CAMBRIDGE TEXTS IN BIOMEDICAL ENGINEERING

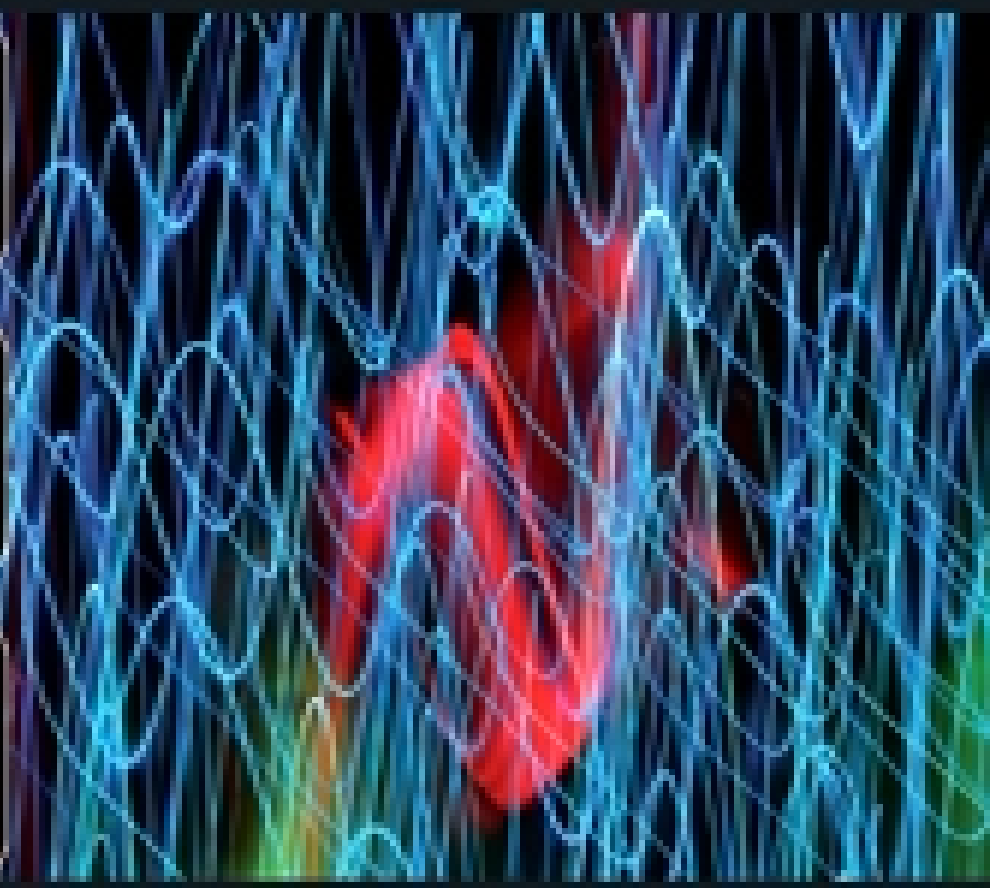
Numerical and Statistical Methods for Bioengineering

Applications in MATLAB

$$\Sigma_i^j = \sigma_i^j K K^T$$

$$x^2 = \sum_{i=1}^{n} \frac{(o_i - e_i)^2}{e_i}$$

$$\frac{|x_{n+1}|}{|x_n|^2} = \frac{|f''(\xi)|}{2|f'(x_n)|}$$



Michael R. King and Nipa A. Mody

CAMBRIDGE www.cambridge.org/9780521871587

This page intentionally left blank

Numerical and Statistical Methods for Bioengineering

This is the first MATLAB-based numerical methods textbook for bioengineers that uniquely integrates modeling concepts with statistical analysis, while maintaining a focus on enabling the user to report the error or uncertainty in their result. Between traditional numerical method topics of linear modeling concepts, non-linear root finding, and numerical integration, chapters on hypothesis testing, data regression, and probability are interweaved. A unique feature of the book is the inclusion of examples from clinical trials and bioinformatics, which are not found in other numerical methods textbooks for engineers. With a wealth of biomedical engineering examples, case studies on topical biomedical research, and the inclusion of end of chapter problems, this is a perfect core text for a one-semester undergraduate course.

