国际著名物理图书——影印版系列

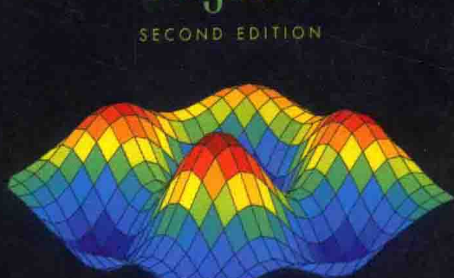
Nicholas J. Giordano, Hisao Nakanishi

计算物理

（第2版）

Computational Physics

SECOND EDITION



Nicholas J. Giordano
Hisao Nakanishi

PEARSON
Prentice Hall

清华大学出版社

PEARSON
Prentice Hall

国际著名物理图书——影印版系列

计算物理
（第2版）

Computational Physics
(Second Edition)

Nicholas J. Giordano
Hisao Nakanishi

清华大学出版社
北京

北京市版权局著作权合同登记号 图字：01-2007-5199

Original edition, entitled COMPUTATIONAL PHYSICS, second edition, 9780031469907 by NICHOLAS J. GIORDANO, HISAO NAKANISHI, published by Pearson Education, Inc, publishing as Pearson Education, Inc., copyright © 2006.

图书在版编目(CIP)数据

计算物理 = Computational Physics：第 2 版：英文/乔达诺（Giordano, N.J.），纳卡尼什（Nakanishi, H.）著. —影印本. —北京：清华大学出版社，2007.12

（国际著名物理图书. 影印版系列）

ISBN 978-7-302-16572-9

Ⅰ. 计… Ⅱ. ①乔… ②纳… Ⅲ. 计算物理学－英文 Ⅳ. O411.1

中国版本图书馆 CIP 数据核字（2007）第 185944 号

责任编辑：邹开颜

责任印制：孟凡玉

出 版 者：清华大学出版社　　地　址：北京清华大学学研大厦 A 座

http://www.tup.com.cn　　邮　编：100084

c-service@tup.tsinghua.edu.cn

社 总 机：010-62770175　　邮购热线：010-62786544

投稿咨询：010-62772015　　客户服务：010-62776969

印 刷 者：北京密云胶印厂

装 订 者：三河市金元印装有限公司

经　销：全国新华书店

开　本：185×230　印　张：35

版　次：2007 年 12 月第 1 版　　印　次：2007 年 12 月第 1 次印刷

印　数：1~3000

定　价：46.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：010-62770177 转 3103　　产品编号：027283-01

Computational Physics (第 2 版)

影 印 版 序

由于计算方法的深入发展和过去几十年中高速计算机的出现和普及，随着物理学基础理论的进一步突破，物理学家们逐步可以应用一些更严格和更全面的复杂模型，来定量研究实际的复杂体系的物理性质。基于物理学基本原理的数值计算和模拟已经成为将理论物理和实验物理紧密联系在一起的一座重要桥梁：它不仅能够弥补简单的解析理论模型难以完全描述复杂物理现象的不足，而且可以克服实验物理中遇到的许多困难，例如直接模拟实验上不能实现或技术条件要求很高、实验代价昂贵的物理系统等。计算机模拟技术已经渗透到物理学的各个领域，包括凝聚态物理、核物理、粒子物理、天体物理等，导致了计算物理这一新学科的突破性发展和成熟。从 20 世纪 40 年代开始，计算物理学家们已经发展了大量新数值方法（如 Monte Carlo 方法、分子动力学方法、快速 Fourier 变换等），由此发现了很多未曾预料到的新现象，并给理论和实验物理学提出了许多新问题。总之，计算物理已成为物理学家揭示多层次复杂体系的物理规律的重要手段，同时也广泛应用于处理实验结果和提出物理解释。对一个成功的物理学家来说，掌握必要的计算物理学知识和手段已变得越来越重要。越来越多的大学已针对将要从事物理学及相关学科研究的研究生和本科生开设了计算物理课程。

