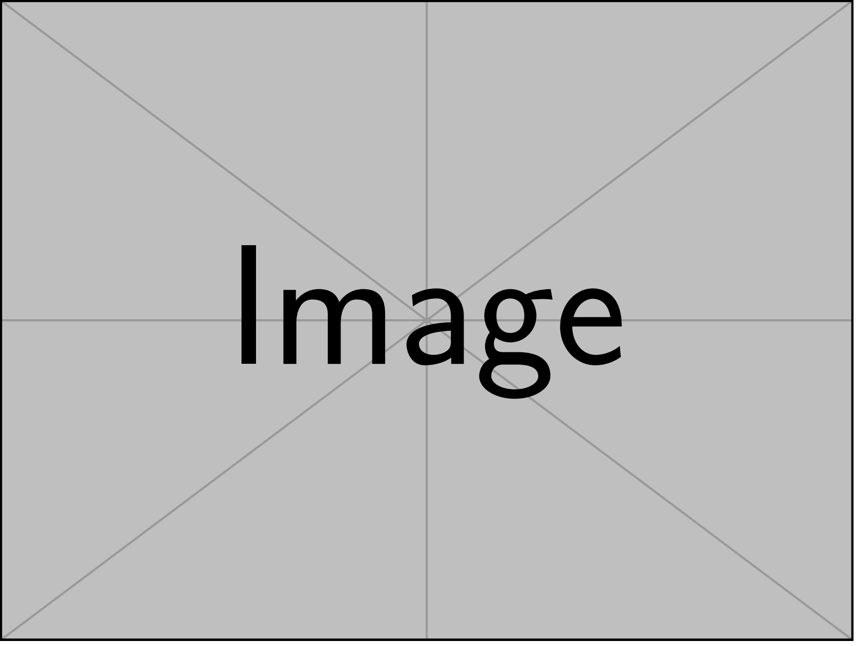


Baby Logic

Lecture Notes

作者：Duōyú Rén

时间：01 April 2023 à 01:28:36 +08:00

Table des matières

Preface

「邏輯」有很多種,「邏輯學」也有很多種。不同的人學習邏輯學有不同的**目的**。本筆記中的「邏輯學」知識主要服務於語言、思維的分析，不追求邏輯學在其他領域的功能。

本筆記大致列出邏輯基礎學習階段的主要材料，主要參考（chāoxí）以下課本攢集而成：

Bibliography

1. 徐明, 2008. **符号逻辑讲义** [M]. 武汉: 武汉大学出版社.
2. 胡龙彪, 黄华新, 2006. **逻辑学教程** [M]. 杭州: 浙江大学出版社.
3. 黄华新, 张则幸, 2011. **逻辑学导论（第二版）**[M]. 杭州: 浙江大学出版社.
4. 安德鲁 · 辛普森, 2005. **离散数学导学** [M]. 冯速, 译. 北京: 机械工业出版社.
5. Tidman P, Kahane H, 2002. ***Logic and Philosophy : A Modern Introduction[M]***. Ninth. Boston : Cengage Learning.
6. Smith N J J, 2012. ***Logic : The Laws of Truth[M]***. New Jersey : Princeton University Press.
7. Copi I M, Cohen C, Rodych V, 2018. ***Introduction to Logic[M]***. New York : Routledge.
8. Bergmann M, Moor J, Nelson J, 2014. ***The Logic Book[M]***. Sixth. New York : McGraw-Hill.

其中，直接取自《符號邏輯講義》的材料最多，取自《邏輯學導論》較多，素樸集合論、表列演算則直接取自完全開源的 Open Logic Project 項目源碼。

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

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 computational biology strategy (mainly from the CLI).

File format

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.

Notation

Mathematical notations

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.

Bioinformatics vs. Computational biology

The extent of computational biology

There so many fields on bioinformarics **?@fig-biofields**, that sometimes its hard to focus on the fundamentals. But this is also an opportunity to discuss the main aspects and differences across the fields.

