

Let the computers do the computation!

Heather Macbeth

Imperial College London/Fordham University

“Mechanization and Mathematical Research”

Lorentz Center

17 September 2025

Today's theme: "Practice"

Today's discussion question:

What is a mathematician? What do mathematicians do?

Then there's everyone's favourite Grothendieck legend:

In a mathematical conversation, someone suggested to Grothendieck that they should consider a particular prime number. "You mean an actual number?" Grothendieck asked. The other person replied, yes, an actual prime number. Grothendieck suggested, "All right, take 57."

(Jackson, Comme appelé du néant . . . , 2004)

Michael Barany told us on Monday that to analyse mathematicians' group identity, we can study mathematicians' shared values.

Computation seems to be a shared **anti-value!**

A widespread feeling is that “real mathematics” is what is left when you take away the computations.

In this talk I will argue that current mathematical practice doesn't live up to this value as thoroughly as it could:

We could be taking away *more* computations, and exposing a smaller, purer core of “real mathematics.”

Example 1: The Jordan–von Neumann theorem

Theorem (Jordan–von Neumann, 1935)

A Banach space E is a Hilbert space if and only if its norm satisfies the “parallelogram law,” i.e.

$$\forall x, y \in E, \quad \|x + y\|^2 + \|x - y\|^2 = 2(\|x\|^2 + \|y\|^2).$$

Proof (hard direction): We define a function $\langle \cdot, \cdot \rangle$ from $E \times E$ to \mathbb{R} as follows:

$$\langle x, y \rangle := \frac{\|x + y\|^2 - \|x - y\|^2}{4}.$$

We propose that $\langle \cdot, \cdot \rangle$ should be the Hilbert space’s inner product.

Key step: additivity of $\langle \cdot, \cdot \rangle$, i.e.

$$\forall x, y, z \in E, \quad \langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle.$$

Proof from the literature¹

Replace f, g in (*) by $f' \pm g, f''$, and subtract. Then

$$\begin{aligned} \|f' + f'' + g\|^2 - \|f' + f'' - g\|^2 + \|f' - f'' + g\|^2 - \|f' - f'' - g\|^2 \\ = 2(\|f' + g\|^2 - \|f' - g\|^2) \end{aligned}$$

obtains, that is (considering (*)):

$$(\S) \quad \Re(f' + f'', g) + \Re(f' - f'', g) = 2\Re(f', g).$$

(*) with $f = 0$ proves $\|-g\|^2 = \|g\|^2$, and so (*) with $f = 0$ gives $\Re(0, g) = 0$. Therefore (§) with $f' = f''$ is $\Re(2f', g) = 2\Re(f', g)$. Thus (§) itself becomes

$$\Re(f' + f'', g) + \Re(f' - f'', g) = \Re(2f', g),$$

or if we replace f', f'' by $\frac{1}{2}(f' + f''), \frac{1}{2}(f' - f'')$:

$$\Re(f', g) + \Re(f'', g) = \Re(f' + f'', g).$$

¹Jordan–von Neumann, “On inner products in linear, metric spaces,” 1935.

Another proof

Proof.

Consider the following four parallelograms:

$$\|(x + y + z) + (x - z)\|^2 + \|(x + y + z) - (x - z)\|^2 = 2 (\|x + y + z\|^2 + \|x - z\|^2)$$

$$\|(x + y - z) + (x + z)\|^2 + \|(x + y - z) - (x + z)\|^2 = 2 (\|x + y - z\|^2 + \|x + z\|^2)$$

$$\|(y + z) + z\|^2 + \|(y + z) - z\|^2 = 2 (\|y + z\|^2 + \|z\|^2)$$

$$\|(y - z) + z\|^2 + \|(y - z) - z\|^2 = 2 (\|y - z\|^2 + \|z\|^2)$$

We vector-space-normalise the E -valued expressions (such as $(x + y + z) + (x - z)$), then run Gaussian elimination to prove that the desired result,

$$\frac{\|x + y + z\|^2 - \|x + y - z\|^2}{4} = \frac{\|x + z\|^2 - \|x - z\|^2}{4} + \frac{\|y + z\|^2 - \|y - z\|^2}{4},$$

is in the linear span of the four normalised identities. □

I prefer my proof!

