

COMPUTATIONAL METHODS FOR PHYSICS

There is an increasing need for undergraduate students in physics to have a core set of computational tools. Most problems in physics benefit from numerical methods, and many of them resist analytical solution altogether. This textbook presents numerical techniques for solving familiar physical problems, where a complete solution is inaccessible using traditional mathematical methods.

The numerical techniques for solving the problems are clearly laid out, with a focus on the logic and applicability of the method. The same problems are revisited multiple times using different numerical techniques, so readers can easily compare the methods. The book features over 250 end-of-chapter exercises. A website hosted by the author features a complete set of programs used to generate the examples and figures, which can be used as a starting point for further investigation. A link to this can be found at www.cambridge.org/9781107034303.

JOEL FRANKLIN is an Associate Professor in the Physics Department of Reed College. He focuses on mathematical and computational methods with applications to classical mechanics, quantum mechanics, electrodynamics, general relativity, and modifications of general relativity.

Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

COMPUTATIONAL METHODS FOR PHYSICS

JOEL FRANKLIN
Reed College



Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town,
Singapore, São Paulo, Delhi, Mexico City

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

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

www.cambridge.org
Information on this title: www.cambridge.org/9781107034303

© J. Franklin 2013

This publication is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without the written
permission of Cambridge University Press.

First published 2013

Printed and bound in the United Kingdom by the MPG Books Group

A catalog record for this publication is available from the British Library

Library of Congress Cataloging in Publication data
Franklin, Joel, 1975–
Computational methods for physics / Joel Franklin, Reed College.
pages cm

Includes bibliographical references and index.

ISBN 978-1-107-03430-3 (hardback)

1. Mathematical physics. 2. Physics – Data processing. 3. Numerical analysis. I. Title.
QC20.F735 2013
530.15 – dc23 2013003044

ISBN 978-1-107-03430-3 Hardback

Additional resources for this publication at www.cambridge.org/9781107034303

Cambridge University Press has no responsibility for the persistence or
accuracy of URLs for external or third-party internet websites referred to
in this publication, and does not guarantee that any content on such
websites is, or will remain, accurate or appropriate.

Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

For Lancaster, Lewis, and Oliver

Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

Contents

<i>Preface</i>	<i>page</i> xiii
1 Programming overview	1
1.1 Arithmetic operations	2
1.2 Comparison operations	3
1.3 Variables	3
1.4 Control structures	5
1.5 Functions	6
1.6 Input and output	8
1.7 Recursion	8
1.8 Function pointers	10
1.9 Mathematica-specific array syntax	11
1.10 Implementations and pseudo-code	12
1.11 Timing and operation counts	14
1.12 Units and dimensions	15
2 Ordinary differential equations	21
2.1 Physical motivation	21
2.1.1 Newton’s second law	21
2.1.2 Relativistic mechanics	28
2.2 The Verlet method	30
2.3 Discretization	33
2.3.1 Euler’s method	35
2.3.2 Improved accuracy	37
2.4 Runge–Kutta methods	39
2.4.1 Adaptive step size	41
2.4.2 Vectors	42
2.5 Stability of numerical methods	44
2.6 Multi-step methods	47

viii	<i>Contents</i>	
3	Root-finding	54
3.1	Physical motivation	54
3.1.1	Roots of functions	55
3.1.2	Shooting for Newton’s second law	62
3.1.3	Shooting for eigenvalues	66
3.2	Finding roots	73
3.2.1	Bisection	73
3.2.2	Newton’s method	74
3.2.3	Newton’s method and shooting	77
3.2.4	Steepest descent	78
4	Partial differential equations	86
4.1	Physical motivation	87
4.1.1	One dimension	88
4.1.2	Two dimensions	89
4.2	Finite difference in one dimension	93
4.3	Finite difference in two dimensions	98
4.4	Examples	102
4.4.1	Empty box	102
4.4.2	Parallel plate capacitor	102
4.4.3	Constant surface charge box	103
4.4.4	Grounded plate with hole	107
5	Time-dependent problems	114
5.1	Physical motivation	114
5.1.1	Conservation laws	115
5.1.2	The wave equation	116
5.1.3	Traffic flow	119
5.1.4	Shock solutions	120
5.1.5	Fluids	122
5.1.6	Quantum mechanics	123
5.2	Exactly solvable cases	124
5.3	Discretization and methods	125
5.3.1	Nonlinear modification	127
5.3.2	Lax–Wendroff	129
5.4	Crank–Nicolson for the Schrödinger equation	131
6	Integration	141
6.1	Physical motivation	141
6.1.1	Direct integrals	142
6.1.2	Integrals that solve PDEs	146
6.2	One-dimensional quadrature	149