Michael R. King is an Associate Professor of Biomedical Engineering at Cornell University. He is an expert on the receptor-mediated adhesion of circulating cells, and has developed new computational and in vitro models to study the function of leukocytes, platelets, stem, and cancer cells under flow. He has co-authored two books and received numerous awards, including the 2008 ICNMM Outstanding Researcher Award from the American Society of Mechanical Engineers, and received the 2009 Outstanding Contribution for a Publication in the International Journal *Clinical Chemistry*.

Nipa A. Mody is currently a postdoctoral research associate at Cornell University in the Department of Biomedical Engineering. She received her Ph.D. in Chemical Engineering from the University of Rochester in 2008 and has received a number of awards including a Ruth L. Kirschstein National Research Service Award (NRSA) from the NIH in 2005 and the Edward Peck Curtis Award for Excellence in Teaching from University of Rochester in 2004.

CAMBRIDGE TEXTS IN BIOMEDICAL ENGINEERING

Cambridge Texts in Biomedical Engineering provides a forum for high-quality accessible textbooks targeted at undergraduate and graduate courses in biomedical engineering. It covers a broad range of biomedical engineering topics from introductory texts to advanced topics including, but not limited to, biomechanics, physiology, biomedical instrumentation, imaging, signals and systems, cell engineering, and bioinformatics. The series blends theory and practice, aimed primarily at biomedical engineering students, it also suits broader courses in engineering, the life sciences and medicine.

Numerical and Statistical Methods for Bioengineering

Applications in MATLAB

Michael R. King and Nipa A. Mody
Cornell University

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore,
São Paulo, Delhi, Dubai, Tokyo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521871587

First published in print format 2010

ISBN-13 978-0-511-90833-0 eBook (EBL)

ISBN-13 978-0-521-87158-7 Hardback

Contents

Preface

Biomedical engineering programs have exploded in popularity and number over the past 20 years. In many programs, the fundamentals of engineering science are taught from textbooks borrowed from other, more traditional, engineering fields: statics, transport phenomena, circuits. Other courses in the biomedical engineering curriculum are so multidisciplinary (think of tissue engineering, Introduction to BME) that this approach does not apply; fortunately, excellent new textbooks have recently emerged on these topics. On the surface, numerical and statistical methods would seem to fall into this first category, and likely explains why biomedical engineers have not yet contributed textbooks on this subject. I mean ... math is math, right? Well, not exactly.

There exist some unique aspects of biomedical engineering relevant to numerical analysis. Graduate research in biomedical engineering is more often *hypothesis driven*, compared to research in other engineering disciplines. Similarly, biomedical engineers in industry design, test, and produce medical devices, instruments, and drugs, and thus must concern themselves with human clinical trials and gaining approval from regulatory agencies such as the US Food & Drug Administration. As a result, statistics and hypothesis testing play a bigger role in biomedical engineering and must be taught at the curricular level. This increased emphasis on statistical analysis is reflected in special "program criteria" established for biomedical engineering degree programs by the Accreditation Board for Engineering and Technology (ABET) in the USA.

