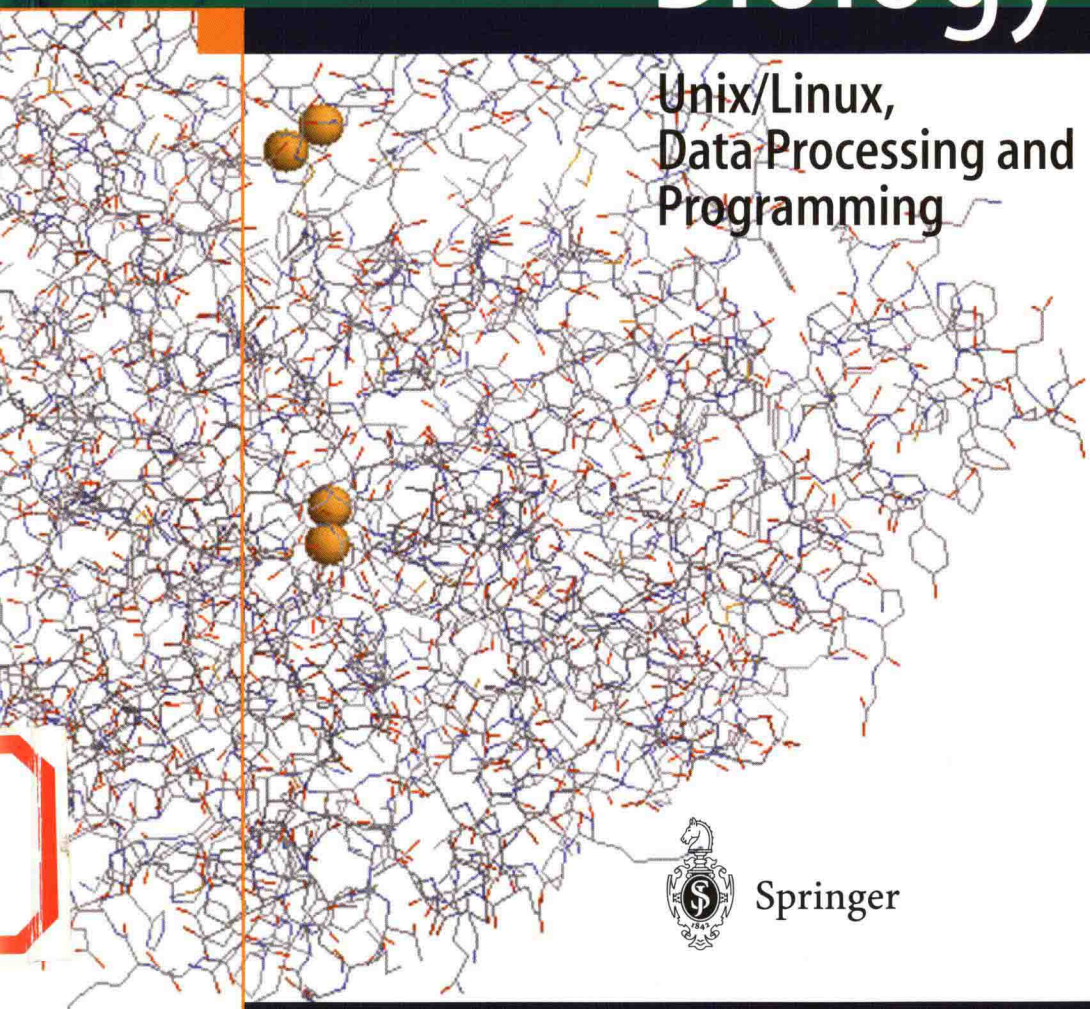
R. Wünschiers

Computational Biology

Unix/Linux, Data Processing and Programming



Springer

Röbbe Wünschiers

Computational Biology –

Unix/Linux, Data Processing and Programming

With 19 Figures and 12 Tables

Springer

Dr. Röbbe Wünschiers
University of Cologne
Institute for Genetics
Weyertal 121
50931 Köln
Germany

