Publishing Computational Mathematics

Tim Daly

Computational mathematics is a new science born of the collision of mathematics with computers. It is a new way of thinking about mathematics. The existing large computer algebra systems are the "Newton's notebooks" of this new field. They contain implementations of algorithms to compute exact symbolic results. For example, most have an implementation of the Risch algorithm for exact definite integration. This algorithm is a decision procedure that will compute the exact antiderivative of a function if it exists and can be expressed in elementary terms. If it cannot, then the failure of the algorithm is a proof that the antiderivative does not exist.

This note is not about the mathematics of the algorithm but about the implementation. This is equivalent to the difference between computability and complexity. The mathematics may be right, but the implementation may be wrong in many ways. Future results will depend on the implementation, as that is the heart of these computer algebra systems. Some areas of mathematics need to rely on machine results because the algorithms cannot be done by hand. But are they correct?

Suppose your research relies on one of these systems for a result. What are the standards of citation? If you have implemented an algorithm to solve a problem, what are the standards of documentation? What does it mean to publish a "proof" of an algorithm in computational mathematics? This is a vital point that is critical for placing computational mathematics on a solid foundation.

Currently, algorithms are described using mathematical notation and, if necessary, pseudocode to illustrate the basic idea. We are told that the algorithm was "implemented" in some system; that it is "better" by some measure of space, time, or breadth; and presented with tables of measurements as "proof". In my opinion, this is pretty but it is not mathematics. It fits the "hand waving" standards of the nineteenth century but not the rigor of the twentieth century.

A viable standard of publication has several requirements. One requirement is the ability to fully understand and reproduce the results. Another requirement is developing a body of reference standards. A third requirement is a paper trail of citations.

Consider the requirement about reproducing the work. Mathematics depends on reviewers to examine publications. Often these reviews uncover mistakes or missing elements in a proof, keeping invalid results out of the literature. What standards should apply to a review of an algorithm?

Clearly, the details of the algorithm are important. No implementation stands alone, as it uses the facilities of existing systems. A journal reviewer needs access to the actual code that claims to implement the algorithm. There is no other way to review the results.

Consider the requirement of reference standards. The Digital Library of Mathematical Functions (DLMF) publishes equations that define functions, like the gamma function, in many different

Tim Daly is the lead developer for the Axiom software project. His email address is daly@axiom-developer.org.

Members of the Editorial Board for Scripta Manent *are Jon Borwein, Thierry Bouche, John Ewing, Andrew Odlyzko, Ann Okerson.*

DOI: http://dx.doi.org/10.1090/noti797

forms. One can find series expansions, or continued fraction expansions, or many others. Try to find a reference standard for an implementation of the gamma function. There is only a table listing the existing computer algebra systems. It seems reasonable to expect that a "digital" reference standard ought to include a reference implementation. Some of these systems are forty years old. Which of these twenty-one implementations is "the reference"?

Consider the requirement of a paper trail of citations. When using algorithms to, say, factor polynomials, what should be the citation? Is it acceptable to "hand wave" at one of these existing systems? Can the reviewer follow the citation, open that algorithm, find the supporting publication, and verify that it handles the cases presented? Clearly not.

This is not an acceptable basis for computational mathematics.

Computational mathematics rests on software. In order to reproduce the results, make reference standards, and create a citation trail, we need to establish ever higher requirements for publication. This is not general computer science; this is mathematics in a new age.

It is technically possible to create a publication format that includes not only the mathematics but the actual implementation. Such a format, called a "literate program", was proposed by Donald Knuth. A literate program contains the detailed documentation and the source code. It is possible to extract the actual, runnable code from the "paper".

Literate programs could change the field. It would be possible to attend a lecture, download the paper as a literate document, and execute it during the talk. Imagine validating the claimed performance by reproducing the results in real time. Imagine testing boundary conditions while the talk is in progress. Imagine following a citation, downloading the literate document, and finding that the citation does not apply. New, more general algorithms with details of limitations and boundary conditions could be collected in the "Software Index" of the DLMF.

This is an acceptable basis for computational mathematics.

There are many objections that could be raised.

Mathematicians do not write code. But computational mathematicians do, and, as I claim, this is a new science. Code is the "coin of the realm".

Current journals would not accept a 300-page paper. This is based on limitations of paper publication, but we live in a digital age. There are no limits to the size of an electronic publication. Perhaps we need a *Literate Journal for Computational Mathematics*.

Current citations which contain source code do not exist. We have to begin to create this body of work. It will take time, and early papers will suffer from rising standards, just as early "proofs" would not meet the standards of today.

There are no language standards for computational mathematics. This is a sign of immaturity in the field. If you read mathematics papers from prior centuries, they are difficult to follow, as they make up notation "on the fly" to suit the problem. Newton and Leibniz had this same discussion, and we still accept either notation.

Implementations include details that have nothing to do with the theory. This is misguided. The implementation details matter a lot. Numerical software shows that details can make or break an algorithm. Whole areas of mathematics, such as numerical analysis, depend on the details.

Algorithm implementations are proprietary. This needs to end. Science is not done behind a curtain. At least, it has not been hidden since Tartaglia and Cardano fought over solving the cubic.

A firm foundation of science rests on fully understandable and reproducible results, reference standards, and citations. We must develop a "proof" standard for publication that goes to the heart of the subject, namely the implementation.

Computational mathematics needs literate software.

As a point of full disclosure, I am one of the original IBM authors of Axiom as well as the lead developer of the open source version. My interest in literate programming stems from the fact that I have personal experience trying to maintain and document hundreds of thousands of lines of computational mathematics. A survey paper discussing issues in greater detail will be posted on the Axiom website `http://axiom-developer.org`.