Baby Logic

Lecture Notes

Baby Logic

Lecture Notes

Duōyú Rén

https://nmfs-opensci.github.io/

Press of SānLǘ University

本課程爲面向新聞學專業的邏輯入門課程, 追求「邏輯」的「把話說清楚」的功能, 並不追求「邏輯學」的其他「用處」。因此, 本講義跳過或刪去了素樸集合論以及公理化系統、元理論等內容, 也不對「邏輯」與「邏輯學」作區分, 納入了部分(被一些專家視爲)非「邏輯學」的材料。所以, 目前所「攢」出的各章材料大多極爲基礎且相對簡易, 這些材料大多直接來自徐明《符號邏輯講義》, 一些案例材料則來自若干邏輯導論或邏輯基礎課本。按照一些課本的觀點, 這些材料已屬於 baby logic, 但爲了便於閱讀、理解, 二狗仍然簡單地增、刪、改寫了部分材料。本文檔仍在攢集中, 且僅用於課程學習, 不作任何其他用途。

ISBN: 9787111153962
https://book.douban.com/subject/1231286/

Developed with Quarto and TeXLive 2023. Composed and typeset with XƎLaTeX v. 0.999995, scrbook class. First printing in June 2020, last printing on **August 22, 2024**, by **Lǐ Èrgǒu**.

Preface

For the course ”R for Statistics” (MAT50303) it is assumed that you have acquired knowledge about statistics. The minimal required level is ”Advanced Statistics” (AS, MAT20306), ”Quantitative Research Methodology and Statistics” (QRMS, MAT22306), or the course ” Advanced Statistics for Nutritionists” (ASN, MAT24306). If you do not have sufficient preknowledge, or your statistical courses were so long ago that memory traces have evaporated, you are strongly advised to follow a statistics course first. AS, ASN and QRMS are presented in English.

The aim of ”R for Statistics” is to provide an introduction to R, a computer language and environment for statistics and graphics. The course will focus on getting familiar with the R environment by means of RStudio, to use R for manipulation and exploration of data and to perform all statistical analyses learned in the basic and advanced statistics courses, i.e. simple linear and multiple linear regression models, analysis of variance for Completely Randomized Designs (CRD), Randomized Complete Block Designs (RCBD) and factorial designs, ANalysis of COVAriance (ANCOVA), non-parametric methods and analysis of categorical data. The correct interpretation and explanation of the results of preformed statistical analyses is expected based on the assumed prior knowledge. Basic programming, visualization, as well as reproducibility of results in R will also be part of this course.

Students with an interest in statistics may benefit from the courses ”Modern Statistics for the Life Sciences” (MSLS, ABG30806), ”Statistics for Data Scientists” (MAT32806), ” Data Science for Plant Breeding and Genetics” (MAT33306) and/or ”Bayesian Data Analysis” (MAT34806) as a follow-up. All these courses are presented in English and build upon AS, or ASN. In MSLS logistic and log linear models (for binary and count data), variance components models (for dependent data, both balanced and unbalanced), general inference techniques (maximum likelihood estimation, Wald test, likelihood ratio test), posterior Bayesian inference using Markov Chain Monte Carlo, and applications in genetics and epidemiology are discussed. Apart from these courses students are also welcome at Biometris for a MSc-thesis Mathematical and Statistical Methods (MAT80418, MAT80439).

The aforementioned courses are organized by or in cooperation with Biometris. Biometris is part of Wageningen University & Research and is responsible for a number of short courses for PhD students and participates in many other courses as well. Biometris is a partner in the M.Sc. course in Applied Statistics at Leiden University. Staff members from Biometris are actively involved in research, mainly in statistical genetics, systems biology and ecology, food health and safety, omics, and big data. We have, however, an interest in other topics as well. If you have an interest in statistics, or work on your master thesis will involve use of advanced statistics, you might consider doing your thesis in cooperation with Biometris. In that case please contact a member of staff of Biometris about this.

Contents

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。

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.

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。

- 徐明, 2008. **符号逻辑讲义[M]**.武汉:武汉大学出版社.
- 胡龙彪,黄华新, 2006. **逻辑学教程[M]**.杭州:浙江大学出版社.

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 備註

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. 1, 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.