	<i>Contents</i>	ix
6.3	Interpolation	151
6.3.1	Arbitrary interval	156
6.4	Higher-dimensional quadrature	158
6.5	Monte Carlo integration	160
6.5.1	Statistical connection	163
6.5.2	Extensions	164
7	Fourier transform	172
7.1	Fourier transform	173
7.2	Power spectrum	176
7.3	Fourier series	177
7.4	Discrete Fourier transform	178
7.5	Recursion	180
7.6	FFT algorithm	182
7.7	Applications	186
7.7.1	Sound input	186
7.7.2	Low pass filter	187
7.7.3	Two-dimensional Fourier transform	189
7.7.4	Spectrogram	192
8	Harmonic oscillators	199
8.1	Physical motivation	199
8.1.1	Linear chains	200
8.1.2	Molecular dynamics	200
8.1.3	Harmonic approximation	201
8.2	Three balls and two springs	201
8.2.1	Equations of motion	202
8.2.2	Solution	204
8.3	Solution for a particular case	207
8.4	General solution	211
8.5	Balls and springs in $D = 3$	211
9	Matrix inversion	220
9.1	Definitions and points of view	221
9.2	Physical motivation	222
9.2.1	Circuits	222
9.2.2	Fitting data	224
9.3	How do you invert a matrix?	227
9.3.1	Easy inversions	227
9.3.2	The QR decomposition	228
9.3.3	QR exists (for square $\mathbb{A}$)	228
9.3.4	The LU decomposition	231

x	Contents	
	9.3.5 LU exists	234
	9.3.6 Computing the LU decomposition	236
	9.3.7 Pivoting	238
9.4	Determinants	239
9.5	Constructing $\mathbb{A}^{-1}$	240
10	The eigenvalue problem	246
10.1	Fitting data	246
10.2	Least squares	248
10.3	The eigenvalue problem	251
10.4	Physical motivation	251
	10.4.1 Linear springs	251
	10.4.2 Quantum mechanics	252
	10.4.3 Image compression	255
10.5	The power method	260
10.6	Simultaneous iteration and QR iteration	262
	10.6.1 Timing	264
10.7	Quantum mechanics and perturbation	265
	10.7.1 Perturbation setup	265
11	Iterative methods	274
11.1	Physical motivation	274
	11.1.1 Particle motion	275
	11.1.2 Advertisement: sparse matrices	276
11.2	Iteration and decomposition	277
	11.2.1 Jacobi's method	279
	11.2.2 Successive over-relaxation (SOR)	279
11.3	Krylov subspace	282
	11.3.1 Definition	282
	11.3.2 Constructing the subspace (Lanczos iteration)	284
	11.3.3 The eigenvalue problem	285
	11.3.4 Matrix inversion	287
	11.3.5 Conjugate gradient	289
12	Minimization	298
12.1	Physical motivation	298
	12.1.1 Variational method in quantum mechanics	299
	12.1.2 Data fitting	300
	12.1.3 Action extremization	301
12.2	Minimization in one dimension	302
12.3	Minimizing $u(\mathbf{x})$	306
	12.3.1 Steepest descent	306
	12.3.2 Newton's method	308

