The Carleson Project: formalization and collaboration

Michael B. Rothgang (he/him)

Formalised mathematics group
Universität Bonn

UNIVERSITÄT BONN

ItaLean workshop
December 12, 2025

Outline

1. Lean and Mathlib
2. Carleson's Theorem
3. The Carleson Formalization Project
4. Design decisions

Lean's mathematical library

Mathlib is Lean's mathematical library.

Lean's mathematical library

Mathlib is Lean's mathematical library.

It is monolithic, containing algebra, analysis, geometry, number theory, probability theory, combinatorics, logic, topology, category theory, . . .

It is large: Mathlib has almost 2 million lines of code, with thousands of definitions and theorems written by over 600 contributors.

It is actively developed: There are more than 150 contributions every week, reviewed by the maintainers and reviewers.

I have worked with Lean since 2023, am a reviewer of `Mathlib` since May 2024 and a maintainer since November 2025.

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Some exciting projects in Lean

- Independence of the Continuum Hypothesis (van Doorn, Han; 2019)
- Perfectoid Spaces (Buzzard, Commelin, Massot; 2020)
- Liquid Tensor Experiment (led by Commelin and Topaz; 2022)
- Polynomial Freiman–Rusza conjecture (led by Terrence Tao; 2023)
- Sphere Eversion (van Doorn, Massot, Nash; 2023)
- Brownian motion (led by Degenne; ongoing)
- Prime Number Theorem+ (led by Kontorovich and Tao; 2025)
- Equational theories (led by Tao, 28 authors; 2025)
- Fermat's Last Theorem project (led by Buzzard; Ongoing)
- Carleson's Theorem (led by van Doorn; 2025)
- Formal Conjectures (Google Deepmind; Ongoing)

Outline

1. Lean and Mathlib
2. Carleson's Theorem
3. The Carleson Formalization Project
4. Design decisions

Partial Fourier Sums

Definition

If $f : \mathbb{R} \to \mathbb{C}$ is a 2π-periodic Borel measurable function, its Fourier coefficients $\hat{f} : \mathbb{Z} \to \mathbb{C}$ are defined by

$$\hat{f}(n) := \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-inx} dx.$$

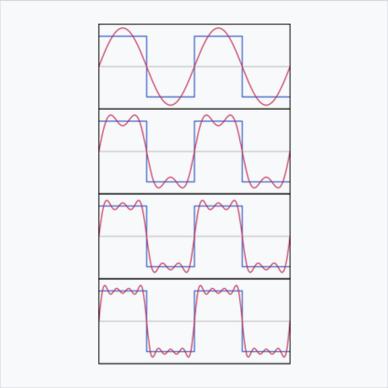
For $N \in \mathbb{N}$, define the N^{th} partial Fourier sum as

$$S_N f(x) := \sum_{-N}^{N} \hat{f}(n) e^{inx}.$$

When f is smooth, $S_N f$ converges uniformly to f.

Partial Fourier Sums: Intuition

- We think of the Fourier series as decomposing a function into the basis elements $x \mapsto e^{inx}$, like splitting a vector into basis components.
- These basis functions are also eigenfunctions of the differentiation operator. In particular we have $\hat{f'}(n) = in\hat{f}(n)$.



Convergence in Norm

What if f is not smooth, but $f \in L^p(\mathbb{R}/2\pi\mathbb{Z})$ for some $1 \le p \le \infty$?

- Recall: this means $\|f\|_{L^p} := \left(\int_0^{2\pi} |f(x)|^p dx\right)^{1/p} < \infty$.

Then $\hat{f}(n)$, $S_N f$ are still well-defined, and we can ask about convergence.

If $1 < p < \infty$, then $S_N f$ converges in $L^p(\mathbb{R}/2\pi\mathbb{Z})$-norm to f (easy).

- That is, $\lim_{N \to \infty} \|S_N f - f\|_{L^p} = 0$.

If $p = 1$ or $p = \infty$, there are counterexamples.

Carleson's Theorem

Theorem (Carleson–Hunt)

Let $f : \mathbb{R} \to \mathbb{C}$ be an $L^p(\mathbb{R}/2\pi\mathbb{Z})$, 2π-periodic function for $1 < p \leq \infty$. Then, for almost every $x \in \mathbb{R}$,

$$\lim_{N \to \infty} S_N f(x) = f(x),$$

where $S_N f$ is the N^{th} partial Fourier sum of f.