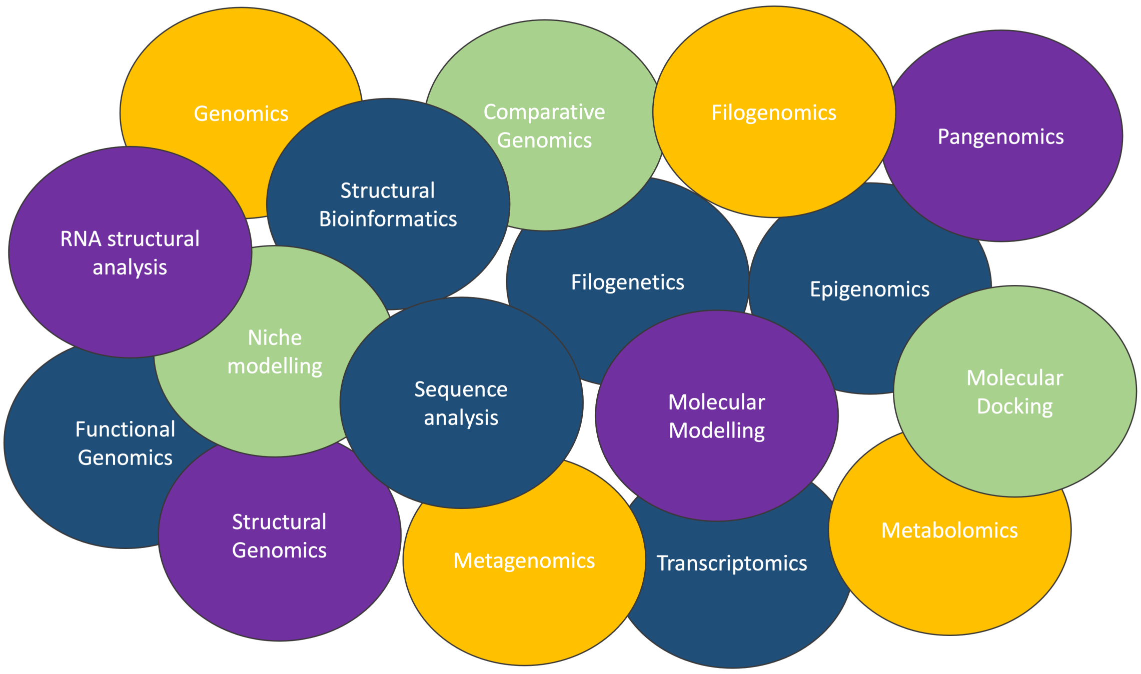


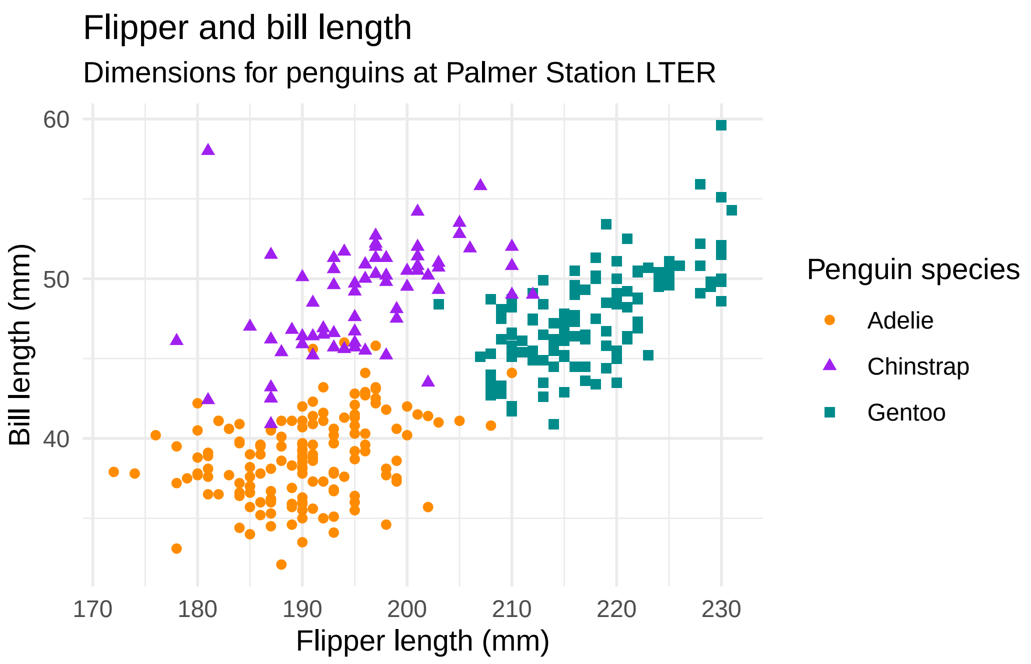
Fig 1: Different areas on bioinfomatics

Propositional

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.

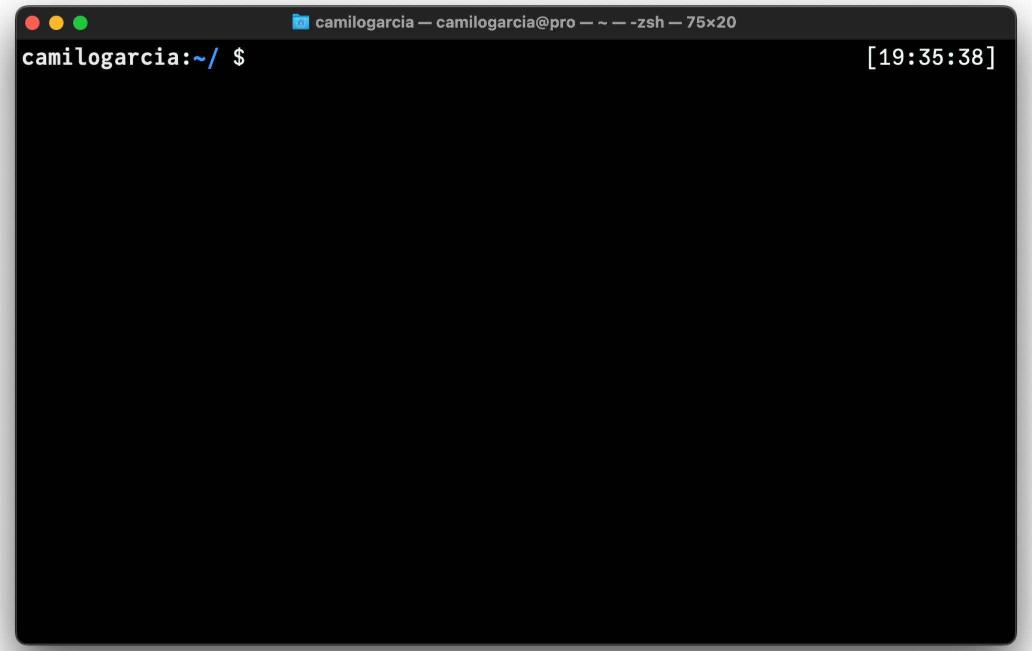
As you will see the CLI is composed of several programs enabling the interaction with the machine, we will discuss some of the basics to navigate your machine, and some advance one that enable complex operations and automating tasks.



