Why I became a professional mathematician

Frank Vallentin*

September 9, 2026†

Abstract

In this personal essay, I try to reconstruct the reasons why I chose to become a professional mathematician. In particular, by writing this note, I want to find out whether I would still find these reasons convincing and motivating today, especially at a time when artificial intelligence is becoming capable of conducting excellent and potentially superhuman mathematical research.

The beginning

As a kid, I liked numbers, counting, and all sorts of numerical data (Bundesliga results, weather statistics, ...).

Throughout elementary and high school, I liked doing mathematics. It came easily to me, but I was not enthusiastic about it. Rather, I found doing homework, including standard computational exercises, relaxing. I guess it was the combination of thinking and writing that had this effect on me, and I also knew that it would never take long.

I was much more obsessed with computers. This began to develop at Christmas 1984, when my grandfather gave me his Sinclair ZX81, one of the first home computers, with 1 KB of main memory. I think he had bought it for himself out of curiosity but found that he could not use it for the tasks he had in mind (he was a master butcher, and I guess he wanted to use it for word processing and spreadsheets; later, it turned out that a Commodore Plus/4 was the right computer for him).

I learned how to operate the ZX81 and played my first computer games (*1K ZX Chess*, *3D Monster Maze*).

Because of the computer's limitations (even after buying a 16 KB memory expansion) and my own limitations (I was 12 years old, and the manual explained BASIC in English), my fascination did not last very long.

In the following years, home computers became more colourful. After a detour with a TI-99/4A (which was powerful but did not have a large user community in Germany), I bought the legendary Commodore 64. I spent quite some time gaming (I was a big fan of Andrew Braybrook's games, particularly *Paradroid* and, above all, *Uridium*, both technically impressive space games). Then my interest shifted completely from playing games to programming them.

I decided to learn 6510-assembly language to program my own games. Special issues of the monthly computer magazine *64'er* helped me learn. Since this was the first programming language I had learned, it took me some time to master the basics. Eventually,

*I am a professor of mathematics (computer science) at the University of Cologne. Email: frank.vallentin@uni-koeln.de, WWW: https://www.mi.uni-koeln.de/opt/frank-vallentin/

†The first draft was written in the evening of Sunday, September 6, 2026.

I wrote a sprite editor, a graphics program for drawing sprites like the beautiful manta ray spaceship in *Uridium*.

But then the era of the C64 came to an end, and we moved on to 16-bit machines—in my case, the Commodore Amiga. At around the same time, in the tenth or eleventh grade, we learned how to program in Pascal at school. We also tried to set up a computer club initiated by our English and geography teacher.

During an evening meeting at his home, he gave me a book about a new programming language, Modula-2. It was still very new and had been invented by Niklaus Wirth at ETH Zurich. The teacher told me that he could not make much sense of the book and thought that perhaps I would have more use for it. And indeed, I became hooked on the beauty of a higher-level programming language that was easy to program in, had been designed by a famous computer science professor, and offered practical software-development techniques: dividing a program into modules in different files and separating a module's interface from its implementation. An elegant implementation for the Amiga, M2AMIGA, was available, and I invested both time and money in it.

Automatic mathematics

In grade 11, school mathematics really annoyed me. For half a year, we did nothing but curve sketching: Looking at functions, computing values, computing derivatives, checking for monotonicity, finding roots, determining minima, maxima, and saddle points. I was annoyed, as it felt completely mechanical. I decided that I wanted to automate it. Write a computer program which, given a function as input, would perform all the desired computations and print the result as LaTeX.[1]

Writing such a program turned out to be a challenge, a much greater challenge than I had expected. It was an excellent challenge: I had to learn what a function is and, especially, how to work with symbolic formulas. I was asking myself the question: when is a formula well formed? If you don't know how to formulate this question of syntactic correctness, it is difficult to find an answer. This was a few years before the World Wide Web, so getting technical information in a remote village in the German countryside was hard. Somehow I found out that I had to learn how to build a compiler, and, by a twist of fate, I took a train to Siegen, the nearest university town, where I went to an academic bookshop and bought the "Dragon Book" (Compilers: Principles, Techniques, and Tools) by Aho, Sethi, and Ullman, a computer science text book explaining all the details of compiler construction. This book was an eye-opener for me in several respects: I learned about syntax and semantics, structural induction, pattern matching, compiler-compilers, and many fascinating facets of computer science. It also helped me to finish my curve-sketching program, but more than half a year too late to make real use of it.

First encounters with the mathematical community

A little later, the first forms of online access became available, opening up new ways to connect with other people. With my 1200-baud modem, for instance, I could dial into the MausNet mailbox MAUS@UN. All of this eventually led me to study computer science at the University of Dortmund.

