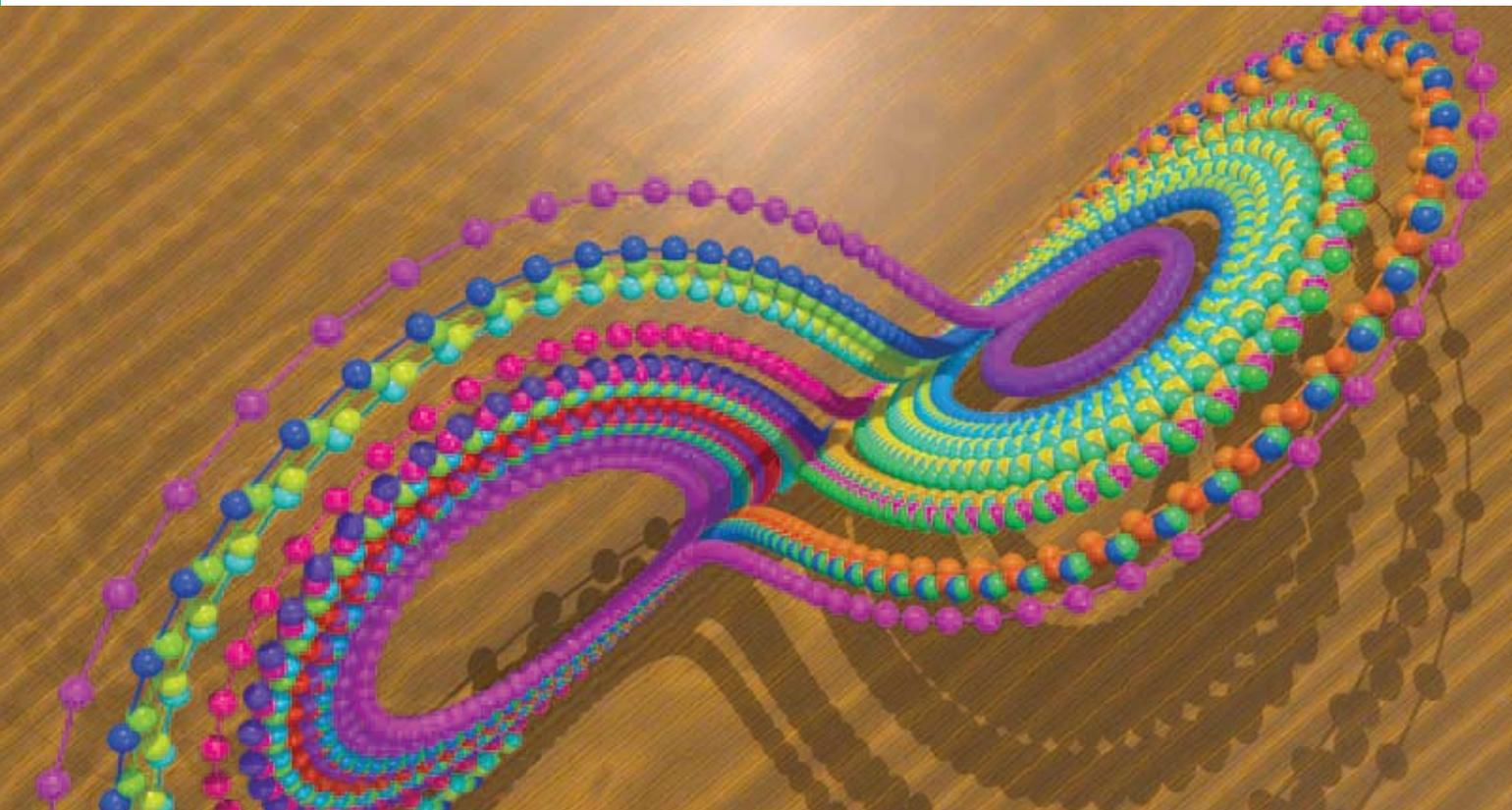


Computational Modeling

AND VISUALIZATION OF PHYSICAL SYSTEMS
WITH PYTHON

JAY WANG



WILEY

VICE PRESIDENT & DIRECTOR	George Hoffman
EXECUTIVE EDITOR	Christopher Johnson
SPONSORING EDITOR	Marian Provenzano
PRODUCT DESIGNER	Beth Tripmacher
ASSISTANT	Tai Wiggins
SENIOR DIRECTOR	Don Fowley
PROJECT MANAGER	Gladys Soto
PROJECT SPECIALIST	Marcus Van Harpen
PROJECT SPECIALIST	Nichole Urban
PROJECT ASSISTANT	Emily Meussner
SENIOR MARKETING MANAGER	Margaret Barrett
ASSISTANT MARKETING MANAGER	Puja Katariwala
SENIOR CONTENT MANAGER	Ellinor Wagner
PRODUCTION EDITOR	Ezhilan Vikraman
PHOTO RESEARCHER	James Russiello

This book was set in 10/12 Times LT Std by SPi Global and printed and bound by Strategic Content Imaging.

Founded in 1807, John Wiley & Sons, Inc. has been a valued source of knowledge and understanding for more than 200 years, helping people around the world meet their needs and fulfill their aspirations. Our company is built on a foundation of principles that include responsibility to the communities we serve and where we live and work. In 2008, we launched a Corporate Citizenship Initiative, a global effort to address the environmental, social, economic, and ethical challenges we face in our business. Among the issues we are addressing are carbon impact, paper specifications and procurement, ethical conduct within our business and among our vendors, and community and charitable support. For more information, please visit our website: www.wiley.com/go/citizenship.

Copyright © 2016 John Wiley & Sons, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923 (Web site: www.copyright.com). Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030-5774, (201) 748-6011, fax (201) 748-6008, or online at: www.wiley.com/go/permissions.

Evaluation copies are provided to qualified academics and professionals for review purposes only, for use in their courses during the next academic year. These copies are licensed and may not be sold or transferred to a third party. Upon completion of the review period, please return the evaluation copy to Wiley. Return instructions and a free of charge return shipping label are available at: www.wiley.com/go/returnlabel. If you have chosen to adopt this textbook for use in your course, please accept this book as your complimentary desk copy. Outside of the United States, please contact your local sales representative..

Library of Congress Cataloging in Publication Data:

Names: Wang, J. (Jianyi), 1962- author.
 Title: Computational modeling and visualization of physical systems with Python / J. Wang.
 Description: Hoboken, NJ : John Wiley & Sons, 2015. | Includes bibliographical references and index.
 Identifiers: LCCN 2015029743 | ISBN 9781118110225 (pbk.)
 Subjects: LCSH: Physics—Computer simulation. | Physics—Graphic methods. | Python (Computer program language)
 Classification: LCC QC30 .W33 2015 | DDC 530.0285/5133—dc23 LC record available at <http://lcn.loc.gov/2015029743>

The inside back cover will contain printing identification and country of origin if omitted from this page. In addition, if the ISBN on the back cover differs from the ISBN on this page, the one on the back cover is correct.

Printed in the United States of America
 10 9 8 7 6 5 4 3 2 1

To my Father and Mother,
who taught me that knowledge is to be revered.

