Computational Biology

A Practical Introduction to BioData Processing and Analysis with Linux, MySQL, and R

Bearbeitet von
Röbbe Wünschiers

1. Auflage 2013. Buch. xxix, 449 S. Hardcover
ISBN 978 3 642 34748 1
Format (B x L): 15,5 x 23,5 cm
Gewicht: 877 g

Weitere Fachgebiete > Chemie, Biowissenschaften, Agrarwissenschaften > Entwicklungsbiologie > Bioinformatik

schnell und portofrei erhältlich bei

beck-shop.de
DIE FACHBUCHHANDLUNG

Die Online-Fachbuchhandlung beck-shop.de ist spezialisiert auf Fachbücher, insbesondere Recht, Steuern und Wirtschaft. Im Sortiment finden Sie alle Medien (Bücher, Zeitschriften, CDs, eBooks, etc.) aller Verlage. Ergänzt wird das Programm durch Services wie Neuerscheinungsdienst oder Zusammenstellungen von Büchern zu Sonderpreisen. Der Shop führt mehr als 8 Millionen Produkte.

Preface to the Second Edition

AWKology at Its Best

This year was full of innovative achievements in the field of computational biology and bioinformatics. I just like to mention two personal highlights: (a) The publication of a whole-cell computational model of the bacterium *Mycoplasma genitalium* that allows prediction of phenotype from genotype (Karr et al 2012) and (b) the coordinated publication of major results from the international ENCODE (Encyclopedia of DNA Elements) project as a set of 30 papers across three different journals that are digitally cross-linked as so-called threads. Each thread consists of theme-specific paragraphs, figures, and tables from across these papers. Both research projects handle a huge amount of complex data. But at the very basis there are a number of tabulator-delimited text files that needed to be filtered, rearranged, reformatted, statistically analyzed, or transferred to relational databases for improved data handling and number crunching.

This is precisely what this book is about. **My aim is to place you in a better position to handle and analyze data**—lots of data. It arose from my own needs and experiences to do so. Once you learn how to play with your datasets, these in turn may change your mindset (as Hans Rosling once put it). The core is data processing and visualization. Thus, this book is about data piping, not pipetting.

The Book's Title. The title of this book is **Computational Biology**. Some would argue that its content is **bioinformatics**. Why is that? I am a biologist and I complement my experiments with computational methods. **To me, computational biology is the complement to experimental biology**. During my professional career in industries I was head of projects which aimed at different things like gene discovery, data integration and visualization, and statistical analyses. I was employed as a bioinformatics manager. But was it bioinformatics that I was doing? Or was it computational biology? Or something else?

There certainly is a difference between computational biology and bioinformatics. However, it heavily depends on whom you are asking (see Sect. 1.3 on page 6). I often hear that bioinformatics is about the development of software tools while computational biology deals with mathematical modeling. A leading journal, PLOS Computational Biology, states that it publishes work that furthers our understanding of living systems at all scales through the application of computational methods. Computational methods in turn involve information processing—and this is what this book is about. Anyway, both terms are frequently used synonymously and the buzz word clearly is bioinformatics—I am glad that you still found this book.)

New Chapters, **Extended Concept and a Dinosaur**. What has changed since the publication of the first edition in 2004? A lot! Next-generation sequencing is not the next generation any more—it is presence. This means that there is a lot more data available (Strasser 2012). High throughput methods became the standard in molecular analysis and provide even more data. More data means that there are more and higher possibilities to find correlations between datasets. But it also implies that there are more datasets that have to be processed.

This new edition grew out of my experience of working with biological data in both academia and industries. I saw the need to add chapters on databases (**MySQL**) and statistical data analysis & visualization (**R**). From my courses on computational biology I learned about the importance of having a tangible problem to solve. This motivates to move on in the command line and likewise demonstrates its power. Therefore, I chose to add **worked examples**.

Since 2004, Linux has become much more comfortable—not only to use, but also to install. Gaining access to a USB-stick was a pain in the neck back then: `mount -t vfat /dev/hde1 /home/Freddy/USB/`. Nowadays, everybody can install a free virtual machine and run almost any operating system on any operating system—in parallel. I take advantage of these developments by showing how to set up **Ubuntu Linux in a VirtualBox**.

My dinosaur is **AWK**. Though there is almost no further development, it is still just amazing to see what one can do. I met several experimentalists who are into computational biology that apply AWK. Why? One of its three developers, Alfred Aho, recently said: *If I had to choose a word to describe our centering forces in language design*, *I'd say Kernighan emphasized ease of learning; Weinberger*, *soundness of implementation*; *and I*, *utility*. *I think AWK has all three of these properties* (Biancuzzi and Warden 2009). That describes it well. Students without any programming experience usually pickup data processing in the command line with AWK within some days. I love it.

