SIAM NEWS

Maxims About Numerical Mathematics, Computers, Science, and Life

January 8, 1998

Lloyd N. Trefethen

Last spring, I taught a new senior/master's-level course at Cornell: "Software Tools for Computational Science," CS 422/522. In this course we tried to blend software and science in an unusual way. Some details are available at the course home page, http://www.cs.cornell.edu/Info/Courses/Spring-97/CS522.

While teaching this class, I handed out a "daily maxim" to the students in each lecture. By the end of the semester, the maxims had covered quite a wide range of topics. At the bottom of each sheet handed out to the class, I had written the following disclaimer: "In order to stimulate thought, these maxims are formulated as concisely as possible, with qualifications and caveats omitted. You may find some of them exaggerated or oversimplified." Of course, the same disclaimer applies to *SIAM News* readers! And, of course, while a few of these ideas may be new, I recognize that many have been expressed before.

Here are my maxims. Reactions to lnt@comlab.ox.ac.uk would be welcome.

1. Scientific progress occurs by revolutions and paradigm shifts on all scales. If Kuhn had written *The Structure of Scientific Revolutions* after the popularization of fractals, he would never have dared to suggest there were just two.
2. There are three great branches of science: theory, experiment, and computation.
3. If no parameters in the world were very large or very small, science would reduce to an exhaustive list of everything.
4. Science is the extraction of underlying principles from given systems, and engineering is the design of systems on the basis of underlying principles.
5. A computational study is unlikely to lead to real scientific progress unless the software environment is convenient enough to encourage one to vary parameters, modify the problem, play around.
6. If the answer is highly sensitive to perturbations, you have probably asked the wrong question.
7. One of the most important of disciplines is one that never assumed its logical name: mathematical engineering. (Instead, it is called applied mathematics.)
8. The big gulf in the mathematical sciences is between the continuous problems (and people) and the discrete ones. Most scientists and engineers are in the continuous group, and most computer scientists are in the discrete one.
9. The fundamental law of computer science: As machines become more powerful, the efficiency of algorithms grows more important, not less.
10. The two most important unsolved problems in mathematics are the Riemann hypothesis and "P = NP?". Of the two, it is the latter whose solution will have the greater impact.
11. No physical constants are known to more than around 11 digits, and no truly scientific problem requires computation with much more precision than this. (OK, throw in another 5 or 6 digits to counter the slow accumulation of rounding errors in very long calculations---using numerically stable algorithms, of course, without which you're sunk in any precision.)
12. Digital arithmetic is to scientific computing as collisions of molecules are to fluid mechanics: crucial for implementation, but irrelevant to most scientific questions.
13. All that it is reasonable to ask for in a scientific calculation is stability, not accuracy.
14. Most problems of continuous mathematics cannot be solved by finite algorithms.
15. Chess is a finite game, hence trivial, but this fact does not seem to dismay those who play it.
16. If rounding errors vanished, 95% of numerical analysis would remain.
17. Just because there's an exact formula doesn't mean it's necessarily a good idea to use it.
18. For large-scale problems of continuous mathematics, the best algorithms are usually infinite even if the problem is finite. In other words, analysis is more useful than algebra, even if the problem is

algebraic.

19. Symbolic computing is mainly useful when you want a symbolic answer.
20. As technology advances, the ingenious ideas that make progress possible vanish into the inner workings of our machines, here only experts may be aware of their existence. Numerical algorithms, being exceptionally uninteresting and incomprehensible to the public, vanish exceptionally fast.
21. Is there an $O(n^{2 + varepsilon})$ algorithm for solving an $n times n$ system $Ax = b$? This is the biggest unsolved problem in numerical analysis, but nobody is working on it.
22. The purpose of computing is insight, not pictures.
23. Twenty years ago, we did not interact with computers graphically, but now, everything is graphical. In the next 20 years, an equally great change will occur as we take to communicating with machines by speech.
24. 109 is a kind of magic number, above which new effects emerge. Think of neurons in the brain, nucleotides in the genetic code, people on earth, or the number of calculations carried out by a computer each time you press Enter.
25. Eventually mankind will solve the problem of consciousness by deciding that we are not conscious after all, nor ever were.
26. In the long run, our large-scale computations must inevitably be carried out in parallel.
27. Nobody really knows how to program parallel computers. Nobody really knows how the brain works. In the next century, related revolutionary developments will occur in both areas.
28. Advanced technological civilizations can last hundreds or maybe thousands of years, but not millions. If this is not so, how can you explain the extraordinary coincidence that you live right at the dawn of this one? And how can you explain the fact that despite millions of years in which to make the journey, civilizations from other parts of the universe have not taken over the Earth?
29. Thanks to digital logic and careful error correction, computers have traditionally behaved deterministically: If you run the program twice, you get the same answer both times. However, as computing becomes ever more intelligent and more distributed in the upcoming century, determinism in any practical sense will fade away. No fully intelligent system can be expected to give you the same answer twice in a row.
30. If the state space is huge, the only reasonable way to explore it is at random.
31. Computational mathematics is mainly based on two ideas: Taylor series and linear algebra.
32. In principle, the Taylor series of a function of n variables involves an *n*-vector, an *n* x *n* matrix, an *n* x *n* x *n* tensor, and so on. Actual use of orders higher than two, however, is so rare that the manipulation of matrices is a hundred times better supported in our brains and in our software tools than that of tensors.
33. When the arithmetic is easy and the challenge lies in efficient ordering of a sequence of operations, computational science turns to graph theory. Two big examples are automatic differentiation and direct methods for sparse linear systems of equations.
34. The two biggest world events in the second half of the 20th century were the end of the Cold War in 1989 and the explosion of the World Wide Web beginning in 1994.
35. Mankind discovered 50 years ago what natural selection discovered 4 billion years ago: If you want to manipulate and copy information without errors, your logic has to be digital.
36. Life on Earth consists of 107 species, each consisting of 1010 individuals, each consisting of 1013 cells, each consisting of 1014 molecules.
37. Computer codes are better written than genetic ones, since there's a programmer in the loop, but as they get bigger, this distinction is fading.
38. Living things are 3D objects, yet they are constructed by folding up 1D pieces. This astonishing method of construction is what makes repair, reproduction, and natural selection practicable. If an infinite intelligence designed an organism from scratch, by contrast, presumably it would use 3D building blocks.
39. Animals are composed of millions of autonomous cells, each performing specialized functions and communicating with the others by prescribed signals. It is almost unimaginable that large creatures could have been engineered without such a modular design. The same object-oriented principles apply in the engineering of large software systems.
40. Once we know the human genome, will that knowledge prove useful? You might as well ask an expert user of a software system, might it be helpful to have access to the source code?

Lloyd N. Trefethen moved in September 1997 from Cornell University to the University of Oxford, where he is a professor of numerical analysis and head of the Numerical Analysis Group in the Computing Laboratory.

- See more news from this issue

3600 Market Street, 6th Floor | Philadelphia, PA 19104-2688 USA
Phone: +1-215-382-9800 | FAX: +1-215-386-7999