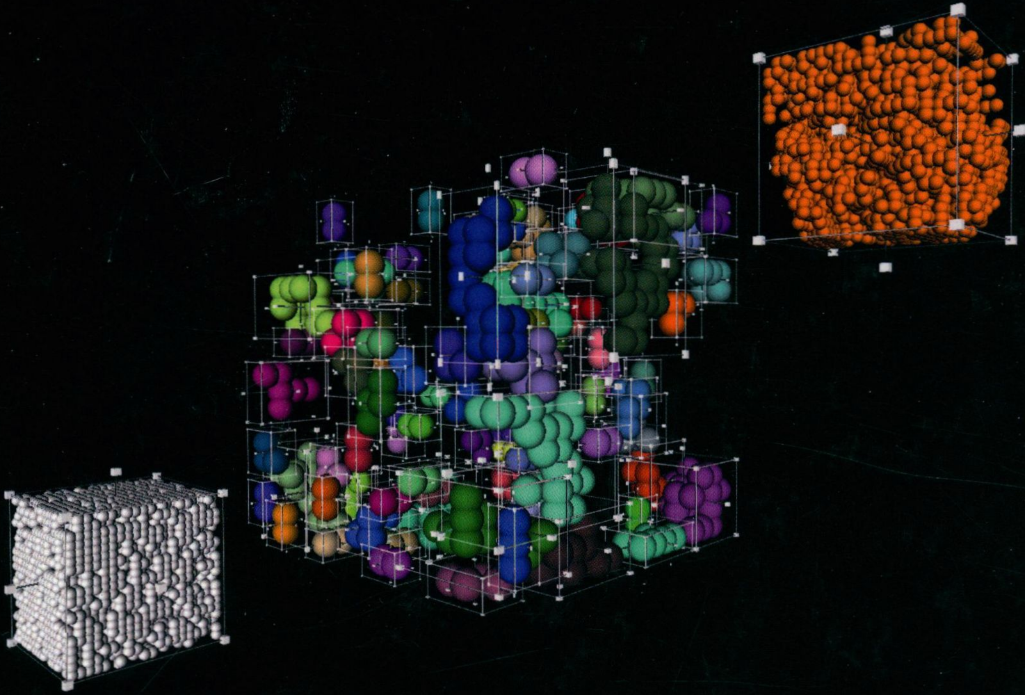
OXFORD



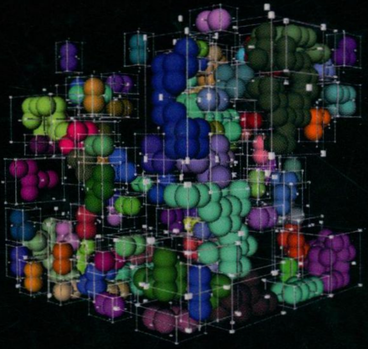
APPLIED COMPUTATIONAL PHYSICS

JOSEPH F. BOUDREAU | ERIC S. SWANSON

with contributions from Riccardo Maria Bianchi

Applied Computational Physics is a graduate-level text stressing three essential elements: advanced programming techniques, numerical analysis, and physics. The goal of the text is to provide students with essential computational skills that they will need in their careers, and to increase the confidence with which they write computer programs designed for their problem domain. The physics problems give them an opportunity to reinforce their programming skills, while the acquired programming skills augment their ability to solve problems in physics. The C++ language is used throughout the text. Physics problems include Hamiltonian systems, chaotic systems, percolation, critical phenomena, few-body and multi-body quantum systems, quantum field theory, simulation of radiation transport, and data modeling. The book, the fruit of a collaboration between a theoretical physicist and an experimental physicist, covers a broad diversity of topics from both viewpoints. Examples, program libraries, and additional documentation can be found at the companion website. Hundreds of original problems reinforce programming skills and increase the ability to solve real-life physics problems at and beyond the graduate level.

JOSEPH F. BOUDREAU and **ERIC S. SWANSON** are both professors of Physics at the Department of Physics and Astronomy, University of Pittsburgh, USA.



Cover image: A simulation of percolation on a cubic lattice, in which distinct clusters are rendered in different colors (center and upper right) and unoccupied sites appear as white spheres (lower left). ©Joe Boudreau and Eric Swanson.

ISBN 978-0-19-870863-6

9 780198 708636

OXFORD
UNIVERSITY PRESS

www.oup.com

BOUDREAU | SWANSON

APPLIED COMPUTATIONAL PHYSICS

OXFORD

Applied Computational Physics

Joseph F. Boudreau and Eric S. Swanson

with contributions from Riccardo Maria Bianchi

OXFORD
UNIVERSITY PRESS

OXFORD
UNIVERSITY PRESS

Great Clarendon Street, Oxford, OX2 6DP,
United Kingdom

Oxford University Press is a department of the University of Oxford.
It furthers the University's objective of excellence in research, scholarship,
and education by publishing worldwide. Oxford is a registered trade mark of
Oxford University Press in the UK and in certain other countries

First Edition published in 2018
Impression: 1

Published in the United States of America by Oxford University Press
198 Madison Avenue, New York, NY 10016, United States of America

British Library Cataloguing in Publication Data
Data available

Library of Congress Control Number: 2017946193

ISBN 978–0–19–870863–6 (hbk.)
ISBN 978–0–19–870864–3 (pbk.)

