Python for Scientists

Second Edition

Scientific Python is a significant public domain alternative to expensive proprietary software packages. This book teaches from scratch everything the working scientist needs to know using copious, downloadable, useful and adaptable code snippets. Readers will discover how easy it is to implement and test non-trivial mathematical algorithms and will be guided through the many freely available add-on modules. A range of examples, relevant to many different fields, illustrate the language's capabilities. The author also shows how to use pre-existing legacy code (usually in Fortran77) within the Python environment, thus avoiding the need to master the original code.

In this new edition, several chapters have been rewritten to reflect the *IPython* notebook style. With an extended index, an entirely new chapter discussing *SymPy* and a substantial increase in the number of code snippets, researchers and research students will be able to quickly acquire all the skills needed for using Python effectively.

Python for Scientists

Second Edition

JOHN M. STEWART
Department of Applied Mathematics & Theoretical Physics
University of Cambridge

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

4843/24, 2nd Floor, Ansari Road, Daryaganj, Delhi – 110002, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781316641231
DOI: 10.1017/9781108120241

First published 2014
Second edition 2017

Printed in the United Kingdom by TJ International Ltd. Padstow Cornwall

A catalogue record for this publication is available from the British Library.

Library of Congress Cataloging-in-Publication Data
Names: Stewart, John, 1943 July 1–
Title: Python for scientists / John M. Stewart, Department of Applied, Mathematics & Theoretical Physics, University of Cambridge.
Description: Second edition. | Cambridge, United Kingdom ; New York, NY, USA : Cambridge University Press, [2017] | Includes bibliographical references and index.
Identifiers: LCCN 2016049298 | ISBN 9781316641231 (paperback)
Subjects: LCSH: Science–Data processing. | Python (Computer program language)
Classification: LCC Q183.9 .S865 2017 | DDC 005.13/3–dc23
LC record available at https://lccn.loc.gov/2016049298

ISBN 978-1-316-64123-1 Paperback

Additional resources for this publication at www.cambridge.org/9781316641231

Contents

Preface to the Second Edition

The motivation for writing this book, and the acknowledgements of the many who have assisted in its production, are included in the topics of the Preface to the first edition, which is reprinted after this one. Here I also need to adjoin thanks to the many readers who provided constructive criticisms, most of which have been incorporated in this revision. The purpose here is to explain why a second edition is needed. Superficially it might appear that very little has changed, apart from a new Chapter 7 which discusses *SymPy*, Python's own computer algebra system.

There is, however, a fundamental change, which permeates most of the latest version of this book. When the first edition was prepared, the reliable way to use the enhanced interpreter *IPython* was via the traditional "terminal mode". Preparations were under way for an enhanced "notebook mode", which looked then rather like the Mathematica notebook concept, except that it appeared within one's default web browser.[1] That project has now morphed into the Jupyter notebook. The notebook allows one to construct and distribute documents containing computer code (over forty languages are supported), equations, explanatory text, figures and visualizations. Since this is also perhaps the easiest software application for a beginner to develop Python experience, much of the book has been rewritten for the notebook user. In particular there is now a lightning course on how to use the notebook in Appendix A, and Chapter 2 has been extensively rewritten to demonstrate its properties. All of the material in the book now reflects, where appropriate, its use. For example, it allows *SymPy* to produce algebraic expressions whose format is unsurpassed by other computer algebra systems.

This change also affects the areas of interactive graphics and visual animations. Their demands are such that the standard Python two-dimensional graphics package *Matplotlib* is having difficulty in producing platform-independent results. Indeed, because of "improved" software upgrades, the code suggested for immediate on-screen animations in the first edition no longer works. However, the notebook concept has a subtle solution to resolve this impasse. Recall that the notebook window is your browser window, which uses modern HTML graphics. The consequent benefits are introduced in Chapter 6.

As a final enhancement, all but the most trivial code snippets listed in this book are now available in electronic form, as a notebook of course, but the website includes

[1] Internet access is neither required nor used.