过去的十年中国际上已涌现出一些很好的计算物理专著和教材。由 Purdue 大学物理系的 Nicholas Giordano 教授和 Hisao Nakanishi 教授在其多年计算物理教学和科研工作基础上合作撰写的 *Computational Physics*（Second Edition）一书就是其中的突出代表。该书紧扣一些非常基本但难以解析求解的物理问题逐步展开，围绕各个物理学专题介绍了物理学研究中各种基本的计算机数值模拟方法，深入浅出地讨论其理论基础和实际应用，着重于解决实际物理问题的基本数值方法。这样可以使读者通过学习，对物理学中应用的主要计算技术有一个全面的了解，从而具有利用计算机进行数值计算解决复杂体系物理问题的能力。该书的另一个特点是包含了很多的物理学专题，这使得该书作为教材使用时教师在教学内容及其深度的选择方面有较大的灵活性。

清华大学出版社将该书引入国内，无疑将有利于从事物理科学及其相关研究的科研工作者和学生掌握必要的计算物理学方法和手段，并促进计算物理学科的的发展。

段文晖

2007 年 10 月

Preface

This book is based on the Computational Physics course that has been developed and taught by the authors at Purdue for more than a decade. The goal of this course is to introduce students to some basic numerical techniques and then apply these techniques to a number of *modern* topics, that is, problems of current interest to physicists. Students with some experience in differential and integral calculus can readily grasp rather sophisticated computational techniques. These students can use computers as tools with which to attack and solve problems that they would not ordinarily encounter in the undergraduate curriculum. We have used this approach to try to convey the excitement of physics, with a variety of problems of current interest.

While there are many texts with the terms "computers" and "physics" in their titles, most of the books in this area tend to focus heavily on numerical methods rather than physics. Since our goal is to teach a course on *physics*, rather than numerical methods, these books are not a good match for our course. While there are a few books that emphasize the physics that can be done with numerical methods, they are either too advanced for use by undergraduates, or (more commonly) they fail to deal with the types of problems that can profit most from a numerical approach. In too many cases they tend to simply treat the standard problems, which are already dealt with in many traditional texts using analytic methods. Hence the basic motivation for creating this book.

The material for our book is taken from a wide variety of "primary" sources, as will become clear from the references at the end of each chapter. In many cases we started with papers from the recent physics literature and then distilled them to produce problems suitable for an undergraduate class. While it is necessary for this book to introduce a variety of numerical methods of interest to physicists, the overriding emphasis is on the physics that can be done with these methods. The majority of the problems described in this book cannot be solved with purely analytic techniques. A computational approach is required in most cases, and we have tried to use the computer to make the *physics* as clear and as interesting as possible.

As readers scan through this new edition, they will notice a number of changes. Perhaps the most important is that there are now two authors. Besides simply sharing the workload of preparing the new edition, this additional expertise has allowed us to add many new topics (and improve old ones!). In fact, we have added much new material on subjects ranging from diffusion on fractals and cellular automata, to the physics of musical instruments (a new chapter) and a new algorithm for doing time-dependent quantum mechanical problems.

This book has also been reorganized in several ways. The first edition gave programming examples in the *True Basic* language. Now, in this new edition the reliance on the programming language *True Basic* has been removed. We recognize that present-day students will likely be using many different languages, so we have chosen to employ a very general pseudocode to illustrate the algorithms. This pseudocode can easily be translated into virtually any language, and hence support the work of students in a wide variety of programming languages. However, for those students who prefer to see programs or routines in a "real" programming language, the *True Basic* programs from the first edition are still available at our website,[1] `www.physics.purdue.edu/~giordano/comp-phys.html`. Our plan is to add more programs, in other languages, to this website in the future, so that it can serve as a useful resource for students (and teachers). Another change in this new edition is that we have moved much of the discussion of the algorithms themselves into the appendices, and have added considerably to the depth and rigor of these discussions. It is our hope that the appendices can serve as reference material for students as they work their way through the physics that is covered in the chapters. This separation also allows the chapters to focus even more on the physics of the various topics.