Contents

Preface **ix**

1 Introduction **1**

- 1.1 Computational modeling and visualization, 1
- 1.2 The science and art of numerics, 2
- 1.3 Fundamentals of programming and visualization, 6
- 1.4 Exercises and projects, 14
- 1.A Floating point representation, 15
- 1.B Python installation, 17
- 1.C The Matplotlib `plot` function, 20
- 1.D Basic NumPy array operations, 21

2 Free Fall and Ordinary Differential Equations **27**

- 2.1 Free fall with Euler's method, 27
- 2.2 The Runge-Kutta (RK) methods, 32
- 2.3 System of first-order ODEs, 37
- 2.4 The leapfrog method, 43
- 2.5 Exercises and projects, 48
- 2.A Area preservation of the leapfrog method, 52
- 2.B Program listings and descriptions, 54

3 Realistic Projectile Motion with Air Resistance **57**

- 3.1 Visualization of ideal projectile motion, 57
- 3.2 Modeling air resistance, 58
- 3.3 Linear air resistance, 62
- 3.4 The Lambert W function, 67
- 3.5 Quadratic air resistance and spin, 70
- 3.6 Physics of ball sports, 73
- 3.7 Shooting methods, 80
- 3.8 Exercises and projects, 83
- 3.A Bisection and Newton's root finders, 87
- 3.B Program listings and descriptions, 89

4 Planetary Motion and Few-Body Problems 92

- 4.1 Motion of a planet, 92
- 4.2 Properties of planetary motion, 94
- 4.3 Precession of Mercury, 99
- 4.4 Star wobbles and exoplanets, 107
- 4.5 Planar three-body problems, 111
- 4.6 The restricted three-body problem, 116
- 4.7 Exercises and projects, 125
- 4.A Rotating frames and rate of change of vectors, 130
- 4.B Rotation matrices, 132
- 4.C Radial velocity transformation, 133
- 4.D Program listings and descriptions, 135

5 Nonlinear Dynamics and Chaos 144

- 5.1 A first model: the logistic map, 144
- 5.2 Chaos, 153
- 5.3 A non-linear driven oscillator, 157
- 5.4 The Lorenz flow, 163
- 5.5 Power spectrum and Fourier transform, 168
- 5.6 Fractals, 170
- 5.7 Exercises and projects, 174
- 5.A Program listings and descriptions, 179

6 Oscillations and Waves 184

- 6.1 A damped harmonic oscillator, 184
- 6.2 Vibrations of triatomic molecules, 188
- 6.3 Displacement of a string under a load, 194
- 6.4 Point source and finite element method, 199
- 6.5 Waves on a string, 204
- 6.6 Standing waves, 210
- 6.7 Waves on a membrane, 212
- 6.8 A falling tablecloth toward equilibrium, 215
- 6.9 Exercises and projects, 217
- 6.A Program listings and descriptions, 222

7 Electromagnetic Fields 226

- 7.1 The game of electric field hockey, 226
- 7.2 Electric potentials and fields, 228

- 7.3 Laplace equation and finite element method, 233
- 7.4 Boundary value problems with FEM, 242
- 7.5 Meshfree methods for potentials and fields, 247
- 7.6 Visualization of electromagnetic fields, 251
- 7.7 Exercises and projects, 256
- 7.A Program listings and descriptions, 261

8 Time-Dependent Quantum Mechanics 272

- 8.1 Time-dependent Schrödinger equation, 272
- 8.2 Direct simulation, 274
- 8.3 Free fall, the quantum way, 281
- 8.4 Two-state systems and Rabi flopping, 289
- 8.5 Quantum waves in 2D, 293
- 8.6 Exercises and projects, 299
- 8.A Numerical integration, 304
- 8.B Program listings and descriptions, 307

9 Time-Independent Quantum Mechanics 313

- 9.1 Bound states by shooting methods, 313
- 9.2 Periodic potentials and energy bands, 319
- 9.3 Eigenenergies by FDM and FEM methods, 320
- 9.4 Basis expansion method, 326
- 9.5 Central field potentials, 331
- 9.6 Quantum dot, 335
- 9.7 Exercises and projects, 343
- 9.A Numerov's method, 348
- 9.B The linear potential and Airy function, 349
- 9.C Program listings and descriptions, 351

10 Simple Random Problems 362

- 10.1 Random numbers and radioactive decay, 362
- 10.2 Random walk, 364
- 10.3 Brownian motion, 367
- 10.4 Potential energy by Monte Carlo integration, 369
- 10.5 Exercises and projects, 372
- 10.A Statistical theory of Brownian motion, 376
- 10.B Nonuniform distributions, 377
- 10.C Program listings and descriptions, 378

11	Thermal Systems	382
11.1	Thermodynamics of equilibrium,	382
11.2	The Ising model,	392
11.3	Thermal relaxation by simulated annealing,	404
11.4	Molecular dynamics,	406
11.5	Exercises and projects,	414
11.A	Boltzmann factor and entropy,	421
11.B	Exact solutions of the 2D Ising model,	422
11.C	Program listings and descriptions,	424
12	Classical and Quantum Scattering	428
12.1	Scattering and cross sections,	428
12.2	Rainbow and glory scattering,	432
12.3	Quantum scattering amplitude,	437
12.4	Partial waves,	439
12.5	Exercises and projects,	450
12.A	Derivation of the deflection function,	456
12.B	Partial wave analysis,	457
12.C	Program listings and descriptions,	459
	List of Programs	463
	Bibliography	467
	Index	471

Preface

Computer modeling has had a significant impact on the way we do science and physics, both in research and in teaching. We are able to study problems of all scales from atoms to galaxies, and of complexities that would be impossible without computer modeling. Computation has even led to an entirely new field of science, chaos. In many ways, computational modeling has become the third pillar of physics alongside experimentation and theory.

In this book, we introduce computational modeling and visualization of physical systems that are commonly found in physics and related areas. Our first and foremost goal is to introduce a framework that integrates model building, algorithm development, and data visualization for problem solving via scientific computing. Through carefully selected problems, methods, and projects, the student is guided to learning and discovery by actively doing rather than just knowing physics. By constructing models and algorithms, programming and testing them, and analyzing the results, we will gain insight, ask new questions, tweak the model as necessary, and change the parameters to test what-if scenarios analogous to turning knobs in a virtual experiment. Another goal is to broaden the scope and depth of problems that may be studied with computational modeling. Many fundamental physical systems, despite their apparent simplicity, are beyond reach without computer simulation. Take projectile motion with air resistance and quantum free fall, for example. Though the problems can be expressed in basic calculus, they are unsolvable analytically in closed-form solutions. Computer modeling enables us to not only solve these problems, but also comparatively study the differences and similarities of their behavior in classical and quantum mechanical realms. We also aim to integrate applied computational tools and methods with effective visualization techniques in the simulations, including in situ data graphing and real-time animation within standard and open-source frameworks.

We take a problem-centric approach to the presentation of the material. For most problems, we include sufficient background information or essential relations to make them self-contained as much as possible, as well as to indicate their general importance and relevance. To streamline the text, many of the details are given in appendices or exercises, allowing readers of diverse backgrounds to skip ahead or study the detail at their own pace. We have also created a Digital Online Companion (DOC) for in-depth and advanced topics. Following the description of a problem, we discuss model building, the appropriate computational methods and tools, and visualization techniques to the simulations. Most results are represented in graphical form directly from the simulation programs for analysis and discussion.

Graphical representation of data and effective visualization techniques, including real-time animations of three-dimensional objects, are standard parts of our simulations because they are essential to help interpret and analyze the results, especially in time-dependent studies. They also bring the simulations alive, making them more dynamic, instructive, and exciting.

The tight integration of advanced graphics and visualization into the simulations is made possible with standard, easy-to-use, open-source packages such as Matplotlib, SciPy, and VPython. These packages only require us to manipulate an object's attributes such as the physical position of a particle, so we can avoid low-level graphics programming and focus on modeling and proper techniques in using these tools. Except for schematic illustrations, most graphical representations and animations are created using techniques in the actual codes contained in the book. However, many programs are written such that the integrated graphics can be decoupled

(disabled), replaced with either Matplotlib output or results written to a file, without affecting the calculations.

