Math 197: Senior Thesis

Formalizing the Gelfand-Naimark-Segal Construction in Lean

Gregory Wickham

Introduction

I formalized the Gelfand-Naimark-Segal (GNS) construction using Lean, a programming language that can verify the correctness of mathematical proofs.

The GNS construction is a method of constructing a *-homomorphism from a C^*-algebra into the algebra of bounded operators on an appropriately constructed Hilbert space. The GNS construction is an essential step in the proof the Gelfand-Naimark theorem, which is a foundational theorem in the theory of C^*-algebras [1].

This formalization has been merged into Mathlib, the canonical, open-source library of mathematics in Lean [2].

Definitions

A **C*-algebra** is an algebra A over $\mathbb{C}$ that is closed with respect to its norm $\|\cdot\|_A$, and has a map $^* : A \to A$ called the **adjoint** such that for all $c \in \mathbb{C}$, and $a, b \in A$, the map *

- is conjugate linear: $(ca + b)^* = \overline{c}a^* + b^*$,
- is anti-multiplicative: $(ab)^* = b^* a^*$,
- is involutive: $(a^*)^* = a$, and
- satisfies the C^*-identity: $\|a\|_A^2 = \|a^* a\|_A$.

A **positive linear functional** is a continuous linear map $f : A \to \mathbb{C}$ such that $0 \le f(a^*a)$ for all $a \in A$.

A ***-homomorphism** is a *-preserving algebra homomorphism (linear, multiplicative map) between *-algebras (algebras with an adjoint).

References

[1] K. R. Davidson, *C*-algebras by example*. American Mathematical Soc., 1996, vol. 6.
[2] The mathlib Community. The Lean Mathematical Library. CPP 2020. https://doi.org/10.1145/3372885.3373824
[3] Patrick Massot, "Why formalize mathematics," N/A, 2021
[4] K. Buzzard and R. Taylor, "A lean proof of fermat's last theorem," Imperial College of London, Tech. Rep., 2025.
[5] Mathlib community, "Completion of the liquid tensor experiment," Lean Community Blog
[6] Kontorovich, A., & Tao, *T. Prime Number Theorem and More*. https://github.com/AlexKontorovich/PrimeNumberTheoremAnd

The GNS Construction

Theorem (Gelfand-Naimark-Segal Construction):

Let A be a C*-algebra.

Let f be a positive linear functional on A.

There exists a pre-inner product space A_f, whose elements are the elements of A, and whose semi-inner product is defined

$$\langle a, b \rangle_f = f(a^* b).$$

There is also a function $\pi_f : A \to B(A_f)$ defined for $a \in A$ and $x \in A$ as

$$\pi_f(a)(x) = ax.$$

Then, if we define H_f to be the Hausdorff completion of A_f with respect to its semi-inner product-induced semi-norm, H_f is a Hilbert space.

And, there exists a *-homomorphism $\overline{\pi}_f(a) : H_f \to H_f$ that is the completion of $\pi_f(a) : A_f \to A_f$.

Formalizing the GNS Construction

```
variable {A : Type*} [NonUnitalCStarAlgebra A]
[PartialOrder A] [StarOrderedRing A]

variable (f : A →ₚ[ℂ] ℂ)

def PreGNS (f : A →ₚ[ℂ] ℂ) := A

abbrev preGNSpreInnerProdSpace :
PreInnerProductSpace.Core ℂ f.PreGNS where
 inner a b := f (star (f.ofPreGNS a) * f.ofPreGNS b)
  ...

def leftMulMapPreGNS (a : A) :
  f.PreGNS →L[ℂ] f.PreGNS := ...

abbrev GNS := Completion f.PreGNS

def gnsNonUnitalStarAlgHom :
   A →⋆ₙₐ[ℂ] (f.GNS →L[ℂ] f.GNS) where
toFun a := (f.leftMulMapPreGNS a).completion
...
```

Why Formalize Mathematics?

In "Why formalize mathematics?," by Patrick Massot [3], he explains five main reasons that people formalize mathematics. Two of those reasons are:

Verifying Correctness

Where conventional mathematical proofs rely on the understanding of a human reader to interpret and be convinced by a proof, formal proofs are machine verified by a program that will only accept a proof that is 100% correct and complete.

This eliminates potential issues from:

- missing edge cases,
- references to inaccessible or incomprehensible literature, and
- proofs being too large or complex for one person to reasonably verify themselves.

Supporting Collaboration

Formal proofs make it easier to subdivide and delegate tasks, because each proof can be developed/edited without impacting the work of any other collaborators and can be verified with no additional human labor.

Examples of collaborative formalization projects include:

- the project to formalize Fermat's Last Theorem [4],
- the Liquid Tensor Experiment [5],
- the Prime Number Theorem (PNT+) project [6], and
- Mathlib [2].

Advisor: Lucas Bang
Reader: Michael Orrison

Special Thanks to: Konrad Aguilar, Jireh Loreaux, and Monica Omar for their feedback and collaboration on this project.