How to Use This Book.

The first edition contained more than could easily be covered in a single course. The second edition contains even more, so it is clearly not possible to cover it all in one semester. The first few chapters rely mainly on elementary mechanics, and can be appreciated with a background at the freshman level. This material can be augmented with selected topics from later chapters (such as on random processes in Chapter 7 and molecular dynamics in Chapter 9) to produce a full-semester course. There is also ample material for a course aimed at advanced undergraduates and beginning graduate students. For example, the material on random processes (Chapter 7) and phase transitions (Chapter 8) can be added to the work on quantum mechanics (Chapter 10) to fill most of a semester at this level. A third way to use the material in this book is for an interdisciplinary course, in which case the chapters on waves (Chapter 6), musical instruments (Chapter 11), and interdisciplinary topics (Chapter 12) could form the core of a course.

The first edition would not have been possible without the help of many people, and we would like to thank them again. The support of Arnold Tubis and the Department of Physics at Purdue, along with that of the National Science Foundation, made our course, and hence this book, possible. Many graduate students helped us teach early versions of this course, including Miguel Castro, Chris Parks, Jan Spitz, Stuart Burnett, Todd Jacobs, and Dan Lawrence. Of course, the undergraduate students who have willingly submitted to the course have provided much useful feedback; there are too many to mention them all here, although Mike Pennington deserves a special thanks. Many colleagues have provided essential advice and encouragement on the manuscript, including Todd Jacobs, Mark Haugan, Paul Muzikar, along with the reviewers Wolfgang Christian, Alejandro Garcia, Jan Tobochnik, and Rodney L. Varley, who were very polite and constructive. The support of the first edition editors Ray Henderson and Alison Reeves was much valued, while the final impetus to actually begin this book was provided by the well-timed encouragement of Earl Prohofsky and Betsy Beasley.

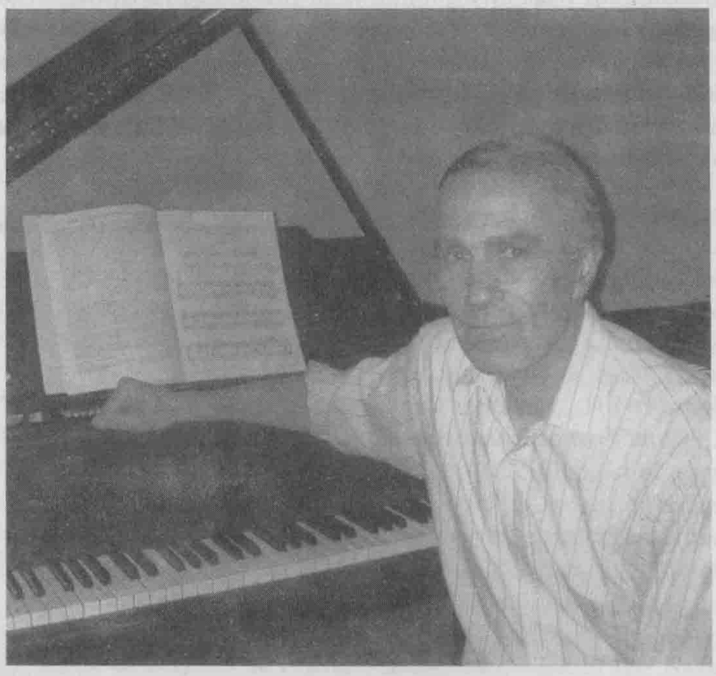
[1] The URL listed for the first edition, `www.physics.purdue.edu/~ng/comp-phys.html`, will take you to this new website.

The second edition owes much to our many colleagues who have sent us suggestions and new ideas for the new edition. In particular, we would like to thank James Behrens, Bob Delaney, Denis Donnelly, Eamin Jamshidi, Michael Oczkowski, Steve Turcotte, and Kobus Visser for alerting us to errors in the first edition, Aaron Montgomery for spotting (and correcting) a mistake in a draft of the second edition, and Eduardo Cuansing, Harvey Gould and Jan Tobochnik for their various contributions and general support. We also greatly appreciate the many constructive comments and suggestions from reviewers Gus Hart, James MacDonald, Micha Tomkiewicz, Thomas Vojta, and Matt Wood concerning drafts of the second edition. And of course, we are grateful to our Editors at Prentice-Hall, Erik Fahlgren and Christian Botting, for patiently guiding (and prodding) us through the preparation of this new edition.