Because coding is essential to understanding an algorithm or to gaining insight to a physical process, and the most effective way to learn to code is by studying examples, the reader is guided in the process of building over ninety fully working sample programs, with detailed explanations for most of them. Many of the coded algorithms form part of basic building blocks in scientific computing. The associated tools and methods, whether well-known or otherwise still being currently researched, share common elements found in computational research of more complex problems. Through building fully working programs complete with graphical output, the student not only will learn how to write codes for computation, but their finished product is suitable for demonstration as well.

We use Python as the default programming language to show concrete, working examples and to take several advantages it offers: being easy to learn and use, readable, flexible, and powerful. One can think of Python as the modern equivalent of BASIC, but with a large and growing body of open source libraries for common tasks in scientific computing such as those mentioned earlier. Even newcomers to Python can learn from examples and grow quickly, writing functional programs and being productive in about two to three weeks. However, the choice of a programming language can be highly personal. Readers with programming background in other languages should find most Python codes to be expressive, pseudocode-like, and readily adaptable.

Sample systems are drawn from across the fundamental areas of physical science including mechanical, electromagnetic, quantum, and statistical systems. Representative problems within a given topic or theme are organized by chapter. Most chapters start with an animated simulation related to the central topic and end with a brief summary. The chapters and topics are organized as follows. After a brief introduction in Chapter 1 to our approach to modeling and basic programming elements, we discuss methods of solving ordinary differential equations (ODEs) in Chapter 2 as it constitutes the beginning of our framework. Thematic studies begin with the familiar example of projectile motion with air resistance (Chapter 3), followed by planetary and few-body motion including exoplanet modeling (Chapter 4), and nonlinear dynamics and classical chaos (Chapter 5). We study waves and oscillations in Chapter 6 and electromagnetic fields in Chapter 7. In Chapters 8 and 9, we present simulations of time-dependent and time-independent quantum mechanics, respectively. We consider simple random systems in Chapter 10 and statistical and thermal dynamics in Chapter 11. Finally, we investigate classical and quantum scattering in Chapter 12.

To effectively study these topics, the student should be familiar with basic calculus and introductory mechanics for mechanical systems (Chapters 2 through 6). For quantum and thermal systems (Chapters 8, 9, and 11), some familiarity with concepts typically covered in a modern physics course such as the wave function, expectation value, eigenstates, and entropy, will be helpful. We also assume the student to have some programming background, as familiarity with Python is a plus but not required. Past experience has shown that students unfamiliar with Python can successfully learn from examples and become productive quickly. By the time Chapter 3 is finished, most students will have grasped the essential programming elements needed to carry on.

Most of the introductory topics are aimed at the undergraduate level (up to Chapter 7). Starting from Chapter 5 and increasingly toward Chapter 12, we include advanced and more in-depth topics in the DOC content. These topics are geared at upper undergraduate or graduate levels, including quantum chaos, particle transport, Bose-Einstein condensation, quantum transitions, inelastic scattering and atomic reactions.

Moreover, we put a special emphasis on the simulation of quantum systems, expanding the coverage over three chapters (Chapters 8, 9, and 12). Ideally, these topics are better studied after a first course in quantum mechanics. However, some sections should be accessible even without a quantum course, such as Sections 8.1 through 8.3 and Section 9.1, for which a basic

understanding of wave motion should be sufficient. Our motivation for including these topics is due to the fact that, unlike classical mechanics, quantum mechanics has far fewer analytically solvable problems and is less intuitive and less visual compared to classical motion. These factors cause considerable difficulties to students' understanding in introductory quantum mechanics. By simulating and visualizing the behavior of quantum systems such as the quantum oscillator or free fall, we are in a position to help effectively address some of the difficulties, pre- or postquantum class. We regularly use computer simulations to augment and enhance traditional physics classes such as quantum mechanics.

Although some chapters are interrelated, most from Chapter 3 on can be independently studied. Figure 1 shows the major dependencies of the chapters where the primary numerical methods (under the chapter number) are discussed, which are used in subsequent chapters. As the book progresses, most chapters are dependent on two previous chapters or less, except Chapters 9 and 12, which are dependent on five. There are several methods spanning multiple chapters, including ODE solvers (discussed in Chapter 2) for most time-dependent simulations in Chapters 3 through 9 and Chapter 11; the finite element method introduced in Chapter 6 and extended in Chapters 7 and 9; and other methods such as root finding (Chapter 3). In many cases, the reader may skip the methodology section without much disruption. For instance, the fast Fourier transform (FFT) discussed in Chapter 5 of DOC is lightly used in Chapter 6 to analyze fundamental oscillation frequencies, but heavily used in Chapter 8 for evolving quantum systems and for extracting momentum distributions. In both instances, the reader initially unfamiliar with FFT can still work through the simulations without worrying about the technical details of FFT, using it as a supplied library. The reader may come back and review it later for a fuller understanding if desired.

The instructor can pick what sections to cover within a given chapter in most cases. For example, the instructor may skip some sections on ODE solvers to save time (see Introduction in Chapter 2). However, this may be limited due to progression in methodology or conceptual

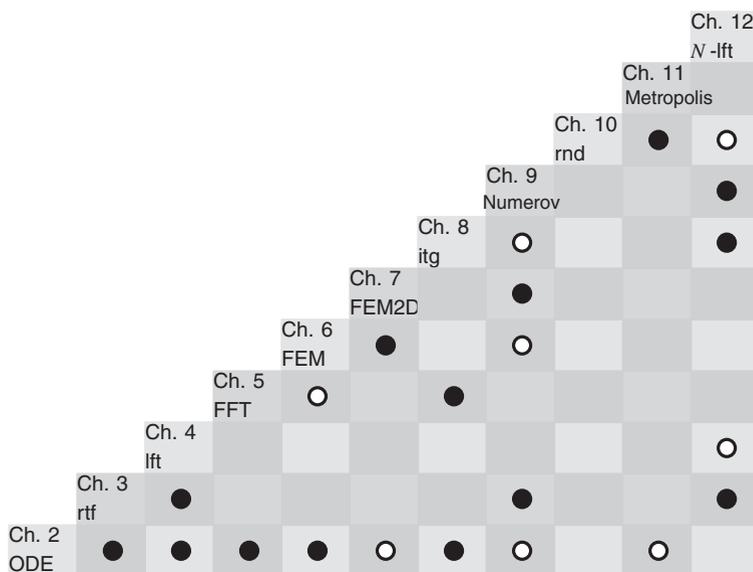


FIGURE 1 Major chapter dependencies at a glance. Open circles indicate minor or Digital Online Companion only dependencies. The abbreviations are: FEM—finite element method; FFT—fast Fourier transform; itg—numerical integration; lft—leapfrog with time-transformation; ODE—ordinary differential equations; rnd—random numbers; and rtf—root finding.

buildup. In such situations, indicated in the narrative and aided by Figure 1, the instructor may choose to discuss the prerequisite section lightly and leave further investigation to a project. Carefully constructed and tested projects given at the end of each chapter are an integral resource enabling the students to extend the models, make predictions, test them, analyze and present the results in project reports. Many in-depth projects, some found in the DOC, are designed for two-person teams to encourage peer learning and collaboration. There are also numerous exercises for the purpose of analytical derivations and proofs or simple programming tasks.

Throughout, references to the DOC content are indicated by the “S:” prefix, including the completely merged index. The DOC content and other electronic resources such as programs, data files, solution manual to select exercises, feedback and contact information, and installation instruction and files can be found at:

<http://www.wiley.com/college/wang> and
<https://github.com/com-py/compy/>

ACKNOWLEDGMENTS

I am indebted to Joachim Burgdörfer from whom I learned much of the craft of the trade during my PhD study. I also benefited from professional and personal interactions with Wolfgang Christian (*Davidson College*), Steven Gottlieb (*Indiana University, Bloomington*), Harvey Gould (*Clark University*), and Jan Tobochnik (*Kalamazoo College*), who have been pioneers in computational physics and education. I am grateful to Bruce Sherwood, the original developer of VPython and GlowScript, for being a constant resource and teacher. I am thankful to Steve Spicklemire (*University of Indianapolis*) for reviewing the manuscript, testing the programs, and making suggestions that substantially changed the structure of the codes on vector operations.

