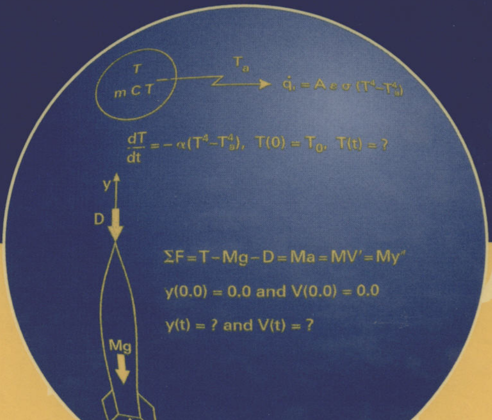
Numerical Methods for Engineers and Scientists

Second Edition
Revised and Expanded



Joe D. Hoffman

Numerical Methods for Engineers and Scientists

Second Edition
Revised and Expanded

Joe D. Hoffman
Department of Mechanical Engineering
Purdue University
West Lafayette, Indiana

MARCEL DEKKER, INC. NEW YORK · BASEL

The first edition of this book was published by McGraw-Hill, Inc. (New York, 1992).

ISBN: 0-8247-0443-6

This book is printed on acid-free paper.

Headquarters
Marcel Dekker, Inc.
270 Madison Avenue, New York, NY 10016
tel: 212-696-9000; fax: 212-685-4540

Eastern Hemisphere Distribution
Marcel Dekker AG
Hutgasse 4, Postfach 812, CH-4001 Basel, Switzerland
tel: 41-61-261-8482; fax: 41-61-261-8896

World Wide Web
http://www.dekker.com

The publisher offers discounts on this book when ordered in bulk quantities. For more information, write to Special Sales/Professional Marketing at the headquarters address above.

Current printing (last digit)
10 9 8 7 6 5 4 3 2 1

PRINTED IN THE UNITED STATES OF AMERICA

To Cynthia Louise Hoffman

Preface

The second edition of this book contains several major improvements over the first edition. Some of these improvements involve format and presentation philosophy, and some of the changes involve old material which has been deleted and new material which has been added.

Each chapter begins with a chapter table of contents. The first figure carries a sketch of the application used as the example problem in the chapter. Section 1 of each chapter is an introduction to the chapter, which discusses the example application, the general subject matter of the chapter, special features, and solution approaches. The objectives of the chapter are presented, and the organization of the chapter is illustrated pictorially. Each chapter ends with a summary section, which presents a list of recommendations, dos and don'ts, and a list of what *you should be able to do after studying the chapter*. This list is actually an itemization of what the student should have learned from the chapter. It serves as a list of objectives, a study guide, and a review guide for the chapter.

Chapter 0, Introduction, has been added to give a thorough introduction to the book and to present several fundamental concepts of relevance to the entire book.

Chapters 1 to 6, which comprise Part I, Basic Tools of Numerical Analysis, have been expanded to include more approaches for solving problems. Discussions of pitfalls of selected algorithms have been added where appropriate. Part I is suitable for second-semester sophomores or first-semester juniors through beginning graduate students.

Chapters 7 and 8, which comprise Part II, Ordinary Differential Equations, have been rewritten to get to the methods for solving problems more quickly, with less emphasis on theory. A new section presenting extrapolation methods has been added in Chapter 7. All of the material has been rewritten to flow more smoothly with less repetition and less theoretical background. Part II is suitable for juniors through graduate students.

Chapters 9 to 15 of the first edition, which comprised Part III, Partial Differential Equations, has been shortened considerably to only four chapters in the present edition. Chapter 9 introduces elliptic partial differential equations. Chapter 10 introduces parabolic partial differential equations, and Chapter 11 introduces hyperbolic partial differential equations. These three chapters are a major condensation of the material in Part III of the first edition. The material has been revised to flow more smoothly with less emphasis on theoretical background. A new chapter, Chapter 12, The Finite Element Method, has been added to present an introduction to that important method of solving differential equations.

A new section, Programs, has been added to each chapter. This section presents several FORTRAN programs for implementing the algorithms developed in each chapter to solve the example application for that chapter. The application subroutines are written in a form similar to pseudocode to facilitate the implementation of the algorithms in other programming languages.

More examples and more problems have been added throughout the book.

The overall objective of the second edition is to improve the presentation format and material content of the first edition in a manner that not only maintains but enhances the usefullness and ease of use of the first edition.