There are many general textbooks on numerical methods available for undergraduate and graduate students in engineering; some of these use MATLAB as the teaching platform. A good undergraduate text along these lines is *Numerical Methods with Matlab* by G. Recktenwald, and a good graduate-level reference on numerical methods is the well-known *Numerical Recipes* by W. H. Press *et al*. These texts do a good job of covering topics such as programming basics, nonlinear root finding, systems of linear equations, least-squares curve fitting, and numerical integration, but tend not to devote much space to statistics and hypothesis testing. Certainly, topics such as genomic data and design of clinical trails are not covered. But beyond the basic numerical algorithms that may be common to engineering and the physical sciences, one thing an instructor learns is that *biomedical engineering students want to work on biomedical problems*! This requires a biomedical engineering instructor to supplement a general numerical methods textbook with a gathering of relevant lecture examples, homework, and exam problems, a labor-intensive task to be sure and one that may leave students confused and unsatisfied with their textbook investment. This book is designed to fill an unmet need, by providing a complete numerical and statistical methods textbook, tailored to the unique requirements of the modern BME curriculum and implemented in MATLAB, which is inundated with examples drawn from across the spectrum of biomedical science.

This book is designed to serve as the primary textbook for a one-semester course in numerical and statistical methods for biomedical engineering students. The level of the book is appropriate for sophomore year through first year of graduate studies, depending on the pace of the course and the number of advanced, optional topics that are covered. A course based on this book, together with later opportunities for implementation in a laboratory course or senior design project, is intended to fulfil the statistics and hypothesis testing requirements of the program criteria established by ABET, and served this purpose at the University of Rochester. The material within this book formed the basis for the required junior-level course "Biomedical computation," offered at the University of Rochester from 2002 to 2008. As of Fall 2009, an accelerated version of the "Biomedical computation" course is now offered at the masters level at Cornell University. It is recommended that students have previously taken calculus and an introductory programming course; a semester of linear algebra is helpful but not required. It is our hope that this book will also serve as a valuable reference for bioengineering practitioners and other researchers working in quantitative branches of the life sciences such as biophysics and physiology.

Format

As with most textbooks, the chapters have been organized so that concepts are progressively built upon as the reader advances from one chapter to the next. Chapters 1 and 2 develop basic concepts, such as types of errors, linear algebra concepts, linear problems, and linear regression, that are referred to in later chapters. Chapters 3 (Probability and statistics) and 5 (Nonlinear root-finding techniques) draw upon the material covered in Chapters 1 and 2. Chapter 4 (Hypothesis testing) exclusively draws upon the material covered in Chapter 3, and can be covered at any point after Chapter 3 (Sections 3.1 to 3.5) is completed. The material on linear regression error in Chapter 3 should precede the coverage of Chapter 8 (Nonlinear model regression and optimization). The following chapter order is strongly recommended to provide a seamless transition from one topic to the next:

Chapter 1 → Chapter 2 → Chapter 3 → Chapter 5 → Chapter 6 → Chapter 8.

Chapter 4 can be covered at any time once the first three chapters are completed, while Chapter 7 can be covered at any time after working through Chapters 1, 2, 3, and 5. Chapter 9 covers an elective topic that can be taken up at any time during a course of study.

The examples provided in each chapter are of two types: Examples and Boxes. The problems presented in the Examples are more straightforward and the equations simpler. Examples either illustrate concepts already introduced in earlier sections or are used to present new concepts. They are relatively quick to work through compared to the Boxes since little additional background information is needed to understand the example problems. The Boxes discuss biomedical research, clinical, or industrial problems and include an explanation of relevant biology or engineering concepts to present the nature of the problem. In a majority of the Boxes, the equations to be solved numerically are derived from first principles to provide a more complete understanding of the problem. The problems covered in Boxes can be more challenging and require more involvement by

the reader. While the Examples are critical in mastering the text material, the choice of which boxed problems to focus on is left to the instructor or reader.