It is easier to grasp at high level: it is clear upfront what facts are being used, and the reader can check by eye that the goal appears to be within the scope of the vector-space normalisation and Gaussian elimination algorithms as run on these facts.

Its black-boxing of the routine algorithm also makes the ideas more transparent—in this case, the choice of the four parallelograms.

Example 2: The Kochen–Specker paradox

Theorem (Kochen–Specker, 1967)

There does not exist a boolean function $s : \mathbb{R}^3 \rightarrow \{0, 1\}$, such that for all triples $u, v, w \in \mathbb{R}^3$ of nonzero mutually-orthogonal vectors, $s(u), s(v), s(w)$ is 0, 1, 1 in some order.

Proof (Peres, 1991)²: Deduce a contradiction from the values of s on the following 33 nonzero vectors in \mathbb{R}^3 :

$$\left| \begin{array}{c|c|c|c|c|c|c|c|c|c|c} \bar{1}\bar{1}2 & \bar{1}02 & \bar{1}\bar{1}2 & \bar{1}\bar{2}\bar{1} & \bar{1}20 & \bar{1}21 & 0\bar{1}\bar{2} & 002 & 012 & 02\bar{2} & 02\bar{1} \\ 020 & 021 & 022 & 1\bar{1}\bar{2} & 102 & 112 & 12\bar{1} & 120 & 121 & 2\bar{2}0 & 2\bar{1}\bar{1} \\ 2\bar{1}0 & 2\bar{1}\bar{1} & 20\bar{2} & 20\bar{1} & 200 & 201 & 202 & 21\bar{1} & 210 & 211 & 220 \end{array} \right|$$

Here $\bar{1}$ is shorthand for -1 , 2 is shorthand for $\sqrt{2}$, and $\bar{2}$ is shorthand for $-\sqrt{2}$. (Sorry!)

²This is an improvement on the original proof, which found a contradiction from a set of 117 vectors.

Proof from the literature³

Proof (continued).

We can determine the colours of some vectors without loss of generality:

By the symmetry	and by the known facts	we can assume
choice of z-axis		001 green; 100, 010 red
choice of x vs $-x$	010 red	101 green; $\bar{1}01$ red
choice of y vs $-y$	100 red	011 green; $0\bar{1}1$ red
choice of x vs y	001 green, thus 110 red	$1\bar{1}2$ green; $\bar{1}12$ red

Now a suitable greedy sequence of deductions (see next slide) forces a contradiction . . . □

³Peres, "Two simple proofs of the Kochen-Specker theorem," 1991.

Another proof⁴

Proof (continued).

Perform the following binary search: split on a vector whose colour is not yet known; then in each case (red or green) greedily make all possible deductions. Stop if a contradiction is found. Recurse if not.

The result of this process is that every branch terminates in a contradiction. □

⁴Harrison, 2005; *Formalizing an analytic proof of the Prime Number Theorem*, 2009.

Both proofs amount to the implementation of a search algorithm.

The search algorithm used in Harrison's proof is very simple, and arguably aesthetically superior: the simpler algorithm is easier for the reader to grasp.

Whereas the symmetry considerations incorporated in the original proof can be considered as baroque optimisations to the search algorithm to get its “runtime” within the scale of human readability.

Showing my hand

For the last five years I have been increasingly absorbed by computer-formalisation of mathematics.

I started thinking about this “computation-outsourced” proof style because I observed it developing among the community of mathematicians who write formalised proofs.⁵

The two examples I just presented started life as formal proofs:

- for the Jordan–von Neumann theorem, Lean code written by Ruben Van de Velde and edited by me;
- for the Kochen–Specker paradox, HOL Light code written by John Harrison.

⁵Macbeth, *Algorithm and abstraction in formal mathematics*, 2024.

This “computation-outsourced” style is particularly compatible with computer-formalised mathematics, because there the outsourcing of computations is seamless: there is no dangerous transition when a prose question gets typed into Sage or when Sage’s answer gets translated back into prose.

In formalised mathematics, there is truly no burden in outsourcing every single computation, no matter how large or how small.

What is left is the “complement of the calculations.”

And this is what we mathematicians think we value.

... Right?