Carleson proved the case $p = 2$ in 1966.

Hunt proved the generalization in 1967.

All known proofs of this theorem are hard.

For $p = 1$ the statement fails badly.

Outline

1. Lean and Mathlib
2. Carleson's Theorem
3. The Carleson Formalization Project
4. Design decisions

Generalized Carleson

- In 2023, Lars Becker, Asgar Jamneshan, Rajula Srivastava and Christoph Thiele proved a generalization of Carleson's theorem.
- The generalized Carleson's theorem holds when the domain of the function is an arbitrary doubling metric measure space.
- The proof was 30 pages long.
- Floris van Doorn joined in late 2023 to set up the formalization.
- The formalization was finished in July 2025.

Blueprint

The harmonic analysis group wrote a blueprint for the proof:

- 120 pages.
- 30 more pages to prove classical Carleson's theorem as a corollary.
- Non-experts can take a single lemma and formalize it.

The blueprint has 11 sections.

- Section 1: statement of the generalized (metric) Carleson's theorem;
- Section 2: statement of 6 propositions used in the proof;
- Section 3: proof of metric Carleson from the propositions;
- Sections 4-9: each section proves one of the 6 propositions;
- Sections 10-11: proof of the classical Carleson theorem.

Blueprint

Carleson operators on doubling metric measure spaces

Theorem 1.0.1. (classical Carleson)✓# L∃∀N

Let f be a 2π-periodic complex-valued continuous function on $\mathbb{R}$. Then for almost all $x \in \mathbb{R}$ we have

$$\lim_{N\to\infty} S_N f(x) = f(x), \tag{1.0.3}$$

where $S_N f$ is the N-th partial Fourier sum of f defined in (1.0.2).

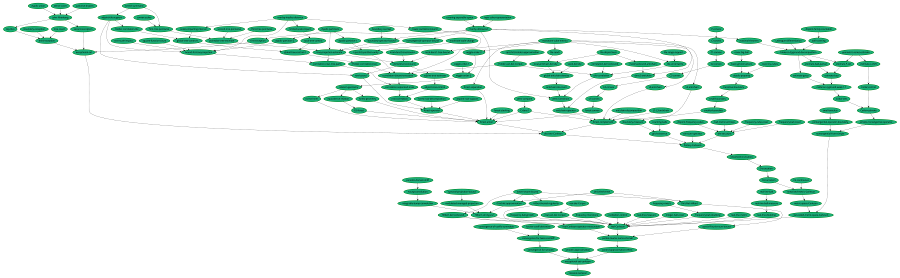
The purpose of e one hand, it prepares computer verification of
Theorem 1.0.1 b l proof as a blueprint for coding in Lean. We pass
through a boun so-called Carleson operator to doubling metric
measure spaces and proving these bounds constitutes the
second purpose of this paper. This generalization incorporates several results from the

Lean declarations ✖

classical_carleson

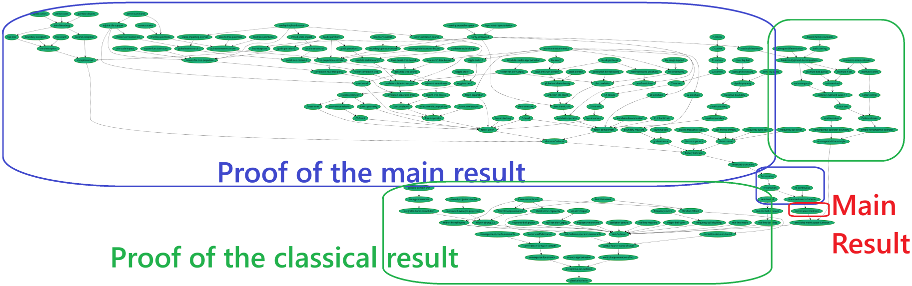
Built using P. Massot's blueprint infrastructure.

Dependency Graph



179 lemmas in total.

Dependency Graph



179 lemmas in total.

