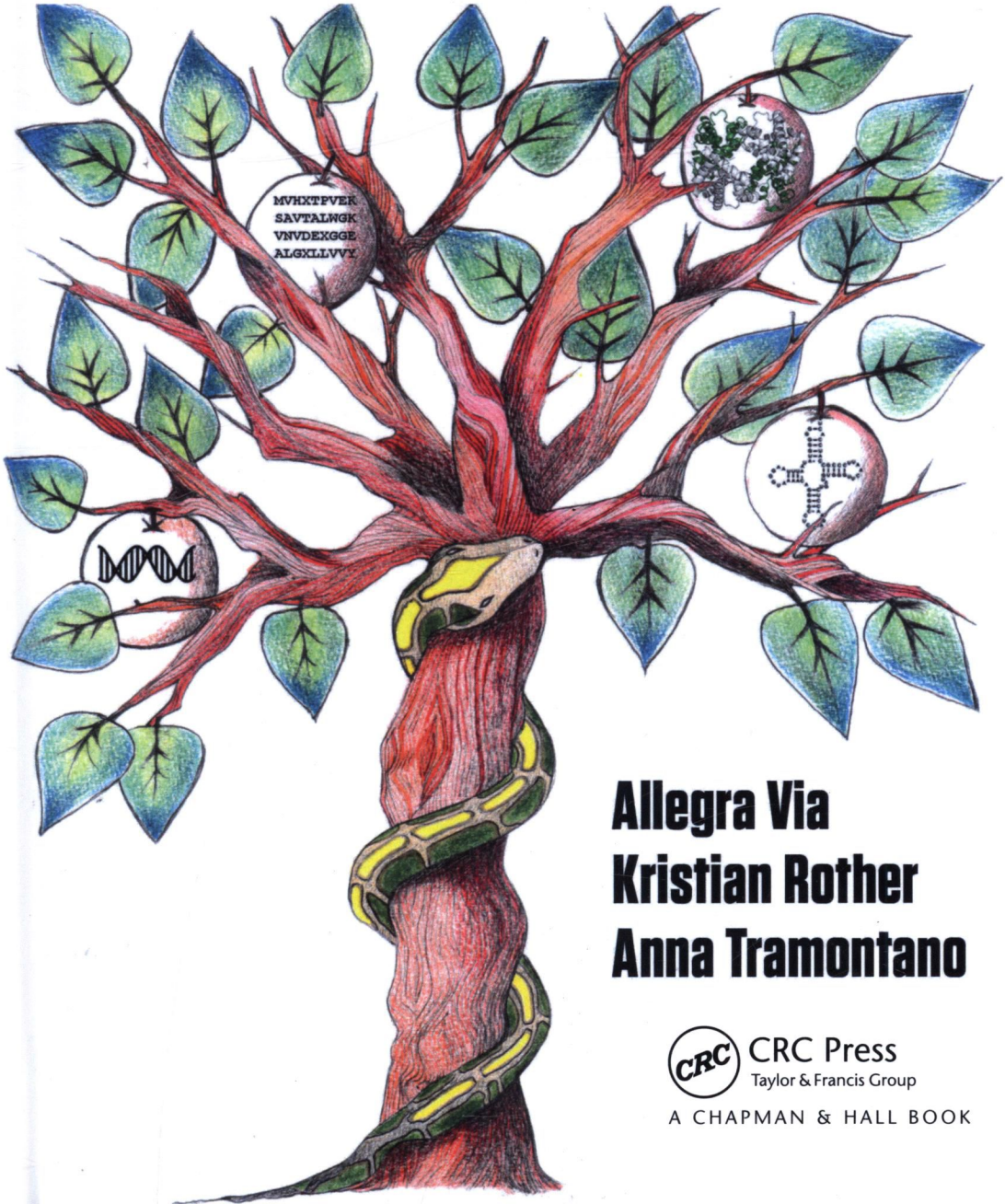
Chapman & Hall/CRC
Mathematical and Computational Biology Series

Managing Your Biological Data with Python



Allegra Via
Kristian Rother
Anna Tramontano

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Chapman & Hall/CRC
Mathematical and Computational Biology Series