As a recurring theme of this book, we illustrate the implementation of numerical methods through programming with the technical software package MATLAB. Previous experience with MATLAB is not necessary to follow and understand this book, although some prior programming knowledge is recommended. The best way to learn how to program in a new language is to jump right into coding when a need presents itself. Sophistication of the code is increased gradually in successive chapters. New commands and programming concepts are introduced on a need-to-know basis. Readers who are unfamiliar with MATLAB should first study Appendix A, Introduction to MATLAB, to orient themselves with the MATLAB programming environment and to learn the basic commands and programming terminology. Examples and Boxed problems are accompanied by the MATLAB code containing the numerical algorithm to solve the numerical or statistical problem. The MATLAB programs presented throughout the book illustrate code writing practice. We show two ways to use MATLAB as a tool for solving numerical problems: (1) by developing a program (m-file) that contains the numerical algorithm, and (2) using built-in functions supplied by MATLAB to solve a problem numerically. While self-written numerical algorithms coded in MATLAB are instructive for teaching, MATLAB built-in functions that compute the numerical solution can be more efficient and robust for use in practice. The reader is taught to integrate MATLAB functions into their written code to solve a specific problem (e.g. the backslash operator).

The book has its own website hosted by Cambridge University Press at www.cambridge.org/kingmody. All of the m-files and numerical data sets within this book can be found at this website, along with additional MATLAB programs relevant to the topics and problems covered in the text.

Acknowledgements

First and foremost, M. R. K. owes a debt of gratitude to Professor David Leighton at the University of Notre Dame. The "Biomedical computation" course that led to this book was closely inspired by Leighton's "Computer methods for chemical engineers" course at Notre Dame. I (Michael King) had the good fortune of serving as a teaching assistant and later as a graduate instructor for that course, and it shaped the format and style of my own teaching on this subject. I would also like to thank former students, teaching assistants, and faculty colleagues at the University of Rochester and Cornell University; over the years, their valuable input has helped continually to improve the material that comprises this book. I thank my wife and colleague Cindy Reinhart-King for her constant support. Finally, I thank my co-author, friend, and former student Nipa Mody: without her tireless efforts this book would not exist.

N.A.M. would like to acknowledge the timely and helpful advice on creating bioengineering examples from the following business and medical professionals: Khyati Desai, Shimoni Shah, Shital Modi, and Pinky Shah. I (Nipa Mody) also thank Banu Sankaran and Ajay Sadrangani for their general feedback on parts of the book. I am indebted to the support provided by the following faculty and staff members of the Biomedical Engineering Department of the University of Rochester: Professor Richard E. Waugh, Donna Porcelli, Nancy Gronski, Mary

Gilmore, and Gayle Hurlbutt, in this endeavor. I very much appreciate the valiant support of my husband, Anand Mody, while I embarked on the formidable task of book writing.

We thank those people who have critically read versions of this manuscript, in particular, Ayotunde Oluwakorede Ositelu, Aram Chung, and Bryce Allio, and also to our reviewers for their excellent recommendations. We express many thanks to Michelle Carey, Sarah Matthews, Christopher Miller, and Irene Pizzie at Cambridge University Press for their help in the planning and execution of this book project and for their patience.

If readers wish to suggest additional topics or comments, please write to us. We welcome all comments and criticisms as this book (and the field of biomedical engineering) continue to evolve.

1 Types and sources of numerical error

1.1 Introduction

The job of a biomedical engineer often involves the task of formulating and solving mathematical equations that define, for example, the design criteria of biomedical equipment or a prosthetic organ or physiological/pathological processes occurring in the human body. Mathematics and engineering are inextricably linked. The types of equations that one may come across in various fields of engineering vary widely, but can be broadly categorized as: linear equations in one variable, linear equations with respect to multiple variables, nonlinear equations in one or more variables, linear and nonlinear ordinary differential equations, higher order differential equations of nth order, and integral equations. Not all mathematical equations are amenable to an analytical solution, i.e. a solution that gives an exact answer either as a number or as some function of the variables that define the problem. For example, the analytical solution for

(1) $x^2 + 2x + 1 = 0$ is $x = \pm 1$, and
(2) $dy/dx + 3x = 5$, with initial conditions $x = 0, y = 0$, is $y = 5x - 3x^2/2$.

Sometimes the analytical solution to a system of equations may be exceedingly difficult and time-consuming to obtain, or once obtained may be too complicated to provide insight.