<i>Contents</i>	xi
12.4 Nonlinear least squares	309
12.5 Line minimization	313
12.6 Monte Carlo minimization	315
13 Chaos	323
13.1 Nonlinear maps	324
13.2 Periodicity and doubling	327
13.2.1 The logistic map	328
13.2.2 Making bifurcation diagrams	332
13.3 Characterization of chaos	333
13.3.1 Period doubling	333
13.3.2 Initial conditions	334
13.4 Ordinary differential equations	338
13.4.1 Linear problems	338
13.4.2 The Lorenz equations	339
13.5 Fractals and dimension	342
14 Neural networks	351
14.1 A neural network model	352
14.1.1 A linear model	352
14.1.2 A nonlinear model	354
14.1.3 The sigmoid function	354
14.2 Training	356
14.2.1 Setting coefficients	357
14.3 Example and interpretation	359
14.4 Hidden layers	361
14.5 Usage and caveats	363
14.6 Signal source	364
15 Galerkin methods	371
15.1 Physical motivation	372
15.1.1 Born–Infeld E&M	373
15.1.2 Nonlinear scalar fields	375
15.1.3 Nonlinear wave equation	376
15.1.4 The heat equation	377
15.2 Galerkin method	378
15.2.1 Nonlinear Klein–Gordon	382
15.2.2 Nonlinear string	384
<i>References</i>	393
<i>Index</i>	395

Cambridge University Press
978-1-107-03430-3 - Computational Methods for Physics
Joel Franklin
Frontmatter
[More information](#)

Preface

This book is meant to teach an advanced undergraduate physics major how to solve problems, that they can easily pose and understand, using numerical methods. It is self-contained (within reason) in its presentation of physical problems,¹ so that a reader need not be previously familiar with all of the physical topics covered. The numerical techniques that are presented are complete, without being exhaustive. I have chosen a set of tools that can be motivated easily – some of these, like Runge–Kutta methods for solving ordinary differential equations, are themselves the “end of the story” for certain problems. In other cases, I choose to present a simplified method that will work for at least some problems, even if it is not what is actually implemented in practice. A homogenous and appropriate level of presentation is the goal. Sometimes that allows for a complete and formal discussion, in other cases, I motivate and inform while ensuring that a successful (if not competitive) method is carefully described.²

The chapters are defined by computational technique, with chapters covering “Partial differential equations,” “Integration,” “Fourier transform,” “Matrix inversion,” etc. Each chapter introduces a disparate set of physical problems, most of which are “unsolvable” (meaning that there is no closed-form solution expressible in terms of simple functions).³ I have attempted to draw problems from as wide an array of physical areas as possible, so that in a single chapter, there may be examples from quantum mechanics, E&M, special relativity, etc. All of these physical setups end in a similar analytically intractable problem. That problem then becomes

¹ This can lead to some repetition, of, for example, the meaning attached to the solutions of Schrödinger’s equation, or the definition of relativistic momentum, etc. But my hope is that repeated local definition allows for easier reading.

² To be clear with an example: presenting a fast, research-grade eigenvalue solver requires more mathematics and programming ability than is available at the undergraduate level – that is not, therefore, a compelling target. But, the power method, and simultaneous iteration, which are easy (and delightful) to describe, will do the job in many cases and inform more sophisticated methods.

³ Of course, many “simple functions” still require numerical evaluation, even sine and cosine have interpolated or otherwise approximated value at most points.

the vehicle for the introduction of a numerical method (or in some cases, a few different methods) in which I focus on the mathematical ideas behind the method, including a discussion of its motivation, its limitations and possible extensions. At the end of each chapter, I include short further reading suggestions,⁴ and two types of exercise. The first, called “Problems,” are pencil-and-paper exercises. These can involve guiding a student through the analysis of a numerical method presented in the chapter, or setting up analytically solvable limiting cases of problems that will be solved numerically. Then there are “Lab problems,” these are physical problems (generally set up at the start of the chapter) that will be solved using the method(s) from the chapter.