"As a beginning structural biologist without any coding experience, this book would have been a welcome companion to quickly get me started on my bioinformatical projects with Python. ... The book introduces you to the basic principles of programming in Python using the many built-in functions. It does so using practical examples that you can start using right away in your day-to-day research. ... I'm confident that reading **Managing Your Biological Data with Python** will quickly allow you to get the most out of your data and start answering those trilling scientific questions you have, and do all of that while having fun."
—Marc van Dijk, Utrecht University

"**Managing Your Biological Data with Python** by Allegra Via et al. teaches Python using biological examples and discusses important Python-driven applications, such as PyMol and Biopython. The book is an excellent resource for any biologist needing relevant programming skills."
—Thomas Hamelryck, University of Copenhagen

"**Managing Your Biological Data with Python** is one of very few user-friendly books for biologists. ... It guides readers from writing simple functions through writing classes to building program pipelines—everything according to Python coding standards and in an easy-to-follow way. This is absolutely the best book to start learning Python. Intermediate Python users can use this book to learn some new tricks that they could implement in their own code. I highly recommend this book to researchers, students, and lecturers."
—Barbara Uszczynska, Centre de Regulació Genòmica

"The book is cleverly designed to cover a wide range of subjects in a pleasant, easy-to-follow sequence of chapters. ... as a single book to support learning Python for problem solvers in the life sciences, this book is certainly a very smart choice. It is also ready for creative teachers to develop more in the same direction."
—Pedro L. Fernandes, Instituto Gulbenkian de Ciência

ISBN 978-1-138-40722-0

CRC Press
Taylor & Francis Group
an informa business
www.crcpress.com

Managing Your Biological Data with Python

Via, Rother, and Tramontano

CRC

Chapman & Hall/CRC Mathematical and Computational Biology Series

Managing Your Biological Data with Python

Allegra Via
Kristian Rother
Anna Tramontano

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

First issued in hardback 2017

ISBN 13: 978-1-138-40722-0 (hbk)
ISBN 13: 978-1-4398-8093-7 (pbk)

Library of Congress Cataloging-in-Publication Data

Via, Allegra.
Managing your biological data with Python / Allegra Via, Kristian Rother, Anna Tramontano.
pages cm. -- (Chapman & Hall/CRC mathematical and computational biology series)
Includes bibliographical references and index.
ISBN 978-1-4398-8093-7 (alk. paper)
1. Biology--Data processing. 2. Python (Computer program language) I. Rother, Kristian. II. Tramontano, Anna. III. Title.

QH324.2.V526 2013
570.285--dc23 2013026177

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

Managing Your Biological Data with Python

CHAPMAN & HALL/CRC
Mathematical and Computational Biology Series

Aims and scope:
This series aims to capture new developments and summarize what is known over the entire spectrum of mathematical and computational biology and medicine. It seeks to encourage the integration of mathematical, statistical, and computational methods into biology by publishing a broad range of textbooks, reference works, and handbooks. The titles included in the series are meant to appeal to students, researchers, and professionals in the mathematical, statistical and computational sciences, fundamental biology and bioengineering, as well as interdisciplinary researchers involved in the field. The inclusion of concrete examples and applications, and programming techniques and examples, is highly encouraged.

Series Editors

Proposals for the series should be submitted to one of the series editors above or directly to:
CRC Press, Taylor & Francis Group
3 Park Square, Milton Park
Abingdon, Oxfordshire OX14 4RN
UK

Published Titles

Algorithms in Bioinformatics: A Practical Introduction
Wing-Kin Sung

Bioinformatics: A Practical Approach
Shui Qing Ye

Biological Computation
Ehud Lamm and Ron Unger

Biological Sequence Analysis Using the SeqAn C++ Library
Andreas Gogol-Döring and Knut Reinert

Cancer Modelling and Simulation
Luigi Preziosi

Cancer Systems Biology
Edwin Wang

Cell Mechanics: From Single Scale-Based Models to Multiscale Modeling
Arnaud Chauvière, Luigi Preziosi, and Claude Verdier

Clustering in Bioinformatics and Drug Discovery
John D. MacCuish and Norah E. MacCuish

