

## Computing for Biologists

### Python Programming and Principles

Computing is revolutionizing the practice of biology. This book, which assumes no prior computing experience, provides students with the tools to write their own Python programs and to understand fundamental concepts in computational biology and bioinformatics.

Each major part of the book begins with a compelling biological question, followed by the algorithmic ideas and programming tools necessary to explore it: the origins of pathogenicity are examined using gene finding, the evolutionary history of sex determination systems is studied using sequence alignment, and the origin of modern humans is addressed using phylogenetic methods. In addition to providing general programming skills, this book explores the design of efficient algorithms, simulation, NP-hardness, and the maximum likelihood method, among other key concepts and methods.

Easy-to-read and designed to equip students with the skills to write programs for solving a range of biological problems, the book is accompanied by numerous programming exercises, available at [www.cs.hmc.edu/CFB](http://www.cs.hmc.edu/CFB).

**Ran Libeskind-Hadas** is the R. Michael Shanahan Professor of Computer Science at Harvey Mudd College, USA, working in the areas of algorithms and computational biology. He is a recipient of both the Iris and Howard Critchell Professorship and the Joseph B. Platt Professorship for teaching, as well as the Distinguished Alumni Educator Award from the University of Illinois Urbana-Champaign Department of Computer Science.

**Eliot Bush** is Associate Professor of Biology at Harvey Mudd College, USA. His main research interest is the study of evolution. Among other things he has modeled the evolution of metabolism, characterized DNA methylation patterns in insects, developed algorithms for studying substitution bias in DNA, and analyzed a 30 million-year-old primate fossil. His teaching interests focus on incorporating computers and programming assignments into biology coursework.

Cambridge University Press  
978-1-107-04282-7 - for Biologists: Python Programming and Principles  
Ran Libeskind-Hadas and Eliot Bush  
Frontmatter  
[More information](#)

---

# Computing

Cambridge University Press  
978-1-107-04282-7 - for Biologists: Python Programming and Principles  
Ran Libeskind-Hadas and Eliot Bush  
Frontmatter  
[More information](#)

# for Biologists

## Python Programming and Principles

Ran Libeskind-Hadas

Department of Computer Science,  
Harvey Mudd College

Eliot Bush

Department of Biology,  
Harvey Mudd College



Cambridge University Press  
978-1-107-04282-7 - for Biologists: Python Programming and Principles  
Ran Libeskind-Hadas and Eliot Bush  
Frontmatter  
[More information](#)

## CAMBRIDGE UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

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](http://www.cambridge.org)

Information on this title: [www.cambridge.org/9781107042827](http://www.cambridge.org/9781107042827)

© R. Libeskind-Hadas and E. Bush 2014

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 2014

Printed in the United States of America by Sheridan Books, Inc.

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

*Library of Congress Cataloguing in Publication data*

Libeskind-Hadas, Ran.

Computing for biologists / Ran Libeskind-Hadas, Department of Computer Science, Harvey Mudd College, Eliot Bush, Department of Biology, Harvey Mudd College.

pages   cm

Includes index.

ISBN 978-1-107-04282-7 (Hardback) – ISBN 978-1-107-64218-8 (Paperback)

1. Biology–Data processing.   2. Python (Computer program language)   3. Computer programming.

I. Bush, Eliot Christen.   II. Title.

QH324.2.L53 2014

570.285–dc23   2014014322

ISBN 978-1-107-04282-7 Hardback

ISBN 978-1-107-64218-8 Paperback

Additional resources for this publication at [www.cambridge.org/c4b](http://www.cambridge.org/c4b)

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.

