Preface

The language of science and engineering is largely mathematical, which, increasingly, requires solving problems that are described by ordinary differential equations (ODEs) and partial differential equations (PDEs). The primary focus of this book is numerical solutions to initial value problems (IVPs) and boundary value problems (BVPs) described by ODEs and PDEs. The solutions are implemented in computer code using the open source R language system.

The intended readership is senior undergraduates and postgraduate students in the subject areas of science, technology, engineering, and mathematics (STEM). The contents should also appeal to engineers and scientists in industry who need practical solutions to real-world problems. The emphasis is on understanding the basic principles of the methods discussed and how they can be implemented in computer code.

The aim of this book is to provide a set of software tools that implement numerical methods that can be applied to a broad spectrum of differential equation problems. Each chapter includes a set of references that provide additional information and insight into the methods and procedures employed. All chapters are more or less complete in themselves, except for a few references to other chapters. Thus each chapter can be studied independently.

It is assumed that the reader has a basic understanding of the R language, although the computer code is annotated to a level that should make understanding clear. Additional discussion is included in the text for more advanced language constructs. Some basic examples of the use of computer algebra systems are also included that make use of the R interface packages `Ryacas` and `rSymPy`.

R is a free, high-level software programming language and software environment that has traditionally been used for statistical computing and graphics. It has been widely used for many years by statisticians for data analysis being particularly effective in handling large data sets. However, recently, packages have been added to R to solve a wider range of numerical problems. In particular, the addition of package `deSolve` [Soe-10] opened up the language for solving differential equations by adding industrial strength integrators. The package `deSolve` is used extensively in the R examples provided in this book. For readers wishing to learn about R, *The Art of R Programming* [Mat-11] is a good introductory text, and *The R Book* [Cra-11] is a comprehensive description of the

R language. Both are highly recommended. The R language can be downloaded from http://cran.r-project.org/. To supplement the R installation, a free interactive development environment (IDE), *RStudio*, can be downloaded from https://www.rstudio.com/. RStudio provides a very user-friendly customizable user interface that greatly enhances programming productivity and graphical display of simulation results. For users familiar with Matlab, a comprehensive cross-reference between Matlab and R commands has been compiled by Dr. David Hiebeler of the University of Maine and can be downloaded from http://www.math.umaine.edu/~hiebeler.

We emphasize two areas of style usage that have been adopted in this book. The first is that the symbol used for *assignment* is `<-` (also known as the `gets` symbol), which is the R purist's form. However, it is acknowledged that `=` is used extensively in the R literature and that choosing between `<-` and `=` could be considered simply a matter of semantics, as there are only a few special situations in which `=` does not work properly. The second area of style usage relates to *global variables*, that is, those variables created outside of a function but that can be used inside a function. Global variables created using the *super-assignment* symbol `<<-` can also be changed within a lower environment function using `<<-`. This means that, in general, they are allocated higher in the *environment hierarchy* than where they are used. Now, like all modern programming languages, variables in R are passed to functions by reference, but global variables are allowed. In this book, we tend to make great use of global variables, as such use leads to cleaner and less cluttered code. It also avoids having to process lists returned from functions where, in the rare situation, globals are actually changed within a function rather than just used. However, we do not use global variables wherever possible and pass values through function arguments where it is appropriate or where it makes the code clearer. For applications described in this book, the scope for global variables is generally between adjacent levels in the environment hierarchy and therefore is not readily prone to programming conflicts. A good discussion relating to the use of global variables can be found in [Mat-11, chap. 7].

OVERVIEW OF CONTENTS

Chapter 1 introduces the subject of ordinary differential equations, and basic integration methods are covered, including Euler, Runge–Kutta, variable step, extrapolation, BDFs, NDFs, and Adams. The Newton and Levenberg–Marquardt convergence methods are introduced in connection with implicit integration. A discussion relating to truncation error and verification of integration order is presented with a number of examples that use the concepts discussed. The chapter concludes with a brief discussion on stiffness.

Chapter 2 explores some of the theory and practical applications of ODE integrator stability analysis. The discussion includes topics on global order of accuracy, A-stable and zero-stable definitions, and the Dahlquist barrier theorems. Using the Dahlquist test problem, we investigate the stability of various integration methods, including Runge–Kutta, variable step, BDFs, NDFs, and Adams. For each method, the stability regions are plotted in the complex plane to illustrate the associated stability margins.

Chapter 3 discusses Cauchy problems in the context of initial value and/or boundary value problems. The basic concepts relating to PDEs and their classification are discussed, along with initial conditions (ICs) and the various types of boundary conditions (BCs). General ideas relating to discretization methods, mesh grid, stencils, upwinding,

and the Courant–Friedrichs–Lewy number are introduced. This leads into a discussion on the method of lines (MOL), one of the major numerical schemes for solving PDEs. A good selection of 1D and 2D example MOL problems are solved. These examples, in both Cartesian and polar coordinates, have been chosen to bring out the main ideas and to show the variety of problems that can be solved using this method. Then follows a brief discussion of the following fully discrete methods: FTBS, implicit FTBS, FTCS, implicit BTCS, leapfrog, Beam–Warming, Lax–Friedrichs, and Lax–Wendroff. Solutions are compared for the different methods applied to sample problems. The ideas behind the finite volume method are presented in preparation for their use in high-resolution schemes, which are discussed subsequently in Chapter 6.