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. 1.1.



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

Fig 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. 1.2

Tip

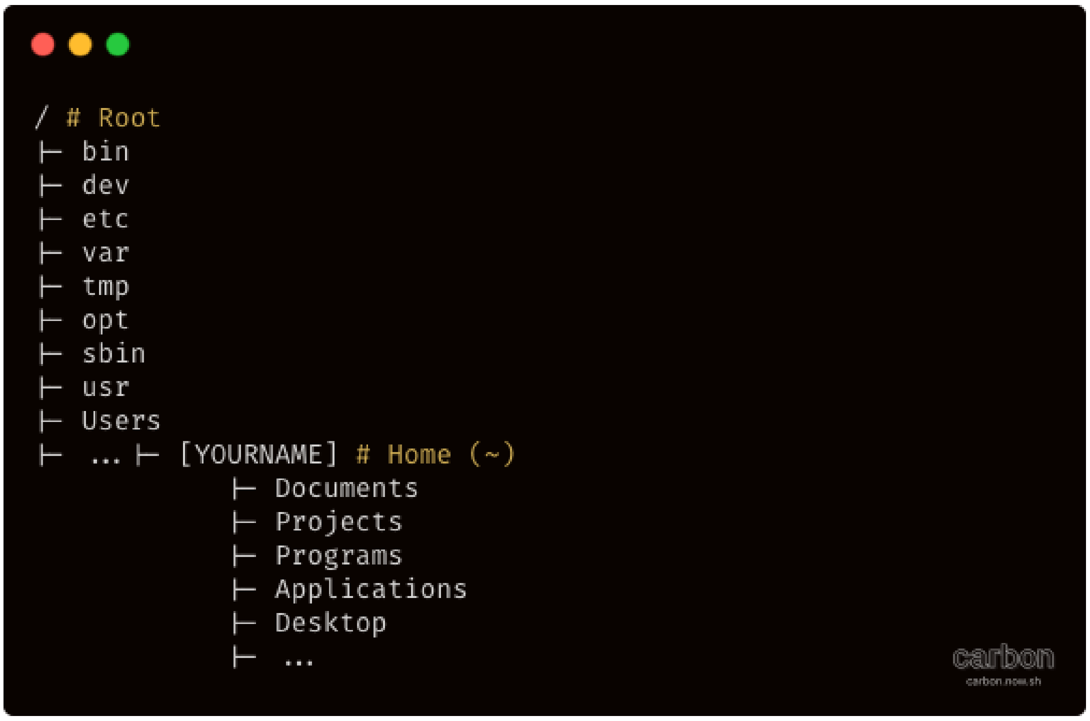
You can inspect the paths of a nested directory tree using treecommand in you cli :

```
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).

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.



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

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

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 pwdcommand.

```
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 touchit just create a file, but do not allow any editing.

