Why standards? Good practices in computational biology

Izaskun Mallona

COST Project Epichembio - Introduction to NGS data analysis

13th March 2019

Acknowledgements

- EU Horizon 2020 COST Project Epichembio
- IJC Carreras Foundation
- Organizers: Sarah, Marguerite-Marie, David, Roberto
- SIB Swiss Institute of Bioinformatics and Univ. Zurich

Notes

- Talk typesetting
  - Commands/options are in `typewriter font`
  - URLs are highlighted in blue
- Exercises
  - Available at the course GitHub repo
  - Please use ad libitum (caution: there are 38 of them, exceeding the workshop workload)

Exercise: Web browsing the genome

- Launch the UCSC Genome Browser
- Specify `Human Assembly hg19`
- Click `go`

By default, the Genome Browser will render a genomic window with many data layers on it. How are these data encoded?

- Click `UCSC Genes` from the `Genes and Gene Predictions` section under the main genomic window.
- Click `View table schema` opens knownGene table schema

knownGene table schema

Schema for UCSC Genes - UCSC Genes (RefSeq, GenBank, CCDS, Rfam, tRNAs & Comparative Genomics)

Database: hg19 **Primary Table:** knownGene **Row Count:** 82,960 **Data last updated:** 2013-06-14
Format description: Transcript from default gene set in UCSC browser

field	example	SQL type	info	description
name	uc001aaa.3	varchar(255)	values	Name of gene
chrom	chr1	varchar(255)	values	Reference sequence chromosome or scaffold
strand	+	char(1)	values	+ or - for strand
txStart	11873	int(10) unsigned	range	Transcription start position (or end position for minus strand item)
txEnd	14409	int(10) unsigned	range	Transcription end position (or start position for minus strand item)
cdsStart	11873	int(10) unsigned	range	Coding region start (or end position if for minus strand item)
cdsEnd	11873	int(10) unsigned	range	Coding region end (or start position if for minus strand item)
exonCount	3	int(10) unsigned	range	Number of exons
exonStarts	11873,12612,13220,	longblob		Exon start positions (or end positions for minus strand item)
exonEnds	12227,12721,14409,	longblob		Exon end positions (or start positions for minus strand item)
proteinID		varchar(40)	values	UniProt display ID, UniProt accession, or RefSeq protein ID
alignID	uc001aaa.3	varchar(255)	values	Unique identifier (GENCODE transcript ID for GENCODE Basic)

knownGene table schema

So they are database entries with **chrom**, **start** and **end** features. This is the most standard data representation in genomics: data refering to genomic coordinates. Why?

Discussion

- Which would be the most efficient file format to store data related to human genomes?