Many people have contributed to the writing of this book. All of the people acknowledged in the Preface to the First Edition are again acknowledged, especially my loving wife, Cynthia Louise Hoffman. My many graduate students provided much help and feedback, especially Drs. D. Hofer, R. Harwood, R. Moore, and R. Stwalley. Thanks, guys. All of the figures were prepared by Mr. Mark Bass. Thanks, Mark. Once again, my expert word processing specialist, Ms. Janice Napier, devoted herself unsparingly to this second edition. Thank you, Janice. Finally, I would like to acknowledge my colleague, Mr. B. J. Clark, Executive Acquisitions Editor at Marcel Dekker, Inc., for his encouragement and support during the preparation of both editions of this book.

Joe D. Hoffman

Contents

0

Introduction

This Introduction contains a brief description of the objectives, approach, and organization of the book. The philosophy behind the Examples, Programs, and Problems is discussed. Several years' experience with the first edition of the book has identified several simple, but significant, concepts which are relevant throughout the book, but the place to include them is not clear. These concepts, which are presented in this Introduction, include the definitions of significant digits, precision, accuracy, and errors, and a discussion of number representation. A brief description of software packages and libraries is presented. Last, the Taylor series and the Taylor polynomial, which are indispensable in developing and understanding many numerical algorithms, are presented and discussed.

0.1 OBJECTIVE AND APPROACH

The objective of this book is to introduce the engineer and scientist to numerical methods which can be used to solve mathematical problems arising in engineering and science that cannot be solved by exact methods. With the general accessibility of high-speed digital computers, it is now possible to obtain rapid and accurate solutions to many complex problems that face the engineer and scientist.

The approach taken is as follows:

1. Introduce a type of problem.

2. Present sufficient background to understand the problem and possible methods of solution.
3. Develop one or more numerical methods for solving the problem.
4. Illustrate the numerical methods with examples.

In most cases, the numerical methods presented to solve a particular problem proceed from simple methods to complex methods, which in many cases parallels the chronological development of the methods. Some poor methods and some bad methods, as well as good methods, are presented for pedagogical reasons. Why one method does not work is almost as important as why another method does work.

0.2 ORGANIZATION OF THE BOOK

The material in the book is divided into three main parts:

I. Basic Tools of Numerical Analysis
II. Ordinary Differential Equations
III. Partial Differential Equations

Part I considers many of the basic problems that arise in all branches of engineering and science. These problems include: solution of systems of linear algebraic equations, eigenproblems, solution of nonlinear equations, polynomial approximation and interpolation, numerical differentiation and difference formulas, and numerical integration. These topics are important both in their own right and as the foundation for Parts II and III.

Part II is devoted to the numerical solution of ordinary differential equations (ODEs). The general features of ODEs are discussed. The two classes of ODEs (i.e., initial-value ODEs and boundary-value ODEs) are introduced, and the two types of physical problems (i.e., propagation problems and equilibrium problems) are discussed. Numerous numerical methods for solving ODEs are presented.

Part III is devoted to the numerical solution of partial differential equations (PDEs). Some general features of PDEs are discussed. The three classes of PDEs (i.e., elliptic PDEs, parabolic PDEs, and hyperbolic PDEs) are introduced, and the two types of physical problems (i.e., equilibrium problems and propagation problems) are discussed. Several model PDEs are presented. Numerous numerical methods for solving the model PDEs are presented.

The material presented in this book is an introduction to numerical methods. Many practical problems can be solved by the methods presented here. Many other practical problems require other or more advanced numerical methods. Mastery of the material presented in this book will prepare engineers and scientists to solve many of their everyday problems, give them the insight to recognize when other methods are required, and give them the background to study other methods in other books and journals.

0.3 EXAMPLES

All of the numerical methods presented in this book are illustrated by applying them to solve an example problem. Each chapter has one or two example problems, which are solved by all of the methods presented in the chapter. This approach allows the analyst to compare various methods for the same problem, so accuracy, efficiency, robustness, and ease of application of the various methods can be evaluated.

Most of the example problems are rather simple and straightforward, thus allowing the special features of the various methods to be demonstrated clearly. All of the example problems have exact solutions, so the errors of the various methods can be compared. Each example problem begins with a reference to the problem to be solved, a description of the numerical method to be employed, details of the calculations for at least one application of the algorithm, and a summary of the remaining results. Some comments about the solution are presented at the end of the calculations in most cases.

0.4 PROGRAMS

Most numerical algorithms are generally expressed in the form of a computer program. This is especially true for algorithms that require a lot of computational effort and for algorithms that are applied many times. Several programming languages are available for preparing computer programs: FORTRAN, Basic, C, PASCAL, etc., and their variations, to name a few. Pseudocode, which is a set of instructions for implementing an algorithm expressed in conceptual form, is also quite popular. Pseudocode can be expressed in the detailed form of any specific programming language.