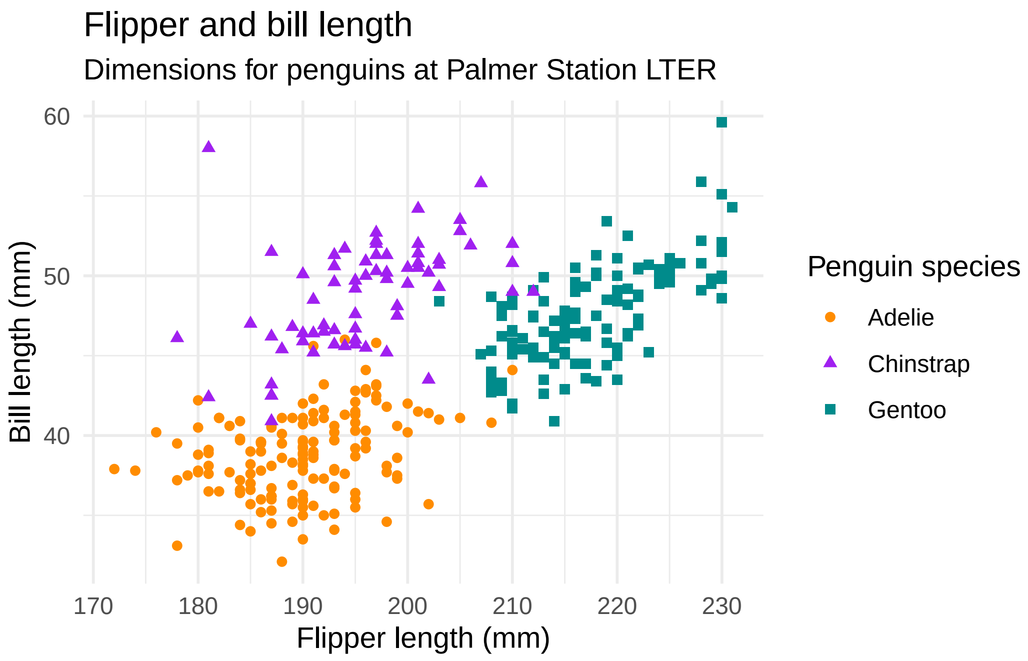
第一部分

Propositional

第 1 章 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.

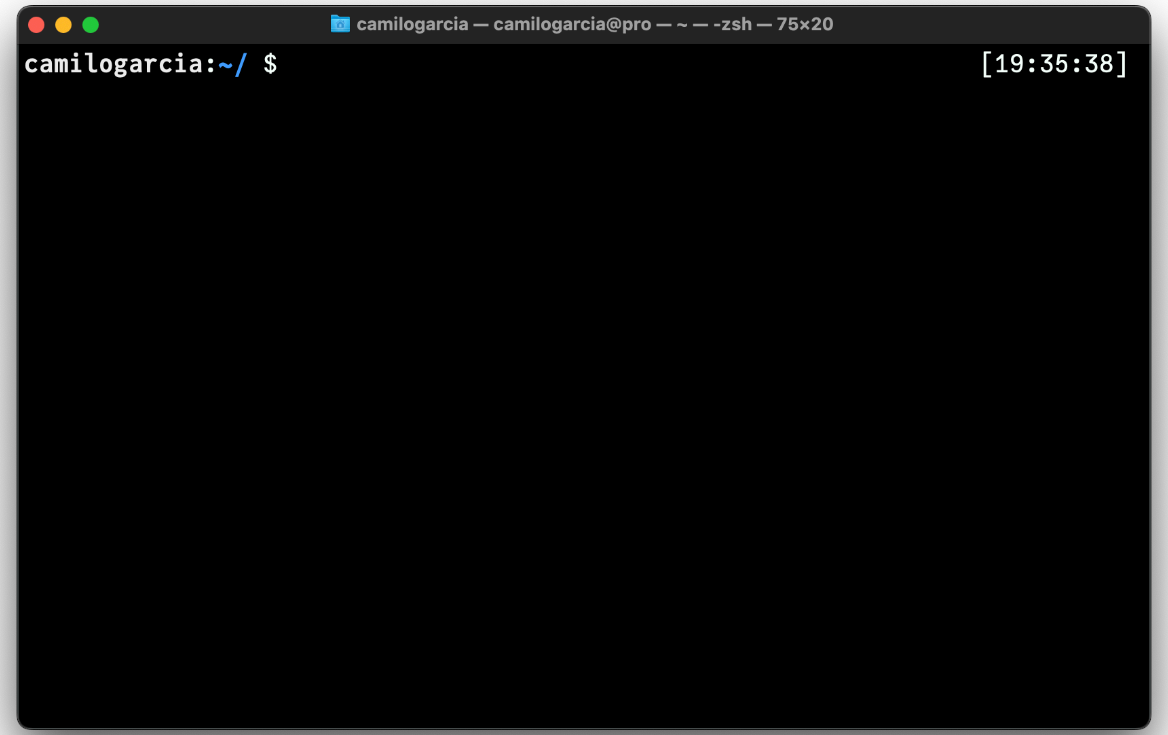
As you will see the CLI is composed of several programs enabling the interaction with the machine, we will discuss some of the basics to navigate your machine, and some advance one that enable complex operations and automating tasks.



