Preface

Modern computing environments support many different tools and languages. We have at our disposal powerful microcomputers, workstations, parallel processors, graphics displays, special-purpose hardware, database tools, plotting packages, symbolic computation languages, and a wide variety of numerical toolboxes. These advances have combined somewhat haphazardly. Nevertheless there is no doubt that the paradigms in scientific computation are evolving. A mathematical model can now be interactively prototyped and explored using symbolic and graphical techniques, and once confidence and understanding has been gained, the model can be transformed into an efficient and robust computational solution. Most researchers take for granted these new tools, but our undergraduate curricula have been slow to change.

This book is an introduction to mathematical representation and computation. We shall draw from a broad array of techniques from scientific, symbolic, and graphical computation. A basic course in single-variable calculus is the only real prerequisite (or corequisite), although some programming experience would be helpful. The material is intended to be accessible to undergraduates at all levels, but it is intended to appeal in different ways depending on needs and interests. It is targeted at all students in the pure and applied sciences, mathematics, and computer science who may be interested in seeing how computation interacts with a continuous world.

The choice of material for this book was (and remains) problematic: the number of topics from which to choose is huge, the literature even larger, and the mathematical foundations can be daunting even to those of us who supposedly should know better. One way to address this diversity is to choose a simple theme and follow some of its paths through various areas of mathematical computation. While we cannot cover all the territory underfoot, we may at least get a sense of the lie of the land. Accordingly, most of the material revolves around the following general questions:

(a) How can we represent a lot by a little?

(b) How can we do a lot with a little?

The first deals with the compact representation of a thing, while the second deals with its efficient computation. These two conundrums will motivate topics such as implicit and parametric representations, interpolation, approximation, sampling, filtering, reconstruction, and integration. Along the way, we shall discuss global and piecewise polynomial interpolation, splines, polynomial and Fourier series, filters, quadrature, the sampling theorem, and the solution of non-linear equations.

Because of the minimal curriculum requirements assumed by this book, namely first-year calculus, but not linear algebra, we devote very little space to problems that are best solved using general matrix formulations. It may therefore be surprising to some that we do not consider problems such as the solution to linear systems of equations (except very briefly in a simple setting), condition numbers, matrix decomposition, eigenvalues, and so on. A tempting case for an introductory book based only on linear-algebraic concepts can be made. I have also avoided the great temptation of grounding this material in one or more of the sciences, opting instead for a general core of ideas that leaves instructors free to ground the techniques to their own specialities.

There are three central beliefs to this book. The first is that graphical and symbolic exploration of concepts is crucial to understanding, and that today's computing environments provide us with sufficiently good tools to learn by doing. The second is that, largely because of the first belief, this material is suitable for first- or second-year undergraduates without significant training in computer science or mathematics, despite the fact that this book is very much about both, and that the technical content of the material we cover is of a high level. Since much of the work in elementary scientific computation and computer graphics is developed from the basic notion of the derivative as learned in high school, I believe that this material works best on students with the definitions of first-principles calculus still fresh in their minds. The success of this approach depends strongly on the availability of workstations or personal computers for students to do lab work. The third belief is that approximation theory (including, of course, interpolation) and signal theory are two sides of the same coin. I am quite mystified as to why students tend to see one topic but not the other, depending on the undergraduate programme the student has chosen to enter. A numerical analyst should understand why a CD player works, and a signal theorist should know why parametric representations are wonderful for modelling teapots!

Deciding among the tools to discuss in this book was almost as difficult as the choice of material. There are many excellent graphics systems, plotting packages, algebraic and symbolic mathematics languages, and scientific toolkits in existence. To ignore them is to ignore the current reality, but neither is it possible to discuss even the main systems. Our programming examples therefore will be illustrated using the Turing and C programming languages, and our symbolic mathematics system will be Maple. Nothing in our presentation actually depends on this choice, and it is quite straightforward to substitute another programming language such as FORTRAN, Pascal, or Modula, or to use another symbolic mathematics system such as Mathematica. Indeed, we expect that our electronic software repository (see below) will eventually support most of these languages.

We cannot be exhaustive in this book. We do not have the time to prove theorems or even be particularly rigorous. This is the domain of more advanced courses in several areas. Our main aim is to introduce some of the problems associated with converting mathematical models into computational solutions. The simple statement of the aim belies its difficulty, especially if we were to derive general solutions. On the other hand, our coverage of some topics will be surprisingly deep and will be good preparation for senior courses in scientific and mathematical computation, computer graphics, image processing, computer vision, and signals and systems.