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

The new-file.txtis 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 + Aand Ctrl + Erespectively.

```
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

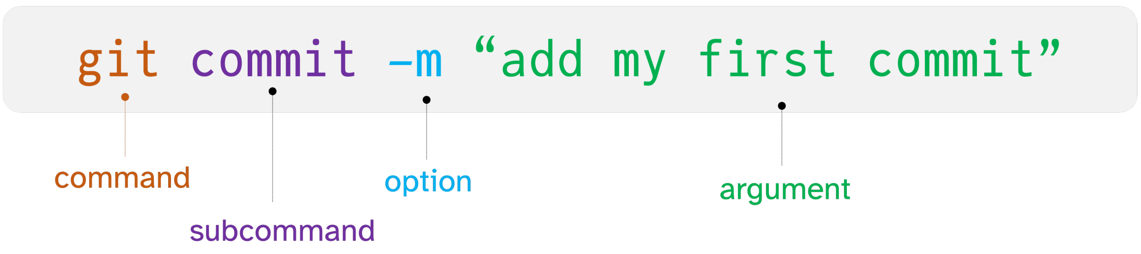


Fig 1.3: A simple command and a convention to call its main components

Some other for instance also tend to call the optionas 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.

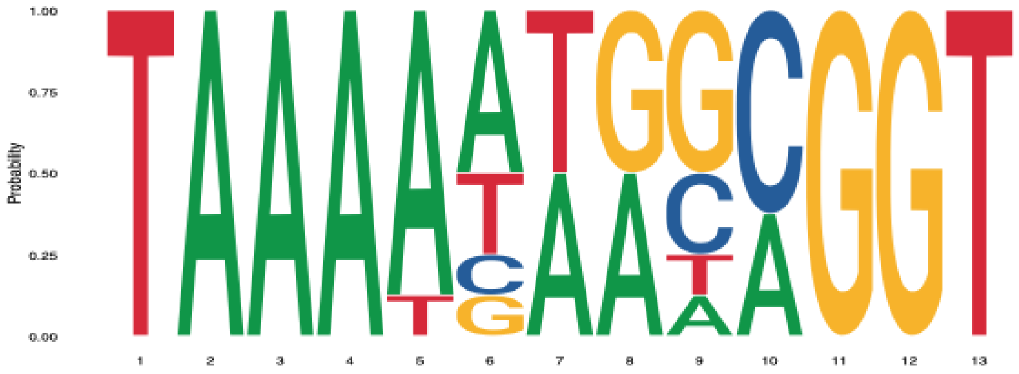


Fig 1.4: A RM-system motif logo

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 $varwill get us 1as 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

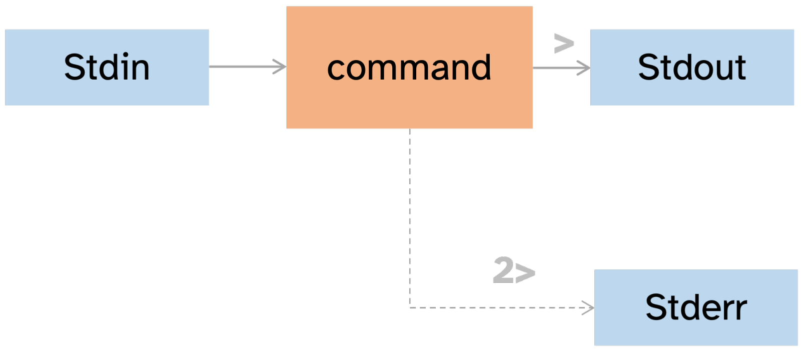


Fig 1.5: Redirecting flow

1.3.3.2 Pipe

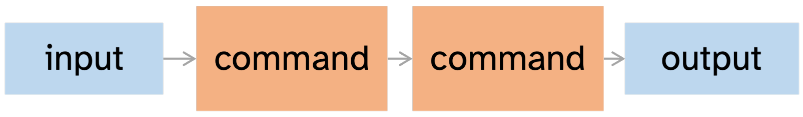


Fig 1.6: Pipe flow

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 awksnippet

⚠ 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 ([**ref-allesina2019**])

A list of reading for this section :
Dudley & Butte ([**ref-dudley2009**])
Perkel et al. ([**ref-perkel2021**])
Brandies & Hogg ([**ref-brandies2021**])

2 Version control

Here we will first address what is version control, its importance and basics on the local workflow.

Later we will introduce GitHub and explore more advanced commands for the collaboration workflow

2.1 Version control systems

As its name suggests, a version control system (VCS) allow you to keep record of the changes happening while working files and directories. Several VCSs have been created but the most popular is git. It is characterized by being a **distributed** VCS, which means that changes history are recorded locally (whether in a user laptop or user account) in contrast to other **centralized** VCSs that changes are saved on a shared machine or server.

So, why bother to learn a VCS in bioinformatics ? Well there are many reasons, but to highlight some of them : i) Since VCSs allow you to record changes, you can always trace back the steps made in ana analysis, which is nice for the **reproducibility** of your work. ii) a system like git could be coupled with a shared-centralized server as it is GitHub (we'll talk about it later and then one could **share and collaborate**, expanding the extent of your research and iii) following the structures and command from git its at first overwhelming and demands consistency and order, then when scaling a project it will payoff this stepping curve of learning by keeping the **efficiency** of your work.

2.2 Git installation and configuration

Installation could proceed from the official page of git. If working from WSL it has the binaries preinstalled, so you can jump directly to the configuration. The second step is to configure your user name and an email. with the following lines :

```
git config --global user.name "Your Name"
git config --global user.email user@eafit.edu.co
```

You could always user the preferred e-mail. More configurations are available, for instance the preferred editor to work with and so on, you can explore by asking for help git config --helpor git config --list.

2.2.1 The basics

There are at least **six** basic commands. Three of them allow recording local changes (git init, git addand git commit) and the other three help you to inspect the state of the changes (git status, git diffand git log), we will dive into the detail in the following lines. So, to start recording changes in a directory you must **initialize** the directory (which will now be called repository) using git init. This is a one-time command to get started.

2.2.2 The local workflow

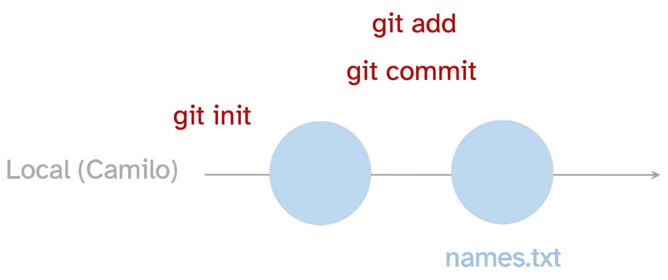


Fig 2.1: Local workflow of a git project adding and committing changes

#local-workflow

2.3 Exploring GitHub

As mention before, one of the great advantages that git can later achieve is to collaborate. However, to get into that a sharable server must allow users to have a common ground, and this is what GitHub allow. GitHub is a web platform where the local repositories become public and any user can access to the controlled versions of an image of your repo. The famous pet is the octocat Fig. 2.2

To enable the communication with a remote repo, git has encoded many specific commands, once the repos are **cloned** a simple workflow from the own local and remote repos is made possible thanks to two simple commands git pushand git pull.

There a several ways to starting out a remote project, whether it starts from a local folder or whether it starts from a remote repo. The second strategy is sometimes easier as you just need to later git clonethe remote into a local folder. To do so every repo has its own code-icon </> to later copy the repo link and later hit git clone <https...>on the desired folder. Now you got a linked copy of the remote on your machine.

Caution

But before working on a remote and pushing your first commits, it is common to find an error regarding the remote branch (also called ***origin***). There are several ways to avoid this caveat, but a very anticipated way is to configure git