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



```
camilogarcia:~/ $                                              [19:35:38]
```

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

computer. Such interface is called the the command-line-interface Fig. 1.1.

1.1.1 File paths

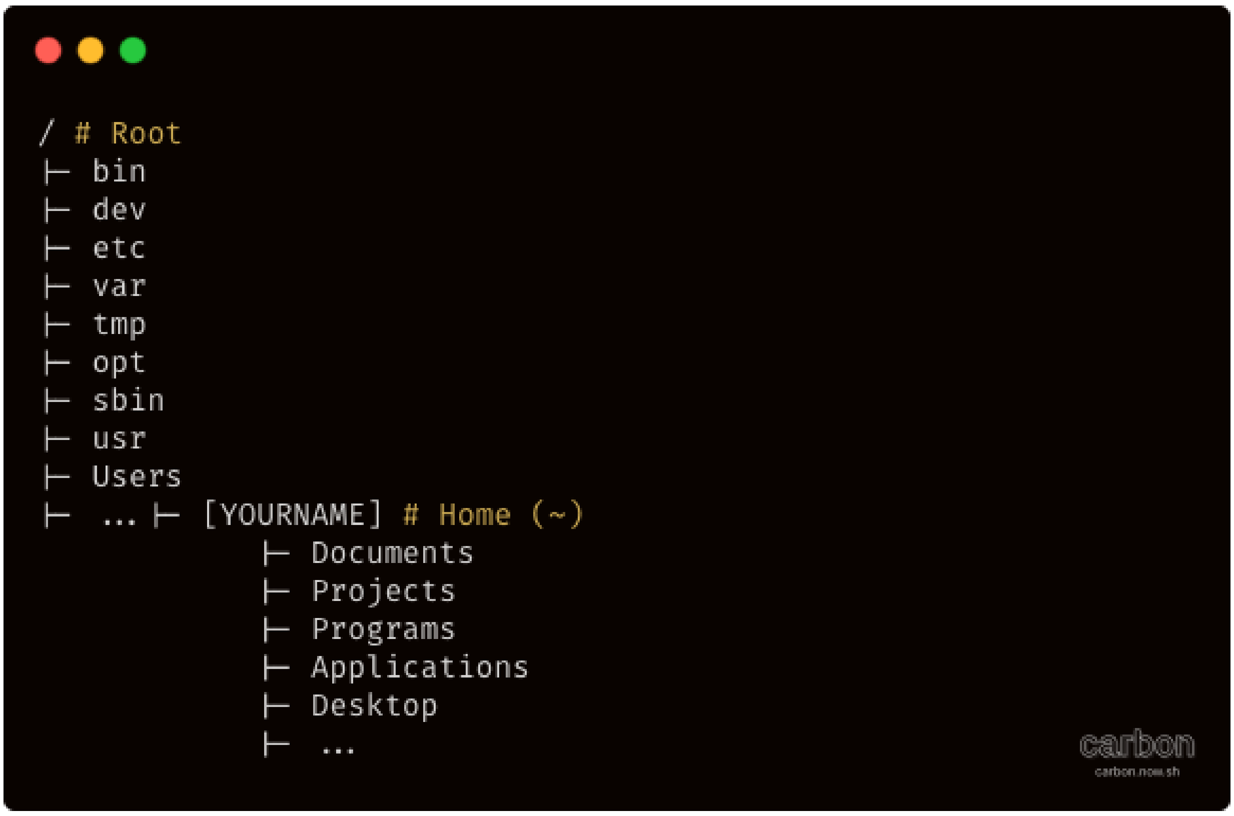
Programs, files and directories on every machine (with Unix-like OS) display hierarchical paths (routes), starting out from the **root** (represented by the back-slash character `/`). The **root** represents the beginning of all the software installed in the machine. And many other files are nested from there forming a tree-like structure for the paths Fig. 1.2

