Scientific Programming 14 (2006) 41–42
IOS Press

Book Review

Accuracy and Reliability in Scientific Computing, by Bo Einarsson, ed., SIAM, Philadelphia, USA, 2005. ISBN: 0 89871 584 9

This book is a collection of 13 articles dealing with accuracy and reliability of scientific computing. Bo Einarsson of Linkoping University in Sweden edited the volume, but there are numerous contributing authors, many of whom I will individually name below since I believe they should receive recognition for their contributions. One very useful feature of the book is an accompanying website containing updates, computer codes and other material: http://www.nsc.liu.se/wg25/book.

The overall theme of the book is the increasingly critical problem of reliability in technical computing. These issues are coming to the fore in part because of the enormous scale of many present-day computations, typically running on hundreds or even thousands of CPUs, and involving many terabytes of data. Such large-scale computations greatly magnify many types of error conditions, ranging from numerical round-off errors and inadequate grid resolution to old-fashioned coding errors. Given the scarcity of highly expert professionals in this area, there is a compelling need to design software, systems and application programs with a significantly greater level of built-in reliability.

The editor Bo Einarsson leads off the volume by listing a number of the classic examples of disastrous results from dealing improperly with the realities of error in technical computing. These include the 1991 Patriot missile malfunction (after 100 hours of operation the projected positions of a Scud missile were off by 573 meters due to round-off error), and the 1991 Norwegian oil platform collapse (shear stress was underestimated by 47% because of misusing some finite element software).

In Chapter 2, Ronald Boisvert, Ronald Cools and Bo Einarsson present an encyclopedic listing of the many different types of errors that can arise, and then mention some specific techniques for code verification (test suites, etc.). In Chapter 3, Cools presents a case study of the potential pitfalls of numerical calculation in the context of a simple example – using the Gauss-Legendre numerical integration facility in Matlab to find the maximum of a parameterized function. In Chapter 4, Sven Hammarling presents a tutorial on condition numbers, stability and error analysis. In Chapter 5, Francoise Chaitin-Chatelin and Elisabeth Traviesas-Cassan discuss the issue of the reliability of scientific computation from a more fundamental perspective, discussing issues such as exact versus inexact computation and effective computability. These two authors continue in Chapter 6 with a discussion of the PRECISE package, which is a toolbox designed to assess the quality of numerical software in scientific and engineering applications. In Chapter 7, Wayne Enright discusses tools for verification of ordinary differential equation software.

Chapter 8 addresses language issues related to scientific computing and reliability. Individual subsections target specific languages, discussing features such as error handling, array layout, dynamic data, user-defined data structures and operator overloading. Brian Wichmann and Kenneth Dritz handle the Ada subsection. Craig Douglas and Hans Petter Langtangen discuss C, C++ and Python. Van Snyder addreses Fortran, including Fortran-77, Fortran-90, Fortran-95 and Fortran-2003. Ronald Boisvert and Roldan Pozo discuss Java. I for one particularly appreciated the frank discussion of various limitations of these languages for scientific computing. For example, Boisvert and Pozo candidly note Java's lack of a complex datatype and multidimensional arrays.

The next two chapters focus on interval arithmetic. In Chapter 9, William Walster describes in considerable detail the actual usage and implementation of interval datatypes. In Chapter 10, Siegfried Rump discusses computer-assisted proofs and self-validating methods, and their connection to interval arithmetic. One particularly interesting note in this chapter is the mention of a "theorem" that was included for some time in textbooks 100 years ago, with the proof dismissed as "obvious," but which was later shown to be false by any of several simple counter-examples. Rump emphasizes that even conventional mathematical proofs involve, to

ISSN 1058-9244/06/$17.00

a large degree, mutual trust among mathematicians that qualified researchers have carefully worked through the proof and are convinced that it is sound. In a similar vein, the technical computing field needs to establish standards of verification that mathematicians and computer scientists collectively agree are effective tests of program soundness.

The book draws to a close with chapters on hardware-assisted algorithms (by Douglas and Lantangen), issues of reliable computing in a parallel computing environment (by William Gropp), and software reliability engineering (by Mladen Vouk).

In general, this is an excellent collection of articles on the topic of reliable scientific computing. Several of the individual chapters are by themselves worth the purchase price. Many persons have worked hard in preparing this work, and are to be congratulated for the quality of the final product.

My only disappointment in reading this material is that there was no significant space given to high-precision computation as a means to detect and rectify numerical difficulties. From my experience, this is the most straightforward solution to this class of problems. There are certainly numerous qualified authors who could have written material in this area. And with any of several readily available software packages, it is quite easy to convert even fairly large codes, particularly in Fortran, C or C++, to perform many or all floating-point operations using some form of higher-precision arithmetic. Double-double (approximately 31 base-10 digits), quad-double (approximately 62 digits) or even arbitrary precision packages are available, and, in most cases, conversion is facilitated by means of translation modules that employ operator overloading and custom datatypes. For that matter, many vendor-supplied Fortran compilers have built-in support for real*16 arithmetic, and many C compilers support a "long double" format. And if the major computer vendors can be coaxed into providing hardware support for the IEEE 128-bit floating-point standard in the widely used microprocessors, then the expansion factor in run time could be greatly reduced, from a typical factor of ten today, to possibly to only a factor of two or three.

In any event, this book is a very nice reference, one that presents in a very accessible and yet effective manner the daunting issues of reliability that we now face in scientific computing. It will do much to raise awareness of these issues and focus attention on what needs to be done all across the scientific computing culture to improve the reliability of our technology.

Hopefully some day we will not only have hardware and software that is inherently more reliable, but even application programmer will recognize the need to include extensive built-in validity checks, recognizing that their code may be used for many years by persons who do not fully appreciate the many ways in which it can be go wrong. When that day comes, we can thank the authors of this book for their efforts in helping to make it happen.

David H. Bailey
Lawrence Berkeley National Laboratory
Berkeley, CA, USA