Combinatorial Pattern Matching Algorithms in Computational Biology Using Perl and R
Gabriel Valiente

Computational Biology: A Statistical Mechanics Perspective
Ralf Blossey

Computational Hydrodynamics of Capsules and Biological Cells
C. Pozrikidis

Computational Neuroscience: A Comprehensive Approach
Jianfeng Feng

Computational Systems Biology of Cancer
Emmanuel Barillot, Laurence Calzone, Philippe Hupé, Jean-Philippe Vert, and Andrei Zinovyev

Data Analysis Tools for DNA Microarrays
Sorin Draghici

Differential Equations and Mathematical Biology, Second Edition
D.S. Jones, M.J. Plank, and B.D. Sleeman

Dynamics of Biological Systems
Michael Small

Engineering Genetic Circuits
Chris J. Myers

Exactly Solvable Models of Biological Invasion
Sergei V. Petrovskii and Bai-Lian Li

Game-Theoretical Models in Biology
Mark Broom and Jan Rychtář

Gene Expression Studies Using Affymetrix Microarrays
Hinrich Göhlmann and Willem Talloen

Genome Annotation
Jung Soh, Paul M.K. Gordon, and Christoph W. Sensen

Glycome Informatics: Methods and Applications
Kiyoko F. Aoki-Kinoshita

Handbook of Hidden Markov Models in Bioinformatics
Martin Gollery

Introduction to Bioinformatics
Anna Tramontano

Introduction to Bio-Ontologies
Peter N. Robinson and Sebastian Bauer

Introduction to Computational Proteomics
Golan Yona

Introduction to Proteins: Structure, Function, and Motion
Amit Kessel and Nir Ben-Tal

An Introduction to Systems Biology: Design Principles of Biological Circuits
Uri Alon

Kinetic Modelling in Systems Biology
Oleg Demin and Igor Goryanin

Knowledge Discovery in Proteomics
Igor Jurisica and Dennis Wigle

Published Titles (continued)

Managing Your Biological Data with Python
Allegra Via, Kristian Rother, and Anna Tramontano

Meta-analysis and Combining Information in Genetics and Genomics
Rudy Guerra and Darlene R. Goldstein

Methods in Medical Informatics: Fundamentals of Healthcare Programming in Perl, Python, and Ruby
Jules J. Berman

Modeling and Simulation of Capsules and Biological Cells
C. Pozrikidis

Niche Modeling: Predictions from Statistical Distributions
David Stockwell

Normal Mode Analysis: Theory and Applications to Biological and Chemical Systems
Qiang Cui and Ivet Bahar

Optimal Control Applied to Biological Models
Suzanne Lenhart and John T. Workman

Pattern Discovery in Bioinformatics: Theory & Algorithms
Laxmi Parida

Python for Bioinformatics
Sebastian Bassi

Quantitative Biology: From Molecular to Cellular Systems
Sebastian Bassi

Spatial Ecology
Stephen Cantrell, Chris Cosner, and Shigui Ruan

Spatiotemporal Patterns in Ecology and Epidemiology: Theory, Models, and Simulation
Horst Malchow, Sergei V. Petrovskii, and Ezio Venturino

Statistical Methods for QTL Mapping
Zehua Chen

Statistics and Data Analysis for Microarrays Using R and Bioconductor, Second Edition
Sorin Drăghici

Stochastic Modelling for Systems Biology, Second Edition
Darren J. Wilkinson

Structural Bioinformatics: An Algorithmic Approach
Forbes J. Burkowski

The Ten Most Wanted Solutions in Protein Bioinformatics
Anna Tramontano

Preface

Only a few years ago, programming was a prerogative of computational scientists. Notwithstanding this, programming is increasingly becoming a need of specialists in other fields such as biology. As a biologist, you are not necessarily interested in becoming an expert programmer, but you want to continue your scientific endeavors using programming as one of many tools. You may already have realized that programming techniques would dramatically speed up the management and analysis of your data. Maybe you want to deal with large amounts of data, repeat the same kind of analysis several times, or parse files with unusual formats. We can assure you that in all these cases programming is very useful. However, you may feel uncomfortable because you never had much interest in a "dry" and "conceptually hard" discipline such as computer science. In that case, this book is for you.