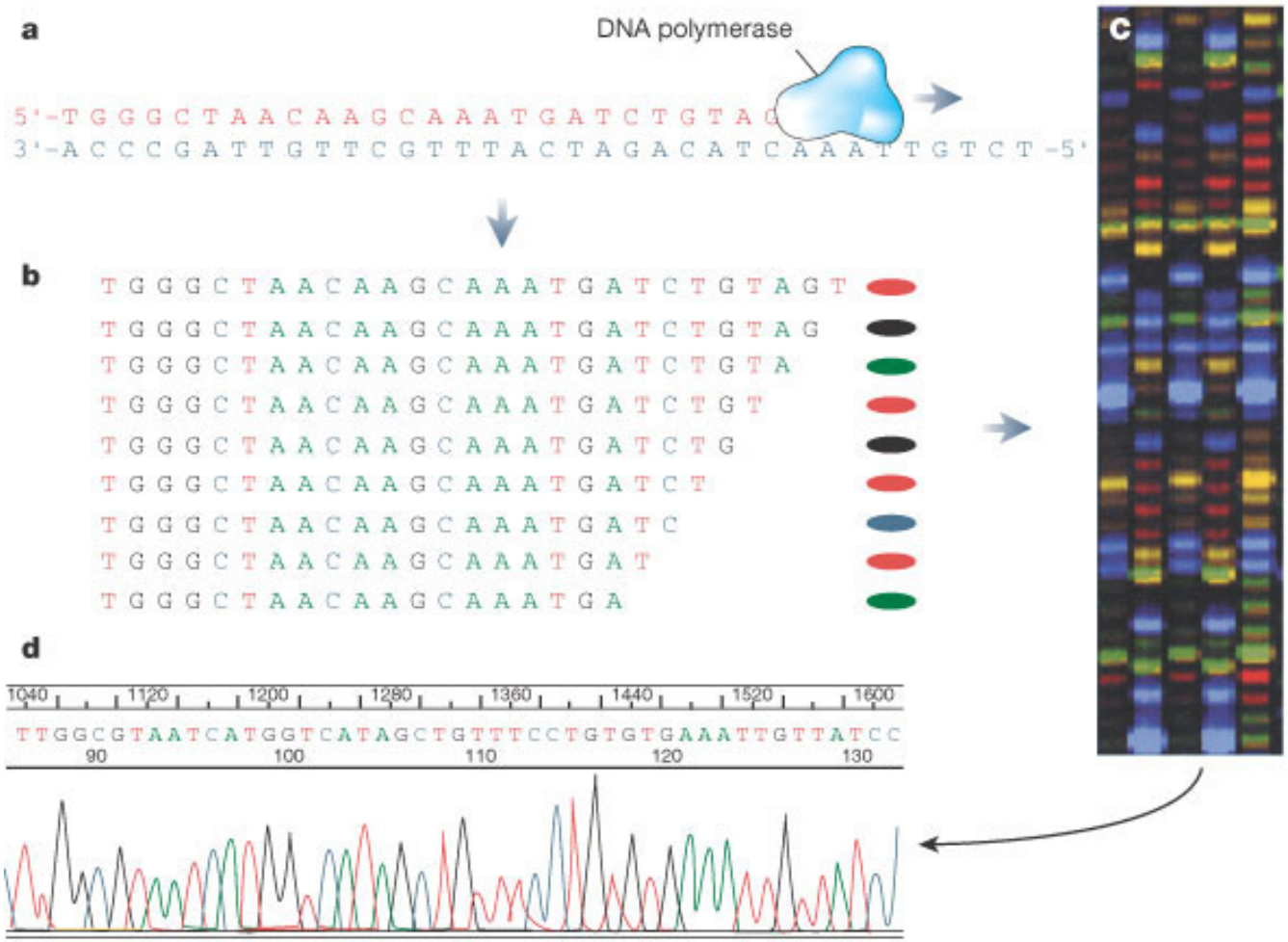
Commonly used formats

- Reference genomes
- Fasta and FastQ (Unaligned sequences)
- SAM/BAM (Alignments)
- BED (Genomic ranges)
- GFF/GTF (Gene annotation)
- BEDgraphs (Genomic ranges)
- Wiggle files, BEDgraphs and BigWigs (Genomic scores).
- Indexed BEDgraphs/Wiggles
- VCFs (variants)