The specific numerical methods I discuss in each chapter were chosen for their coverage (of a wide range of physical problems), and presentability. By the latter, I mean that any method should be transparent in its functioning (and implementation), and easy to analyze. Certain types of Monte Carlo “simulations” are omitted, since it is harder to prove (or in some cases even indicate) that they will succeed in solving a particular problem. One can motivate and a posteriori verify that such methods work, and they have a certain physically inspired allure, but my primary goal in choosing methods is to introduce students to ideas whose advantages and limitations can be judged easily. In addition to numerical methods, I discuss “traditional” mathematical methods relevant to the physical sciences. I have chosen specific mathematical methods to present (like separation of variables, shock solutions to time-dependent PDEs, or time-independent perturbation theory) based on the availability of a comparative discussion with numerical methods, and to set up limiting cases so that we can check the functioning of numerical methods on concrete test problems where the answer is known. I hope, then, that the book is more than “just” a numerical methods book, but one that can be used to learn some physics, learn methods for solving physical problems, and build confidence that those methods are working (in the appropriate regime).

Structure and teaching

Structurally, there are three sections to the book:

1. ODEs, PDEs, and integration. These are the topics covered in the first seven chapters. The idea is to get a bulk of the problem-solving tools out quickly, and most physical problems benefit from an ability to solve ODEs in initial

⁴ For arguments or examples that come directly from a particular source, I cite the source directly within the text, and those citations refer to the references at the end of the book – suggestions for further reading represents that subset of the full bibliography that provides useful additional information and/or inspiration local to the chapter.

value (Chapter 2) and boundary value form (Chapter 3). There are two chapters on finite difference for PDEs: Chapter 4 covers static, linear operators (mainly the Poisson problem), and in Chapter 5, we see time-dependent PDEs (both linear, like Schrödinger's equation, and nonlinear forms). Chapter 6 develops integration methods from interpolating polynomials, and connects the statistical Monte Carlo integration method to simple box sums. Finally, I count the chapter on the FFT as an integration topic.

2. Numerical linear algebra. Chapters 9–12 discuss numerical techniques for matrix inversion, least squares solution, and the eigenvalue problem. Some of the material in these chapters is required by topics in the first section (for example, solving Poisson's problem in discretized form involves matrix inversion) where we used canned routines. Now we come back and pick up these topics, their motivation coming in part from previous discussions. Chapters 9 and 10 focus on “direct” solution of the matrix inverse and eigenvalue problem, while Chapter 11 introduces iterative techniques (Jacobi, SOR, and Krylov subspace methods).⁵ Chapter 12 is not, strictly speaking, a part of linear algebra – but minimization is usefully applied in the nonlinear least squares problem, so must come after the linear form of that problem.
3. Additional topics. The first two sections provide methods that cover many physical problems. The remaining chapters give interesting additional topics, but are slightly different in their coverage from the rest of the book. Chapter 13: “Chaos,” for example, has no numerical methods attached to it. The neural network chapter (14) presents these models in terms of their predictive ability for physical phenomena, with emphasis on the promise provided by the Cybenko theorem rather than the physiological inspiration normally encountered. Chapter 15 returns to PDEs and describes the Galerkin approach with some additional nonlinear PDE targets.

Two of the chapters do not fall neatly into these categories – the first chapter provides a review of the minimal programming tools that we need to use to implement the methods presented in the other chapters – it also provides some orienting *Mathematica* examples of these tools in action. The eighth chapter acts as a pivot between the first and second sections, and is meant to set up our interest in the eigenvalue problem.