FORTRAN is one of the oldest programming languages. When carefully prepared, FORTRAN can approach pseudocode. Consequently, the programs presented in this book are written in simple FORTRAN. There are several vintages of FORTRAN: FORTRAN I, FORTRAN II, FORTRAN 66, 77, and 90. The programs presented in this book are compatible with FORTRAN 77 and 90.

Several programs are presented in each chapter for implementing the more prominent numerical algorithms presented in the chapter. Each program is applied to solve the example problem relevant to that chapter. The implementation of the numerical algorithm is contained within a completely self-contained *application subroutine* which can be used in other programs. These *application subroutines* are written as simply as possible so that conversion to other programming languages is as straightforward as possible. These subroutines can be used as they stand or easily modified for other applications.

Each *application subroutine* is accompanied by a *program main*. The variables employed in the *application subroutine* are defined by comment statements in *program main*. The numerical values of the variables are defined in *program main*,which then calls the *application subroutine* to solve the example problem and to print the solution. These main programs are not intended to be convertible to other programming languages. In some problems where a function of some type is part of the specification of the problem, that function is defined in a *function subprogram* which is called by the *application subroutine*.

FORTRAN compilers do not distinguish between uppercase and lowercase letters. FORTRAN programs are conventionally written in uppercase letters. However, in this book, all FORTRAN programs are written in lowercase letters.

0.5 PROBLEMS

Two types of problems are presented at the end of each chapter:

1. Exercise problems
2. Applied problems

Exercise problems are straightforward problems designed to give practice in the application of the numerical algorithms presented in each chapter. Exercise problems emphasize the mechanics of the methods.

Applied problems involve more applied engineering and scientific applications which require numerical solutions.

Many of the problems can be solved by hand calculation. A large number of the problems require a lot of computational effort. Those problems should be solved by writing a computer program to perform the calculations. Even in those cases, however, it is recommended that one or two passes through the algorithm be made by hand calculation to ensure that the analyst fully understands the details of the algorithm. These results also can be used to validate the computer program.

Answers to selected problems are presented in a section at the end of the book. All of the problems for which answers are given are denoted by an asterisk appearing with the corresponding problem number in the problem sections at the end of each chapter. The **Solutions Manual** contains the answers to nearly all of the problems.

0.6 SIGNIFICANT DIGITS, PRECISION, ACCURACY, ERRORS, AND NUMBER REPRESENTATION

Numerical calculations obviously involve the manipulation (i.e., addition, multiplication, etc.) of numbers. Numbers can be integers (e.g., 4, 17, -23, etc.), fractions (e.g., 1/2, $-2/3$, etc.), or an inifinite string of digits (e.g., $\pi = 3.1415926535\ldots$). When dealing with numerical values and numerical calculations, there are several concepts that must be considered:

1. Significant digits
2. Precision and accuracy
3. Errors
4. Number representation

These concepts are discussed briefly in this section.

Significant Digits

The **significant digits**, or figures, in a number are the digits of the number which are known to be correct. Engineering and scientific calculations generally begin with a set of data having a known number of significant digits. When these numbers are processed through a numerical algorithm, it is important to be able to estimate how many significant digits are present in the final computed result.

Precision and Accuracy

Precision refers to how closely a number represents the number it is representing. **Accuracy** refers to how closely a number agrees with the true value of the number it is representing.

Precision is governed by the number of digits being carried in the numerical calculations. Accuracy is governed by the errors in the numerical approximation. Precision and accuracy are quantified by the errors in a numerical calculation.

Errors

The **accuracy** of a numerical calculation is quantified by the **error** of the calculation. Several types of errors can occur in numerical calculations.

1. Errors in the parameters of the problem (assumed nonexistent).
2. Algebraic errors in the calculations (assumed nonexistent).
3. Iteration errors.
4. Approximation errors.
5. Roundoff errors.

Iteration error is the error in an iterative method that approaches the exact solution of an exact problem asymptotically. Iteration errors must decrease toward zero as the iterative process progresses. The iteration error itself may be used to determine the successive approximations to the exact solution. Iteration errors can be reduced to the limit of the computing device. The errors in the solution of a system of linear algebraic equations by the successive-over-relaxation (SOR) method presented in Section 1.5 are examples of this type of error.