I owe a debt of deep gratitude to Michael Carlozzi (Writing and Reading Center, UMass Dartmouth) for his great generosity in time and effort to offer expert critique of the manuscript over two semesters that helped to significantly improve the writing, organization, and presentation of the material.

Many reviewers have provided critical and constructive feedback that improved the manuscript. I thank Brad Marston (*Brown University*) for extensive comments and valuable suggestions on not only modeling, but the validation and interpretation of computational results. I also thank Johnathan Barnes (*Salt Lake Community College*) for careful reading and annotation of part of the manuscript and Larry Engelhardt (*Francis Marion University*) for many useful discussions and suggestions, including code optimization. I sincerely acknowledge useful input and viewpoints from other reviewers, including Eric Ayers (*California State University, Chico*), Simon Billinge (*Michigan State University*), Alexander Godunov (*Old Dominion University*), Rainer Grobe (*Illinois State University*), Dawn Hollenbeck (*Rochester Institute of Technology*), Lev Kaplan (*Tulane University*), James McDonald (*University of Delaware*), Brianislav Nikolic (*University of Delaware*), Michael Roth (*University of Northern Iowa*), Steffen Vojta (*Missouri University of Science and Technology*), and Haowen Xi (*Bowling Green State University*).

In addition, I would like to thank Anne Caraley (*State University of New York at Oswego*), Henry Greenside (*Duke University*), and Timothy Roach (*College of the Holy Cross*) for their generous time and effort providing helpful input out of personal interest in the manuscript.

I wish to thank all my colleagues in the Department of Physics for their strong support and encouragement for the book project, including the granting of a sabbatical. I am very grateful to Robert Fisher and Gaurav Khanna who gave specific comments and valuable suggestions, and to Glenn Volkema, who tested and advised on the installation procedures. I also wish to thank Adam Hausknecht and Alfa Heryudono in the Department of Mathematics for helpful ideas and feedback, Andreas Ernst for discussions on fewbody problems, and Amit Tandon for source

references on fluid motion. I thank my own students in the computational physics course over the years who left their mark on the project.

Furthermore, I would like to express my thanks to the Wiley team for their able support and guidance: to editors Stuart Johnson for initiating and developing the project and to Jessica Fiorillo for expertly and energetically steering it to completion, to Mary O'Sullivan, Amanda Rillo, Gladys Soto, Marcus Van Harpen, Ezhilan Vikraman and Karen Slaght for production and logistics support.

Finally, I wish to thank my family for their unconditional love and support. I want to say thanks to my sons, Gregory and Adam, for giving advice and opinion whenever I asked, for helping with literature research, and for trying out some topics and projects in the book. Last but not least, a heartfelt thank-you to my wife, Huihua, for showing great understanding and patience throughout the long project, without which this project would not have been possible.

J Wang
Dartmouth, Massachusetts
www.faculty.umassd.edu/j.wang/

List of Programs

We have built many programs in this book. The following table summarizes these programs and dependencies. Standard libraries such as Matplotlib (2D), NumPy, and SciPy are omitted. Library abbreviations are: `fft`—fast Fourier transform (`fft.py`, Chapter 5); `fem`—finite element method (`fem.py`, Chapter 9); `fileio`—file input/output (`fileio.py`, Chapter 9); `itg`—numerical integration (`integral.py`, Chapter 8); `ode`—ordinary differential equation (`ode.py`, Chapter 2); `rtf`—root finders (`rootfinder.py`, Chapter 3); and `vpm`—VPython Modules (`vpm.py`, Chapter 6). In addition, we also note the use of 3D and animation libraries: `Axes3D`—Matplotlib 3D plotting; `am`—Matplotlib animation; and `vp`—VPython.

Page numbers with the “S:” prefix are entries from the Digital Online Companion.

Program	Description	Dependence
3body, 139	Three-body motion	ode, rtf, vp
balltoss, 58	Ideal projectile motion	vp
baseball, 89	Flight of baseball	ode, vp
bdipole, 268	Magnetic dipole field	vp
bem, 354	Basis expansion method	itg
bisect, 88	Bisection root finder	
boltzmann, 424	Boltzmann distribution	
bouncing_ball, 27	Bouncing ball	vp
brownian, 379	Brownian motion	am
coupled, S:112	Quantum transitions	ode, vp, vpm
ctmc, S:263	Classical ion-atom collisions	ode, rtf, vp, vpm
deflect, 459	Deflection function	itg, rtf
dipole, 270	Dipole distribution	Axes3D
earth, 93	Planetary motion	ode, vp
edipole, 269	Electric dipole radiation	vp
eigh, 194	Eigenvalue problem	
einsteinsolid, 383	Einstein solid class	
entropy, 387	Entropy of an Einstein solid	
equishare, 424	Thermal energy sharing	am
fem, 357	FEM library	
fft, S:44	Fast Fourier transform	
fractal, 182	Mandelbrot fractal	
freefall_euler, 39	Free fall with modular Euler	
freefall_plot, 30	Free fall with inline Euler	
gauss_elim, S:57	Gauss elimination	
gaussxw, 312	Gauss abscissa and weight	
hockey, 261	Electric field hockey	ode, vp
hydrogen, 355	Atomic structure	rtg
integral, 311	Numerical integration	

Continued on next page

Continued from previous page

Program	Description	Dependence
ising1d, 425	Ising model	
lambertw, 69	Lambert W function	
laplace, 262	Relaxation solutions	Axes3D, vp
laplacefem, 264	Laplace equation by FEM	Axes3D
laplacerbf, 266	Meshfree method for PDEs	
leapfrog, 45	Leapfrog method	
leapfrog_tt, 103	Time-transformed leapfrog	
leapfrog_ttN, S:224	N -body transformed leapfrog	
life, S:149	Game of life	
logisdiff, 179	Difference of logistic map	
logisticmap, 146	Logistic map iterates	
longwire, 269	Magnetic fields of a long wire	vp
lyapunov, 180	Lyapunov exponent	
mcint, 380	Monte Carlo integration	
md, 426	Molecular dynamics	ode, vp, vpm
mdf2py, S:192	Molecular dynamics F2Py	ode, nbodyf
mercury, 135	Precession of Mercury	ode, vp
meshhex, 360	FEM meshes of a hexagon	fileio
metropolis2, 400	Ising model in 2D	
motion, 7	1D motion and graphing	
nbody, S:266	N -body Coulomb interaction	
nbodyf, S:189	N -body interaction, Fortran	
newton, 89	Newton's root finder	
nns, S:135	Energy spectrum unfolding	
nonlindro, 180	Nonlinear pendulum	ode
odewrap, 55	SciPy ODE wrapper	
perioddbl, 150	Bifurcation diagram	
plane, 270	Plane electromagnetic wave	vp, vpm
poincare, 181	Poincaré map	ode
projectile, 62	Motion with linear drag	
qmdot, 358	Quantum dot	Axes3D, fem, fileio
qmfem, 353	Eigenenergies by FEM	
qmplane, 307	Quantum plane wave	am
qmscatt, 460	Central field scattering	ode
qmshoot, 352	Quantum shooting method	ode, rtf
qmwaves2d, 310	Quantum waves in 2D	vp, vpm
r3body, 142	Restricted 3-body motion	ode, vp
r3body_veff, 140	Lagrange points	Axes3D
range, 66	Range with linear drag	rtf
relax, S:59	Relaxation of a string	Axes3D, ode
relaxtd, S:187	Thermal relaxation	
rk2, 35	Runge-Kutta order 2	
rk4, 36	Runge-Kutta order 4	
rk45n, 55	Runge-Kutta-Fehlberg	
rk4n, 54	Nonvector RK4	
rvfit, 136	Radial velocity modeling	