```
git config --global push.autoSetupRemote true
```

This will save you from every time typing git push --set-upstream origin <main>when working on a new remote repo.

Cloning is therefore the process of creating a local copy of remote repository, that is a machine version of the remote repo, later all the common local workflow is carried out normally Fig. 2.3

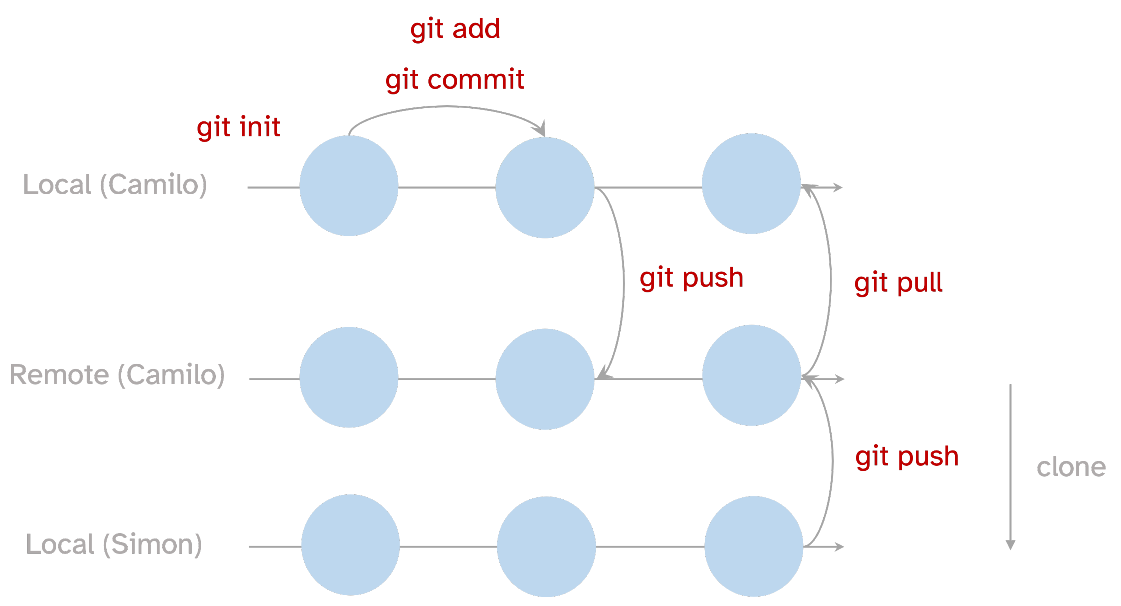


Fig 2.3: The cloning workflow in brief

Tip

If you want to keep a file out of synchronization git provides a simple way to do so by creating a .gitignorefile having the the paths to the files to keep in your local machine

2.3.1 Forking and collaborate

Basic collaboration on an open repository is a three-step process. First you need to forkthe repo, this will create a mirror copy of the repo in your GitHub profile. In a second step, a simple cloneof the repo will generate a copy of the forked repo on your local machine, so you can freely work and make your mistakes and push them to your forked repo that belongs to your account. In a third step, once they are on the remote repo you will have to create a pull request(PR), as its name suggest : you are **asking** the owners of the original to consider your changes Fig. 2.4.

2.3.2 Branching and merging

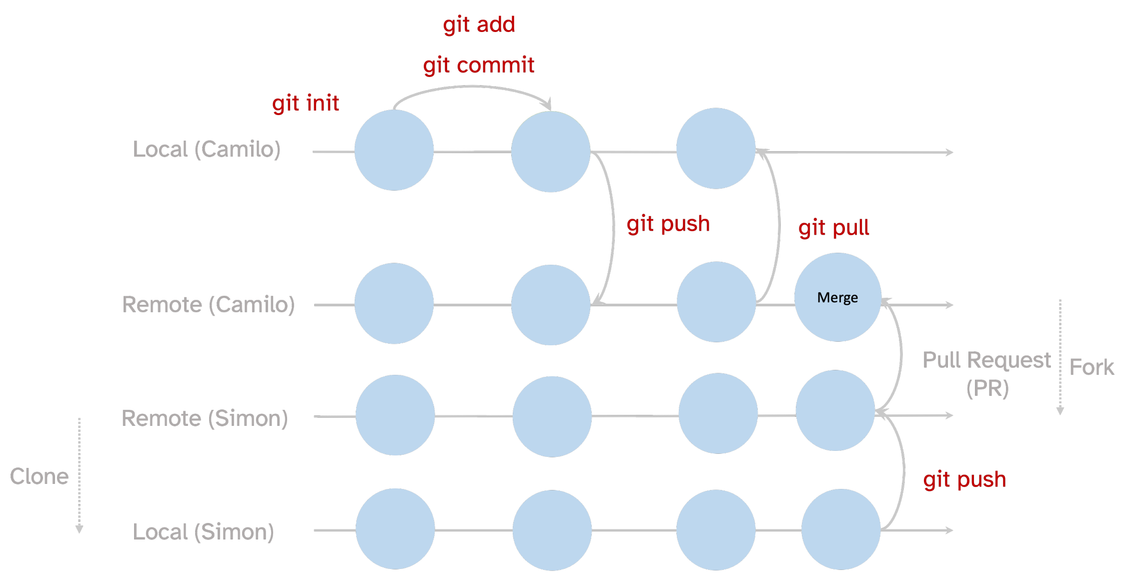


Fig 2.4: A common collaborative workflow from GitHub, using fork, clone and pull requests

II

Predicate

3 Algorithm thinking

III Modal

4 Sequence analysis

In this chapter we we will consider several biological concepts that appear central to understand the manipulation of biological data.

We are also covering the transition of a biological (organic) information to digital bits.

We will explore some databases in which biological information is stored.

4.1 Biological information

Since the origin, organisms (or molecules) have been the result of different selective processes. An emergent property of successive iterations of survival/decease was the ability of molecules to keep a record of its past in a very stable manner, so that it will pass generation to generation. Although this might not be the first property or molecule to ever exist (see the :origin of life), the innovation of organisms to pack information of previous events became so advantageous that almost any form of living organism has this property : that is DNA.

