FORMALIZING CARLESON'S THEOREM IN LEAN

AUTHORS

Abstract. We present the formalization of Carleson's theorem in the proof assistant *Lean*. This paper describes the mathematical content, organization of the project, the blueprint, and the main design decisions behind the formalization. It is the result of a large collaborative effort, written and developed in public.

Contents

1. Introduction

1.1. Brief history of Carleson's theorem and significance.

1.2. Overview of the formalization project.

Date: November 26, 2025.

1.3. **Related work.**

- Other large-scale formalizations in Lean.
- Other harmonic analysis formalizations.

Acknowledgement. The authors acknowledge contributions in the form of small formalization additions, pointing out corrections to the blueprint, or supplying ideas to the Lean efforts by the following people: Michel Alexis, Bolton Bailey, Julian Berman, Joachim Breitner, Martin Dvořák, Georges Gonthier, Aaron Hill, Austin Letson, Bhavik Mehta, Eric Paul, Clara Torres, Dennis Tsar, Andrew Yang, and Ruben van de Velde.

L.B., M.I.d.F.F., L.D., F.v.D., M.R., R.S., and C.T. were funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2047/1 – 390685813. L.B., R.S., and C.T. were also supported by SFB 1060. A.J. is funded by the TÜBITAK (Scientific and Technological Research Council of Türkiye) under Grant Number 123F122. J.R. was supported in part by NSF grant DMS-2154835 and a HIM fellowship for the Fall 2024 trimester program in Bonn.

2. The Metric Space Carleson theorem

2.1. **Mathematical background.**

- Prerequisites: Fourier series, Real interpolation, Hardy–Littlewood maximal function, doubling measures (`IsDoubling`), `WeakType`, `wnorm`.

2.2. **Statement of the main results.**

- Statement of Theorem 1.0.2.
- Main propositions in Chapter 2.

3. Project organization

- Project structure and division of tasks.
- Floris responsible for lemma statements (pros and cons).
- Interaction with harmonic analysis group.
- Contributors and roles.
- ToMathlib directory and contribution standards.
- Communication channels: Zulip, GitHub, Blueprint.

4. Working with a blueprint

4.1. **The blueprint writing process.**

- Blueprint writing process and finitary arguments.

4.2. **Changes and refinements.**

- $I \leq J$ vs. $I \subset J$.
- Real interpolation theorem.
- Hardy–Littlewood maximal function for finite vs. countable balls.
- Just one top cube.
- Constant tweaking.

4.3. **Dealing with mistakes.**
- Lemma 11.1.6.
- Lemma 6.3.3/4.
- Hölder cancellation (radius $R \to 2R$).
- Lemma 6.2.3: additional hypothesis.

4.4. **Lessons learned.**
- Impact of blueprint choices on errors.
- Lack of definition environments ⇒ absence from dependency graph.

5. Design decisions

5.1. **Treatment of constants.**

5.2. **The `ProofData` pattern.**

5.3. **Working with real numbers.**
- `Real` vs. `NNReal` vs. `ENNReal`:
  - Tactic support issues (`norm_num`, `field_simp`, `ring`).

5.4. **Use of `ENorm`.**

5.5. **Working with L^p functions.**
- Working with `MemLp`, not functions on `Lp`.

5.6. **The `BoundedCompactSupport` structure.**
- `BoundedCompactSupport` and packaging conditions.
- Ongoing experiments with `fun_prop`. TODO: consider whether this should be a separate subsection.

5.7. **Common pitfalls.**
- Using `Real`.
- `Set.indicator` vs. `Measure.restrict`.
- `Finsets` vs. `Sets` in a `Fintype`.

6. Conclusion

- Project statistics (e.g. size of ToMathlib and total project).
- Summary of lessons learned:
  - Refer to general results early on.
  - Generalize during blueprint writing.

References