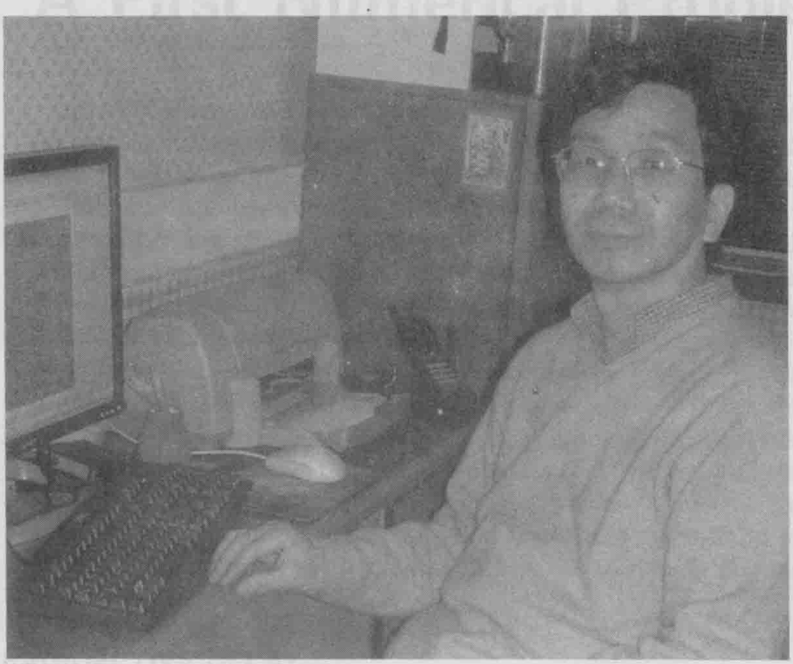
In closing we would like to reaffirm that our goal has been to write a book that uses computational methods to do interesting *physics*. While numerical methods can be fun, they are *not* our primary purpose. We hope that this book helps the student in all of us learn about and enjoy doing physics.

Nicholas J. Giordano and Hisao Nakanishi

About the Authors



Nicholas Giordano obtained his B.S. at Purdue University and his Ph.D. at Yale University. He has been on the faculty at Purdue since 1979, served as an Assistant Dean of Science from 2000-2003, and is currently the Hubert James Distinguished Professor of Physics. His research interests include electrical conduction, superconductivity, and magnetism in ultra-small metallic structures, along with musical acoustics and the physics of the piano. Ideas for this book grew out of the course on computational physics that he developed and taught in the early 1990s. Professor Giordano earned a Computational Science Education Award from the Deparment of Energy in 1997, and in 2004 was named Indiana Professor of the Year by the Carnegie Foundation for the Advancement of Teaching and the Council for the Advancement and Support of Education.



Hisao Nakanishi earned his B.S. from Brown University and his Ph.D. from Harvard University. His Ph.D. research concerned scaling and universality in a geometric phase transition called percolation and he has been interested in scale-invariance ever since. During his first postdoctoral work at Cornell he was introduced to the problem of surface critical phenomena such as wetting phase transitions, and later at the University of California, Santa Barbara, he started working on the statistics of diffusion and polymers in earnest. In 1992 Professor Nakanishi was a part of the team that won a Gordon Bell Prize for the application of parallel computing to a problem in polymer statistics. More recently he has also put on another hat as a developer of a computer-based interactive exercise system which is used by a few thousand students at Purdue each year.