Acknowledgments. I wish to thank all colleagues and students who have read, commented upon, and corrected various chapters of this book. This second edition benefited a lot from the waking eyes of the students attending my computational biology course, especially Sebastian Gustmann and Robin Schwarzer at Cologne University, Germany.

Quedlinburg, Germany, September 2012 Röbbe Wünschiers

References

Karr et al (2012) A whole-cell computational model predicts phenotype from genotype. Cell 150:389. simtk.org.
Strasser BJ (2012) Data-driven sciences: from wonder cabinets to electronic databases. Stud Hist Philos Biolo Biomed Sci 43:85.
Biancuzzi F, Warden S (eds) (2009) Masterminds of programming. O'Reilly, Sebastopol, p 104.

Preface to the First Edition

Welcome on Board!

With this book I would like to invite you, the scientist, to a journey through terminals and program codes. You are welcome to put aside your pipette, culture flask, or rubber boots for a while, make yourself comfortable in front of a computer (do not forget your favourite hot alcohol-free drink), and learn some unixing and programming. *Why?* Because we are living in the information age and there is a huge amount of biological knowledge and databases out there. They contain information about almost everything: genes and genomes, rRNAs, enzymes, protein structures, DNA-microarray experiments, single organisms, ecological data, the tree of life, and endless more. Furthermore, nowadays many research apparatuses are connected to computers. Thus, you have electronic access to your data. However, in order to cope with all this information you need some tools. This book will provide you with the skills to use these tools and to develop your own tools, i.e., it will introduce Unix and its derivatives (Linux, Mac OS X, CygWin, etc.) and programming (shell programming, awk, perl). These tools will make you independent of the way in which other people make you process your data—in the form of application software. What you want is open functionality. You want to decide how to process (e.g., analyze, format, save, correlate) data and you want it now—not waiting for the lab programmer to treat your request; and you know it best—you understand your data and your demands. This is what open functionality stands for, and both Linux and programming languages can provide it to you.

I started programming on a Casio PB-100 hand-held built in 1983. It can store 10 small Basic programs. The accompanying book was entitled "Learn as you go" and, indeed, in my opinion this is the best way to learn programming. My first contact to Unix was triggered by the need to copy data files from a Unix-driven Bruker EPR-Spectrometer onto a floppy disk. The real challenge started when I

tried to import the files to a data-plotting program on the PC. While the first problem could be solved by finding the right page in a Unix manual, the latter required programming skills—Q-Basic at that time. This problem was minor compared to the trouble one encounters today. A common problem is to feed one program with the output of another program: you might have to change lines to columns, commas to dots, tabulators to semicolons, uppercase to lowercase, DNA to RNA, FASTA to GenBank format, and so forth. Then there is that huge amount of information out there on the Web, which you might need to bring into shape for your own analysis.

You and This Book. This book is written for the total beginner. You need not even know what a computer is, though you should have access to one and find the power switch. The book is the result of (a) the way I learned to work with Unix, its derivatives, and its numerous tools and (b) a lecture which I started at the Institute for Genetics at the University of Cologne/Germany. Most programming examples are taken from biology; however, you need not be a biologist. Except for two or three examples, no biological knowledge is necessary. I have tried to illustrate almost everything practically with so-called terminals and examples. You should run these examples. Each chapter closes with some exercises. Brief solutions can be found at the end of the book.

Why Linux? This book is not limited to Linux! All examples are valid for Unix or any Unix derivative like *Mac OS X*, *Knoppix* or the free Windows-based *CygWin* package, too. I chose Linux because it is open source software: you need not invest money except for the book itself. Furthermore, Linux provides all the great tools Unix provides. With Linux (as with all other Unix derivatives) you are close to your data. Via the command line you have immediate access to your files and can use either publicly available or your own designed tools to process these. With the aid of *pipes* you can construct your own data-processing pipeline. It is great.

Why `awk` and `perl`? `awk` is a great language for both learning programming and treating large text-based data files (contrary to binary files). For 99 % you will work with text-based files, be it data tables, genomes, or species lists. Apart from being simple to learn and having a clear syntax, `awk` provides you with the possibility to construct your own commands. Thus, the language can grow with you as you grow with the language. I know bioinformatic professionals entirely focusing on `awk`.`perl` is much more powerful but also more unclear in its syntax (or flexible, to put it positively), but, since `awk` was one basis for developing `perl`, it is only a small step to go once you have learned awk – but a giant leap for your possibilities. You should take this step. By the way, both `awk` and `perl` run on all common operating systems.

Acknowledgments. Special thanks to Kristina Auerswald, Till Bayer, Benedikt Bosbach, and Chris Voolstra for proofreading, and all the other students for encouraging me to bring these lines together.

Hürth, Germany, January 2004 Röbbe Wünschiers