The grouping above is informed by the course that inspired this book, and for which it was written – in it, the students present a final project of their own design,

⁵ Divorcing the finite difference setup from the iterative solutions (Jacobi and SOR) to the matrix inverse problem is not the traditional approach. Typically, authors will combine the discretization of the Laplacian with an iterative update. But I believe that the two pieces are logically separate – discretizing the Laplace operator gives us a matrix inverse problem to solve, how we go about solving it is a separate matter.

and they begin working on this at about the middle of the semester. That is when we have covered the bulk of item one above, so they have seen a number of different kinds of problems and understand numerical methods for solving them. The section on numerical linear algebra is interesting and informative, but students who need to, for example, invert matrices for their final projects can use a built-in routine (few students have made the process of matrix inversion the central idea of their projects) until the relevant material is covered (and even after, in most cases). The additional topics section has then come at a time when students can take advantage of the information in them without relying on it for their project (coupled, driven oscillators can be solved and studied numerically before the nonlinear analysis tools that are typically used to describe them are introduced).

While I use a subset of the programming language of *Mathematica* both in the book, and in the course, I have found that students can re-render the content in other languages relatively easily, and have had students work out the lab problems using python, Java, C, Sage, and matlab. For those students with less of a programming background, I provide, for each chapter, a *Mathematica* “notebook” that contains all the commands used to generate the figures and example solutions for the chapter. Then much of the programming can be done by re-working examples from the chapter notebook. My intention is for students who are not as interested in programming to have a way of learning the methods, without worrying about their implementation as much. The computational methods are based on implementation-independent mathematical ideas, and those are the core targets. For students who enjoy the implementation side (I always did, although I was never particularly good at it): optimizing and ordering commands efficiently, don’t look at the chapter notebooks, work out your own routines from scratch.

This course is one of my favorites to teach – there are limitless physical problems to set up, so it’s a great place for me to learn about new physics. Then the extent to which it is difficult to solve most problems (even simple ones) analytically is always surprising. And, finally, the numerical solutions that allow progress are often simple and fast. I usually cover each chapter in a week with three lectures: on Monday, we just set up physical problems, ones that come from different physics, but all end in a common, fundamental “problem.” On Wednesday, we develop a method that solves that problem, and then, during “Friday potpourri,” we return to some of the problems from Monday and solve them, and discuss limitations or extensions of the method. The chapter structure mimics these lectures, with physical problems presented first, then methods, then additional items, sometimes additional physics, other times, more in-depth discussion of the method or its variants.

What is notably missing from the weekly lectures is any sort of implementation discussion – in addition to the three lectures, I also run three-hour “labs” every week. Students come to these to work on the lab problems (I assign three or four

of these each week), and this is where implementation can be carefully discussed, with the relevant programming ideas built up on an individual basis, or through group discussion. Having a concrete problem to work on (the lab problems) gives natural focus to the programming. This separation between physics, the idea behind the method, and its implementation, is intentional. The goal is to help identify points of confusion as arising from the statement of a physical problem, the proper functioning of a numerical method, or its implementation. Keeping those three separate makes clearing up the confusion easier.

Website and materials

The book has a website, and on it, students can find all of the chapter notebooks (the `Mathematica` code that generated all the examples and figures in the book), a few sample “lab notebooks,” and some project ideas that students have come up with over the years. Some of the lab problems require mocked up data or other information, and those are also available on the website.

Acknowledgements

It is a pleasure to thank my colleagues in the Reed College physics department, especially Darrell Schroeter, who developed and taught an early version of the computational methods class at Reed with me, Nelia Mann who contributed useful comments and problems, David Latimer, a great sounding board, and David Griffiths for his usual outstanding commentary, criticism and encouragement. Outside the department, Jim Fix gave me some great image rotation advice, Olivia Schelly took the pictures that are used in Chapter 7, and Matt Sayre wrote the Gravitea-Time song for filtering. Simon Capelin and Lindsay Barnes at Cambridge University Press have once again made the publishing process enjoyable, and I thank them for their help at all stages of the project.

I would like to thank Sebastian Doniach and the Stanford University physics department for their hospitality in the winter of 2012: aside from a nice change of scene, and stimulating new environment, I got to teach this material to an entirely new audience, for which I am very grateful. The Writer’s Bloc has provided a wonderful setting for revising the manuscript. Finally, a special thanks to the Reed College students who helped me during the process of editing and refining the book: Tom Chartrand, Todd Garon, and Reuven Lazarus.