We wrote this book for life scientists who want to have more control of their data and, for this, need to learn some programming. It is aimed at empowering biologists without prior programming experience to work with biological data on their own using Python.

In the Preface, you will find a summary of what you can learn reading this book and an introduction of what a program is, followed by an overview of the Python programming language.

We hope that this book on programming is tailored to your needs as a biologist and will help you analyze your data and thus increase the likelihood to make better discoveries.

WHAT YOU CAN LEARN FROM THIS BOOK

In this book you will learn not only how to program but also how to manage your data, which means reading data from files, analyzing and manipulating them, and writing the results to a file or to the computer screen. Every single piece of code described in the book is aimed at solving

biological problems; every example deals with biological questions. The book proposes as many different cases as possible; covers many strategies to organize, analyze, and present data; and solves biological problems in the form of "programming recipes." Exercises that you can use to test yourself or include in a programming course for biologists appear at the end of each chapter.

The book is organized in six parts and contains twenty-one chapters in total. Part I introduces the Python language and teaches you how to write your first programs. Part II introduces all the basic elements of the language, enabling you to write small programs independently. Part III is about creating bigger programs using techniques to write well-organized, efficient, and error-free code. Part IV is devoted to data visualization. You will learn how to plot your data, or draw a figure for an article or a slide presentation. It also introduces PyMOL, a program to visualize macromolecular structures. Part V introduces you to Biopython, a programming library that helps with reading and writing several biological file formats and facilitates querying the NCBI databases online and retrieving biological records from the web. Part VI is a cookbook containing twenty specific programming "recipes," ranging from secondary structure prediction and multiple sequence alignment analyses to superimposing protein three-dimensional structures.

Furthermore, the book has four appendices. Appendix A provides an overview of both Python and UNIX commands. Appendix B lists several links to Python resources freely available on the web. Appendix C contains sample file formats cited throughout the book, such as a sequence in FASTA format, a sequence in GenBank format, a PDB file, an MSA example, etc. Finally, Appendix D is a short UNIX tutorial.

WHAT IS PROGRAMMING?

This book will teach you how to write programs. What exactly is a program? A program is conceptually similar to a cooking recipe. Like a recipe lists ingredients and kitchenware at the beginning, a program needs to define what objects (data and functions) are necessary. For instance, you could define a given DNA sequence as your data and define a function that calculates the GC-content in it. A recipe also contains a list of actions that must be carried out to use ingredients and kitchenware to prepare a dish. Likewise, a program contains a written list of elementary instructions such as "read the DNA sequence from a file," "calculate the GC-content,"

or "print the GC-content to the screen." Creating a program means writing instructions in a suitable language (e.g., Python), typically to a text file. Running a program means executing the instructions (i.e., the lines of code) listed in the program.

There is one big difference between kitchen recipes and computer programs, though: a human cook can divert from the recipe and add ingredients creatively or react to unexpected mishaps, which is important to obtain a tasty meal! A computer, however, is never creative. It reads the instructions in the program one by one and executes them by the letter. On one hand, the lack of computer creativity makes it necessary for you to explicitly tell it every tiny step, which can sometimes be unnerving. Imagine you are talking to a cook who is intellectually disabled but incredibly fast. On the other hand, computer predictability makes it easy to precisely repeat instructions many times. Imagine what a cook would say to an order of 100,000 identical dishes! Programming means using the rigid logics of computers to your advantage.

You must be aware that most of programming happens in your head. When you struggle to write a program, it may be helpful to formulate small step-by-step instructions in human language first. When the overall structure of your program is ready and you know exactly what you want it to do, it is time to start writing instructions. To do this, you need a programming language. In fact, programming basically consists of writing instructions in a given language to a text file or to a special terminal shell and telling your computer to execute them. The lines containing instructions are commonly called source code. Accordingly, programming or coding means writing source code. Since computers do not understand English, Italian, or German, you need to use a programming language to write source code. Our favorite language for answering biological questions is Python.

WHY PYTHON?

Python is simple to learn. It is a high-level programming language that is interpreted and object oriented. Let's analyze these concepts one by one.

