LESSONS IN SCIENTIFIC COMPUTING

Norbert Schörghofer

Last updated December 9, 2015

Contents

About This Document

Fundamental scientific discoveries have been made with the help of computational methods. For example, commonalities in the behavior of chaotic systems, most prominently Feigenbaum universality, have been discovered by means of numerical calculations. This required only simple and small computations. An example at the opposite extreme, using large and complex computations, is the prediction of the mass of the proton from fundamental theories of matter; only with numerical computations was it possible to evaluate the theoretical expressions sufficiently accurate for comparison with experimental measurement. Such examples highlight the crucial role of numerical calculations in basic science.

Even before the dawn of electronic computers, to name but one example, Milutin Milankovitch spent one hundred full days, from morning until night, with paper and fountain pen calculations on one of his investigations into Ice Ages. Today, the same computation can be done in seconds with electronic computers. Even small numerical calculations can solve problems not easily accessible with mathematical theory.

While today's students and researchers are trained in the theoretical, experimental, and observational methods of their field, comparatively little educational literature is available about the computational branch of scientific inquiry. Moreover, the rapid advances in computer technology have changed the way we do scientific computing. As a consequence of such developments, part of the conventional wisdom about computing has become outdated.

This book is a collection of concise writings that expose the reader to a wide range of approaches, conceptional ideas, and practical issues. Often when working on the manuscript, it grew shorter, because less relevant material was discarded. It is written with an eye on usefulness, longevity, and breadth. Short-lived but useful pointers to specific resources are listed toward the end of each chapter or section.

The book takes a broad and interdisciplinary approach. Numerical methods, computer technology, and their interconnections are treated with the goal to facilitate scientific research. Viewpoints from several subject areas are often compounded within a single chapter.

The book can serve as supplementary reading material for upper-level college and graduate classes in computational physics or scientific computing in general. It can also be used as sole textbook for a one hour per week graduate class. Although the book is focused on physicis, all but a few chapters are accessible to and relevant for a much broader audience in the physical sciences. Sections with a star symbol (*) are specifically intended for physicists and chemists. For better readability, references and footnotes within the text are entirely omitted.

I am indebted to my teachers and colleagues, and to many textbook authors who have, knowingly or unknowingly, contributed to the substance of this book. They include Prof. Leo Kadanoff, who taught me why we compute, Prof. James Sethian, on whose lectures chapter 15 is based on, and the authors of “Numerical Recipes”, to name but a few. Many friends and colleagues have provided valuable feedback on draft versions of this material.

Norbert Schörghofer

Honolulu, Hawaii
December, 2015

To the Instructor

This material is intended as a modernized, broad, and compact introduction into scientific computing. It is appropriate as main textbook or as supplementary reading in upper-level college and in graduate courses on scientific computing or computational physics, astrophysics, chemistry, meteorology, etc. (the hard sciences). Prerequisites are basic calculus, linear algebra, and introductory physics. Prior knowledge of numerical analysis and a programming language are optional, although students will have to pick up programming skills during the course. In a 1-credit (1 academic hour per week) graduate course, I covered 12 of the 16 chapters. A few homework problems are posted on the website, not yet enough to cover most possible variants of the course.

Instead of slides, I use the pdf version of the manuscript to display figures and tables. The lectures involved interactive exercises with a Mac computer connected to the same LCD projector. These interactive exercises are mentioned in the text and based on files in the Programs folder. Each chapter is designed to cover about one academic hour, but chapter 7 is too long, and if optional material is left out, several other chapters are too short.

The content of the book can be roughly divided into two parts. The first half deals with small computations. Chapter 1 illustrates how insight can be gained from numerical calculations, introducing binary floating-point numbers along the way. The next three chapters provide an overview of numerical analysis, go into the nature of numbers on computers, and discuss programming languages, which readies the student to carry out the sample problem in chapter 5. Examples and homework try to establish the process of computational research as routine: problem setup, program implementation, testing, analysis of results, and iteration of these steps. The subsequent chapter elaborates on elements of this process, and forms a gradual transition into the second part of the book that deals mainly with large computations. Performance is discussed first on a hardware level, which then leads into the classical operation count of numerical analysis. Topics discussed next are random numbers, algorithms, and symbolic computations. Topics about data are collected in one chapter, followed by two chapters on partial differential equations applied to fluids and quantum systems. By the end, the student has been exposed to a wide range of methods, concepts, and practical material. Even some graduate level physics is introduced in optional sections, such as almost integrable systems, diagrammatic perturbation expansions, and density functionals.