Contributors

- In June 2024, Floris van Doorn publicly launched the formalization, and asked for volunteers to contribute.
- van Doorn set up definitions common to the whole project.
- Statements are formalized by Lean experts (most often van Doorn).
- Contributors formalize the proofs, following the blueprint.
- Most contributors do not have a background in harmonic analysis.
- Contributors: María Inés de Frutos-Fernández, Leo Diedering, Floris van Doorn, Sébastien Gouëzel, Evgenia Karunus, Edward van de Meent, Pietro Monticone, Jasper Mulder-Sohn, Jim Portegies, Joris Roos, MR, James Sundstrom, Jeremy Tan, and others.

Coordination over Zulip

Carleson > Outstanding Tasks, V5 JAN 27

Tasks (including uncompleted tasks from last post):

L6. ▶️ (**María Inés de Frutos Fernández**) Proposition 2.0.3, proven in chapter 6.

55 ✅ (**Michael Rothgang** and **Sébastien Gouëzel**) Prove Lemma 8.0.1 (long)
56 ✅ (**Michael Rothgang** and **Sébastien Gouëzel**) Prove Proposition 2.0.5 (quite long)
73 ✅ (**James Sundstrom**) Prove Lemma 7.2.2
74 ✅ (**Jeremy Tan**) Prove Lemma 7.2.3
76 ✅ (**James Sundstrom**) Prove Lemma 7.3.1
77 ✅ (**Jeremy Tan**) Prove Lemma 7.3.2
78 ✅ (**James Sundstrom**) Prove Lemma 7.3.3
82 🆕 Prove Lemma 7.4.4
83 🆕 Prove Lemma 7.4.5
84 🆕 Prove Lemma 7.4.6
87 ✅ (**Jeremy Tan**) Prove Lemma 7.5.2
89 🆕 Prove Lemma 7.5.4
90 ✅ (**Jeremy Tan**) Prove Lemma 7.5.5
92 ✅ (**Jeremy Tan**) Prove Lemma 7.5.7
94 ✅ (**Jeremy Tan**) Prove Lemma 7.5.9
95 ▶️ (**Evgenia Karunus**) Prove Lemma 7.5.10

Collaboration with Lean

- Individual contributors work on their own part of the proof.
- Statements are formalized by experts to ensure correct translation.
- No need to trust proof authors, since Lean checks the proofs.
- Lean guarantees consistency between project parts.
- Safe refactoring: when a definition or theorem is reformulated, Lean will inform you of all the places that have to be adapted.

Carleson and Mathlib

- The Carleson Project extensively uses Mathlib: integration, metric spaces, measure theory, topology, ...
- Preliminary results are proved in high generality, so that they can be upstreamed to Mathlib.
  - Examples are the Marcinkiewicz Interpolation Theorem and the Hardy–Littlewood Maximal Principle.
- Project-specific results aren't necessarily done in the proper generality, and their proofs do not follow Mathlib standards.

Errors in the Blueprint

- Most errors are very minor (e.g., use $<$ instead of $\leq$, wrong constant) and can be fixed by the formalizer.
- For less trivial issues:
  - Bonn contributors can directly contact the harmonic analysis group.
  - All contributors can ask on Zulip.
  - Lars Becker answered questions, and made small fixes to the blueprint.
- No significant mistakes were found.

Outline

1. Lean and Mathlib
2. Carleson's Theorem
3. The Carleson Formalization Project
4. Design decisions

Constants

Analysis statements often take the form $\exists C \geq 0, \forall x, f(x) \leq C \cdot g(x)$.

The formalization works better with explicit constants.

In the Carleson project, constants are defined in terms of $D := 2^{100a^2}$.

Without effort, the formalization shows $D := 2^{7a^2}$ suffices.

Real numbers (I)

In analysis/measure theory, three types are very important:

- $\mathbb{R}$ (`Real`): the real numbers;
- $\mathbb{R}{\geq}0$ (`NNReal`): `{x : ℝ // 0 ≤ x}`;
- $\mathbb{R}{\geq}0\infty$ (`ENNReal`): `WithTop ℝ≥0`.

Often you need to use the canonical maps between these types.

It is a major pain to reason about these casts/cancel them in proofs.

Real numbers (II)

Proposal: in the Carleson project, just use $\mathbb{R}$ everywhere.