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`).



```
/ # Root
├ bin
├ dev
├ etc
├ var
├ tmp
├ opt
├ sbin
├ usr
├ Users
├  ... ├ [YOURNAME] # Home (~)
            ├ Documents
            ├ Projects
            ├ Programs
            ├ Applications
            ├ 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. 1.3

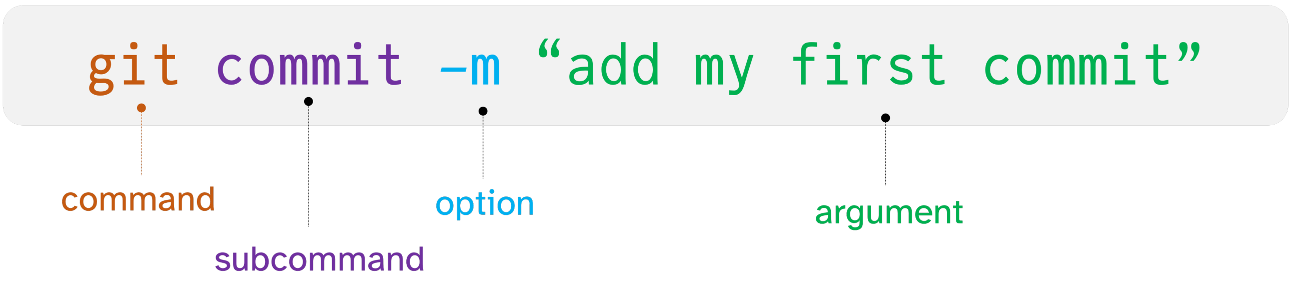


Figure 1.3: A simple command and a convention to call its main components

Some other for instance also tend to call the `option` as `flag`. This conventions are powerful because almost any command line interface display this structure (complex one add some other features and simple one tend to lack subcommands).

Challenge

Bacterial defense mechanisms to avoid bacteriophage infections are abundant. One of these is the :restriction-modification system (RM-System), which works by targeting a specific sites called *motifs*, shared by the phage and bacteria, with methylations. Motifs are commonly represented as a :sequence logo which is a probabilistic representation of the nucleotides at each position. The challenge consists of finding the number of times the motif from Fig. 1.4 appears on *B. tequilensis* EA-CB0015 genome using a command. Assume that probabilities are equal when multiple bases appeared at one site.

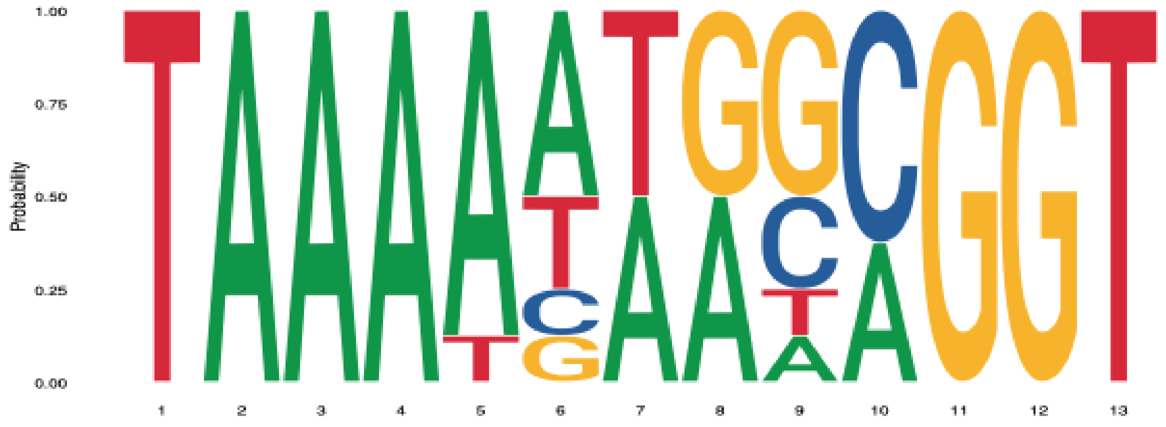


Figure 1.4: A RM-system motif logo

Before diving into an :answer take your time to think and solve it by your own.

1.2 Most important skills

When facing the CLI several issues or problems will arise. As for any other unintuitive challenge, a complete text interface Handling errors. Getting help Patience

1.3 Intermediate Unix

1.3.1 Special operators or metacharacters

Some operator or metacharacters have special functions in bash. For instance the `*` or *wildcard* is a regular expression character (sometimes called as a placeholder) that will turn in *any character, many times*, similarly the `?` represents *any character, once*. Whereas the `$` (dollar sign or operator) is intended for an special task : call *environmental variables* which means that once a variable is defined (e.g., `var=1`) this variable can be called via the `$` operator anytime `echo $var` will get us `1` as the standard output

1.3.2 Intermediate commands