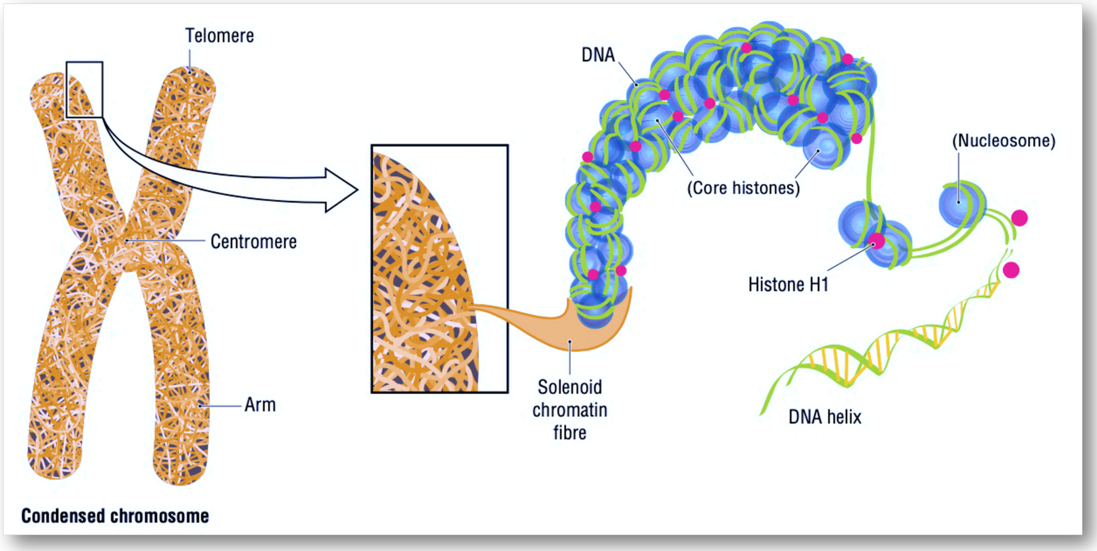


Fig 4.1: The different scales of information packed in a biological fashion. Figure adapted from Meneely et al. ([**ref-meneely2017**]) book showing how *DNA in a eukaryote is packaged with chromosomal proteins to form the much more compact chromatin. Chromatin is then folded into higher-order structures and eventually into the chromosome*

Almost any living organism harbor DNA in a nucleus or chromosome, which is a very compacted structure that could be actually seen to naked eye in some species. It so arranged or condensed Fig. 4.1 that it is the result of multiple compacting strategies.

4.2 The duality of DNA

DNA is an organic molecule of :nucleic acids that is mainly build out of nitrogen-containing compounds (:nucleobases) or just bases.The four bases are adenine (A), cytosine (C), guanine (G) and (T) and they bind strictly as $A = T$ and $C \equiv G$ where each of the lines represents an hydrogen bond.

DNA nitrogenous bases of are arranged in a very compacted helix structure and too much can be understood from its molecular nature. It is of course governed by the physicochemical properties of the atoms, therefore DNA is a physicochemical entity. But those common bricks have also a very beautiful emergent property that comes out from their order (mainly), and that is to keep the instructions that build the organism from which it belongs. These instructions are nothing else than information, which also follow some informational rules. Then DNA is also an informational entity.

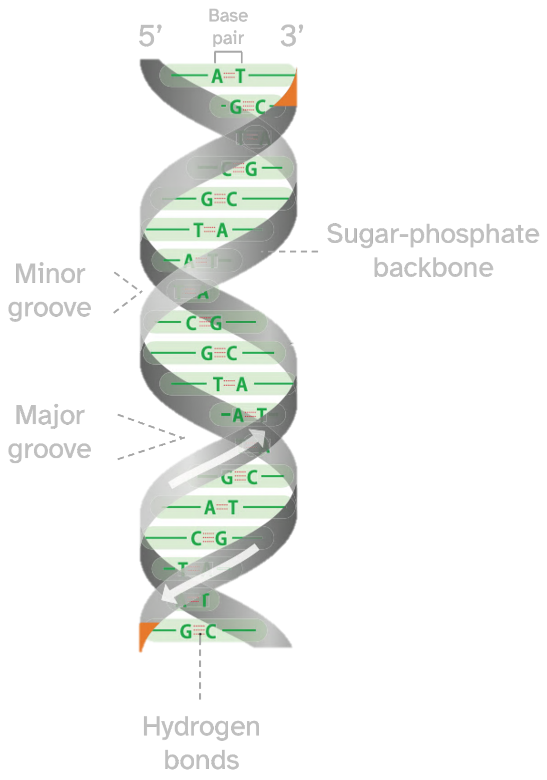
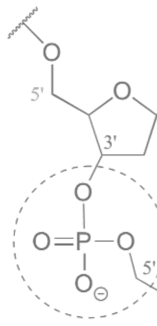
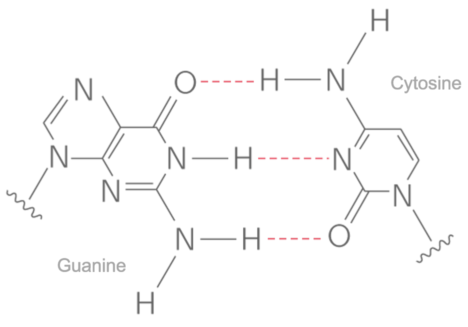


Fig 4.2: DNA double helix structure and main characteristics. Image adapted from Brown ([**ref-brown2018**]) showing the structure with the *sugar–phosphate backbone of each polynucleotide drawn as a gray ribbon with the base pairs in green*