## CONTENTS

Preface	<i>page ix</i>
<b>0 Meet Python</b>	<b>1</b>
0.1 Getting Started	1
0.2 Big Numbers	3
0.3 Strange Division	3
0.4 Naming Things	4
0.5 What's in a Name?	6
0.6 From Numbers to Strings...	6
0.7 Slicing	8
0.8 Adding Strings	9
0.9 Negative Indices	10
0.10 Fancy Slicing	10
0.11 And Now to Lists...	11
0.12 Lists for Free!	13
0.13 Changing Values	14
0.14 More on Mutability	17
0.15 Booleans	18
Putting it All Together	21
<b>Part I Python versus Pathogens</b>	<b>23</b>
<b>1 Computing GC Content</b>	<b>25</b>
1.1 Representing DNA on a Computer	26
1.2 Python Functions	26
1.3 A Short Comment on Docstrings	28
1.4 Bigger and Better Functions	28
1.5 Why Write Functions?	29
1.6 Making Decisions	29
1.7 A Potential Pitfall	32
1.8 GC Content of Strings of Length 1, 2, 3, or 4: <code>if</code> , <code>elif</code> , <code>else</code>	34
1.9 Oops!	36
1.10 The "Perfect" GC Content Function: <code>for</code> Loops	38

1.11	Another Example of <code>for</code> Loops: Converting DNA to RNA	39
	Putting it All Together	40
<b>2</b>	<b>Pathogenicity Islands</b>	<b>41</b>
2.1	How <i>Salmonella</i> Enters Host Cells	41
2.2	Investigating Pathogenicity Islands	43
2.3	Looping Over Lists	44
2.4	Looping Over Lists with <code>range</code>	45
2.5	From Gum to CAT Boxes	47
2.6	Functions Can Call Other Functions!	49
	Putting it All Together	52
<b>3</b>	<b>Open Reading Frames and Genes</b>	<b>53</b>
3.1	Open Reading Frames and the Central Dogma	53
3.2	GC Content and ORFs	54
3.3	The <code>countStarts</code> Function	56
3.4	The <code>genString</code> Function	57
3.5	<code>while</code> Loops and Population Genetics	59
	Putting it All Together	64
<b>4</b>	<b>Finding Genes (at last!)</b>	<b>65</b>
4.1	From ORFs to Genes	65
4.2	Genes Occur on Both Strands	67
4.3	Determining the Function of a Protein	67
	Putting it All Together	69
	<b>Part II Sequence Alignment and Sex Determination</b>	<b>71</b>
<b>5</b>	<b>Recursion</b>	<b>75</b>
5.1	A Brief Diversion Before Recursion	76
5.2	And Now For Recursion!	77
5.3	The Factorial Function	80
5.4	How to Write a Recursive Function	82
5.5	Another Example: Recursive Reverse	85
	Putting it All Together	85
<b>6</b>	<b>The Use-It-Or-Lose-It Principle</b>	<b>87</b>
6.1	Peptide Fragments	87
6.2	Making Change	92

6.3	Longest Common Subsequence	96
	Putting it All Together	99
<b>7</b>	<b>Dictionaries, Memoization, and Speed</b>	<b>100</b>
7.1	Dictionaries	101
7.2	Using a Dictionary	102
7.3	What Kinds of Things Can be Keys and Values?	103
7.4	Optional Section: How Dictionaries Work (and why you should care)	105
7.5	Memoization	109
7.6	Memoizing LCS	113
	Putting it All Together	115
<b>8</b>	<b>Sequence Alignment and the Evolution of Sex Chromosomes</b>	<b>116</b>
8.1	The Sequence Alignment Score	116
8.2	From Scores to Alignments	121
8.3	Sequence Alignment Scoring with Variable Rewards and Penalties	123
8.4	Optional Section: How Substitution Matrices are Computed	126
8.5	Optional Section: Getting the Actual Sequence Alignment	129
8.6	Identifying Orthologs	135
8.7	Comparing Chromosomes	136
	Putting it All Together	138
	<b>Part III Phylogenetic Reconstruction and the Origin of Modern Humans</b>	<b>141</b>
<b>9</b>	<b>Representing and Working with Trees</b>	<b>145</b>
9.1	Representing Trees	146
9.2	Computing with Trees	148
	Putting it All Together	152
<b>10</b>	<b>Drawing Trees</b>	<b>154</b>
10.1	Drawing Fractal Trees	158
10.2	Drawing Phylogenetic Trees	159
	Putting it All Together	162
<b>11</b>	<b>The UPGMA Algorithm</b>	<b>163</b>
11.1	The Algorithm	164
11.2	Implementing UPGMA in Python	169
11.3	Calibrating Trees	171
	Putting it All Together	172