- All measures, integrals and suprema we work with should be finite.

Real numbers (II)

Proposal: in the Carleson project, just use $\mathbb{R}$ everywhere.

- All measures, integrals and suprema we work with should be finite.

This made the problem worse, so we switched to $\mathbb{R}_{\geq 0}\infty$.

- Mathlib likes operations in $\mathbb{R}_{\geq 0}\infty$.
- Even when a supremum (integral/measure) is provably finite, it is often still easier to work with the version that lands in $\mathbb{R}_{\geq 0}\infty$.
- Downside: some algebraic operations require finiteness hypotheses.

L^p-spaces

L^p-spaces: (certain) integrable functions quotiented by a.e. equality.

Basically everything we do respects a.e. equality.

However, it is painful to work with these quotients in Lean, e.g.

$$(f+g)(x) = f(x) + g(x)$$

will only hold for almost every x.

Much nicer: work with actual functions, not with quotients.

ENorm (I)

Let $f : \mathbb{R}^d \to \mathbb{C}$, the Hardy–Littlewood maximal function of f is

$$\begin{aligned} Mf : \mathbb{R} &\to \mathbb{R}{\geq}0\infty \\ x &\mapsto \sup_{r>0} \frac{1}{|B(x,r)|} \int_{B(x,r)} |f(y)|\, dy. \end{aligned}$$

If f is integrable, then Mf is (weak) L^1, and hence a.e. finite.

ENorm (I)

Let $f : \mathbb{R}^d \to \mathbb{C}$, the Hardy–Littlewood maximal function of f is

$$Mf : \mathbb{R} \to \mathbb{R}{\geq}0\infty$$
$$x \mapsto \sup_{r>0} \frac{1}{|B(x,r)|} \int_{B(x,r)} |f(y)|\, dy.$$

If f is integrable, then Mf is (weak) L^1, and hence a.e. finite.

Before: $f \in L^p$ was only defined in Lean for $f : X \to E$ where E was a normed vector space.

Hence, we could only *state* in Lean that $x \mapsto$ `(Mf(x)).toReal` is L^1.

So we could not conclude from this that Mf is a.e. finite.

ENorm (II)

Solution: introduce a notation class

```
class ENorm (E : Type*) where
  enorm : E → ℝ≥0∞
```

Normed spaces and ℝ≥0∞ are both instances of this class.

Definitions like `Integrable` and `MemLp` and many lemmas about them can be generalized to functions where the codomain has a `ENorm` class.

ENorm (II)

Solution: introduce a notation class

```
class ENorm (E : Type*) where
  enorm : E → ℝ≥0∞
```

Normed spaces and $\mathbb{R}_{\geq 0}\infty$ are both instances of this class.

Definitions like `Integrable` and `MemLp` and many lemmas about them can be generalized to functions where the codomain has a `ENorm` class.

To generalize results, we need more structure, e.g., `ENormedSpace`.

Conclusion

(Recent) results from many areas — such as harmonic analysis — are being formalized in Lean.

Formalization can help can be efficiently divided into small parts.

With a detailed blueprint, many people can efficiently contribute.

Outlook: formalising harmonic analysis

Will mathematical research results be verified by computers in the future?

Christoph Thiele and Floris van Doorn have been awarded an ERC Synergy Grant of 6.4 million euros

Will it be possible in future to prepare proofs developed in cutting-edge mathematical research with a reasonable amount of human effort so that they can be verified by computers in real time? Christoph Thiele and Floris van Doorn from the Hausdorff Center for Mathematics (HCM), a Cluster of Excellence at the University of Bonn, want to help make this possible. The two researchers submitted a joint application for a coveted Synergy Grant from the European Research Council (ERC). Following the award of the grant, the European Union will now provide total funding of 6.4 million euros to the "Harmonic Analysis with Lean Formalization" (HALF) project over the next six years. Lean is a relatively new programming language that is increasingly establishing itself as the standard for mathematical formalization.


Conclusion

(Recent) results from many areas — such as harmonic analysis — are being formalized in Lean.

Formalization can help can be efficiently divided into small parts.

With a detailed blueprint, many people can efficiently contribute.

Thanks for listening! Any questions?