Contents

Preface ix

About the Authors xii

1 A First Numerical Problem 1
- 1.1 Radioactive Decay 1
- 1.2 A Numerical Approach 2
- 1.3 Design and Construction of a Working Program: Codes and Pseudocodes 3
- 1.4 Testing Your Program 11
- 1.5 Numerical Considerations 12
- 1.6 Programming Guidelines and Philosophy 14

2 Realistic Projectile Motion 18
- 2.1 Bicycle Racing: The Effect of Air Resistance 18
- 2.2 Projectile Motion: The Trajectory of a Cannon Shell 25
- 2.3 Baseball: Motion of a Batted Ball 31
- 2.4 Throwing a Baseball: The Effects of Spin 36
- 2.5 Golf 44

3 Oscillatory Motion and Chaos 48
- 3.1 Simple Harmonic Motion 48
- 3.2 Making the Pendulum More Interesting: Adding Dissipation, Nonlinearity, and a Driving Force 54
- 3.3 Chaos in the Driven Nonlinear Pendulum 58
- 3.4 Routes to Chaos: Period Doubling 66
- 3.5 The Logistic Map: Why the Period Doubles 70
- 3.6 The Lorenz Model 75
- 3.7 The Billiard Problem 82
- 3.8 Behavior in the Frequency Domain: Chaos and Noise 88

4 The Solar System 94
- 4.1 Kepler's Laws 94
- 4.2 The Inverse-Square Law and the Stability of Planetary Orbits 101
- 4.3 Precession of the Perihelion of Mercury 107
- 4.4 The Three-Body Problem and the Effect of Jupiter on Earth 113
- 4.5 Resonances in the Solar System: Kirkwood Gaps and Planetary Rings 118
- 4.6 Chaotic Tumbling of Hyperion 123

5 Potentials and Fields 129
5.1 Electric Potentials and Fields: Laplace's Equation 129
5.2 Potentials and Fields Near Electric Charges 143
5.3 Magnetic Field Produced by a Current 148
5.4 Magnetic Field of a Solenoid: Inside and Out 151

6 Waves 156
6.1 Waves: The Ideal Case 156
6.2 Frequency Spectrum of Waves on a String 165
6.3 Motion of a (Somewhat) Realistic String 169
6.4 Waves on a String (Again): Spectral Methods 174

7 Random Systems 181
7.1 Why Perform Simulations of Random Processes? 181
7.2 Random Walks 183
7.3 Self-Avoiding Walks 188
7.4 Random Walks and Diffusion 195
7.5 Diffusion, Entropy, and the Arrow of Time 201
7.6 Cluster Growth Models 206
7.7 Fractal Dimensionalities of Curves 212
7.8 Percolation 218
7.9 Diffusion on Fractals 229

8 Statistical Mechanics, Phase Transitions, and the Ising Model 235
8.1 The Ising Model and Statistical Mechanics 235
8.2 Mean Field Theory 239
8.3 The Monte Carlo Method 244
8.4 The Ising Model and Second-Order Phase Transitions 246
8.5 First-Order Phase Transitions 259
8.6 Scaling 264

9 Molecular Dynamics 270
9.1 Introduction to the Method: Properties of a Dilute Gas 270
9.2 The Melting Transition 285
9.3 Equipartition and the Fermi-Pasta-Ulam Problem 294

10 Quantum Mechanics 303
10.1 Time-Independent Schrödinger Equation: Some Preliminaries 303
10.2 One Dimension: Shooting and Matching Methods 307
10.3 A Matrix Approach 323
10.4 A Variational Approach 326
10.5 Time-Dependent Schrödinger Equation: Direct Solutions 333
10.6 Time-Dependent Schrödinger Equation in Two Dimensions 345
10.7 Spectral Methods 349

11 Vibrations, Waves, and the Physics of Musical Instruments **357**
11.1 Plucking a String: Simulating a Guitar 357
11.2 Striking a String: Pianos and Hammers 362
11.3 Exciting a Vibrating System with Friction: Violins and Bows 367
11.4 Vibrations of a Membrane: Normal Modes and Eigenvalue Problems 372
11.5 Generation of Sound 382