HTML and PDF versions, see Section 1.2. The explanatory text surrounding the text is not included. For that you have to read the book, in hard copy or ebook format!

Note added in proof:

John died shortly after the completion of the Second Edition, and is much missed by colleagues, friends and family, especially the "Python widow".

Preface to the First Edition

I have used computers as an aid to scientific research for over 40 years. During that time, hardware has become cheap, fast and powerful. However, software relevant to the working scientist has become progressively more complicated. My favourite textbooks on Fortran90 and C++ run to 1200 and 1600 pages respectively. And then we need documentation on mathematics libraries and graphics packages. A newcomer going down this route is going to have to invest significant amounts of time and energy in order to write useful programmes. This has led to the emergence of "scientific packages" such as Matlab® or Mathematica® which avoid the complications of compiled languages, separate mathematics libraries and graphics packages. I have used them and found them very convenient for executing the tasks envisaged by their developers. However, I also found them very difficult to extend beyond these boundaries, and so I looked for alternative approaches.

Some years ago, a computer science colleague suggested that I should take a look at Python. At that time, it was clear that Python had great potential but a very flaky implementation. It was, however, free and open-source, and was attracting what has turned out to be a very effective army of developers. More recently, their efforts have coordinated to produce a formidable package consisting of a small core language surrounded by a wealth of add-on libraries or *modules*. A select group of these can and do replicate the facilities of the conventional scientific packages. More importantly an informed, intelligent user of Python and its modules can carry out major projects usually entrusted to dedicated programmers using Fortran, C etc. There is a marginal loss of execution speed, but this is more than compensated for by the vastly telescoped development time. The purpose of this book is to explain to working scientists the utility of this relatively unknown resource.

Most scientists will have some computer familiarity and programming awareness, although not necessarily with Python, and I shall take advantage of this. Therefore, unlike many books which set out to "teach" a language, this one is not just a brisk trot through the reference manuals. Python has many powerful but unfamiliar facets, and these need more explanation than the familiar ones. In particular, if you encounter in this text a reference to the "beginner" or the "unwary", it signifies a point which is not made clear in the documentation, and has caught out this author at least once.

The first seven chapters, plus Appendix A, cover almost everything the working scientist needs to know in order to get started in using Python effectively. My editor and some referees suggested that I should devote the second half of the book to problems in

a particular field. This would have led to a series of books, "Python for Biochemists", "Python for Crystallographers", ..., all with a common first half. Instead I have chosen to cover just three topics, which, however, should be far more widely applicable in many different fields. Chapter 8 covers four radically different types of ordinary differential equations and shows how to use the various relevant black boxes, which are often Python wrappers around tried and trusted Fortran codes. The next chapter while ostensibly about pseudospectral approaches to evolutionary partial differential equations, actually covers a topic of great utility to many scientists, namely how to reuse legacy code, usually written in Fortran77, within Python at Fortran-like speeds, without understanding Fortran. The final chapter about solving very large linear systems via multigrid is also a case history in how to use object-oriented programming meaningfully in a scientific context. If readers look carefully and critically at these later chapters, they should gain the practical expertise to handle problems in their own field.

Acknowledgments are due to the many Python developers who have produced and documented a very useful tool, and also to the very many who have published code snippets on the web, a great aid to the tyro, such as this author. Many of my colleagues have offered valuable advice. Des Higham generously consented to my borrowing his ideas for the last quarter of Chapter 8. I am especially grateful to Oliver Rinne who read carefully and critically an early draft. At Cambridge University Press, my Production Editor, Jessica Murphy and my Copy Editor, Anne Rix have exhibited their customary expertise. Last but not least I thank the Department of Applied Mathematics and Theoretical Physics, Cambridge for continuing to offer me office space after my retirement, which has facilitated the production of this book.

Writing a serious book is not a trivial task and so I am rather more than deeply grateful for the near-infinite patience of *Mary*, the "Python-widow", which made this book possible!