The need to obtain a solution to these otherwise unsolvable problems in a reasonable amount of time and with the resources at hand has led to the development of **numerical methods**. Such methods are used to determine an approximation to the actual solution within some tolerable degree of error. A numerical method is an iterative mathematical procedure that can be applied to only certain types or forms of a mathematical equation, and under usual circumstances allows the solution to converge to a final value with a pre-determined level of accuracy or tolerance. Numerical methods can often provide exceedingly accurate solutions for the problem under consideration. However, keep in mind that the solutions are rarely ever exact. A closely related branch of mathematics is **numerical analysis**, which goes hand-in-hand with the development and application of numerical methods. This related field of study is concerned with analyzing the performance characteristics of established numerical methods, i.e. how quickly the numerical technique converges to the final solution and accuracy limitations. It is important to have, at the least, basic knowledge of numerical analysis so that you can make an informed decision when choosing a technique for solving a numerical problem. The accuracy and precision of the numerical solution obtained is dependent on a number of factors, which include the choice of the numerical technique and the implementation of the technique chosen.

Errors can creep into any mathematical solution or statistical analysis in several ways. Human mistakes include, for example, (1) entering incorrect data into

a computer, (2) errors in the mathematical expressions that define the problem, or (3) bugs in the computer program written to solve the engineering or math problem, which can result from logical errors in the code. A source of error that we have less control over is the quality of the data. Most often, scientific or engineering data available for use are imperfect, i.e. the true values of the variables cannot be determined with complete certainty. Uncertainty in physical or experimental data is often the result of imperfections in the experimental measuring devices, inability to reproduce exactly the same experimental conditions and outcomes each time, the limited size of sample available for determining the average behavior of a population, presence of a bias in the sample chosen for predicting the properties of the population, and inherent variability in biological data. All these errors may to some extent be avoided, corrected, or estimated using statistical methods such as confidence intervals. Additional errors in the solution can also stem from inaccuracies in the mathematical model. The model equations themselves may be simplifications of actual phenomena or processes being mimicked, and the parameters used in the model may be approximate at best.

Even if all errors derived from the sources listed above are somehow eliminated, we will still find other errors in the solution, called **numerical errors**, that arise when using numerical methods and electronic computational devices to perform numerical computations. These are actually unavoidable! **Numerical errors**, which can be broadly classified into two categories – round-off errors and truncation errors – are an integral part of these methods of solution and preclude the attainment of an exact solution. The source of these errors lies in the fundamental approximations and/or simplifications that are made in the representation of numbers as well as in the mathematical expressions that formulate the numerical problem. Any computing device you use to perform calculations follows a specific method to store numbers in a memory in order to operate upon them. Real numbers, such as fractions, are stored in the computer memory in floating-point format using the binary number system, and cannot always be stored with exact precision. This limitation, coupled with the finite memory available, leads to what is known as round-off error. Even if the numerical method yields a highly accurate solution, the computer round-off error will pose a limit to the final accuracy that can be achieved.

You should familiarize yourself with the types of errors that limit the precision and accuracy of the final solution. By doing so, you will be well-equipped to (1) estimate the magnitude of the error inherent in your chosen numerical method, (2) choose the most appropriate method for solution, and (3) prudently implement the algorithm of the numerical technique.

The origin of round-off error is best illustrated by examining how numbers are stored by computers. In Section 1.2, we look closely at the floating-point representation method for storing numbers and the inherent limitations in numeric precision and accuracy as a result of using binary representation of decimal numbers and finite memory resources. Section 1.3 discusses methods to assess the accuracy of estimated or measured values. The accuracy of any measured value is conveyed by the number of significant digits it has. The method to calculate the number of significant digits is covered in Section 1.4. Arithmetic operations performed by computers also generate round-off errors. While many round-off errors are too small to be of significance, certain floating-point operations can produce large and unacceptable errors in the result and should be avoided when possible. In Section 1.5, strategies to prevent the inadvertent generation of large round-off errors are discussed. The origin of truncation error is examined in Section 1.6. In Section 1.7 we introduce useful termination