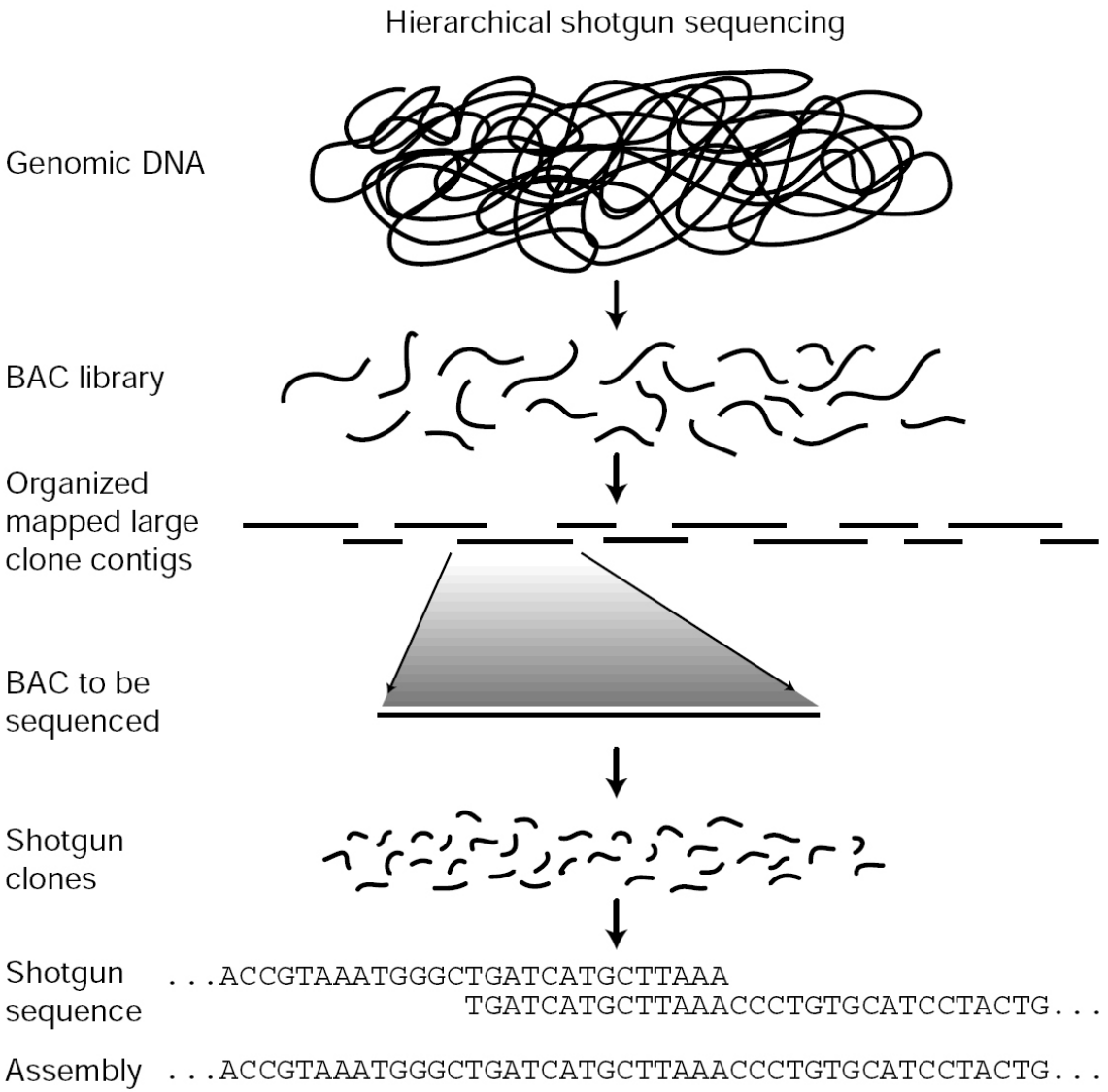
Reference genomes

- Reference genomes describe the 'consensus' DNA sequence
- (The human genome from whom? What does consensus mean?)
- Human variation aside, multiple assemblies have been released (i.e. hg18, hg19...)

Sanger sequencing Nature 409, 863 (2001)



Hierarchical shotgun Nature 409, 863 (2001)



Reference genomes

GRCh stands for 'Genome Reference Consortium'

- Human GRCh37 (hg19)
- Human GRCh38
- Mouse mm10
- Mouse GRCm38
- Zebrafish, chicken and others: https://www.ncbi.nlm.nih.gov/grcThe Genome Reference consortium

Activity: sequence retrieval

- Retrieve the sequence of the human chromosome 7...
- ... and make it traceable, replicable and reproducible

Automation

- Using a Web browser to retrieve genomic sequences is not efficient nor reproducible: programmatic alternatives exist
- Need of standardizing data analysis using reproducible workflows
  - Scripts for data retrieval (bash, R, python...)
  - Keeping track of data analysis steps and avoiding manual editing
  - Data storage: standards (fasta, fastq, sam, vcf...)