Chapter 4 discusses the concept of a system being *well-posed* and ways of using this idea to investigate system stability. The matrix stability method is outlined, and detailed stability calculations are included for semi-discrete PDE schemes. A number of examples are included that analyze PDEs, and results are presented in graphical form. The von Neumann stability method is discussed with the fundamental ideas developed using Fourier transforms. Various semi-discrete and fully discrete PDE systems are analyzed, with results presented graphically.

Chapter 5 introduces the idea of the *dispersion relation* and its relationship to wave number and wave frequency. The accuracy of numerical schemes is discussed in relation to numerical amplification and exact amplification factors. Also discussed is how these factors can be used to provide an indication of phase (dispersion) and amplitude (dissipation) errors. The chapter concludes with a discussion on phase and group velocities.

Chapter 6 introduces the idea of *high-resolution* schemes for the solution of PDEs. The Riemann problem is discussed, along with Godunov's method of providing an approximate solution. A discussion follows on the principle of total variation diminishing (TVD) and how Godunov's order barrier theorem places constraints on monotonic PDE solution methods. An introduction is then provided to two major methods that are employed widely to solve these types of problems. The first is the flux limiter method, whereby fifteen different flux limiter functions are presented. The second is the weighted essentially nonoscilliatory (WENO) method, whereby the associated weights and smoothness indicator calculations are presented. These methods are discussed within a finite volume and MOL framework. They are based on the monotone upstream-centered schemes for conservation laws (MUSCL) method, with the particular implementation being the Kurganov and Tadmor central scheme. A variety of 1D and 2D problems are solved using these methods, and the results are presented graphically. These include, advection, Burgers, Buckley–Everett, Euler equations, Sod's shock tube, Taylor–Sedov detonation, Woodward–Colella interacting blast wave, and frontogenesis: all computationally demanding PDE evolution problems.

Chapter 7 introduces the concept of *meshless methods* using radial basis functions (RBFs). These methods represent important tools for the numerical analyst and are becoming very popular for solving otherwise difficult problems. One of the main advantages of meshless methods is that they can be used on irregular grids and therefore are applicable to problem geometries of any shape. The ideas are presented from an introductory-basics level, with examples showing how the method is used to interpolate scattered data and also for solving partial differential equations. For a number of examples, the results are compared to analytical solutions to demonstrate the accuracy of the

results obtained. The Halton sequence, which produces pseudo-random data, is introduced to demonstrate how the method readily handles irregular grids and/or scattered data. A number of globally and compactly supported RBFs are defined and their use illustrated with examples in 1D, 2D, and 3D. This chapter also includes discussion on the use of local RBFs, which allow the method to be used on very large problems. Local RBFs result in the system matrices becoming sparse, which facilitates the application of sparse matrix methods, which are provided by the R package `Matrix`.

Chapter 8 introduces the concept of *conservation laws* in the context of evolutionary PDEs. Under the assumption of certain decay conditions, it is shown how conserved quantities can be identified for particular PDEs. It is then shown that conserved quantities can be used to calculate associated constants of motion or invariants. These constants of motion are very useful in numerical analysis as they can be used to provide an indication of calculation accuracy. The ideas are discussed mainly in terms of the 1D Korteweg-de Vries (KdV) equation, where a number of the commonly known conservation laws applicable to this equation are derived. It is then shown using the Miura and Gardener transformations that, actually, the KdV equation possesses an infinity of conservation laws. The discussion is then extended to the 2D KdV equation and then to the KdV equation with variable coefficients (vcKdV). This chapter also includes conservation law discussions and example calculations in relation to the nonlinear Schrödinger (NLS) equation and the Boussinesq equation.

Chapter 9 is a case study into the analysis of a golf ball in fight. It has been the subject of many theoretical investigations, with some of the earliest being published in the late 1890s. This case study provides an in-depth study of this subject, with computer simulation results presented and compared to published performance data. The various forces acting on a golf ball in flight are discussed, namely, drag, Magnus, and gravitational. The differential equations that describe the golf ball flight are then derived. The effects of compression, spin, ambient conditions, wind, launch angle, bounce, and roll are all taken into account in the simulation calculations. The latest coefficients from the literature for drag and lift are used. The results are presented graphically and compared with measured statistics for club head speed, carry, and trajectory height for various classes of player, using different golf clubs. The effect of push, pull, fade, draw, slice, and hook shots are investigated, and the results are discussed. The magnitude of wind shear at ground level is calculated, and its effect on a static golf ball is investigated.