Program	Description	Dependence
sdlf, 307	Quantum wavepacket motion	ode, vp, vpm
sho, 184	Simple harmonic oscillator	vp
sho_lf, 46	Oscillator with leapfrog	ode
shoot, 82	Shooting method	ode, rtf
slinky, S:60	Motion of a slinky	ode, vp, vpm
splitop, 309	Quantum free fall	vp, vpm
squarewell, 351	Visualizing eigenstates	ode, vp, vpm
stringfdm, 197	Displacement of a string	
stringfun, 222	Fundamental frequencies	
tablecloth, 224	Falling tablecloth	ode, vp, vpm
transport, S:188	Particle transport	
triatom, 222	Triatomic vibration	vp
tripwf, 359	Triangular mesh plots	fileio
vpm, S:61	VPython modules	vp
walk2d, 378	Random walk in 2D	
waves, 208	Waves on a string	vp, vpm
waves2d, 223	Waves on a membrane	vp, vpm

Index

Page numbers with the “S:” prefix are entries from the Digital Online Companion.

- accelerated relaxation, 231
 - Gauss-Seidel method, 231
- air resistance, 58–60
 - linear, 62
 - quadratic, 70
- animation
 - ant raids, S:145
 - atomic collision, S:228
 - baseball, realistic flight, 75
 - bouncing ball, 27
 - dipole radiation field, 253
 - electric field hockey, 226
 - falling tablecloth
 - mechanical, 215
 - thermal, 405
 - game of life, S:138
 - Halley’s comet, S:15
 - laser-driven coherent state, S:94
 - magnetic field, 252
 - Mandelbrot fractals, 172
 - N*-body simulator, 410
 - oscillation of slinky, S:52
 - plane electromagnetic wave, 254
 - planetary motion
 - Earth, 93
 - Mercury, 98
 - precession of Mercury, 101
 - projectile motion, 57
 - quantum revival, 2D, 293
 - quantum scattering from a barrier, S:79
 - quantum wavepacket
 - in free fall, 285
 - in SHO, 278
 - relaxation of electric potential, 229
 - shooting for eigenenergies, 314
 - simple harmonic oscillator, 184
 - soccer, 80
 - spin flip, 400, 418
 - strange butterfly attractor, 166
 - thermal equilibrium, 424
 - Thomson problem, S:67
 - three-body motion
 - choreography, 114
 - collinear, 113
 - Trojan asteroids, 122
 - wave on a membrane, 214
 - ants raiding pattern, S:142, S:147
 - atomic form factor, S:214
 - elastic, S:220
 - atomic reaction, S:211–S:221
 - antiproton impact, S:247
 - capture, S:229, S:232
 - cross section, S:230, S:233
 - Thomas mechanism, S:234
 - excitation, S:162, S:214
 - cross section, S:215, S:216, S:230
 - electron impact, S:245
 - ionization, S:162, S:218
 - cross section, S:218, S:230
 - electron impact, S:248
 - free-particle model, S:220
 - positron impact, S:248
 - atomic structure, S:65, *see also*
 - hydrogenic atom
 - atomic units, 273
 - attractor, 147, 160, 165
 - dimension, *see* fractal
 - band matrix, 283
 - representation, 284
 - solver, *see* SciPy
 - baseball, 73
 - animation, 75
 - curveball, 76
 - drag coefficient, 61
 - lift coefficient, 73
 - basis expansion method, 327
 - box basis, 328
 - half-open space, 331
 - SHO basis, 329
 - bifurcation, 150, *see also* chaos
 - binomial
 - coefficient, 414
 - distribution, 365
 - Bohr model, 334, S:225, S:232
 - Boltzmann distribution, 391, 399
 - Bose-Einstein condensation, S:170
 - chemical potential, S:171
 - critical temperature, S:172
 - scattering length, S:206
 - bound states, 313
 - central field potentials, 332
 - double square well, 318
 - gerade and ungerade, 327
 - Morse potential, 331
 - periodic multiple wells, 320
 - square well, 318
 - boundary value problem
 - Dirichlet boundary condition, 202, 228, 322, 337
 - mixed boundary conditions, S:135
 - Neumann boundary condition, 202, 228, 256
 - Brownian motion, 276, 364, 367–369, 376
 - simulator, 380
 - C/C++, 11, 26, 276, S:193
 - catenary, S:47
 - celestial mechanics, 92
 - central field approximation, 104, 332, S:165
 - central field motion, 94
 - centrifugal
 - force, 117, 119, 132
 - potential, 95, 117, 119, 334, S:127
 - chaos, 153
 - bifurcation, 151, 162
 - kicked rotor, S:19, S:27
 - Lorenz model, 164
 - Lyapunov exponent, 155
 - nonlinear driven oscillator, 157
 - Poincaré map, 160, 167, S:134
 - Poincaré surface of section, 162
 - stadium billiard, S:22, S:28
 - strange attractor, 165, 178
 - time scale, 159
 - weather system, 165
 - chemical potential, S:170
 - circle
 - approximation of π , 375
 - midpoint drawing algorithm, S:71
 - quantum dot, S:118
 - classical scattering, *see* scattering
 - classical trajectory Monte Carlo, S:221–S:236
 - animation, S:228
 - microcanonical ensemble, S:225, S:260
 - straightline approximation, S:229, S:248
 - coherent state, 293, S:88
 - measurement problem, 293
 - comet, S:15–S:18
 - Halley’s, S:15
 - ISON, S:15
 - commutator, S:76
 - Coriolis effect, 117, 120
 - coupled channel method, S:86, S:90, S:110