```
wc
```

```
tr
```

```
grep
```

```
sed
```

1.3.3 Unix flows

Tip

When using the CLI at first its common to feal quite slow. Then, a very useful tip to boost the productivity from the command line is the autocompletion of commands by hitting `<tab>` after the initial command.

1.3.3.1 Redirection

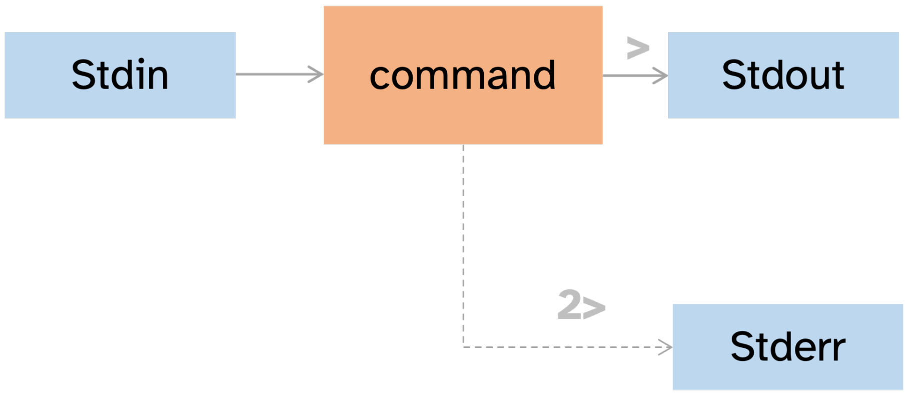


Figure 1.5: Redirecting flow

1.3.3.2 Pipe

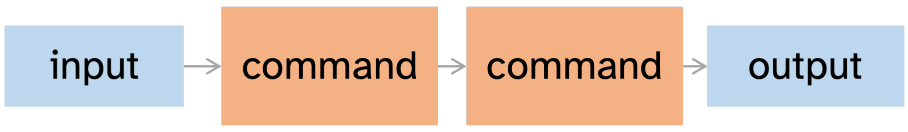


Figure 1.6: Pipe flow

Tip

When having a long command, it is also useful to jump by lines instead of character by character. To do so you can use the key combination `Alt + <-` and `Alt + ->` respectively.

1.3.4 loops, conditionals and script variables

Challenge

A second part of this challenges consists of create a script out from r the motif-search one-line command that recursively search the motifs in all genomes from a zip file that contains 10 bacterial genomes. The script should include the shebang, loops, conditionals and environmental variables.
See a possible script that solve the challenge :here

1.4 Advance Unix stuff

1.4.1 System permissions

1.4.2 Aliasing

1.4.3 The `.bashrc`

1.4.4 `awk` snippet

Challenge

:Restriction endonucleases (RE) cleave the DNA by digesting the :phosphodiester bond between two nucleotides. Many RE are directed to specific DNA motifs normally palindromic. There are mainly three types according to it digestive mechanism. RE have been widely used in molecular biotechnology because its specificity and versatility to carry out different experiments.
One of the main uses of RE is to generate a pattern of restricted fragments from different organisms so that samples of organisms, sequences or genes could be distinguished, as long as they display differences in the number of recognition motifs. This is normally done in the lab, where an RE is mixed with a DNA sample and later an :electrophoresis gel is run to see a separation pattern according to the fragments size.

Professor Javier has sequenced the genome of a sampled SARS-CoV2 and want to see the band pattern that the genome would display if it were digested with the RE EcoRV. He has asked you to help him with this problem. The expected output is a text file with the sizes of the fragments, where the size is the number of nucleotides of each fragment.

For more explanations on the basic commands in the command line we suggest to visit the first chapters of *Computing skills for biologist* from Allesina & Wilmes (2019)

A list of reading for this section :

Dudley & Butte (2009)

Perkel et al. (2021)

Brandies & Hogg (2021)

第二部分

Predicate

第 2 章 Algorithm thinking

2.1 What is an algorithm

2.2 Solving a problem step by step

2.3 Algorithms in computing

2.4 Relevance in biology

2.5 Pseudocode and notations

第三部分

Appendix

History

Edition 0.0.11 (2023-01-19)

- Adding some quarto code annotations!
- New chapter on algorithms
