Fundamentals of computational biology

Lecture notes

Camilo García-Botero

2022-08-31

Table of contents

Preface

We started this book with the aim of compiling the lectures of the course Fundamentals of Computational Biology offered at EAFIT University for undergrad students in Biology. The course has been taught from different perspectives from its creation. All main units are depicted in this book, although the course is mainly composed from at least four of them. We suggest to cover the first three units/parts and vary the last. Units in this book are partitioned as follows: i) Unix, ii) sequence analysis, iii) genomics, iii) metagenomics iv) RNA seq v) structural biology.

Lectures are focused on a theoretical-practical approach where basic concepts from biology, bioinformatics and computer science and interleave with the practice to solve challenges. Exercises or challenges are designed to improve students' abilities that are likely to be involved in real-life problems in computational biology.

This book is a work in progress. If you find issues, typos or relevant information to share with, anything is welcome.

Learning features

> **Note**
>
> Sometimes other fields might add interested value to the understanding of the computational biology area. This feature remarks some of them and aim to explain these intersections.

> **Tip**
>
> As you move forward in the computational biology field you will find that there are several tips and tricks (mainly from the command line) as well as some random CLI programs that can leverage your daily workflow as a researcher. Using this feature we highlight some of those that appeared to linger on the field.

> **Important**
>
> To help you consolidate your understanding we end most chapters with important messages or concepts that help you evaluate yourself as you move forward on the lessons.

> **Caution**
>
> When experimenting with the CLI and many other computational tools it is common to face several known errors and drawbacks. Then, we present some of them and how to sort them out.

> **Challenges**
>
> Since focused on a competences learning approach we have highlighted several real-life (but basic) *challenges* a researcher faces when approaching computational biology problem (from tool selection, usage and result analysis). Therefore the book section *challenges* presents a selection of these problems that will later be approached by a com-

putational biology strategy (mainly from the CLI).

As many analysis specialize on data analysis, many formats arise that optimize the processing steps or the data storing steps. Some of these formats are keystones of bioinformatic analyses. We present examples of some formats an describe its main elements.

Introduction

The present book gathers the lecture notes of the undergrad course in Fundamentals of Computational Biology that takes place in EAFIT University. The first part will focus on how to use the the command line interface (aka CLI). It includes a long-format chapter about the Unix tools and concepts that is taught during the first four class lessons. The second chapter of covers the basics of git and the principal workflows to work daily on a collaborative manner this is actually the second lesson of the course. The third lessons, highlights the importance of package manager systems (such as homebrew and conda or scoop) and briefly introduce the main concepts and relevant commands. Later, in a fourth lesson, we introduce important concepts about how computers handle the tasks or jobs, and the parallelization of them. We talk about the computer architectures, the cluster architectures and the job scheduling software called Slurm, which is used on our local HPC cluster.

The second part is dedicated to sequence analysis and will cover several topics related to sequencing technologies, sequence alignments. All this topics are covered from the biological perspective, mathematical notions and the practical computing approach. Some of the main bioinformatics file extensions are covered and the git

Third part is focused genomics. It will cover main algorithms for genome assembly and some relevant programs. Also the biological nuances of gene calling and gene annotation, and finally will variant calling analysis, which introduces the gene mapping concept and some of the most important bioinformatics file extensions.

Next parts are currently optionals for the course per-se and some iterations may only include one of those or some combinations. They are mainly dedicated to introduce metagenomics,

differential gene expression analysis and structural bioinformatics.

Course schedule

Bioinformatics vs. Computational biology

The extent of computational biology

?@fig-biofields.

Part I

The command line

1 The command line

In this chapter we will explore the fundamentals of the command line interface (aka CLI). We will distinguish the differences between Unix, CLI, Bash and Terminal and other concepts from the computer sciences.

As you will see the CLI is composed of several programs enabling the interaction with the machine, we will discuss some of the basics to navigate your machine, and some advance one that enable complex operations and automating tasks.

1.1 Command line basics

Before landing into the CLI let us consider the Unix concept. The first question that comes in this section is: what is Unix? It simply is an :operating system (OS). In other words, it is a set of programs that inter-operate with each other to let you communicate with the machine. A very important variant (or clone) of Unix is the very well known OS :Linux, which was created by :Linus Torvalds from scratch. The most important idea behind Unix based systems is the idea that we can use it to access information and hardware programmatically. Other main feature from Unix-like OS systems is the fact that data is usually stored as text files and the interface by which users communicate with the machine is also text-based (TUI: text user interface as opposed to GUI, graphic user interface).

Almost every computer has a way to interact with or access to the inner elements of the computer. Such interface is called the the command-line-interface Fig. **??**.