- Crank-Nicolson method, 282
 Curie's law, 416
 Cython, 11, 276, 427, S:110, S:193
- debugging, 13
 deflection function, *see* scattering
 density of states, S:171, S:258
 differentiation operator, 248, *see also* radial basis function
 dipole selection rule, S:93
 Dirac δ atom, 325, 348
 δ comb, S:129
 δ molecule, 325, 345, S:128
 cusp, 325
 displacement of a string, 194
 drag force, *see also* air resistance
 Brownian motion, 367
 coefficient, 60
 empirical formula, 61
 quadratic, 60
 viscosity, 59
- eccentric anomaly, 109, S:260
 eccentricity, 96, 456
 Ehrenfest theorem, 280, 299, S:96
 eigenvalue problem, 192, 211, 322, 328
 generalized, 191, 324, 337, 360
 Jacobi transformation, 192
 Einstein solid, 382–390
 energy distribution, 384
 entropy, 387, 388, 416
 interacting systems, 388, 417
 temperature, 392
 electric field hockey, 226
 electric potentials and fields, 228
 disk in a box, 249
 parallel plates, 229
 unit square, 244
 electromagnetic waves
 dipole radiation, 253
 plane waves, 254
 electrostatic equilibrium
 on a sphere, S:65
 plum-pudding model, S:65
 electrostatic potential energy, 369, 375
 energy band, 321
 entropy, 387, 422
 Einstein solid, 387
 heat and temperature, 398
 Ising model, 397
 paramagnetic system, 416
 envelope function, S:92, S:96
 equipartition theorem, 376, 412
 error, *see* numerical error
 Euler method, 29
 Euler rotation, S:261
 Euler-Cromer method, 48, 185, 222
 evolution operator, 282
 approximate, S:77
- exoplanets, 107–111
 HD 139357, 108
 HD 3651, 110, 129
 modeling RV datasets, 109
 radial velocity method, 107
 expectation value, 279, 324, S:90, S:96
- F2Py, 11, 276, 427, S:110, S:191, S:193, S:247, S:266
- falling tablecloth
 mechanical, 215
 thermal, 405
- fast Fourier transform, 170, S:30–S:44
 aliasing, S:40
 iterative FFT, S:38
 Nyquist frequency, 208, S:44
 Parseval's relation, 169
 positive and negative frequencies, S:42
 recursive FFT, S:34
 two-dimensional, 295
 wave function, 287
- fast multipole method, S:193
- finite difference method
 displacement of a string, 196
 error, 245
 Laplace equation, 229
 quantum eigenenergies, 321
 standing waves, 210
 waves on a membrane, 212
 waves on a string, 206
- finite element method
 accuracy, 245
 basis functions, 199, 233
 building system matrix, 239
 data structure, 241
 Dirac δ atom, 325, 348
 Dirac δ molecule, S:129
 displacement of a string, 199
 error, 340
 FEM library, 338
 Laplace equation, 233
 mesh file format, 362
 mesh generation, 237, 241, 265, 338
 mixed boundary conditions, S:135
 nodes and elements, 235
 quantum dot, 336
 Schrödinger equation, 322
 stiffness matrix, 203
 tent functions, 233, 325
- fitting, 110
- fixed-point number, 15
- floating point, 16
 bit shifting, 17
 byte string, 16
 machine accuracy, 3, 17
 mantissa, 3, 16
 phantom bit, 16
 round-off error, 17
- football, 86
 Fortran, 11, 194, 276, S:191
- Fourier transform, 169, *see also* fast Fourier transform
- fractals, 170
 Cantor set, 175
 correlation dimension, 172
 Hausdorff dimension, 171
 Julia set, 176
 Koch curve, 171
 Mandelbrot fractals, 172
 Sierpinski carpet, 176
- free fall, 29
 animation, 27
 Euler's method, 30
 momentum profile, 289
 quantum mechanical, 281, 285
 Runge-Kutta methods, 35
- game of life, S:137
- Gauss elimination, S:55
 Gauss-Jordan method, S:56
 Gauss-Seidel method, S:56
- Gaussian distribution, 374, S:140
- golden mean, 5, 314
 recursion, 5
- golf, 77
 drag and lift, 77
- Green's function, S:199
- Halton sequence, 249
 heat capacity, 391
 Heisenberg's uncertainty principle, 208, 273, 289, 342, S:44
- Hilda asteroids, *see* restricted three-body problem
- Hooke's law, 99, 188, 218
 hydrogen molecule, vibrational states, 331
 hydrogenic atom, 332–336
 l degeneracy, 332
 angular probability density, 335
 Hulthén potential, 349
 modified potential, $1/r^{1+\epsilon}$, 332, 348
 radial equation, 332
 radial probability density, 334
 radial wave function, 333, S:106, S:256
 screened Coulomb potential, 348
 shell structure, 334
- ideal gas law, 413
 importance sampling, 371
 installation, 17–18
 integral equation, S:198
 IPython, 2, 17, 19, 21
 IVisual, 12
 Matplotlib inline, 11
 Ising model, 392–399
 1D, 392
 2D, 399–404, 419
 3D, S:179
 antiferromagnetism, 392, S:178, S:187
 critical temperature, 400, 403

- energy, 395, 400
- entropy, 397, 398, 416, 418, 419
 - computation, 418
- exact solution in 2D, 422
- ferromagnetism, 392
- heat capacity, 397, 402
- hysteresis, 419, S:181
- magnetization, 395, 400
 - staggered, 424, S:178
- mean field approximation, S:177, S:180, S:181, S:184
- partition function, 416
- phase transition, 400
- spin domains, 400
- toward equilibrium, 395
- IVisual, 12, 17
- Jupiter, *see* precession of Mercury, *see also* restricted three-body problem
 - pull on the Sun, 107
- Kansa's method for PDEs, 248, *see also* radial basis function
- Kepler orbit, 96, 108
 - Kepler's third law, 98
 - planets, 96
- Kepler's equation, 110, S:260
- Lagrange points, 119, S:233
- Lambert W function, 67–70, 345, S:129
 - approximate formulas, S:7
 - Bose-Einstein condensation, S:171
 - Dirac δ molecule, 326
 - evaluation, 68
 - projectile motion, linear drag, 69
- laminar flow, 60
- Langevin equation, 367, 376
- Laplace equation, 228
 - additivity rule, 258
- Laplace operator
 - nine-point discretization, 258
- laser-electron interaction, S:92
 - strong fields, S:98
- leapfrog method, 43–48, 52
 - area-preserving, 44
 - space discretized, 274
 - time dependent, S:261
 - time transformation, 101
 - N -body system, S:222
- least square fitting, 110, 137
- Lennard-Jones potential, 407, S:132
- Levinson theorem, 441, 448, 451, 454
- lift force, 72, *see also* Magnus force
- linear combination of atomic orbitals (LCAO), 320, 327, S:129
- linear interpolation, 85, 347
- Lippmann-Schwinger equation, S:200
- logistic map, 144–153
 - bifurcation, 151
 - Feigenbaum number, 151, 152
 - fixed points, 147
 - Lyapunov exponent, 156
 - period doubling, 150
 - renormalization, S:28
- Lorenz flow, 163, *see also* chaos
- Lyapunov exponent, 155, *see also* chaos
- magnetic field, 251
 - closed loop, 251
 - long wire, 252
- magnetization, 395, 400, 416
- Magnus force, 72
 - lift coefficient, 73
- Matlab, 11
- Matplotlib, 11, 20
 - 2D plots, `plot`, 8, 20–21, 31
 - 3D plots
 - `Axes3D.plot_surface`, 142, 263, 266, 270
 - scatter, 178, S:60
 - animation, 307, 380, 424, S:148
 - aspect ratio, 142, 270
 - axis
 - label, 8
 - limit, 138, 182
 - off, 268
 - semilog scale, 179
 - width, 21
 - bitmap images, `imshow`, 183, 268, 311, 424, S:150
 - color, 20
 - colorbar, 268
 - configuration records, 21
 - contour
 - filled, `contourf`, 454, S:96
 - lines, `contour`, 142, 263, 266
 - error bar, 138, 385
 - font size, 21
 - frame
 - off, 268
 - spacing, 21
 - histogram, 425, S:136
 - IPython inline, 11
 - legend, 198, 223, S:114
 - frame off, S:136
 - line
 - style, 20
 - width, 21, 142
 - marker, 20
 - color, S:60
 - multiple plots, `subplot`, 181, 182, 311
 - polar plot, 452
 - pylab, 11
 - step plot, `step`, S:117
 - text label, 20
 - \LaTeX math mode, 142
 - tick marks, 311, 461
 - triangular mesh plot, `tripcolor`, 259, 266, 360
 - triangular mesh surface, `plot_trisurf`, 360
 - vector fields, `quiver`, 142, 263, 269, 379
- Maxwell distributions, 410
- mean free path, S:161
 - energy dependent, S:183
- Mercury, *see* precession of Mercury
- meshfree method, 247, *see also* radial basis function
- MeshPy, 266, 338
- Metropolis algorithm, 393–394, 398–399
- model building, 60
- molecular dynamics, 406–410, S:222
 - close-neighbor interaction, 409
 - equipartition theorem, 412
 - ideal gas law, 413
 - initial condition, 410, 427
 - Maxwell distributions, 412, 420
 - optimization, 427, S:191
 - periodic boundary condition, 408–410
 - pressure, 412
 - second virial coefficient, 413, 421
 - units, 407
- Monte Carlo integration, 307, 369, 375
 - error, 371
 - hit-or-miss method, 375
- Monte Carlo simulation
 - ants, S:145
 - Einstein solid, 382
 - nuclear decay, 364
 - particle transport, S:160
 - simulated annealing, 404
 - falling tablecloth, 405
 - hanging chain, S:153
 - hanging tablecloth, 419
 - traveling salesman problem, S:157, S:181
- Morse potential, 331, 407, S:131
- Navier-Stokes equations, *see* Lorenz flow
- Newton
 - second law, 27
 - third law, 72
- nuclear decay, 363
- Numba, 11, 183, 276, S:110, S:193
- numerical differentiation
 - first order, 29
 - midpoint method, 33
 - second order, 196
- numerical error, 3
 - global error, 31
 - in energy, 47
 - round-off, 4
 - truncation, 4
- numerical integration, 304, *see also* Monte Carlo integration
 - Gaussian, 306, S:107
 - abscissa and weight, 312
 - multiple integral, 307