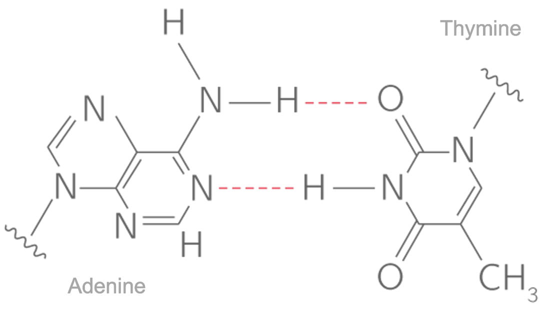


The description of DNA bases helped the elucidation of the helicoidal structure as well as the elucidation of the base pairing rules of the nucleotides Fig. 4.4, also called the Chargaff's rules of base pairing. The first outstanding feature of the DNA structure was indeed highlighted by Watson and Crick in its famous paper about a subtle mechanism of replication. The

This idea of the duality of the DNA (and of course of other biological molecules as well) is the source of the study of many bioinformatic fields. But a question remains open : how do we capture the informational nature of a DNA sequence so that we end up with a sequential file of characters ATCGCTATC.... This is not a trivial question in fact.



(a) Guanine and cytosine base pair



(b) Adenine and thymine base pair

Fig 4.4: Most common nucleotides present in DNA and the chemical interactions that bond each pair.

4.3 The central ~~dogma~~ theory of molecular biology extended

So far we have addressed that DNA is a very stable molecule that stores biological (i.e., evolutionary) information. Also, that DNA could represent a sequential object of characters as in a computer digital object or file as well. But how is it that this order has an underlying biological sense ? This was a very though question that required the accumulation of many experimental discoveries and the meeting of genetics and molecular biology.

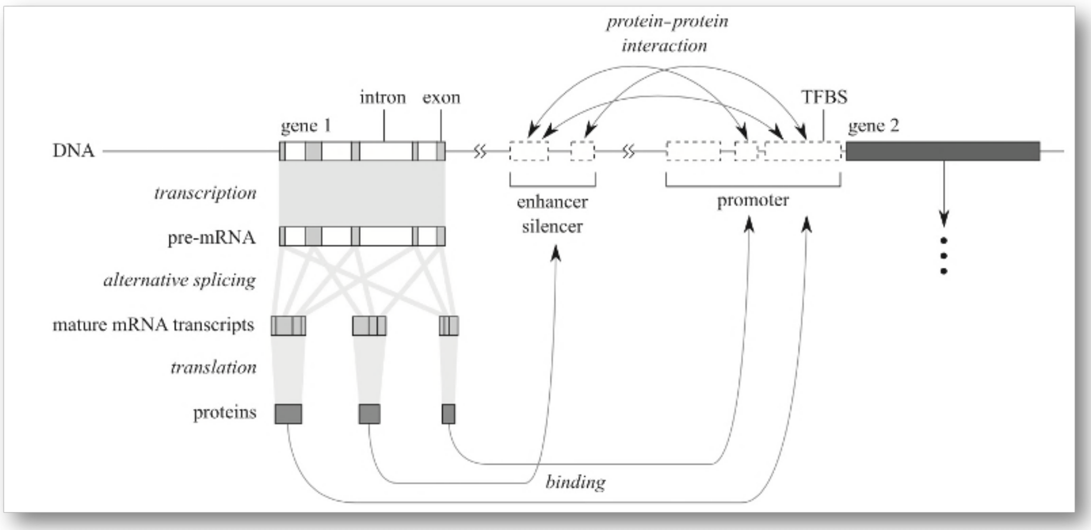


Fig 4.5: An extended representation of the central theory of molecular biology. Image from Mäkinen et al. ([**ref-makinen2015**])

4.4 Sequencing strategies

4.5 Sequencing over time

4.6 Some insights from sequencing genomes

4.7 Biological information databases

4.8 Retrieving data from NCBI

Inforation or databases inside the NCBI comes in many different ways. Perhaps the most used file format in bioinformatics is the FASTAformat. It is a very simple format that consist of two lines, the header which is basically noted by starting with the >symbol and is always preceding the sequence, that is present in the second line.

FASTA