Python Is Simple to Learn

A program can be written in one of many programming languages: C, C++, Fortran, Perl, Java, Pascal, etc. Every programming language has

formal rules and keywords (the syntax) and semantics (meaning). A key advantage of Python is that code is easy to read. Code can be more or less comprehensible to humans; for example, the Python instruction

```
print 'ACGT'
```

is quite intuitive (the computer will print the text ACGT to the screen), whereas the Perl instruction

```
$cmd = "imgcvt -i $intype -o $outtype $old.$num";
```

is less intuitive. Python is, compared to other programming languages, relatively similar to English and has a very simple syntax. We think this makes Python easy to learn for biologists.

Python Is a High-Level Programming Language

Python can also be used to do very complex things. You can represent complex data types like trees and networks, start other programs (e.g., bioinformatics applications) from Python, and download web pages. You also have tools to detect and handle errors in your programs. Finally, Python is not optimized for any particular purpose; it is therefore well apt to glue together other programs, web services, and databases in order to build customized scientific pipelines with a few lines of source code.

Python Is Interpreted

Some programming languages are interpreted, and some are compiled. For computers to execute a program, they need to translate the instructions to binary machine code, which is unreadable even for experienced programmers. In an interpreted language, each line is translated and executed one after another. In a compiled language, first the whole program is translated and only then executed. Execution of compiled languages is generally much faster than execution of interpreted ones. However, you need to compile the program each time you change something. With an interpreted language, you can see the effect of your changes immediately and, as a result, write programs faster. Therefore, we think that an interpreted language like Python is much easier to start with.

Python Is Object Oriented

In Python, everything is an *object*. Objects are independent program components representing data and instructions. They allow you to connect

data with useful functionalities (e.g., you could have a sequence object that contains a DNA sequence and functions for transcribing and translating this sequence). Objects help to structure complex programs and make program components reusable.

Using Python, many developers have made reusable objects available in programming libraries. For instance, reading and parsing a FASTA sequence file using Biopython can be done in two lines of code. Without the library, you would have to write ten to thirty lines, depending on the programming language. Therefore, object orientation in Python helps you to write short programs.

In conclusion, we believe that Python is an ideal language for those who want to have fun with little or no pain and learn programming to pragmatically manage biological data, solve biological problems, and widen the horizon of their scientific discoveries. We hope you will enjoy using this book at least as much as we enjoyed writing it!

Code Downloads

All code examples presented in this book are available online at https://bitbucket.org/krother/python-for-biologists, following the "Source" link.

Acknowledgements

We would like to thank the students and trainees to whom we had the privilege to teach Python. Your questions, problems, and ideas during Python courses over the past seven years are the main source of inspiration for this book. We can't name all of you, but we want you to know that we learned much from your enthusiasm, cheerfulness, frustration, and success.

Special thanks go to Pedro Fernandes, a great course organizer, who provided us with the opportunity to condense existing material into a five-day course at the Gulbenkian Institute in Portugal. We learned many of the key questions of this book during these courses and during after-dinner discussions in Astrolabio.

Additional credit goes to Janusz M. Bujnicki, Artur Jarmolowski, Jakub Nowak, Edward Jenkins, Amelie Anglade, Janick Mathys, and Victoria Schneider for providing various Python training opportunities.

We are also grateful to Francesco Cicconardi for his help with the RNA-Seq output parser and the NGS pipeline on which Chapters 6 and 14 are respectively based. He not only suggested us a typical NGS pipeline but also provided code and verified that the biological and computational discussions of the problem were correct and exhaustive.

We would like to thank Justyna Wojtczak, Katarzyna Potrzebowska, Wojciech Potrzebowski, Kaja Milanowska, Tomasz Puton, Joanna Kasprzak, Anna Philips, Teresa Szczepinska, Peter Cock, Bartosz Telenczuk, Patrick Yannul, Gavin Huttley, Rob Knight, Barbara Uszczynska, Fabrizio Ferre', Markus Rother, and Magdalena Rother for providing examples and constructive feedback.

Finally, many thanks to Alba Lepore for discussions during the realization of the book and for key help in accomplishing the book's cover.