- numerical integration, (*continued*)
 - Simpson's rule, 305
 - trapezoid rule, 304
- Numerov's method, 320, 349
 - first derivative, 350
 - logarithmic scale, S:127
- NumPy, 10, 21–26
 - advanced indexing, 23
 - row or column swap, 25, 264, 362
 - with boolean array, 24, 138, 142, 410, 427
 - with integer array, 23, 265
- argmin, 138
- array creation, 21
- broadcasting, 22, 76, 225, 410, 427
- concatenate, 138, 266, 289, 354, 361, S:43
- conversion to list, 362
- copy, 23
- data type, 21, 284
- diagonal, 198, 223, 355, 356
- dot product, 103, 266, 355
- element insertion or deletion, 22, 198, 223, 265, 271, 356, 360
- element-wise operations, 22, 25, 138, 209, 212–214, 225, 229, 268, 277, 311, 410, 420, 427, S:58, S:60
- F2Py, 11
- FFT, S:86
 - in 2D, 295
- flatten, 265
- gradient, 142, 263
- histogram, 420, S:189
- linspace, 137, 141
- matrix multiplication, 355
- maximum element, S:58
- meshgrid, 141
- nearest difference, 138
- outer method, 138
- outer product, 311, S:58
- random distribution, 363, 380
- reshape, 264–266
- row and column convention, 142, 263
- row and column insertion or deletion, 25, 266, S:130
- shape, 24
- slicing, 22, 142, 209, 212–214, 225, 229, 263, 265, 284, 352, S:60
- sorting, 266
- stacking, S:60
 - column, 268, 271, 361
 - depth, 224, 264
- summing arrays, 224, 301, 325, 410, 420, 427
- take, 266
- transpose, 263
- truth array, 23, 138, 142, 264, 362
- universal functions (ufunc), 10, 25, 26, 141, 325
- vector operations, 75
- vectorizing functions, 26, 325, 352
- object-oriented programming, 9, 380, 383
- orbiting, S:195, *see also* scattering
- ordinary differential equation, 27, *see also* Euler, leapfrog, Numerov, and Runge-Kutta methods
 - implicit method, 43
- oscillation
 - damping, 185
 - resonance, 187
 - RLC circuit, 185
- paramagnetic system, 390, 415, 417
- partial differential equation, 184, *see also* Laplace, Schrödinger, and wave equations
- particle transport, S:160
 - angular scattering, S:182
 - energy deposition, S:161, S:169
 - energy-dependent mean free path, S:183
 - range distribution, S:164, S:168
- partition function, 391
 - harmonic oscillator, 392
- phase transition, 400
 - Ising model, 400
- Ping-Pong, 78
 - spin effects, 79
- planetary motion, 92–99
 - open orbits, 98
 - properties, 94
 - simulation, 93
 - units, 97
- Pluto, *see* restricted three-body problem
- Poincaré map, *see* chaos
- Poisson distribution, 364, 377, S:122
- Poisson equation, 228
- power spectrum, 168, 189
- precession of Mercury, 99–107
 - by other planets, 104
 - oscillations, 104
 - relativistic, 100, 103
 - scaling law, 105
- probability density, 281
- profiling, 260, 277, S:109
- program profiling, S:85
- programs list, 463
- projectile motion, 57
 - linear drag, 62
 - quadratic drag, 70
 - visualizing, 57
- pseudospectral method, S:78
- Python, 7
 - 2.7x vs. 3.xx compatibility, 9
 - assignment and type, 8
 - complex number, 182, 307
 - conditional, 8
 - deep copy, S:148
 - eval function, 137
 - exception, 360
 - file I/O, 9, 137
 - formatting string, 136, 145
- global variables, 9, 36, 318
- IDLE, 20
- indentation, 8
- inline if, 9, 138, 145
- input, 8
- installation, 17
- lambda function, 325, 361
- list, 8
 - append, 8
 - concatenate, 267, 384
 - count, 425
 - delete element, S:148
 - nested, 265
 - slicing, 23
 - sorting, 266
 - vs. ndarray, 23, 384
 - online help, 10, 21
 - operator overloading, 384
- pickle file handler, 360
- profiling, S:109
- random integer, 384
- random number, 362
- speed boost, 11, 183, 216
- with-as statement, 137
- quantum chaos, S:117–S:125
 - chaoticity, S:125
 - energy level statistics, S:121
 - nearest neighbor spacing, S:121
 - histogram, S:124
 - scars, S:125
 - spectrum unfolding, S:122
 - stadium billiard, S:117
- quantum dot, 336–344
 - circle, 349, S:128
 - degeneracy, 342
 - energy level distribution, S:117
 - hexagon, 341
 - isosceles right triangle, 338
 - stadium, S:118
 - triangle, 346
 - wave function, 340, 343, S:119, S:125
- quantum mechanics, *see* Schrödinger equation
- quantum quilt, 302
- quantum revival, 295, 297
 - revival time, 297
 - semiclassical limit, 298
- quantum scattering, 437
 - T*-matrix, S:212
 - amplitude, 438, 442, S:200, S:211
 - atomic form factor, S:214
 - Born approximation, 448, 450, S:201, S:206, S:209, S:213
 - Buckingham potential, 455
 - cross section, 442, 444, 449, S:212
 - elastic, S:245
 - Fermi's golden rule, S:201
 - Gaussian potential, 455
 - hard sphere, 443
 - shadow effect, 446