Box 1.1A Oxygen transport in skeletal muscle

Oxygen is required by all cells to perform respiration and thereby produce energy in the form of ATP to sustain cellular processes. Partial oxidation of glucose (energy source) occurs in the cytoplasm of the cell by an anaerobic process called glycolysis that produces pyruvate. Complete oxidation of pyruvate to CO_2 and H_2O occurs in the mitochondria, and is accompanied by the production of large amounts of ATP. If cells are temporarily deprived of an adequate oxygen supply, a condition called hypoxia develops. In this situation, pyruvate no longer undergoes conversion in the mitochondria and is instead converted to lactic acid within the cytoplasm itself. Prolonged oxygen starvation of cells leads to cell death, called necrosis.

The circulatory system and the specialized oxygen carrier, hemoglobin, cater to the metabolic needs of the vast number of cells in the body. Tissues in the body are extensively perfused with tiny blood vessels in order to enable efficient and timely oxygen transport. The oxygen released from the red blood cells flowing through the capillaries diffuses through the blood vessel membrane and enters into the tissue region. The driving force for oxygen diffusion is the oxygen concentration gradient at the vessel wall and within the tissues. The oxygen consumption by the cells in the tissues depletes the oxygen content in the tissue, and therefore the oxygen concentration is always lower in the tissues as compared to its concentration in arterial blood (except when O_2 partial pressure in the air is abnormally low, such as at high altitudes). During times of strenuous activity of the muscles, when oxygen demand is greatest, the O_2 concentrations in the muscle tissue are the lowest. At these times it is critical for oxygen transport to the skeletal tissue to be as efficient as possible.

The skeletal muscle tissue sandwiched between two capillaries can be modeled as a slab of length L (see Figure 1.1). Let $N(x, t)$ be the O_2 concentration in the tissue, where $0 \leq x \leq L$ is the distance along the muscle length and t is the time. Let D be the O_2 diffusivity in the tissue and let Γ be the volumetric rate of O_2 consumption within the tissue.

Performing an O_2 balance over the tissue produces the following partial differential equation:

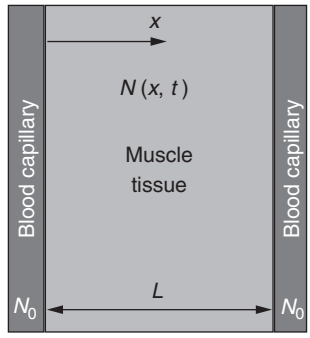
$$\frac{\partial N}{\partial t} = D\frac{\partial^2 N}{\partial x^2} - \Gamma.$$

The boundary conditions for the problem are fixed at $N(0, t) = N(L, t) = N_0$, where N_0 is the supply concentration of O_2 at the capillary wall. For this problem, it is assumed that the resistance to transport of O_2 posed by the vessel wall is small enough to be neglected. We also neglect the change in O_2 concentration along the length of the capillaries. The steady state or long-term O_2 distribution in the tissue is governed by the ordinary differential equation

$$D\frac{\partial^2 N}{\partial x^2} = \Gamma,$$

Figure 1.1

Schematic of O_2 transport in skeletal muscle tissue.



whose solution is given by

$$N_s = N_0 - \frac{\Gamma x}{2D}(L - x).$$

Initially, the muscles are at rest, consuming only a small quantity of O_2, characterized by a volumetric O_2 consumption rate Γ_1. Accordingly, the initial O_2 distribution in the tissue is $N_0 - \frac{\Gamma_1 x}{2D}(L - x)$.

