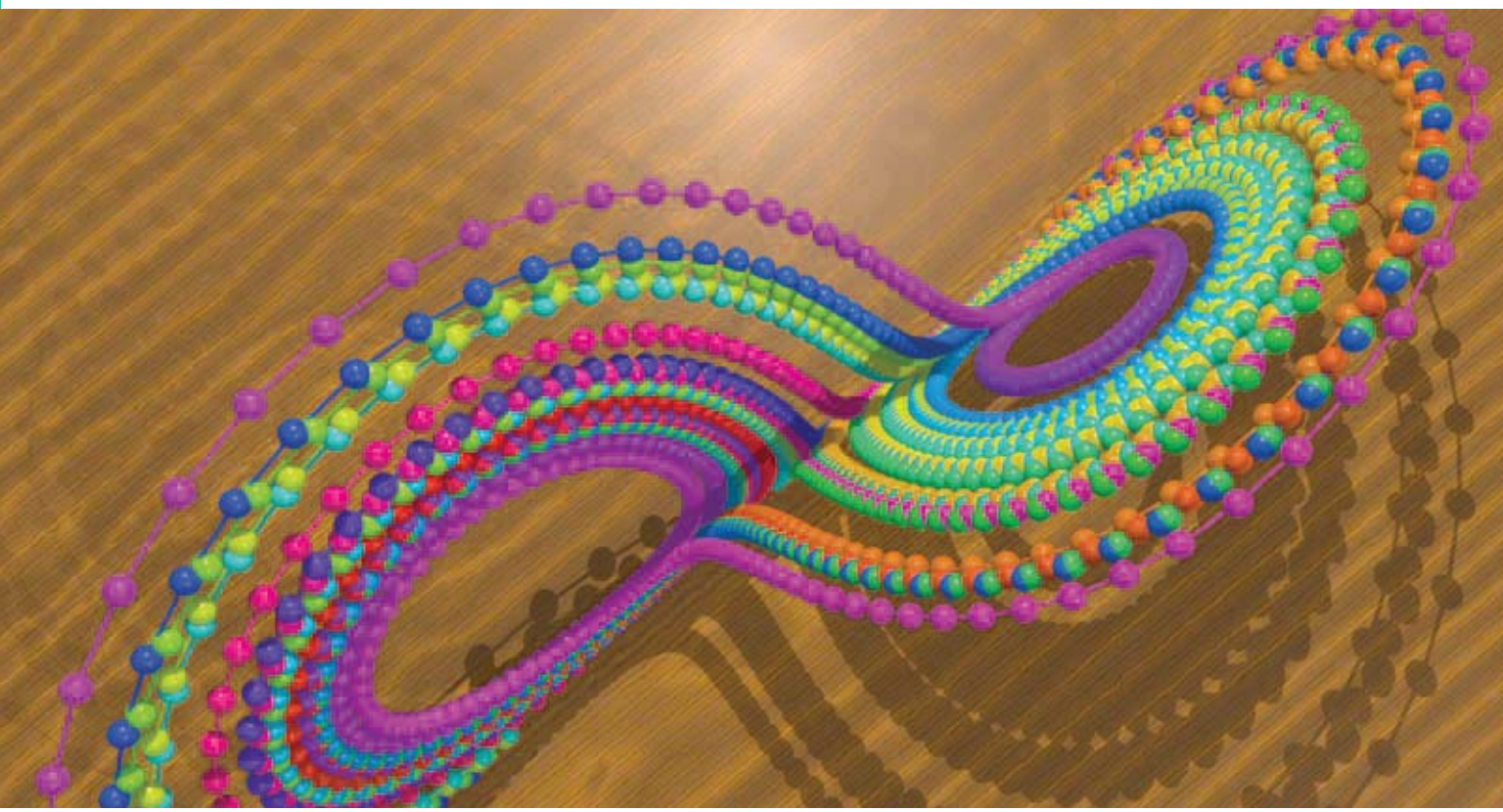
Computational Modeling

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.

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://lccn.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

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

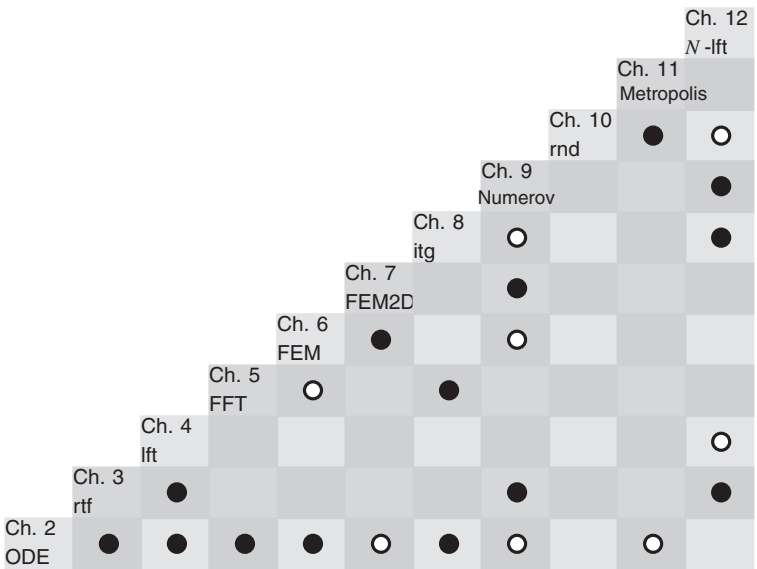


FIGURE 1 Major chapter dependencies at a glance. Open circles indicate minor or Digital Online Companion only dependencies. The abbreviations are: FEM—finite element method; FFT—fast Fourier transform; itg—numerical integration; lft—leapfrog with time-transformation; ODE—ordinary differential equations;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