12 Interdisciplinary Topics **389**
12.1 Protein Folding 389
12.2 Earthquakes and Self-Organized Criticality 405
12.3 Neural Networks and the Brain 418
12.4 Real Neurons and Action Potentials 436
12.5 Cellular Automata 445

APPENDICES

A Ordinary Differential Equations with Initial Values **456**
A.1 First-Order, Ordinary Differential Equations 456
A.2 Second-Order, Ordinary Differential Equations 460
A.3 Centered Difference Methods 464
A.4 Summary 467

B Root Finding and Optimization **469**
B.1 Root Finding 469
B.2 Direct Optimization 472
B.3 Stochastic Optimization 473

C The Fourier Transform **479**
C.1 Theoretical Background 479
C.2 Discrete Fourier Transform 481
C.3 Fast Fourier Transform (FFT) 483
C.4 Examples: Sampling Interval and Number of Data Points 486
C.5 Examples: Aliasing 488
C.6 Power Spectrum 490

D Fitting Data to a Function **493**
D.1 Introduction 493
D.2 Method of Least Squares: Linear Regression for Two Variables . . . 494
D.3 Method of Least Squares: More General Cases 497

E Numerical Integration **500**
E.1 Motivation 500
E.2 Newton-Cotes Methods: Using Discrete Panels to Approximate an Integral 500
E.3 Gaussian Quadrature: Beyond Classic Methods of Numerical Integration 504

E.4 Monte Carlo Integration 506

F Generation of Random Numbers **512**
F.1 Linear Congruential Generators 512
F.2 Nonuniform Random Numbers 516

G Statistical Tests of Hypotheses **520**
G.1 Central Limit Theorem and the χ^2 Distribution 521
G.2 χ^2 Test of a Hypothesis 523

H Solving Linear Systems **527**
H.1 Solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, $\mathbf{b} \neq \mathbf{0}$ 528
H.1.1 Gaussian Elimination 528
H.1.2 Gauss-Jordan elimination 530
H.1.3 LU decomposition 531
H.1.4 Relaxational method 533
H.2 Eigenvalues and Eigenfunctions 535
H.2.1 Approximate Solution of Eigensystems 537

Index **541**

CHAPTER 1

A First Numerical Problem

Many problems encountered in physics involve ordinary differential equations. Examples include projectile motion, harmonic motion, and celestial mechanics, topics we will be discussing extensively in the next few chapters. We therefore begin with a problem involving a first-order differential equation and use it to introduce some computational techniques that will be employed extensively in later chapters. We will also proceed step by step through the construction of a program to deal with this problem, so as to illustrate in detail how a numerical approach is translated into a (working) computer program.

In this chapter it is not possible to provide a complete introduction to programming for students who have no previous exposure to the subject. Rather, our goal is to enable students with some (even limited) experience in programming to begin writing programs to treat the physics that will be encountered in this book. However, those students with no prior experience should not give up hope! With some extra effort and access to a good instructor or book on computer programming (or both), such students should be able to handle the material in this and later chapters.

1.1 RADIOACTIVE DECAY

It is well known that many nuclei are unstable. A typical example is the nuclear isotope ^{235}U (the uranium nucleus that contains 143 neutrons and 92 protons, for a total of 235 nucleons), which has a small, but not insignificant, probability for decaying into two nuclei of approximately half its size, along with an assortment of protons, neutrons, electrons, and alpha particles. This process of radioactive decay is random in the following sense. If you were given a single ^{235}U nucleus, you would not be able to predict precisely when its decay would take place. The best you could do would be to give the *probability* for decay. An equivalent way to describe such a process would be to give the average time for decay; for ^{235}U the mean lifetime is approximately 1×10^9 years.