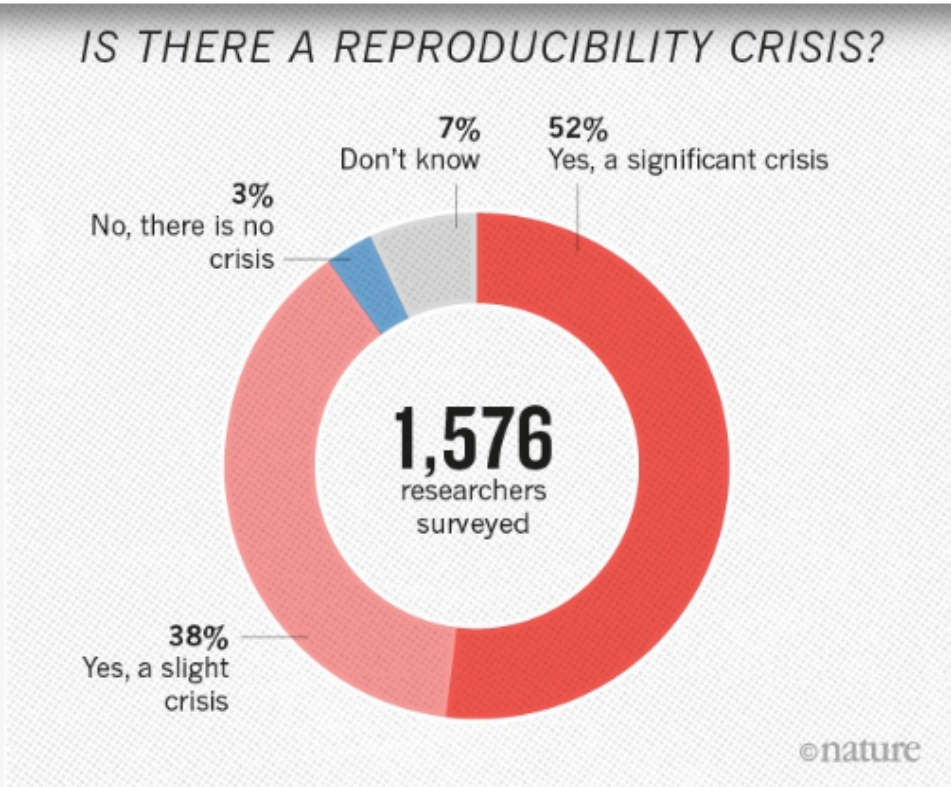
Reproducibility

- What do we mean by data science reproducibility? and replicability? and repeatability?
- In data science: avoid manual steps of data analysis using scripts plus version control systems
- Magics, blackboxes, untraceable stuff include:
  - Spreadsheet manual editing
  - Find-and-replace using a text editor
  - Antyhing that involves mouse clicks without any log/macro

Is there a reproducibility crisis?

