An Interview with the Authors: Log-Lightning Computation, and Some Instructive Mathematical Errors

ANNIE A. M. CUYT, University of Antwerp, Belgium

An interview with the authors of “Log-lightning computation of capacity and Green’s function”, *Maple Trans.* 1, 1, Article 14124 (July 2021), and the author of “Some Instructive Mathematical Errors” *Maple Trans.* 1, 1, Article 14069 (July 2021). This interview was conducted by Annie Cuyt, with authors Peter Baddoo & Nick Trefethen and with Richard Brent, on Wednesday Sep 22, 2021 7am – 8am (EDT) via Zoom.

**Recommended Reference Format:**
Annie A. M. Cuyt. 2021. An Interview with the Authors: Log-Lightning Computation, and Some Instructive Mathematical Errors. *Maple Trans.* 1, 1, Article 14466 (July 2021), 5 pages. https://doi.org/10.5206/mt.v1i1.14466

**Annie:** Since all of us love numerics so much and we therefore have so many things that connect us, I think the conversation will come up quite naturally. One of the first things we could start with, is that each of you tell us what you love about numerics. When did you realize that it was the branch in mathematics where you wanted to have a career?

**1. Pure, applied, computational mathematics?**

**Nick:** When I was a freshman at Harvard, I assumed I would be a pure mathematician. I thought I was more serious than the others and would be an expert at both analytic and algebraic number theory. A year later everything changed and I realized that this was ridiculous and what mattered was applied mathematics, because it connects with so much more of the scientific landscape. The pure mathematics always felt to me a little bit isolated and applications seemed more connected.

**Richard:** Thinking back, it was probably when I was a graduate student at Stanford and I had to take the qualifying exams. I went to courses by Gene Golub and other people. Then later I got involved in pure math, but using numerics and computational number theory.

**Annie:** Both of you use the terms pure and applied mathematics, which is kind of like a very traditional division. I have been switching to the wording impactful mathematics which can cover parts of both pure and applied mathematics.

---

Annie: Since all of us love numerics so much and we therefore have so many things that connect us, I think the conversation will come up quite naturally. One of the first things we could start with, is that each of you tell us what you love about numerics. When did you realize that it was the branch in mathematics where you wanted to have a career?

1. Pure, applied, computational mathematics?

Nick: When I was a freshman at Harvard, I assumed I would be a pure mathematician. I thought I was more serious than the others and would be an expert at both analytic and algebraic number theory. A year later everything changed and I realized that this was ridiculous and what mattered was applied mathematics, because it connects with so much more of the scientific landscape. The pure mathematics always felt to me a little bit isolated and applications seemed more connected.

Richard: Thinking back, it was probably when I was a graduate student at Stanford and I had to take the qualifying exams. I went to courses by Gene Golub and other people. Then later I got involved in pure math, but using numerics and computational number theory.

Annie: Both of you use the terms pure and applied mathematics, which is kind of like a very traditional division. I have been switching to the wording impactful mathematics which can cover parts of both pure and applied mathematics.

---

Permission to make digital or hard copies of all or part of this work for any use is granted without fee, provided that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. © 2021 Copyright held by the owner/author(s). Publication rights licensed to Maple Transactions, under Creative Commons CC-BY 4.0 License. 

https://doi.org/10.5206/mt.v1i1.14466

Maple Trans., Vol. 1, No. 1, Article 14466. Publication date: July 2021.
Peter: Similarly, early on in my career I thought I was going to be a pure mathematician but then I realized that I didn’t know what pure mathematics was. The first year at Oxford was kind of a wake-up call because I realized what I did at school was not very pure, it was actually applied even when they called it pure. My interest in computational mathematics began relatively recently in my career. During my undergrad and PhD we kind of disparaged numerical methods. We thought of it as just a computer solving the problem. Real thoughts or physical insight could not be gained from that. In the last two or three years I really came to see the value and the beauty of numerical methods. Now I can really appreciate how numerical methods illustrate something fundamental about the problem at hand.

Richard: There need not be such a great distinction between pure and applied. I think it’s a continuum and there’s a road in the middle. Some of the pure mathematicians I knew were really interested in applications. Maybe now they specialize too much. I was really a computer scientist for most of my career and not a mathematician at all.