DOI 10.1093/oso/9780198708636.001.0001

Printed and bound by
CPI Group (UK) Ltd, Croydon, CR0 4YY

APPLIED COMPUTATIONAL PHYSICS

For Pascale.
For Lou, Gordy, Suzy, Kris, Maura, Vin.
For Gordon V.

For Erin, Megan, Liam, and Drew.
For Max and Gudrun.

reor ut specular

Preface

New graduate students often experience something like shock when they are asked to solve real-world problems for the first time. These problems can be only rarely solved with pen and paper and the use of computational techniques becomes mandatory. The role of computation in any scientific endeavor is growing, and presents an increasing set of challenges. Numerical algorithms play a central role in theoretical prediction and in the analysis of experimental data. In addition, we see an increasing number of less numerical tasks taking on major importance in the life of young scientists. For example, how do you blend together two computing languages or split a computation between multiple computers? How does one design program libraries of numerical or scientific code for thousands of users? How is functionality added to a million-line reconstruction program? How can complicated datasets be visualized? What goes into a monitoring program that runs in a control room? These tasks are not particularly numerical or even scientific, but they are nonetheless important in the professional lives of scientists. From data acquisition systems to solving quantum field theory or presenting information, students face an intimidating computational environment with many languages and operating systems, multiple users with conflicting goals and methods, and complex code solving subtle and complicated problems.

Unfortunately, the typical student is marginally prepared for the challenges faced in modern computational ecosystems. Most students have had some exposure to a programming language such as C, C++, Java, or Fortran. In their first contact with "real" code, they may well be exposed to a proliferation of legacy software that in some cases is better used as a counterexample of good modern coding practices. Under these circumstances the usual solution is to learn on the fly. In a bygone era when the computing environment was simple this learning process was perfectly satisfactory, but today undirected learning leads to many false starts and some training has become indispensable. The search for help can be difficult because the nearby senior physicist probably grew up in an era preceding the explosive development of languages, paradigms, and computational hardware.

This book aims to fill some of the holes by introducing students to modern computational environments, object-oriented computing, and algorithmic techniques. We will rely on 'canned' code where reasonable. However, canned code is, by definition, incapable of solving research problems. It can at best solve portions of problems. At worst, it can lead the student researcher to false or incomplete conclusions. It is therefore imperative that the student understands what underlies his code. Thus an explanation of the numerical issues involved in common computational tasks will be presented.

Sometimes the numerical methods and applications will be quite technical; for this reason we regard this book as appropriate for newly graduated students. Our examples will be drawn primarily from experimental and theoretical physics. Nevertheless, the book is also useful for students in chemistry, biology, atmospheric science, engineering, or any field in which complex analytical problems must be solved.

This text is meant for advanced (or graduate) students in the sciences and engineering. The physics ranges from advanced undergraduate topics in classical and quantum mechanics, to very advanced subject matter such as quantum field theory, renormalization, and scaling. The concepts of object oriented computing are introduced early in the text and steadily expanded as one progresses through the chapters. The methods of parallel computation are also introduced early and are applied in examples throughout. Since both the physics and the coding techniques can be replete with jargon, we attempt to be practical and provide many examples. We have not made any effort to prune away any discussion of fairly pedestrian material on the pretext that it is not advanced enough for a sophisticated audience. Our criterion is that the topics we discuss be *useful*, not that they be graduate-level, particularly since some topics are interdisciplinary in nature.

The numerical algorithms we consider are those applied in the major domain areas of physics. Classical problems involving a finite number of degrees of freedom are most often reduced to a coupled set of first-order differential equations. Those involving an infinite number of degrees of freedom require techniques used to solve partial differential equations. The study of quantum mechanical systems involves random processes, hence the temporal evolution of the system is handled though simulation of the underlying randomness. The computation of physical processes thus can be generally categorized according to the number of degrees of freedom and the stochastic or deterministic nature of the system. More complicated situations can mix these. For example, to follow a charged particle through a magnetic field in the presence of multiple scattering involves both deterministic and stochastic processes.

The flip side of simulation is data modeling. This is the procedure by which a mathematical description of data, often along with the values of physically interesting parameters, is obtained. Data modeling is an activity that consumes much of the time and creativity of experimental physicists working with datasets, large or small. While many treatises exist on the statistical analysis of data, the goal here is to explore, in somewhat greater detail than is usually found, the computational aspects of this field.

