathbb{R}$ (or $f \in \mathbb{R}^{\mathbb{R}}$)
- Minimize it !
  Find $m$ so that $\forall x,\ f(x) \geqslant f(m)$ (if it exists!)

## What is it about ?

The task :

- Objective function $f : \mathbb{R} \to \mathbb{R}$ (or $f \in \mathbb{R}^{\mathbb{R}}$)
- Minimize it !
  Find $m$ so that $\forall x,\ f(x) \geqslant f(m)$ (if it exists!)

The idea

- Start at some $x_0$
- Compute sequence $x_{n+1} = x_n + Magic(x_n, f)$
- Hope that $(x_n)_{n\in\mathbb{N}}$ converges, and to $m$

**Subgradient**

A *subgradient* for $f$ at $x$ is a $g_{x,f}$ such that:

$$\forall y, f(y) - f(x) \geqslant g_{x,f}(y - x)$$

**Subgradient descent**

For a sequence $(s_n)_{n\in\mathbb{N}}$ and a way to compute subgradients a $g_{x,f}$, we seek the minimum via:

$$x_{n+1} \ = \ x_n \ - \ s_n g_{x_n,f}$$

**Lemma**

If $y$ is such that $f(y) \leqslant f(x_n)$,

$$(x_{n+1} - y)^2 \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$$

**Lemma**

If $y$ is such that $f(y) \leqslant f(x_n)$,

$$(x_{n+1} - y)^2 \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$$

**Proof:**

$(x_{n+1} - y)^2 = (x_n - s_n g_{x_n,f} - y)^2$

**Subgradient descent**

For a sequence $(s_n)_{n\in\mathbb{N}}$ and a way to compute subgradients a $g_{x,f}$, we seek the minimum via:

$$x_{n+1} = x_n - s_n g_{x_n,f}$$

**Lemma**

If $y$ is such that $f(y) \leqslant f(x_n)$,

$$(x_{n+1} - y)^2 \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$$

**Proof:**

$(x_{n+1} - y)^2 = (x_n - s_n g_{x_n,f} - y)^2$

$\ldots = (x_n - y)^2 - 2s_n g_{x_n,f}(x_n - y) + s_n^2 g_{x_n,f}^2$

**Lemma**

If $y$ is such that $f(y) \leqslant f(x_n)$,

$$(x_{n+1} - y)^2 \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$$

**Proof:**

$\ldots = (x_n - y)^2 - 2s_n g_{x_n,f}(x_n - y) + s_n^2 g_{x_n,f}^2$

$\ldots \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$

**Subgradient**

A *subgradient* for $f$ at $x$ is a $g_{x,f}$ such that:

$$\forall y, f(y) - f(x) \geqslant g_{x,f}(y - x)$$

**Lemma**
If $y$ is such that $f(y) \leqslant f(x_n)$,

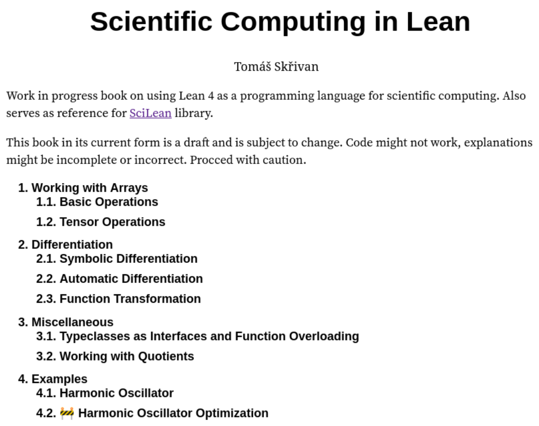
$$(x_{n+1} - y)^2 \leqslant (x_n - y)^2 - 2s_n(f(x_n) - f(y)) + s_n^2 g_{x_n,f}^2$$

**Theorem**
If $y$ is such that $f(y) \leqslant f(x_n)$ and $0 < s_n < \dfrac{2(f(x_n) - f(y)}{g_{x_n,f}^2}$,
then $(x_{n+1} - y)^2 < (x_n - y)^2$

## Next, formally

## IRL

- SciLean



# Scientific Computing in Lean

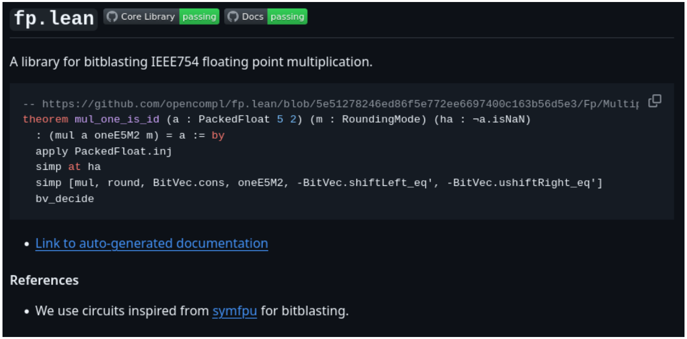
Tomáš Skřivan

Work in progress book on using Lean 4 as a programming language for scientific computing. Also serves as reference for SciLean library.

This book in its current form is a draft and is subject to change. Code might not work, explanations might be incomplete or incorrect. Procced with caution.

## IRL

- FPLean



fp.lean Core Library passing Docs passing

A library for bitblasting IEEE754 floating point multiplication.

```
-- https://github.com/opencompl/fp.lean/blob/5e51278246ed86f5e772ee6697400c163b56d5e3/Fp/Multip
theorem mul_one_is_id (a : PackedFloat 5 2) (m : RoundingMode) (ha : ¬a.isNaN)
  : (mul a oneE5M2 m) = a := by
  apply PackedFloat.inj
  simp at ha
  simp [mul, round, BitVec.cons, oneE5M2, -BitVec.shiftLeft_eq', -BitVec.ushiftRight_eq']
  bv_decide
```

- Link to auto-generated documentation

**References**

- We use circuits inspired from symfpu for bitblasting.

Thank you for your attention.

Do you have any questions ?