- Add a challenge on genome assembly
- Rendering with Q v.1.3

Edition 0.0.10 (2022-12-17)

- Update book with new quarto and extension versions
- Challenge on sequence alignment
- Experimenting new covers
- Some typos cleaning
- Change giscus theme

Edition 0.0.9 (2022-09-08)

- Starting the Sanger chapter
- More on Seq. analysis
- Keep history on book and add dates
- Change the cover
- Add new parts and chapters outlines (phylogenetics and metagenomics)
- New challenges on seq analysis solved

Ed. 0.0.8 (2022-08-29)

- New render w/ Quarto 1.1
- Adding more on sequence analysis
- Including new extension for videos
- Improve some images
- Book has now Giscus to enable discussions
- New info on download genomics data

- New foot page with license and more info

Ed. 0.0.7 (2022-08-19)

- Changes in git chapter
- More images in different chapters
- Start using new filters : lightbox for images and nutshell for expandable explanations
- Some images in HPC chapter
- Some outlines in Package manager chapter
- Some paragraphs on the sequence analysis chapter
- Additional challenges

Ed. 0.0.6 (2022-08-09)

- More updates to CLI chapter.
- Update the Version Control chapter.
- Decoupled genome annotation section into separate chapter.
- Add a motif search challenge
- Add cover image
- Many typos corrected.

Ed. 0.0.5 (2022-07-28)

- Update CLI chapter.
- Decoupled Seq analysis and Genomics into separate chapters.
- Chapters were reorganized in its own dirs to make easier navigation. They also were renamed for later easier addition of other chapters.
- New learning features using Quarto callouts. This will hopefully improve learning and enjoy the book.
- Chapters sections will have numbering.

Ed. 0.0.4 (2022-04-24)

- Minor typos and outline for the structural biology section
- Start improving crossreferences

Ed. 0.0.3 (2022-04-18)

- Citation and reference of the book is now almost complete

Ed. 0.0.2 (2022-04-17)

- This Ed. includes more information of the book and author
- Adds DOI and Zenodo
- Includes Social commenting

Ed. 0.0.1 (2022-04-12)

- This preliminary Ed. contains the basic outline of the book
- Contains the first draft solutions to challenges demos

Bibliographie

Allesina, S., & Wilmes, M. (2019). *Computing skills for biologists.* https://doi.org/10.2307/j.ctvc77jrc

Brandies, P. A., & Hogg, C. J. (2021). Ten simple rules for getting started with command-line bioinformatics. In *PLoS Computational Biology* (No. 2; Vol. 17, p. e1008645). Public Library of Science San Francisco, CA USA.

Dudley, J. T., & Butte, A. J. (2009). A quick guide for developing effective bioinformatics programming skills. In *PLOS computational biology* (No. 12; Vol. 5, p. e1000589). Public Library of Science San Francisco, USA.

Perkel, J. M. et al. (2021). Five reasons why researchers should learn to love the command line. *Nature*, *590*(7844), 173–174.