Now the muscles enter into a state of heavy activity characterized by a volumetric O_2 consumption rate Γ_2. The time-dependent O_2 distribution is given by a Fourier series solution to the above partial differential equation:

$$N = N_0 - \frac{\Gamma_2 x}{2D}(L - x) + \frac{4(\Gamma_2 - \Gamma_1)L^2}{D} \sum_{n=1,\ n \text{ is odd}}^{\infty} \left[\frac{1}{(n\pi)^3} e^{-(n\pi)^2 Dt/L^2} \sin\left(\frac{n\pi x}{L}\right) \right].$$

In Section 1.6 we investigate the truncation error involved when arriving at a solution to the O_2 distribution in muscle tissue.

criteria for numerical iterative procedures. The use of robust convergence criteria is essential to obtain reliable results.

1.2 Representation of floating-point numbers

The arithmetic calculations performed when solving problems in algebra and calculus produce exact results that are mathematically sound, such as:

$$\frac{2.5}{0.5} = 5,\ \frac{\sqrt[3]{8}}{\sqrt{4}} = 1, \text{and } \frac{d}{dx}\sqrt{x} = \frac{1}{2\sqrt{x}}.$$

Computations made by a computer or a calculator produce a true result for any integer manipulation, but have less than perfect precision when handling real numbers. It is important at this juncture to define the meaning of "precision." Numeric **precision** is defined as the exactness with which the value of a numerical estimate is known. For example, if the true value of $\sqrt{4}$ is 2 and the computed solution is 2.0001, then the computed value is precise to within the first four figures or four significant digits. We discuss the concept of significant digits and the method of calculating the number of significant digits in a number in Section 1.4.

Arithmetic calculations that are performed by retaining mathematical symbols, such as 1/3 or $\sqrt{7}$, produce exact solutions and are called symbolic computations. Numerical computations are not as precise as symbolic computations since numerical computations involve the conversion of fractional numbers and irrational numbers to their respective numerical or *digital* representations, such as $1/3 \sim 0.33333$ or $\sqrt{7} \sim 2.64575$. Numerical representation of numbers uses a finite number of digits to denote values that may possibly require an infinite number of digits and are, therefore, often inexact or approximate. The precision of the computed value is equal to the number of digits in the numerical representation that tally with the true digital value. Thus 0.33333 has a precision of five significant digits when compared to the true value of 1/3, which is also written as $0.\overline{3}$. Here, the overbar indicates infinite repetition of the underlying digit(s). Calculations using the digitized format to represent all real numbers are termed as **floating-point arithmetic**. The format

Box 1.2 Accuracy versus precision: blood pressure measurements

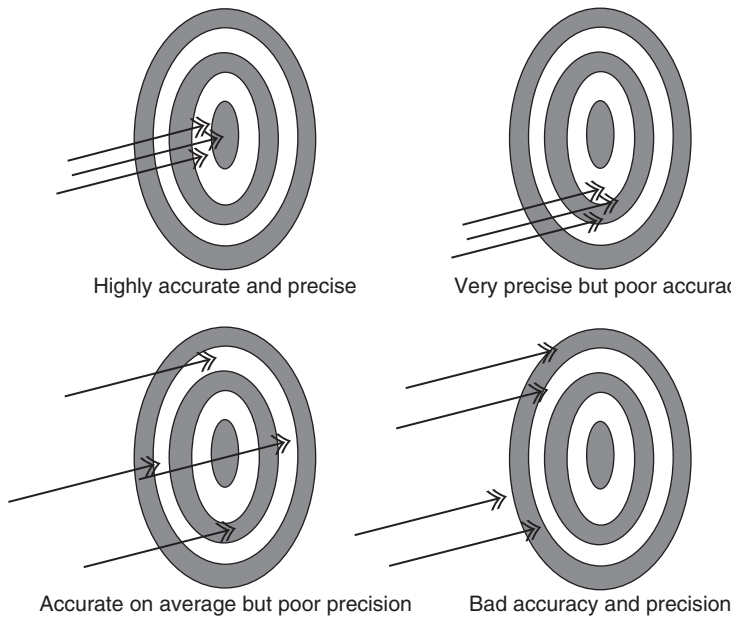
It is important that we contrast the two terms *accuracy* and *precision*, which are often confused. **Accuracy** measures how close the estimate of a measured variable or an observation is to its true value. **Precision** is the range or spread of the values obtained when repeated measurements are made of the same variable. A narrow spread of values indicates good precision.