Approximation error is the difference between the exact solution of an exact problem and the exact solution of an approximation of the exact problem. Approximation error can be reduced only by choosing a more accurate approximation of the exact problem. The error in the approximation of a function by a polynomial, as described in Chapter 4, is an example of this type of error. The error in the solution of a differential equation where the exact derivatives are replaced by algebraic difference approximations, which have truncation errors, is another example of this type of error.

Roundoff error is the error caused by the finite word length employed in the calculations. Roundoff error is more significant when small differences between large numbers are calculated. Most computers have either 32 bit or 64 bit word length, corresponding to approximately 7 or 13 significant decimal digits, respectively. Some computers have extended precision capability, which increases the number of bits to 128. Care must be exercised to ensure that enough significant digits are maintained in numerical calculations so that roundoff is not significant.

Number Representation

Numbers are represented in number systems. Any number of bases can be employed as the base of a number system, for example, the base 10 (i.e., decimal) system, the base 8 (i.e., octal) system, the base 2 (i.e., binary) system, etc. The base 10, or decimal, system is the most common system used for human communication. Digital computers use the base 2, or binary, system. In a digital computer, a binary number consists of a number of binary bits. The number of binary bits in a binary number determines the precision with which the binary number represents a decimal number. The most common size binary number is a 32 bit number, which can represent approximately seven digits of a decimal number. Some digital computers have 64 bit binary numbers, which can represent 13 to 14 decimal digits. In many engineering and scientific calculations, 32 bit arithmetic is adequate. However, in many other applications, 64 bit arithmetic is required. In a few special situations, 128 bit arithmetic may be required. On 32 bit computers, 64 bit arithmetic, or even 128 bit arithmetic, can be accomplished using software enhancements. Such calculations are called **double precision** or **quad precision**, respectively. Such software enhanced precision can require as much as 10 times the execution time of a single precision calculation.

Consequently, some care must be exercised when deciding whether or not higher precision arithmetic is required. All of the examples in this book are evaluated using 64 bit arithmetic to ensure that roundoff is not significant.

Except for integers and some fractions, all binary representations of decimal numbers are approximations, owing to the finite word length of binary numbers. Thus, some loss of precision in the binary representation of a decimal number is unavoidable. When binary numbers are combined in arithmetic operations such as addition, multiplication, etc., the true result is typically a longer binary number which cannot be represented exactly with the number of available bits in the binary number capability of the digital computer. Thus, the results are rounded off in the last available binary bit. This rounding off gives rise to **roundoff error**, which can accumulate as the number of calculations increases.

0.7 SOFTWARE PACKAGES AND LIBRARIES

Numerous commercial software packages and libraries are available for implementing the numerical solution of engineering and scientific problems. Two of the more versatile software packages are Mathcad and Matlab. These software packages, as well as several other packages and several libraries, are listed below with a brief description of each one and references to sources for the software packages and libraries.

A. Software Packages

Excel Excel is a spreadsheet developed by Microsoft, Inc., as part of Microsoft Office. It enables calculations to be performed on rows and columns of numbers. The calculations to be performed are specified for each column. When any number on the spreadsheet is changed, all of the calculations are updated. Excel contains several built-in numerical algorithms. It also includes the Visual Basic programming language and some plotting capability. Although its basic function is not numerical analysis, Excel can be used productively for many types of numerical problems. Microsoft, Inc. www.microsoft.com/office/Excel.

Macsyma Macsyma is the world's first artificial intelligence based math engine providing easy to use, powerful math software for both symbolic and numerical computing. Macsyma, Inc., 20 Academy St., Arlington, MA 02476-6412. (781) 646-4550, webmaster@macsyma.com, www.macsyma.com.

Maple Maple 6 is a technologically advanced computational system with both algorithms and numeric solvers. Maple 6 includes an extensive set of NAG (Numerical Algorithms Group) solvers for computational linear algebra. Waterloo Maple, Inc., 57 Erb Street W., Waterloo, Ontario, Canada N2L 6C2. (800) 267-6583, (519) 747-2373, info@maplesoft.com, www.maplesoft.com.

Mathematica Mathematica 4 is a comprehensive software package which performs both symbolic and numeric computations. It includes a flexible and intuitive programming language and comprehensive plotting capabilities. Wolfram Research, Inc., 100 Trade Center Drive, Champaign IL 61820-7237. (800) 965-3726, (217) 398-0700, info@wolfram.com, www.wolfram.com.

Mathcad Mathcad 8 provides a free-form interface which permits the integration of real math notation, graphs, and text within a single interactive worksheet. It includes statistical and data analysis functions, powerful solvers, advanced matrix manipulation,