```
>sequence
ATCCCTAGCTAGCTCGCTATCGATCATCCCTAGCTAGCTCGCTATCGATCATCCCTAGCTAGCTCG...
```

4.8.1 The manual way

4.8.2 Entrez direct

We first need to install the Entrez Direct command line utilities. This could be done in several ways, one suggested by the manual and a very simple one using conda (mamba).

```
mamba create -y -n entrez entrez
```

This line will create a new conda environment called ***entrez*** and will automatically install all dependencies. We later activate the environment with conda activate entrezand then we got the entire Entrez CLI utilities.

There are several commands and its function is to interact with the NCBI databases system. It is, however, a very huge set of utilities and commands to cope with at first, so the learning curve is somehow steppe. Here is a way to download the ***Bacillus tequilensis*** EA-CB0015 genome with a line of code that interleave three of the main commands of Entrez :

```
esearch -db assembly -query GCF_012225885.1 |
elink   -target nucleotide -name assembly_nuccore_refseq |
efetch  -format fasta > GCF_012225885.1.fasta
```

4.8.3 Simpler download programs

Since the manual way is clumsy when scaling and automating, and the entrez-directutilities are somehow hard at first hand, some bioinformaticians have already thought ways to ease users the hard of knowing the NCBI internal database structure and to resume many of the utlities to just a 'download' command out from what in entrez is distributed in search, filter, and download commands.

A very nice command to download genomes is the bit-dl-ncbi-assemblies from Mike Lee, who has also made a remarkable work to bring many bioiformatic knowledge to newcomers. Similarly and quite more robust is the ncbi-genome-download CLI (or abbreviated as ngd) developed by Kai Blin

e NCBI databases
eing curated and
by the execution
tools. The refseq
ly the one that is
case of genomes
ber of those that
ategory are start
as the annotated
enbank category
play the **GCA** ids

Using ngdto download a complete genome using the refeseq or genbank accession numbers, would be as easy as :

```
ngd --format fasta --flat-output -A GCF_002055965.1 bacteria
```

The one-line will download the refeseq genome of the ***Bacillus tequilenis*** EA-CB0015 in a fasta nucleotide format. the --flat-optionfrom the script will handle out extra metadata of the search and will download strictly the data.

It is important to highlight that these simpler scripts are focused on genomic information, so that direct downloads of genes, proteins, transcripts, and many other kind of data must be done with entrezor other programs. Some recent efforts have been made to create a more robust CLI to interact with different databases, but in a very easy and intuitive way, such as the gget project. It is more focused on vertebrate databases, but also has a port to NCBI, UniProt and other databases. And is very simple. Here is an example to download all the isoforms in a multi-fasta format of the cancer-related gene BRCA2 in a human genome (GRCh38.p13) :

```
gget seq -iso -o BRCA2.fasta ENSG00000139618
```

It is establishing a connection with the Ensemble database, which is focused on vertebrate genome and transcriptome data.

Challenge

Professor Camilo is interested on knowing how many complete genomes of ***Bacillus subtilis*** are there in the NCBI databases. He asks you later to count the number of features (genes, CDS, ncRNA, rRNA, etc.) in the genome of ***Bacillus subtilis*** NCIB 3610 (GCF_002055965.1). And tell you to document each of the steps and how did you end up with the answer. Saving the file with your initials (e.g., CG-activity01.md)

IV Intuitionistic

5 Phylogenetics

5.1 What is a phylogenetic tree

5.2 Mehtods for phylogenetic reconstruction

5.3 Building a phylogenetic reconstruction

5.3.1 Evolutionary substitution model

5.3.2 Maximum likelihood

5.3.3 Bayesian inference

V

Meta

6 Genome assembly

6.1 The problem of assembling genomes

6.2 Main algorithms for genome asssembly

6.2.1 Overlay, Layout, Consensus (OLS)

6.2.2 De Bruijn graphs

6.3 Main concepts of an assembly

6.3.1 Contigs, Unitigs, Scaffolds

6.4 A complete workflow for assembling genomes

6.5 Assessing genomes

6.5.1 Inspecting genome graphs

6.5.2 Genome completeness

6.6 Understanding genome difficulties

- End of chromosomes
- Erros
- Lack of coverage
- Heterozigozity
- repeats

VI Application

7 Metagenomic sequencing

an experimental design

Metagenomes is the study of whole DNA of a biological communities.

Metagenomic of a single gene (16S, ITS, etc.) is also called metataxonomic (amplicon sequencing). In contrast all the genes/ genomes in the sample is actually metagenomics.

Some differences :

Metagenomic | Metataxonomic expensive | cheap more samples| less specific regions | random regions

Torsten Seeman, er - A guide from ing data analysis

7.1 Designing the experiment

Stablish the basic experimental unit and the appropriate replicates, control, randomization !

It is important to have metadata of the experiment in a plant microbial community is common to have pH, geo-reference, temperature, etc.

In metataxonomic we analyses hypervariable regions (V3-V4) and compare relative abundance to identify which is the most informative (abundance, richness)

7.2 DNA extraction and Sequencing (Illumina)

Followed common sequencing is also important to make quality check of the data

7.3 Denoising

There are several way to define OTUs (97-99 % of similarity) for instance :

- It is defined operational unit of species or species groups (be careful to assign a sequence as an species)
- Taxonomic level of sampling selected in the study⋯ individual, population, species,⋯

Pipelines for OTUs or ASVs are Quimme and Silva and deNBI

7.4 Chimera

Reads that result from combinations in the sequencing

7.5 Taxonomic annotation

Using Silva or deNBI

VII

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

References

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.

Brown, T. A. (2018). ***Genomes 4***. Garland science.

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.

Mäkinen, V., Belazzougui, D., Cunial, F., & Tomescu, A. I. (2015). ***Genome-scale algorithm design***. Cambridge University Press.

Meneely, P. M., Hoang, R. D., Okeke, I. N., & Heston, K. (2017). ***Genetics: Genes, genomes, and evolution***. Oxford University Press.

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