For example, a digital sphygmomanometer consistently provides three readings of the systolic/diastolic blood pressure of a patient as 120/80 mm Hg. If the true blood pressure of the patient at the time the measurement was made is 110/70 mm Hg, then the instrument is said to be very precise, since it provided similar readings every time with few fluctuations. The three instrument readings are "on the same mark" every time. However, the readings are all inaccurate since the "correct target or mark" is not 120/80 mm Hg, but is 110/70 mm Hg.

An intuitive example commonly used to demonstrate the difference between accuracy and precision is the bulls-eye target (see Figure 1.2).

Figure 1.2

The bulls-eye target demonstrates the difference between accuracy and precision.



standards for **floating-point number representation** by computers are specified by the IEEE Standard 754. The binary or base-2 system is used by digital devices for storing numbers and performing arithmetic operations. The binary system cannot precisely represent all *rational numbers* in the decimal or base-10 system. The imprecision or error inherent in computing when using floating-point number representation is called **round-off error**. Round-off thus occurs when real numbers must be approximated using a limited number of significant digits. Once a round-off error is introduced in a floating-point calculation, it is carried over in all subsequent computations. However, round-off is of fundamental advantage to the efficiency of performing computations. Using a fixed and finite number of digits to represent

each number ensures a reasonable speed in computation and economical use of the computer memory.

Round-off error results from a trade-off between the efficient use of computer memory and accuracy.

Decimal floating-point numbers are represented by a computer in standardized format as shown below:

$$\pm 0.f_1 f_2 f_3 f_4 f_5 f_6 \ldots f_{s-1} f_s \times 10^k,$$

significand — 10 raised to the power k

where f is a decimal digit from 0 to 9, s is the number of significant digits, i.e. the number of digits in the significand as dictated by the precision limit of the computer, and k is the exponent. The advantage of this numeric representation scheme is that the **range** of representable numbers (as determined by the largest and smallest values of k) can be separated from the degree of **precision** (which is determined by the number of digits in the significand). The power or exponent k indicates the **order of magnitude** of the number. The notation for the order of magnitude of a number is $O(10^k)$. Section 1.7 discusses the topic of "estimation of order of magnitude" in more detail.

This method of numeric representation provides the best approximation possible of a real number within the limit of s significant digits. The real number may, however, require an infinite number of significant digits to denote the true value with perfect precision. The value of the last significant digit of a *floating-point number* is determined by one of two methods.

(1) **Truncation** Here, the numeric value of the digit f_{s+1} is not considered. The value of the digit f_s is unaffected by the numeric value of f_{s+1}. The floating-point number with s significant digits is obtained by dropping the $(s + 1)$th digit and all digits to its right.

(2) **Rounding** If the value of the last significant digit f_s depends on the value of the digits being discarded from the floating-point number, then this method of numeric representation is called rounding. The generally accepted convention for rounding is as follows (Scarborough, 1966):
   (a) if the numeric value of the digits being dropped is greater than five units of the f_{s+1}th position, then f_s is changed to $f_s + 1$;
   (b) if the numeric value of the digits being dropped is less than five units of the f_{s+1}th position, then f_s remains unchanged;
   (c) if the numeric value of the digits being dropped equals five units of the f_{s+1}th position, then
      (i) if f_s is even, f_s remains unchanged,
      (ii) if f_s is odd, f_s is replaced by $f_s + 1$.

This last convention is important since, on average, one would expect the occurrence of the f_s digit as odd only half the time. Accordingly, by leaving f_s unchanged approximately half the time when the f_{s+1} digit is exactly equal to five units, it is intuitive that the errors caused by rounding will to a large extent cancel each other.