Most of the exercises should be worked through as part of the reading process, and working through them using a symbolic mathematics system is encouraged. A few of the exercises involve a longer stay at a workstation. The ordering of the chapters beyond Chapter 2 is due more to taste than to dependencies. Potential instructors may be surprised to see that floating-point representations are not formally presented until Chapter 5. If students have had prior exposure to series representations from calculus, then this material could be presented earlier, although it is not essential to do so.

Acknowledgements

I am grateful to Wayne Enright, Sherif Ghali, Michael McCool, and Chris Williams, for reading, criticising and improving this book. Earlier drafts of this book were inflicted on the students of CSC160 and CSC260 at the University of Toronto, and many students took the time offer suggestions and corrections. I thank them for their help and patience. Allan Borodin encouraged me to design a course that got this whole thing started; blame me for the implementation. Alain Fournier has inspired me in many ways, but in this book he has specifically inspired my treatment of polynomials and parameterisation in Chapters 1 and 2. The gang at AK Peters, namely Alice and Klaus Peters, Alexandra Benis and Joni McDonald have been a treat to work with.

Suggestions for Further Reading

The main reason for the existence of this book is that there are few, if any, books on mathematical computation that are directed at the beginning undergraduate. However, a great many useful, excellent, but more specialised books exist at a higher level. The list below is certainly not exhaustive, and is largely an indication of my influences and tastes.

Advanced Calculus and Real Analysis

These are both excellent introductions to real analysis. The first has a particularly attractive discussion of Fourier series.

J.E. Marsden, *Elementary Classical Analysis*, W.H. Freeman and Co., San Francisco, 1974.

W. Rudin., *Principles of Mathematical Analysis*, Second Edition, McGraw-Hill, New York, 1964.

Scientific Computation

The first is a good introduction to many of the issues in scientific computation, while the second is thorough treatment of quadrature, including Monte Carlo integration.

D. Kahaner, C.M. Moler, and S. Nash, *Numerical Methods and Software*, Prentice-Hall, New York, 1989.

P. Davis and P. Rabinowitz, *Methods of Numerical Integration*, Second Edition, Academic Press, New York, 1984.

Approximation Using Polynomials

The first two books are standard texts on polynomial curves and surfaces as relevant to computer-aided geometric design and computer graphics. The third is a comprehensive and remarkably well written presentation of similar material that is my current favourite. The last reference is an advanced but accessible presentation of approximation methods for many problems in scientific computation.

R. Bartels, J. Beatty and B. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann, Los Altos, CA, 1987.

G. Farin, *Curves and Surfaces for Computer-Aided Geometric Design* Third Edition, Academic Press, Boston, 1993.

J. Hoschek and D. Lasser (translated by L.L. Schumaker), *Fundamentals of Computer-Aided Geometric Design*, A.K. Peters, Wellesley, MA, 1993.

P. Prenter, *Splines and Variational Methods*, Wiley, New York, 1975.

Signal Theory and Processing

The first book is an authoritative introduction to the theory and processing of general signals. The second is an advanced treatment of digital image processing. Both are wonderful books.

A.V. Oppenheim, A.S. Willsky, and I.T. Young, *Signals and Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

W. Pratt, *Digital Image Processing*, Second Edition, John Wiley and Sons, New York, 1990.

Language References

Useful books containing illustrative examples of the use of symbolic programming languages are appearing almost daily. While Maple is used extensively in this book, the material does not depend on this choice for a symbolic computation language. Nevertheless, two references are included in this list. Note that the book by Burbulla and Dodson has an equivalent version for Mathematica. Introductions to Mathematica and to MATLAB are also listed.

N. Blachman, *Mathematica: A Practical Approach*, Prentice-Hall, Englewood Cliffs, NJ, 1992.

D.C.M. Burbulla and C.T.J. Dodson, *Self-Tutor for Computer Calculus using Maple*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

B. Char *et al.*, *Maple V Language Reference Manual*, Springer-Verlag, New York, 1992.

The Math Works, Inc., *The Student Edition of MATLAB*, Prentice-Hall, Englewood-Cliffs, NJ, 1992.

Internet Connectivity, or How You Can Make a Difference

One of the remarkable outgrowths of the Internet is the increasing availability of electronic mechanisms for information dissemination. A repository of exercises, sample software, and applications exists. A separate repository of solutions is also available for instructors, as is an initial distribution of software for courseware. Instructors and students alike are welcome to contribute to the repository. An information page can be found via the World-Wide Web at

http://www.dgp.toronto.edu/people/elf/elf.html

or by FTP at

ftp:dgp.toronto.edu:pub/elf/README.

The author’s e-mail address is

elf@dgp.toronto.edu.

A synonym for dgp.toronto.edu is dgp.utoronto.ca.