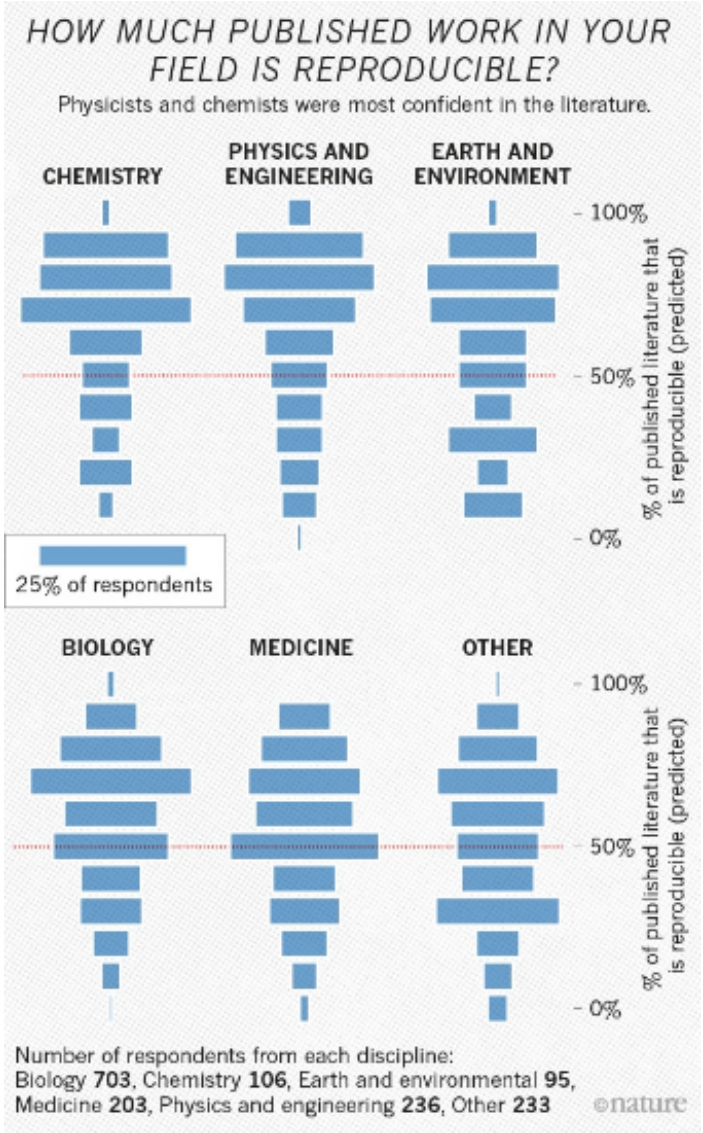
NATURE | NEWS FEATURE

1,500 scientists lift the lid on reproducibility



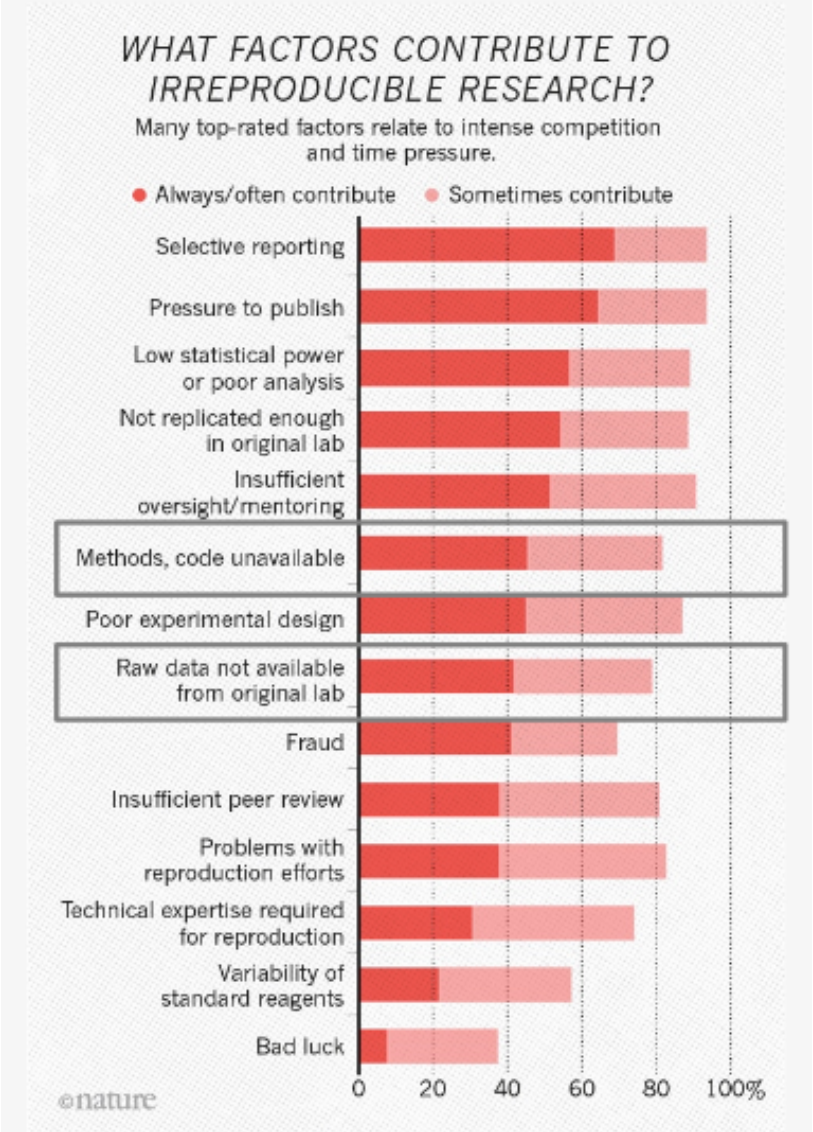
Baker M (2016) Is there a reproducibility crisis? Nature 533:452–454 9

Is there a reproducibility crisis?