ISBN 3-540-21142-X Springer-Verlag Berlin Heidelberg New York

Library of Congress Control Number: 2004102411

Springer-Verlag is a part of Springer Science + Business Media
springeronline.com

Springer-Verlag Berlin Heidelberg 2004
Printed in Germany

Cover design: Design & Production, Heidelberg
Typesetting: Camera ready by the author
31/3150WI - 5 4 3 2 1 0 - Printed on acid-free paper

Röbbe Wünschiers

Computational Biology - Unix/Linux, Data Processing and Programming

Springer

Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Dedicated
to
Károly Nagy and to the Open-Source Community

Foreword

A shift in culture

Only a decade ago, the first thing a molecular biologist would have had to learn when he or she started the lab work was how to handle pipettes, extract DNA, use enzymes and clone a gene. Now, the first thing that he or she should learn is how to handle databases and to extract all the information that is already known about the gene that he or she wants to study. In all likelihood, he or she will find that the gene has already been sequenced from several organisms, that it was recovered in a variety of EST projects, that expression data are available from microarray and SAGE studies, that it was included in linkage studies, that proteomics data are rapidly accumulating, that lists of interacting proteins are being compiled, that domain structure data are available and that it is part of a network of genetic interactions which is intensively modelled. He or she will discover that all this information resides in many different databases with different data formats and with different levels of analyses and linking. Starting to work on this gene will make sense only, if all this information is put together in a project-specific manner and set into the context of what is known about related genes and processes. At this point he or she may decide to walk up to the bioinformatics group in house and ask for help with arranging the data in a useful manner. This will then turn into the first major frustration in his or her career, since the last thing a scientific bioinformatics group wants to do is to provide a service for data retrieval and management.

Molecular biology is currently going through a dramatic cultural shift. The daily business of pipetting and gel running has increasingly to be complemented with data compiling and processing. Large lists of data are produced by sequencers, microarray experiments or real-time PCR machines every day. Working with data lists has become as important as extracting DNA. Every bench scientist needs proficiency in computing; discoveries are made both at the bench and on the screen. It would be completely wrong to think that computers in molecular biology are the business of bioinformaticians only.

Bioinformatics has become a scientific discipline of its own and should not be considered to be a service provider. The day-to-day computing will always have to be done by the experimentalist himself or herself.

Of course, there are now also a lot of helpful and fancy program packages for the bench scientist; but these will only perform routine tasks and all too often they are only poorly compatible. A scientist needs the freedom to develop his or her own ideas and to link things that have previously not been linked. Being able to go back to the basics of computing and programming is therefore a vital skill for the experimentalist, as important as making buffers and setting up enzyme reactions. It allows him or her to handle and analyze the data in exactly the way it is required for the project and to pursue new avenues of research, rather than trotting old paths.

Unix is the key to basic computing. If one is used to Windows or Mac operating systems, this might at first sound like going back into the stone age; but the dramatic recent shift of at least the Mac operating system to a Unix base should teach us otherwise. Unix is here to stay and it allows the largest flexibility for bioinformatics applications. Those who have learned Unix will soon discover the myriad of little "progies" that are available from colleagues all over the world and that can make life much easier in the lab.

This book gives exactly the sort of introduction into Unix, Unix-based operating systems and programming languages that will be a key competence for experimentally working molecular biologists and that will make all the difference for the successful projects of the future. It has been written by a bench scientist, specifically with the needs of molecular biologists in mind. It can be used either for self-teaching or in practical courses. Every group leader should hand over this book to new students in the lab, together with their first set of pipettes.

Cologne/Germany,
January 2004

Prof. Dr. Diethard Tautz
(Institut für Genetik der Universität zu Köln)

Preface

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 on 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 in 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 to 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 how 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). To 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.

Acknowledgements – 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

Contents