CONTENTS

<b>Part IV Additional Topics</b>	<b>175</b>
12 RNA Secondary Structure Prediction	177
13 Gene Regulatory Networks and the Maximum Likelihood Method	186
Putting it All Together	195
14 Birds, Bees, and Genetic Algorithms	196
14.1 Fast Algorithms	198
14.2 Slow Algorithms	199
14.3 Genetic Algorithms	201
<b>Where to go from here</b>	<b>205</b>
Index	206



## PREFACE

### What's this book about?

The computer is the most powerful general-purpose tool available to biologists.

In part, this is due to the continuing rapid growth of biological data. For example, at the time of writing, the GenBank database had over 100 *million* genetic sequences with over 100 *billion* DNA characters. Among the contents of that database are genes from many organisms, annotated with what's known about their function.

Imagine that you're studying a bacterium and wish to understand what causes it to be infectious. One promising approach is to identify genes in the bacterium and compare these to known genes in GenBank. If you're able to find similar genes whose function is known, it will tell you a great deal about the role of the genes in your bacterium. This approach represents a computational challenge, and is, in fact, the topic of Part I of this book.

But searching enormous databases is not the only reason that computers are so useful to biologists. Many biological problems have a large number of different possible solutions and only a computer – programmed with carefully designed computational recipes or “algorithms” – has any chance of finding the right one. For example, biological molecules such as proteins and RNA fold into complex shapes that strongly impact their function. Computational techniques have been developed to predict how these molecules fold. Such techniques help us understand how proteins and RNA work and can even help us design new molecules to treat disease.

Simply put, computing is revolutionizing the practice of biology.

In order to fully appreciate and exploit the power of computation, biologists must be trained to “think” computationally. In practical terms, this means understanding fundamental computing concepts that recur in many applications *and* being able to write programs.

Why? Consider, for example, that there are well over 400 different software packages for phylogenetics (the study of the evolutionary relationships among organisms). An increasing level of computational sophistication is needed to select the appropriate software for a given application, use it correctly, and understand its

## PREFACE

abilities and limitations. This is true not just of phylogenetics, but also for many other areas of biology.

But using existing software will not be enough. The number and variety of computational problems that arise in biology are rapidly outpacing the functionality of software tools. Eventually, most biologists are likely to encounter problems that cannot be solved with existing software. Therefore, it is imperative for biologists to have the ability to write their own programs.

This book seeks to provide biology students with both an exposure to major computational ideas *and* practical programming skills. It requires no specific biology or computer science background. It is designed as a first-year college-level course and has been taught to that audience at Harvey Mudd College since 2009. The authors hereby acknowledge the generous grant support of HHMI for the development of that course.

In contrast to a typical introductory bioinformatics book, this book emphasizes programming over the use of existing software. By the time you've completed this book, and the online homework problems, you should feel comfortable writing programs for a wide array of applications in biology and beyond. You'll also have an understanding of computational ideas like "heuristics," "memoization" and "dynamic programming," "NP-completeness," and others, that will allow you to understand and compare the technical aspects of existing bioinformatics tools.

This book begins with Chapter 0, which offers a first introduction to the Python programming language. The rest of the book is organized into four parts, each comprising several chapters. Each part begins with a "large" biological question and the chapters in that part provide the computational and programming tools to answer that question.

At the end of each chapter – and sometimes even within a chapter – you'll see a question icon pointing you to one or more recommended problems. You'll find those problems at the url:



[www.cs.hmc.edu/CFB](http://www.cs.hmc.edu/CFB)

Let's get started!