It is useful to imagine that we have a sample containing a large number of ^{235}U nuclei, which would usually be the case if we were actually doing an experiment to study radioactive decay. If $N_U(t)$ is the number of uranium nuclei that are present in the sample at time t, the behavior is governed by the differential equation

$$\frac{dN_U}{dt} = -\frac{N_U}{\tau} , \tag{1.1}$$

where τ is the "time constant" for the decay. You can show by direct substitution that the solution to this differential equation is

$$N_U = N_U(0)\, e^{-t/\tau} , \tag{1.2}$$

where $N_U(0)$ is the number of nuclei present at $t = 0$. This solution may be familiar to you; similar equations and similar solutions are found in many other contexts.[1] We note that at time $t = \tau$ a fraction e^{-1} of the nuclei that were initially present has not yet decayed. It turns out that τ is also the mean lifetime of a nucleus.

1.2 A NUMERICAL APPROACH

While the differential equation (1.1) can be solved without resorting to a numerical approach, this problem is useful for introducing several computational methods that will be used extensively in later chapters. With that in mind we now consider a simple method for solving this problem numerically. Our goal is to obtain N_U as a function of t. Given the value of N_U at one particular value of t (usually at $t = 0$), we want to estimate its value at later times. This is called an initial value problem, and various general approaches for solving such ordinary differential equations are discussed in Appendix A. Here we will describe one particularly useful line of attack that is based on the Taylor expansion for N_U,

$$N_U(\Delta t) \; = \; N_U(0) \; + \; \frac{dN_U}{dt}\,\Delta t \; + \; \frac{1}{2}\,\frac{d^2 N_U}{dt^2}\,(\Delta t)^2 \; + \cdots \; , \tag{1.3}$$

where $N_U(0)$ is the value of our function at time $t = 0$, $N_U(\Delta t)$ is its value at $t = \Delta t$, and the derivatives are evaluated at $t = 0$. If we take Δt to be small, then it is usually a good approximation to simply ignore the terms that involve second and higher powers of Δt, leaving us with

$$N_U(\Delta t) \; \approx \; N_U(0) \; + \; \frac{dN_U}{dt}\,\Delta t \; . \tag{1.4}$$

The same result can be obtained from the definition of a derivative. The derivative of N_U evaluated at time t can be written as

$$\frac{dN_U}{dt} \equiv \lim_{\Delta t \to 0} \; \frac{N_U(t + \Delta t) \; - \; N_U(t)}{\Delta t} \; \approx \; \frac{N_U(t + \Delta t) \; - \; N_U(t)}{\Delta t} \; , \tag{1.5}$$

where in the last approximation we have assumed that Δt is small but nonzero. We can rearrange this to obtain

$$N_U(t + \Delta t) \; \approx \; N_U(t) \; + \; \frac{dN_U}{dt}\,\Delta t \; , \tag{1.6}$$

which is equivalent to (1.4). It is important to recognize that this is an *approximation*, which is why it contains the $\approx$ symbol, not the $=$ symbol. The error terms that were dropped in deriving this result are of order $(\Delta t)^2$, which makes them at least one factor of Δt smaller than any of the terms in (1.6). Hence, by making Δt small, we would expect that the error terms can be made negligible. This is, in fact, the case in many problems, but there are situations in which the error terms can

[1] For example, an equation of this kind describes the time dependence of the voltage across a capacitor in an *RC* circuit.

still make life complicated. Therefore, it is important to be careful when discussing the errors involved in this numerical approach; we will return to this point later in this chapter, and in more detail in Appendix A.

From the physics of the problem we know the functional form of the derivative (1.1), and if we insert it into (1.6) we obtain

$$N_U(t + \Delta t) \approx N_U(t) - \frac{N_U(t)}{\tau} \Delta t . \tag{1.7}$$

This approximation forms the basis for a numerical solution of our radioactive decay problem. Given that we know the value of N_U at some value of t, we can use (1.7) to *estimate* its value a time Δt later.[2] Usually we are given, or can manage to discover, the initial value of the function, that is, the value at time $t = 0$. We can then employ (1.7) to estimate its value at $t = \Delta t$. This result can be used in turn to estimate the value at $t = 2\Delta t$, $3\Delta t$, etc., and thereby lead to an approximate solution $N_U(n\Delta t)$ at times $n\Delta t$ where n is an integer.[3] We cannot emphasize too strongly that the numerical "solution" obtained in this way is only an *approximation* to the "true," or exact, solution. Of course, one of our goals is to make the difference between the two negligible.