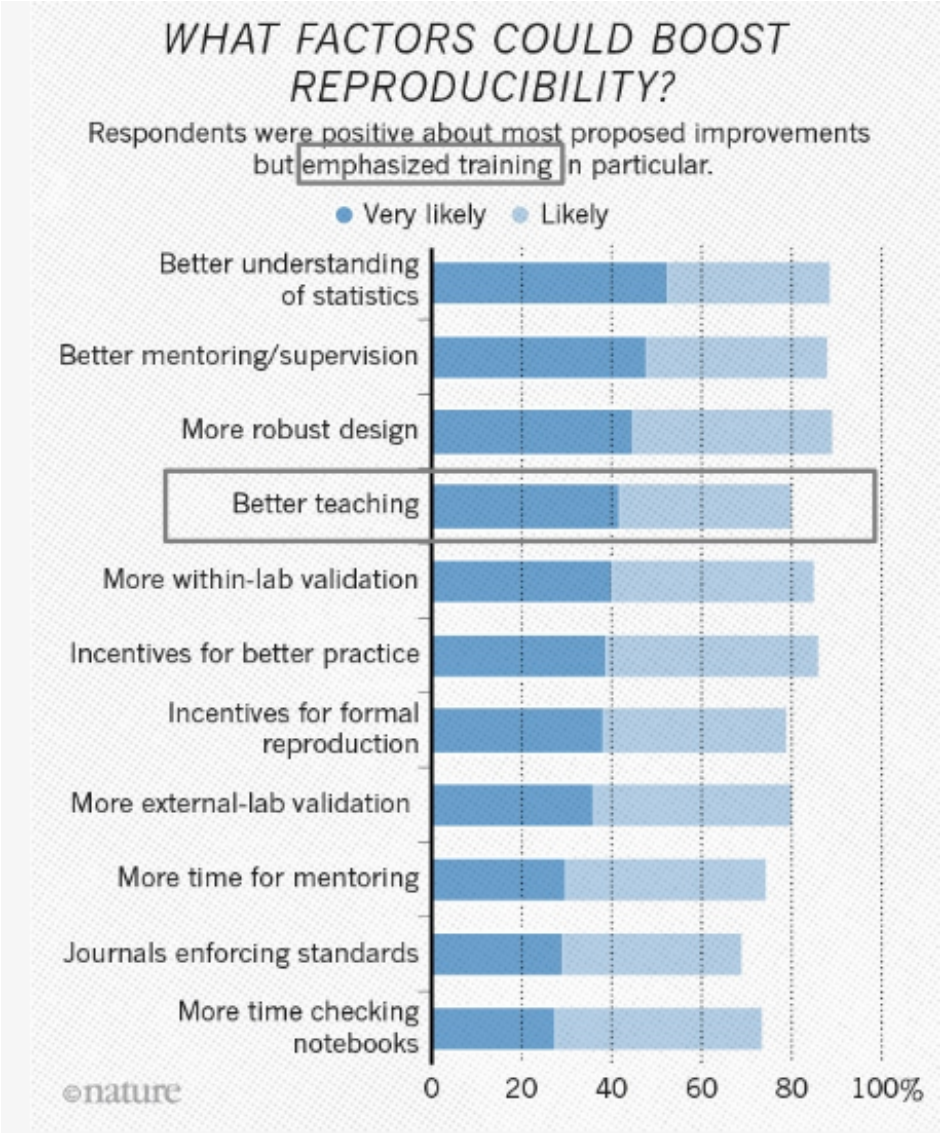
Baker M (2016) Is there a reproducibility crisis? Nature 533:452–454 9

