The forthcoming

golden age

of mathematics software

Benjamin Antieau
Northwestern University

Deven Manam
University of Copenhagen

Accelerating Mathematical Discovery
30 October 2026

Background

- interested in Brauer groups, derived categories, algebraic K-theory, and higher category theory;
- work in the last five years on algorithms related to algebraic K-theory using syntomic cohomology with Achim Krause and Thomas Nikolaus;
- work on a much bigger project to make computations with spectra effective;

I've followed AI from the beginning: genetic algorithms and game theory over 20 years ago, and helping undergrads train a neural network to play a card game around 2018. LLMs I found largely a curiosity until February of this year.

Plan

I What I've worked on since Feb 2026.
II Show-and-tell.
III The development process.
IV Challenges and future work.

I.a. `snforacle`

On a cold winter evening in Chicago, I installed Claude Code and had it create plumbing between software projects I wanted to interact. FLINT has very fast integer matrices, but did not, at the time, return transformation matrices for Smith form.

That was my default package for integer matrices and I didn't want to swap it for something else. Pari can do this, and so can Magma. But Magma is often not installed. So, I wanted an oracle with a ladder of preferred options for getting this information. Hence, `snforacle`.

I.b. Bourbaki

The prompt was to create a new language for type theory completing the analogy

$$\textit{Bourbaki} : \textit{LEAN} :: \textit{Python} : \textit{assembly}.$$

This quickly took me from a $20/month Claude Code account to a $200/month account.

The architecture is that this compiles into low-level LEAN that can be checked by the LEAN kernel. I view the LEAN kernel as analogous to a microprocessor; LEAN is its assembly language.

l.c. `Noether` (with D. Manam)

Two previous attempts at creating new CASs from scratch: `Carrara`, a pure-Rust system built with grad students at NU; never got off the ground. And, `Jacamar`, a Python library using FLINT for fast arithmetic. Built for

(i) a faster version of the $K(\mathbf{Z}/p^n)$ algorithm;

(ii) the insane complexity of the `effective spectra` project.

Agentic coding allowed me to revisit this in a huge way, leading to `Noether`. The first attempt reached around 1M LOC. The first public release is much smaller and more curated. More on this later.

See also forthcoming `Syntomic.jl of Gabriel Ong`.

l.c. `Noether`: benchmarks

Computation	**Noether**	**Sage**	**Magma***
Cyclic-8 Gröbner basis	**0.887**	20.233	0.950
Katsura-9 Gröbner basis	**0.221**	3.625	0.110
Triangle count 30,000 vertices, 1.92M edges	**0.991**	1.380	—

Times in seconds on an Apple M3 Max, one thread.

*Magma 2.29-5 under Rosetta.

l.c. `Noether`: scale

Dense $\mathbf{F}_2$ matrix multiplication on Quest with `m4rst-distributed`.

200 GiB
packed left operand

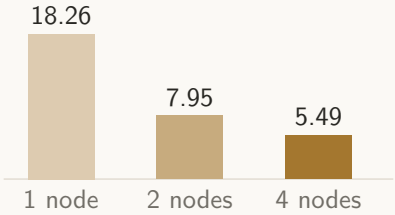
$26{,}214{,}400 \times 65{,}536$
times $65{,}536 \times 256$

8 nodes, 16 processes

89.8 s
multiplication time

Validated 10 GiB campaign

Median time (seconds)



3.33× **speedup**

I.d. The IGP24 SAIR competition

This summer, the SAIR Foundation ran a competition about making the inverse Galois problem in degree 24 concrete. At the start this was the first degree with open problems, except for M_{23} in degree 23. There are 25,000 possible Galois groups T of degree 24 extensions. Each of these can occur for a field with varying signatures r_1. There are 165,836 possible pairs (T, r). The goal was to produce as many examples as possible, hopefully completely solving IGP24. I found **92,175**, good enough for 7th out of 256.

Tools: Noether, Pari, AWS, GCP, Azure.

https://antieau.github.io/2026/08/27/gunter-malle-juergen-klueners-ai.html

I.e. `Reasoning` (with D. Manam)

Our first harness, `Reasoning`, was built in summer 2026. It provides an architecture for proving theorems with math. Lots of iterations and versions.

The first public release was earlier this month. It is in active use as part of a big SLMath collaboration, `stanconj.org`. It turns out to be similar in spirit to SAIR's Open Math OS. It has been used to prove and disprove conjectures by now.

Should be easy to be used by teams using DigitalOcean, Modal, OpenRouter, and Resend.

I.f. `aiTeX` (with D. Manam)

A fast implementation of the TeX compiler written in pure Rust. These slides were compiled with it. The goal is to create something that could be compiled to WASM (after a conversation with Hood Chatham).

Not easy: the TeX compiler is super complicated; Knuth is a genius.

It also has an experimental *micro-harness*, i.e., the ability to automatically repair TeX compilation errors live using LLMs.

I.g. TypeBase

A database for homotopy theory,

to solve my LMFDB-envy.

I.h. Hexagon (with J. Schmitt)

Runs on AWS and CloudFlare.



https://hexagonmath.org/2610.00294v1

II. Show-and-tell

(1) `Reasoning.`
(2) `aiTeX.`
(3) `Noether.`
(4) `TypeBase.`

III. The development process

(i) Hetzner: I pay around $125/month for a **fast** machine.

(ii) ~~AWS~~ Modal: containerized CPU and GPU compute.

(iii) ~~CloudFlare~~ DigitalOcean: hosting.

(iv) Resend for sending emails.

(v) GitHub, especially CI, although I now self-host those on the Hetzner box above because it costs money.

(vi) `claudedsp` and `codexdsp`.

(vii) OpenRouter.

(viii) Browser automation.

IV.1. Correctness

(a) Test against existing software oracles.
(b) Property-based testing.
(c) Ablation tests.
(d) LEAN certificates. (See also Hex.)

IV.2. Human attention

Attention is all you need.

By far the biggest factor in determining the success of these projects is very large amounts of human attention. Still less than programming them yourself. In addition, having multiple users is key.

IV.3. Hard algorithms

Three algorithms I was not able to one-shot (as of the summer, so pre-Fable).

(I) Cohen's class group algorithm.

(II) The Gröbner basis techniques in `msolve`.

(III) the Rubi rules-based integration system used in `Mathematica`.

Documentation slop.

IV.5. The goal

The goal for Noether is to provide arithmetic primitives and algorithms for running

GPU and cluster-scale computations

across pure mathematics.

Thank you

tertius.cc