Chapter 10 is a case study into the problem of an intense explosion. In 1945, Sir Geoffrey Ingram Taylor was asked by the British *Military Application of Uranium Detonation* (MAUD) Committee to deduce information regarding the power of the first atomic explosion at the *Trinity site* in the New Mexico desert. He was able to estimate, using only public domain photographs of the blast, that the yield of the bomb was equivalent to between 16.8 and 23.7 kilotons of TNT. This case study discusses the subsequent seminal papers that Ingram published and traces the perceptive calculations that he made. A systems analysis starts with a form of the Euler equations, from which, using similarity analysis, certain important relationships are deduced. With the aid of a set of high-speed photographs of the detonation, and assuming spherical symmetry, the underlying characteristics of the blast are gradually revealed by a sequence of thermodynamic calculations. The thermodynamic gas laws required to unravel this puzzle are discussed, along with shock wave analysis using the Hugonoit–Rankine relations. Kinetic energy

and heat energy integrals are evaluated, leading to a full description of the blast. This includes blast wave speed, pressure, and temperature over time. Closed-form analytical solutions to the problem, subsequently published by Sedov in 1959, are presented that provide a useful check on the accuracy of Ingram's numerical calculations. A similarity analysis is included, covering spherical, cylindrical, and planar blast situations.

Chapter 11 is a case study into the *global carbon cycle* and how increased concentrations in atmospheric carbon dioxide have implications for climate change. A simplified model is presented that considers four fossil-fuel emission scenarios based on the work of Caldeira and Wickett and how the atmosphere and oceans respond. Air–ocean interaction is modeled using the wind-driven gas–sea exchange relationship due to Wanninkhof, with the subsequent dispersion of gaseous CO_2 into $CO_2(aq)$, HCO_3^-, CO_3^{2-}, and H^+ in accordance with carbonate chemistry equilibria. It is shown how an increasing concentration of positive hydrogen ions causes seawater acidity to rise, that is, a fall in pH. Seawater buffering calculations are also introduced to demonstrate how the ocean's ability to absorb atmospheric CO_2 is diminished as seawater concentration of dissolved inorganic carbon increases. A discussion of solar and terrestrial radiation modeling is also included, along with calculations that show how increasing CO_2 concentrations in the atmosphere can lead to increases in the Earth's surface temperature, that is, to the so-called greenhouse effect.

All chapters include worked examples, many of which generate animations, along with annotated computer code, which is available for download. As an additional resource for some chapters, computer code is provided with the downloads for symbolic computer algebra analysis using Maple and Maxima/wxMaxima systems. Maxima is a free open-source program available for different operating systems that can be downloaded from http://maxima.sourceforge.net/. A GUI-based version, wxMaxima, can be downloaded from http://andrejv.github.io/wxmaxima/.

To maximize the benefit of studying numerical computing, the learning process should not be regarded as a passive pursuit. Rather, it should be regarded as a *hands-on* experimental activity. Because many problems are nonlinear, analytical or well-tried and tested solutions may not exist. In these situations it may be necessary to try various different options to find the most appropriate solution technique, with the best parameter values being deduced by trial and error. This book includes many and varied solutions to a wide range of problems, and it is hoped that the reader will find some that can be applied directly, or adapted, to the particular problem of interest.

To conclude, the major focus of this book is the numerical solution of initial value problems (IVPs) and boundary value problems (BVPs) described by ODEs and PDEs. The general approach and content complement the author's earlier books ([Gri-11] and [Sch-09]) and also the excellent book *Solving Differential Equations in R* [Soe-12].

ACKNOWLEDGMENTS

I would like to thank Professor William (Bill) E. Schiesser of Lehigh University for collaboration in the area of numerical solutions to differential equations over many years and for suggesting that I look into using the R language for solving ODEs and PDEs; Professor Alan M. Nathan of the University of Illinois for useful discussions on the flight of projectiles and for clarifying some concepts; and Professor Duncan Murdoch of the

University of Western Ontario for advice on using the R package `rgl` and for responding positively to requests for changes to the 3D graphics function `persp3d()`.

Graham W. Griffiths
Nayland, Suffolk, United Kingdom
June 2015
graham@griffiths1.com
www.pdecomp.net

REFERENCES

[Cra-11] Crawley, M. J. (2011), *The R Book*, John Wiley.
[Gri-11] Griffiths, G. W. and W. E. Schiesser (2011), *Traveling Wave Solutions of Partial Differential Equations: Numerical and Analytical Methods with Matlab and Maple*, Academic Press.
[Mat-11] Matloff, N. (2011), *The Art of R Programming*, No Starch Press.
[Sch-09] Schiesser, W. E. and G. W. Griffiths (2009), *A Compendium of Partial Differential Equation Models: Method of Lines Analysis with Matlab*, Cambridge University Press.
[Soe-10] Soetaert, K., T. Petzoldt and R. W. Setzer (2010), Solving Differential Equations in R: Package **deSolve**, *Journal of Statistical Software* **33**-9, 1–25.
[Soe-12] Soetaert, K., J. Cash and F. Mazzia (2012), *Solving Differential Equations in R*, Springer-Verlag.