This text is neither a treatise on numerical analysis nor a guide to programming, but rather strives to develop practical skills in numerical and non-numerical methods applied to real world problems. Because of the emphasis on practical skills, students should expect to write programs and to refine and develop their programming techniques along the way. We assume a basic knowledge of C++ (the part of the language taken directly from C, minus the anachronisms), and treat the newer features in dedicated chapters. We do not, however, give a complete lesson on the syntax and symantics of any language, so we advise the reader who has not mastered C++ to learn it in parallel using any one of a number of sources. Our emphasis can then fall on using the language *effectively* for problems arising in physics.

A key ingredient to effective programming nowadays is mastery of object-oriented programming techniques—we strive to develop that mastery within the context of greatest interest to the target audience, namely physics. As noted in *Numerical Recipes*

(Press 2007), object-oriented programming has been "recognized as the almost unique successful paradigm for creating complex software". As a result, object-oriented techniques are widespread in the sciences and their use is growing. The physicist appreciates object oriented programming because his day-to-day life is filled with a rich menagerie of interesting objects, not just integers and real numbers, but also vectors, four-vectors, spinors, matrices, rotation group elements, Euclidean group elements, functions of one variable, functions of more than one variable, differential operators, etc. One usually gets more from a programming paradigm that allows user-defined datatypes to fill in gaps left at the language level. Encapsulation and polymorphism can be effectively used to build up a more functional set of mathematical primitives for a physicist—and also to build up an important set of not-so-mathematical objects such as are found in other nonscientific code.

Many books are devoted to object oriented analysis and design, and while some of these treatises are perfect for their target audience, a typical scientist or engineer most likely gets tired of examples featuring the payroll department and looks for a discussion of object oriented programming that "speaks his language". Accordingly, we include three chapters on object oriented programming in C++: Encapsulation, Polymorphism, and Templates. Other chapters of the book provide excellent examples of object-oriented techniques applied to various practical problems.

A companion web site has been established for this text at:

- `http://www.oup.co.uk/companion/acp`

The site includes example code (`EXAMPLES` area), skeletons which students can use as a starting point for certain exercises appearing in the text (`SKELETONS` area), and raw data for other exercises (`DATA` area). In addition the site provides user's guides, reference manuals, and source code for software libraries provided free of charge and used within this text. This software is licensed under the GNU Lesser General Public License. In referencing examples, skeletons, data, etc., we generally omit the full URL and refer simply to the directory, e.g. `EXAMPLES`, `SKELETONS`, `DATA`, etc.

Course organization

It is our experience that most of the material in this text can be covered in two terms of teaching. There are three main strands of emphasis: computational, numerical, and physical, and these are woven together, so the reader will find that emphasis alternates, though the book begins with more computational topics, then becomes more numerical, and finally more physical.

Computational topics include: Building programs (Chap 1), Encapsulation (Chap 2), Some useful classes (Chap 3), How to write a class (Chap 6), Parallel computing (Chap 9), Graphics for physicists (Chap 10), Polymorphism (Chap 12), and Templates, the standard library, and modern C++ (Chap 17).

Numerical topics include Interpolation and extrapolation (Chap 4), Numerical quadrature (Chap 5), Monte Carlo methods (Chap 6), Ordinary differential equations (Chap 11).

Topics related to applications in experimental and theoretical physics are Percolation and universality (Chap 8), Nonlinear dynamics and chaos (Chap 14), Rotations and Lorentz transformations (Chap 14), Simulation (Chap 15), Data modeling (Chap 16), Many body dynamics (Chap 18), Continuum dynamics (Chap 19), Classical spin systems (Chap 20), Quantum mechanics (Chap 21 and 23), Quantum spin systems (Chap 22), and Quantum field theory (Chap 24).

Finally, there are nearly 400 exercises of widely varying difficulty in the text. To assist students and instructors in selecting problems, we have labelled those exercises that are meant to be worked out without the aid of a computer as *theoretical* [T]; exercises which are more open-ended and require significant effort are elevated to the status of a *project* and labelled with a [P].

Acknowledgments

We are grateful to many people for encouragement and for lending their expertise. Among these are Jiří Čížek, Rob Coulson, Paul Geiger, Jeff Greensite, Ken Jordan, Colin Morningstar, James Mueller, Kevin Sapp, Søren Toxvaerd, and Andrew Zentner. J. Boudreau wishes to thank Petar Maksimovic and Mark Fishler for inspiration and help with the development of Function Objects (Chapter 3), Lynn Garren for support of the original package which appeared in the CLHEP class library, and Thomas Kittelmann and Vakho Tsulaia for their collaboration over the years. Preliminary versions of this text, and particularly the exercises, have been inflicted on our graduate students over the years–their help has been instrumental in effecting many important revisions. We thank them wholeheartedly.

Contents