- Hulthén potential, 455
- inelastic, S:211
- optical theorem, S:238
- partial wave expansion, 441–443
- phase shift, 439, 442, 447, 448, S:206, S:208, S:209
- potential barrier, S:79
- potential well, S:83
- Ramsauer-Townsend effect, 455, S:205
- resonance, 440, S:84
- scattering length, S:203, S:242
- square spherical barrier, 447, 451
- square spherical well, 451
- WKB approximation, S:207, S:209
- Yukawa potential, 446, S:166, S:201, S:209
- quantum transitions, S:86
 - amplitudes, S:88
 - dipole allowed, S:98, S:216
 - dipole forbidden, S:98, S:216
 - in hydrogen, S:105
 - in the SHO, S:104
 - laser driven, S:91
 - multiphoton transition, S:98
 - occupation probability, 290, S:88, S:96
 - Rabi flopping, 291
 - two-photon transition, S:98
 - two-state system, 289
- Rabi flopping, 291
 - in hydrogen, S:106
 - Rabi frequency, 292
 - rotating wave approximation, 292
- radial basis function, 247
 - collocation method, 248, S:71
 - differentiation operator, 249, 250
 - Gaussian and multiquadric RBF, 247
 - scattered data interpolation, 247
 - shape parameter, 247, 250
- radial velocity method, *see* exoplanets
- radial velocity transformation, 133
- random number, 362
 - correlation and moment tests, 362
 - integer, 384
 - nonuniform distribution, 374, 377
 - Lorentzian, 374
 - rejection method, 378
 - transform method, 377
 - seed, 362
 - uniform range, 374
- random walk, 364
 - binomial distribution, 365
 - in 2D, 366
- recursion
 - stability, 6, 207, S:238
- reflection coefficient, S:81, S:83
- restricted three-body problem, 116–121
 - Earth-Moon system, 119
 - Hilda asteroids, 123, 130
 - Lagrange points, 119
 - orbital resonance, 123
 - Pluto libration, 123, S:14
 - Pluto’s motion, 123
 - Sun-Jupiter system, 121
 - Sun-Neptune system, 123
 - units, 117
- Reynolds number, 60
- root finding, 64
 - bisection, 64, 87, 318
 - false position, S:6
 - Newton’s method, 65, 89, 176
 - SciPy equation solver, 64, 83, 344
 - secant method, S:5
- rotating frame, 130
- rotation matrix, 132
- round-off error, *see* numerical error
- Runge-Kutta methods, 32
 - characteristic time, 42
 - non-vectorized, 42
 - SciPy wrapper, 55
 - step size control, 42
- Runge-Kutta-Fehlberg method, 42, 350
- Runge-Lenz vector, 100, 103
- Rutherford scattering, 431, S:66
 - cross section, 432
- Rydberg states, 334, S:236
- scattering, 428, *see also* quantum scattering
 - cross section, 430, S:253
 - deflection function, 428, 456, S:208
 - Yukawa potential, 436
 - glory, 436, 449
 - impact parameter, 428, S:197
 - orbiting, S:195, S:240
 - plum potential, 433, 460
 - rainbow, 433, 435, 436, 449, 450, 452
 - Snell’s law, 450, 453
 - square spherical barrier, 453
 - square spherical well, 452
 - Yukawa potential, 436
- scattering length, S:203, *see also* quantum scattering
- Schrödinger equation, time dependent, 272, *see also* wavepacket
 - average position, 279, 298
 - boundary effects, 286
 - coupled channel method, S:86
 - direct simulation, 274
 - periodic boundary condition, 277
 - split evolution operator, S:78
 - split-operator method, 283
- Schrödinger equation, time independent, 313, *see also* bound states
 - animated eigenstates, 314
 - basis expansion method, 327
 - discrete energies, 212, 315
 - integral equation, S:199
 - matching condition, 316
 - pseudo-continuum states, 322, 329
 - shooting methods, 315
- Schrödinger’s cat, 293
- SciPy, 10
 - Airy function and its zeros, 351, 356
 - band eigenvalue solver, 349, 355, 360
 - band matrix solver `solve_banded`, 283
 - Bessel function
 - spherical, 447
 - zeros, S:128
 - BLAS and LAPACK, 194
 - combination, 416, 417
 - eigenvalue solver `eigh`, 194, 223, 355
 - elliptic integral, 423
 - gamma function, 415
 - Hermite polynomial, 356, S:131
 - integration, 312, S:243
 - Lambert *W* function, 68
 - least square fitting, 110, 137
 - Legendre polynomial, 461
 - linear system solver `solve`, 197, 198, 266
 - ODE solvers, 27, 42, 55
 - orthogonal polynomial, 312
 - root solver `fsolve`, 64, 83, 344
 - sparse eigenvalue solver `eigsh`, 322, 325, 355–356, 360
 - Weave, 11
- self-consistent methods, 228, S:48
 - relaxation error, 232
- shooting methods, 80, 315, 318
- simple harmonic oscillator
 - animation, 184
 - classical, 45
 - quantum mechanical, 278
- Simpson’s rule, 280
- Snell’s law, *see* scattering
- snub cube, S:69
- soccer, 79
- space discretized leapfrog method, 274, 276
 - normalization error, 281
 - stability, 276
- sparse matrix, 243
- special function
 - Airy function, 330, 346, 350, S:130
 - zeros, 351
 - Bessel function, 342, S:128
 - modified spherical, 451
 - recurrence, 457
 - spherical, 441, 447, 457, S:251
 - zeros, S:128
 - elliptic integral, 402, 423
 - Hermite polynomial, 329, S:131
 - Laguerre polynomial, S:127
 - Lambert *W* function, 326
 - Legendre polynomial, 306, S:108
 - in plane waves, 441
- spectral staircase, S:115
- spherical harmonics, 332, S:105
 - addition, S:251
 - in plane waves, S:251
 - orthogonality, S:251
- spinning balls, 72
- spin parameter, 73

- split evolution operator, S:75, S:78
- split-operator method, first order, 282
 - error, 286
- stiff differential equation, 43
- Stirling's approximation, 415
- Stokes' law, 59
- stopping power, S:162
- symplectic methods, 47, *see also* leapfrog method
 - first order, 48
- SymPy, 11, 17, 20, 217
 - factorization, 175
 - integrate, 49, 450
 - Lambert W function, 68
 - physics
 - hydrogen atom, S:106
 - quantum oscillator, 329
 - series, 126
 - solve, 84, 344, 345
- table tennis, 78, *see also* Ping-Pong
- temperature
 - Curie, 402
 - Einstein solid, 390, 392
 - negative, 390, 418
- thermodynamics, 382
 - second law, 388
 - third law, 397
- Thomson model, S:65
- three-body problem, 111–116
 - choreography, 114
 - dynamics, S:228, *see also* classical trajectory Monte Carlo
 - Euler's collinear motion, 111
 - Euler's quintic equation, 113
 - planar motion, 111
- traffic flow, S:139
 - fundamental diagram, S:140
 - hybrid model, S:146
- transmission coefficient, S:81, S:83, S:103
- tridiagonal matrix, *see* band matrix
- Trojan asteroids, 121
- truncation error, *see* numerical error
- tunneling, S:82, S:234
- turbulent flow, 60
- unitarity, S:76
- Verlet, *see* leapfrog method
- vibration, 188
 - normal modes, 190
 - string, 204
 - triatomic molecules, 188
- virial theorem, 334, 349
- viscosity, 59–60
 - air, 83
- visualization, 2, 27, 75, 253
- von Neumann stability, 207
- VPython, 12
 - arrow, 90, 136, 140, 269
 - axis flip, 271
 - box, 12, 28, 90
 - camera angle, 13
 - curve, 90
 - faces, 258
 - helix, 185
 - in GlowScript, 20
 - IVisual, 12
 - key detection, 90, 127
 - label, 90, 126, 136, 262, 311
 - box, 271
 - light source, 94
 - make_trail, 94, 261
 - making movies, 20
 - opacity, 90, S:67
 - rate requirement, 12
 - retain, 58
- ring, 143
- rotate, 13, 90
- sphere, 28
- vector operations, 75, 136, 268, 270
- VIDLE, 20
- VPython modules (VPM), 9, 94, 209, 224, 225, 258, 308, 311, S:52, S:61–S:64
- wave function, 272
 - laser driven, S:96
 - momentum space, 287
 - normalization, 281
 - conservation, 276
 - plane wave, 273
 - scarring, S:125
 - scattering, 444
- wavepacket, 278, 285
 - broadening, 285
 - in 2D, 294
 - momentum distribution, 287, 295
 - optical diffraction, 296
 - refocusing, 278
 - scattering from a barrier, S:79
 - scattering from a well, S:85
 - self interference, 286
- waves, 204–210
 - on a membrane, 212
 - standing, 206, 209, 210, S:85
 - traveling, 205, 209, 220
 - wave equation, 205, 212
- Weave, 11, 276, 427, S:110, S:193
- Weyl formula, S:116
- Wigner distribution, S:122
- WKB approximation, S:207, *see also* quantum scattering
- Yukawa potential, 348, 436, 446, S:201