The approach to calculating $N_U(t)$ embodied in (1.6) and (1.7) is known as the *Euler method* and is a useful general algorithm for solving ordinary differential equations. We will use this approach, and closely related methods, extensively in this book. Other methods for solving equations of this kind will be discussed in later chapters, and more systematic discussions of all of these approaches and the typical errors associated with them are the subject of Appendix A. For now, the reader should realize that while the Euler method arises in a very natural way, it is certainly not the only algorithm for dealing with problems of this sort. We will see that the different approaches have their own strengths and weaknesses, which make them more or less suitable for different types of problems.

1.3 DESIGN AND CONSTRUCTION OF A WORKING PROGRAM: CODES AND PSEUDOCODES

In the previous section we introduced the Euler method as the basis for obtaining a numerical solution to our radioactive decay problem. We now consider how to translate that algorithm into a working computer program. Perhaps the first choice that one must make in writing a program is the choice of programming language. From the authors' experiences, there are many programming languages that are well suited for the kinds of problems we address in this book, and it is impossible for us to give example programs in all of these languages. However, it is possible to describe the structure of a program in a general way that is useful to users of many different languages. We will do this using a "language" known as *pseudocode.* This is not a precise programming language, but rather a description of the essential

[2] As you might expect, the quality of this estimate, i.e., its *accuracy*, will depend on the value of Δt. This is a very important issue that we will be discussing in some detail below and in Appendix A.

[3] Note that errors made each at each time step, i.e., each time (1.7) is used, will accumulate.

parts of an algorithm, expressed in "common" language. The idea is to give enough detail so that you (the readers of this book) can see how to translate each piece of pseudocode into the specific instructions of your favorite programming language. In most of this book, we will give our examples only in pseudocode. However, in this chapter we will work through an example using pseudocode along with actual codes in two popular languages, `Fortran` and `C`, so that you can see how the translation from pseudocode to an actual programming language can be done.[4] Working programs for many of the problems in this book are available in `Fortran`, `C`, and `Basic` at our Web site.[5].

While programming, like handwriting, is a highly individualized process, there are certain recommended practices. After all, as in handwriting, it is important that we be able to understand programs written by others, as well as those we ourselves have written! With that in mind, this book will try to promote proper programming habits. The (admittedly very loose) analogy between handwriting and programming can be carried one step further. The first thing you should do in writing any program is to *think*. Before writing any detailed code, construct an outline of how the problem is to be solved and what variables or parameters will be needed. Indeed, the pseudocode version of a program will often provide this outline. For our decay problem we have already laid the foundation for a numerical solution in our derivation of (1.7). This equation also contains all of the variables we will need, N_U, t, τ, and Δt. Our stated goal was to calculate $N_U(t)$, but since the numerical approximation (1.7) involves the values of N_U only at times $t = 0$, $t = \Delta t$, $t = 2\Delta t$, etc., we will actually calculate N_U at just these values of t. We will use an array to store the values of N_U for later use. An array is simply a table of numbers (which will be described in more detail shortly). The first element in our array, that is, the first entry in the table, will contain N_U at $t = 0$, the second element will be the value at $t = \Delta t$, and so on. Our general plan is then to apply (1.7) repetitively to calculate the values of $N_U(t)$.

The overall structure of the program consists of four basic tasks: (1) declare the necessary variables, (2) initialize all variables and parameters, (3) do the calculation, and (4) store the results.

EXAMPLE 1.1 Pseudocode for the main program portion of the radioactive decay problem

- Some comment text to describe the nature of the program.
  - ▷ Declare necessary variables and arrays.
  - ▷ *initialize* variables.
  - ▷ Do the actual *calculation.*
  - ▷ *store* the results.

---

[4]We are certainly not implying that everyone should use `Fortran` or `C`, but these are the authors' favorites.

[5]`www.physics.purdue.edu/~giordano/comp-phys.html`