2. Or experimental mathematics?

Nick: Forgetting the pure versus applied for a moment, I like hands-on things, I’m on the computer all the time. I don’t like the phrase experimental mathematics: it sounds so amateuristic. If you look at physics: everyone knows that physics has an experimental side and a theoretical side and that one needs both for the field to advance effectively. Even if a theorist feels remote from experiments, no theorist would say that physics shouldn’t have experiments. Somehow mathematics has ended up in a weirdly different state and it’s pretty easy to see why. Mathematics has the possibility of proof. So in some abstract sense, it almost seems as if it doesn’t need experiments. But that’s crazy. Proof is a wonderful thing that mathematics has in addition. But if you want to find out what phenomena are out there, once you’re at the level of complexity that we all are at now, you need the experiments. So I believe that experimental mathematics, although I hate the phrase, is intrinsically as important as experimental physics and it just amazes me that so few people are operating in that mode. It need not be just the numerical analysts, although we’re the ones for whom the tools are easy. And let me say one more thing on why writing for this journal was so appealing for me. It’s clear that the Maple Transactions style and Rob Corless’ style is very hands-on and computational. Even though I happen to be a Matlab person, the hands-on style of computing is very much what I relate to.

Annie: I associate experimental mathematics a lot with computer algebra systems. These allow you to investigate to very high precision some conjectures that you would like to be able to prove. That is how I have been active in experimental mathematics. And I don’t hate the word as much as you do ;-)
numbers on the computer. I had to do a lot of pattern matching and then sit down and try to prove it, which was actually difficult. I was convinced that the result was correct because the numbers were equal to a hundred digits or more.

**Peter:** Sometimes there are situations where the experiments can be misleading, or can lead you down the wrong route. The interplay between theory and experiments is very important: the theory can refine the experiments and the experiments can refine the theory.

**Nick:** We should be careful to not go down the wrong path as a result of errors in experiments.

**Annie:** And this is often where computer algebra systems come to the rescue.

### 3. What about computer algebra systems?

**Annie:** What was your first encounter with computer algebra systems? I vividly remember mine: it was 1984, in Stanford, at a conference I went from one computer algebra system stand to the next with a particular integral, to check whether the system could return the answer.

**Nick:** Well, I’d like to tell the story of my first encounter with Maple. I don’t remember the year, I suppose it was early eighties. I was visiting Waterloo or Toronto or something and a bunch of us at a conference were put in a room. We were allowed to play with Maple. This is like 1982, the computers were not like now and I thought I would ask Maple for ten thousand digits of $\pi$. So I entered the command and I typed return and instantly out came ten thousand digits of $\pi$! It turned out that Maple had prestored exactly ten thousand digits of $\pi$. If you asked ten thousand and one, it ran out of memory.

**Peter:** When I was first getting interested in mathematics, I was thinking along the lines of how I could improve my mathematical ability. As a sixteen year old, my conclusion was to get Maple and to experiment with that. There were tutorials on different mathematical concepts. I remember going through a few of those and this really ignited my interest in mathematics at the time. That was my first experience with computer algebra, I think even with any programming or mathematical software.

**Richard:** As a graduate student at Stanford in the early seventies, I used some program, coded in Lisp, probably Macsyma, but I didn’t use it extensively. But even before that, as an undergraduate, some of my friends were students of a famous group theorist and he wanted them to compute a lot of things by hand. I wrote a little program for them. So I guess, in some sense I wrote a symbolic computing program, because their professor didn’t believe in computers.
4. On communicating mathematics through code.

Nick: Another thing I’m passionate about is that, for communicating certain simple ideas mathematically, a code segment is very powerful. It doesn’t work for all endeavours, of course. Often, for the kind of things I do, ten or fifteen or twenty lines of code are a beautiful and precise way to say exactly what I did. The trouble is that journals by and large hate this. They want you to describe your algorithms but not actually give the details. This again is one reason Maple Transactions appeals to me, because obviously the style here is very welcoming to code. I realize that a lot of code that people might want to publish is a mess and ugly and counterproductive. But if the journals would more generally encourage excellent bits of code, then the standards might go up.

Annie: You actually touch on a point that was on my list of items to discuss. I think that all of us like to explain very clearly and particularly how algorithms work and why they do such a beautiful job. What do you want to add about teaching numerical mathematics? Or transmitting new research?