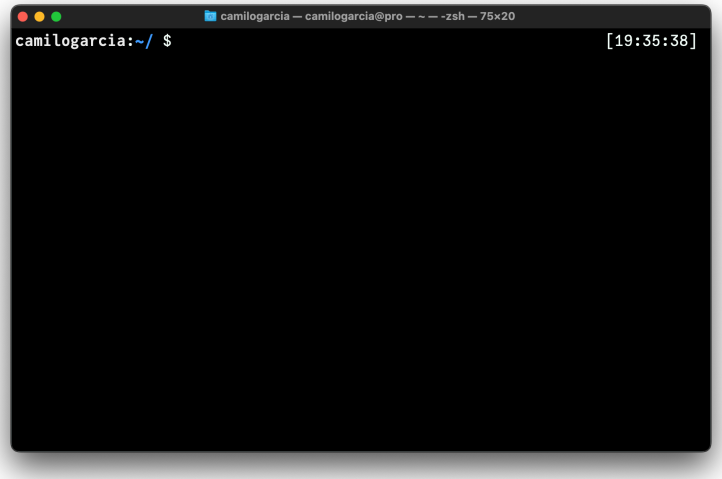

Figure 1.1: A **terminal** app displaying common features of the command line interface

1.1.1 File paths

Programs, files and directories on every machine (with Unix-like OS) display hierarchical paths (routes), starting out from the **root** (represented by the back-slash character `/`). The **root** represents the beginning of all the software installed in the machine. And many other files are nested from there forming a tree-like structure for the paths Fig. **??**

> **Tip**
>
> You can inspect the paths of a nested directory tree using `tree` command in you cli:
>
> ```
> tree -d -L 1
> ```

There are basically **two** ways to explore or navigate your file system. If you always represent it from the root, then you are presenting an **absolute path**. For instance the absolute path to my desktop is (`/Users/camilogarcia/Desktop`).

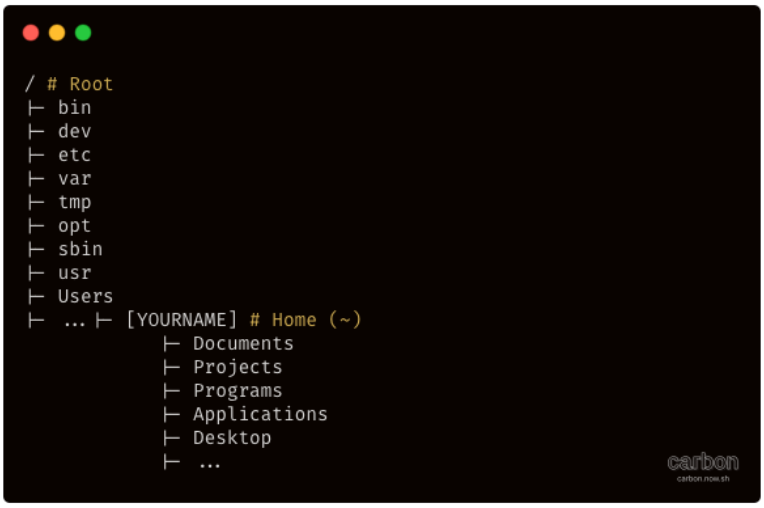

Figure 1.2: A terminal displaying tree-like structure of the programs in a machine with macOS

1.1.2 Basic Unix commands

Given that the vast majority of file systems are organized in file paths, the first question when starting with the CLI is "Where am I?". So Unix tool system is equipped with a bunch of commands but its basic ones are pretty much oriented to answer that question and navigating this text-based interface of files. The following three commands (`pwd`, `cd`, `ls`) will help you conquer the CLI.

1.1.2.1 Printing your working directory

To know where you are you can see your current location, that is to *print your working directory* using the `pwd` command.

```
pwd
```

1.1.2.2 Change to other directory

```
cd test-dir
```

> Tip
>
> Some basic arguments to navigate across your terminal:
>
> ```
> cd .. # change backwards
> cd ~  # change to the home
> cd /  # change to the root
> cd -  # change to previous dir
> ```

1.1.2.3 Listing files

```
ls
```

> Tip
>
> You can navigate your executed commands by typing ↑ or ↓.

1.1.2.4 Making new directories

```
mkdir test-dir
```

1.1.2.5 Creating a file

A simple command to create any file inside your terminal is `touch` it just create a file, but do not allow any editing.

```
touch new-file.txt
```

The `new-file.txt` is empty and created on your current location unless you assign another path when creating it. We suggest to take a look at Allison Horst illustrations, especially

on how to name files depending on the *case* see **?@fig-naming-files**

1.1.2.6 Printing files or inputs

```
cat new-file.txt
```

```
some
lines
that
were
written
```

```
echo "This will be printed"
```

```
This will be printed
```

1.1.2.7 Removing files or directories

```
rm
```

> **Tip**
>
> When having a long command, it becomes practically to go to the beginning or to the end of it. To do so you can use the key combination `Ctrl + A` and `Ctrl + E` respectively.

```
rmdir
```

1.1.3 Anatomy of a command

There is still many conventions by which the parts of a command line might be called, yet a very standard convention is presented in Fig. **??**