In the first semester, we learned Standard ML, a functional programming language; it directly supported structural recursion and pattern matching, and writing a correct

[1] Of course, there were already computer algebra programs, for instance *Derive*, which could do this, but I wasn't aware of them.

program amounted to writing an inductive proof that the program was indeed correct. I chose psychology as my subsidiary subject, with the idea that I could specialize in artificial intelligence later.

For mathematics, in particular linear algebra, we had a specialized classes in the computer science curriculum. It was taught by Eberhard Becker, who chose topics close to the standard mathematics curriculum, but with a bit more emphasis on structural, algebraic, and algorithmic aspects. I loved going to these lectures.

Becker's lectures were perfect performances: I learned beautiful mathematical content presented in a way that was similar to a theatrical performance; but it was neither a drama nor a comedy. It was a celebration of beauty in which everything, the voice, the handwriting, the movements, came together to create this effect. Later I found out that this kind of performance is typical of mathematicians who are passionate about their subject. Beyond this, the lectures helped me to gain a better understanding of the mathematics of computation.

The tutorials accompanying Becker's lectures added another dimension. The tutors were young, cool, heavy-smoking, and knowledgeable, and we, the students, got the feeling that mathematics is quite democratic: especially when working together in teams, contributions from all sides are welcome.

In the introductory lecture on theoretical computer science, we discussed questions of complexity, also touching on Gödel's incompleteness results: What makes a (computational) problem difficult or even impossible to solve? The notion that one can reason about computational complexity made another deep impression on me.

Of course, the university had a library. Somehow, the library of the University of Dortmund was extraordinarily well equipped, and even students could borrow all kinds of books for long periods. I made extensive use of this opportunity. Through reading these books, I got a sense of how deep modern mathematics is, how it is rooted in more than 200 years of history, and how there has been steady progress.

I reserved Sunday evenings for writing down the solutions to the exercises. To get into the flow, I listened to the offbeat mix of independent music, drum and bass, and electronic sounds on the radio show *1 Live Fiehe* (formerly *Raum und Zeit*). This habit of doing mathematics on Sunday evenings and getting into a flow is something I have tried to keep up over the last thirty years.

Overall, I think this combination of theatre, performance, beauty for all the senses, empowerment, depth, democracy, teamwork, and history made me want to become a professional mathematician.

The next level

What will change for me if AI becomes superhuman at solving mathematical problems? Not much... The things that drew me to mathematics (beauty, understanding, and working with others) would still draw me to it today. Maybe it will become even more exciting. We are getting power-ups, and we are getting the chance to move to the next level.

My high school dream of making mathematics mechanical is materializing on a grand scale. Mathematical theorems might be digraded to become data points in the landscape of mathematical theory.

Before AI, it took a great deal of human labor to obtain these data points. Now AI provides a proof factory. Proofs are becoming relatively easy to produce, especially those lying in the convex hull of existing mathematical knowledge. *But the laws of computational complexity still apply!*

Behind such a data point there is a mathematical proof. Maybe even a complete "space" of proofs. Over the next few years, I want to explore this space of proofs.

Talking to LLMs, I have the feeling that finding a proof is like solving an optimization problem. By prompting an LLM, one explores a relaxed space of candidate proofs from the outside. This space contains both feasible points—correct proofs—and infeasible points—arguments with gaps or errors. Through (auto-)formalization, one then tries to enter the feasible region by turning candidate arguments into formally verified proofs. Ideally, this interaction produces a feasible solution, a first formally verified proof.

But the work does not end with a first proof: Which proofs provide human insight? Which are particularly elegant? Which are from the BOOK? What is the shape of the space of all proofs? Where are the bottlenecks? How robust are certain parts of arguments? Can one extract useful pieces from a proof as lemmas? How does one move forward to develop a new theory?

I am confident that asking and answering these questions will take our mathematical understanding to a higher level. This deeper understanding will be reflected in the intensity of the data points representing theorems. Some data points will glow only faintly, while others will shine brightly because there is a proof that is understood, admired, and internalized by many humans. It is a privilege of our profession that, in the end, it is mainly this brightness that matters.

Again, machines will help us answer these questions, and I wonder which power-ups we will have to invent and what the next level will be. I want the exploration of this next level to be, once again, a democratic adventure, open to everyone who wants to contribute and respects the values of mathematics. I believe that we, as a mathematical community, must build our own independent AI infrastructure. If mathematical exploration is to remain open, we must retain some control over our tools, training data, and standards of verification.

This will require more teamwork than ever, and for precisely that reason, I am sure that it will be fun. And if we, as teachers, have good reason to be passionate about mathematics, our enthusiasm will be infectious: students will catch it, follow us, and take over.

Acknowledgements

I thank Michael Emmerich, David Gross, David de Laat, Philippe Moustrou, Fernando Oliveira, Achill Schürmann, for feedback on an earlier version of this essay.

I used ChatGPT to help polish the English. The personal experiences and views expressed here are my own.