Peter: It’s connected to how much impact your research has. If you have a concept that is very, very complicated and you don’t explain it well, then it’s hard to have some impact, in the sense that other people will probably not use your algorithm. But if you can distill it to ten or fifteen lines of code and then explain it clearly so that they get the basics, then you can be more impactful and can help other people in the field. For teaching, I’m not yet using pieces of code, but I could see the benefit of it.

Richard: I haven’t taught such a course for a few years now. When I did, I certainly used bits of code, not necessarily in Maple or Matlab, but in some pseudo-code that was easy to convert. You don’t have ambiguities in code, which you might have in English.

Annie: In my computational science classes, I often first introduce a concept theoretically and then afterwards I take the subject up again but in a computer algebra environment. Take Lagrange interpolation. I would introduce the basis functions, explain the interpolation algorithm and then program some visual illustration in Maple and make that code available with the course text.

Nick: I’ve taught for twenty years on each side of the Atlantic ocean and I find, at Oxford at least, computing is a problem for two reasons. One is that we get such outstanding students, and with society being what it is, the very best mathematicians often make it a point of pride that they’re above computing. Also, at least in the Oxford system, it’s very hard to mix computing into courses because it doesn’t get assessed. We’ve not been very successful at making computing a part even of our numerical analysis courses. Maybe at other universities in the UK it’s a bit more free. Certainly in the US it’s a bit more free and I have to say that I miss that.

Richard: I think that in Australia too, the students that are enrolled in maths, don’t do a lot of computing. And the computing students generally dislike maths. So there’s a dichotomy here. Something related to what we discussed, is the following. When students graduate, a lot of them find jobs in programming. So the more programming they do in their course work, up to a reasonable limit of course, the better. Not a lot of mathematicians are going to get a job as teachers of pure maths.

Annie: That’s a pity, of course, because there are also a lot of modelling jobs in banks, pharmaceutical companies, engineering enterprises that need good mathematicians.

5. And there’s also extended precision.

Nick: I have a Maple - Matlab question to bring up. There is of course symbolic computing, but there’s also extended precision computing. It’s a funny accident of history that many numerical
people are stuck in this 64-bit mode. I wish I could have a knob I could turn and crank up the computation to 32, 64, 128 bits of precision, and yet we all know that when we do this in a system like Maple, things slow down enormously. Any comments on this weird schism between the numerical world and the Maple world?

Richard: It would be nice if you had a dial you could turn, and in a sense you do if you don’t mind about the speed of the program. You need to be careful though, because not everything works as you think. You may have defined some constants in standard precision and that mucks everything up. Unfortunately, because of the computer’s design, operations in standard hardware precision will run a lot faster than operations programmed in software. It’s more or less inevitable. Even in packages that are optimized, software is a factor of 10 times slower.

Nick: We worry about this issue a lot in relation with our paper in the Maple Transactions, which is very much motivated by high precision. The algorithm is spectacularly good if you want a large number of digits. We don’t really know how feasible that is in a system like Maple. For instance, besides computing in a large number of digits, you also need to do numerical linear algebra in a solid robust way in high precision. Is that feasible?

Richard: I don’t see why it is not feasible. But I don’t know whether it is implemented at the moment. There’s no reason why you cannot do numerical linear algebra in high precision, besides the storage requirements of course. If you would do it using a precompiler idea, then the debugging becomes very difficult if the line of code is compiled into some other code so that one may be looking for the error in the wrong place. I was actually involved in precompilers for multiprecision arithmetic and complex arithmetic back in the seventies. But they never really took off. But nowadays with languages such as Python it may become easier.

Nick: Peter, what’s the attitude of the students steeped in Julia at MIT towards these things? These things being: computing, precision, numeric versus symbolic, Python versus Julia versus Matlab versus Maple.

Peter: The computer science students at MIT are really incredible at coding. There are also incredible mathematicians here. The students are highly motivated. They’re very supportive of Julia, partly because they can meet the people who developed Julia. They’re very good and fast at learning languages and are probably much more willing to learn Julia than in the UK.

Annie: Thank you all for your time and interesting comments. I am sure that we all look forward to a time when we can meet again, in person, to continue this discussion!