The causes



Baker M (2016) Is there a reproducibility crisis? Nature 533:452–454 9

The alternatives



Baker M (2016) Is there a reproducibility crisis? Nature 533:452–454 9

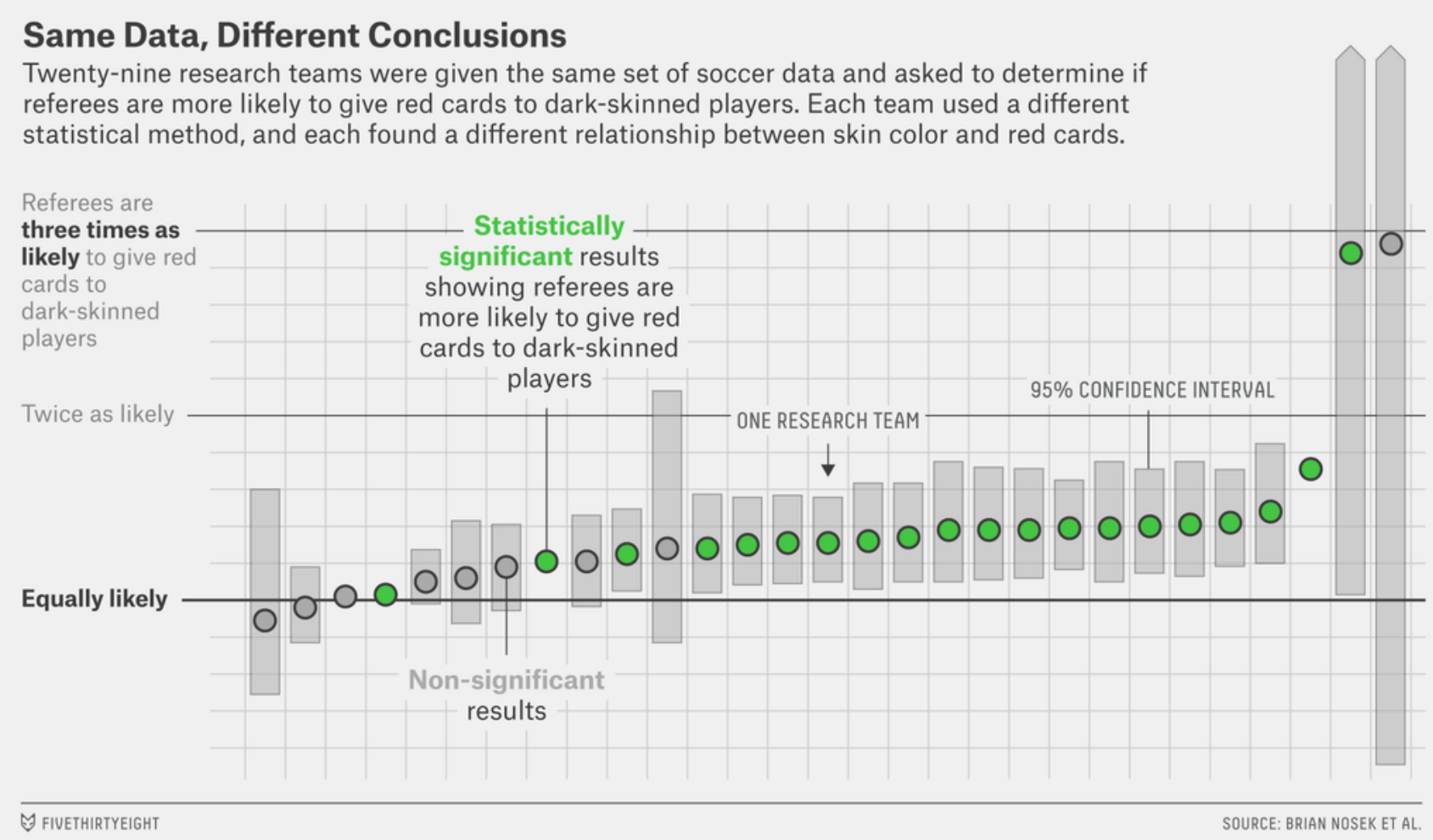
Good practices

- Data
  1. Using data standards
  2. Raw data availability
  3. Metadata
  4. Intermediate datasets availability (mid-processed, i.e. BED files)
- Analysis
  1. Scripting everything
  2. Version control
  3. Trace software versions/automate installs
  4. Release all code as supplementary information

Good practices

- What if we don't know how to program?
  - Still, switching to command-line tools and keeping track of the commands used
  - Request the source code when collaborating with computational biologists

Cautionary note: science is science



https://fivethirtyeight.com/features/science-isnt-broken

The terminal

- Simple command line interface
- Present in MacOS and GNU/Linux
- Interprets the Unix shell language (commonly bash)
- Even though can be used in an interactive manner, commands can be written and stored as a script (=workflow, =pipeline)
- (A bash script is, probably, the simplest way of making a workflow repeatable)

UNIX

- Efficient
- Scalable
- Portable
- Open

Unix philosophy

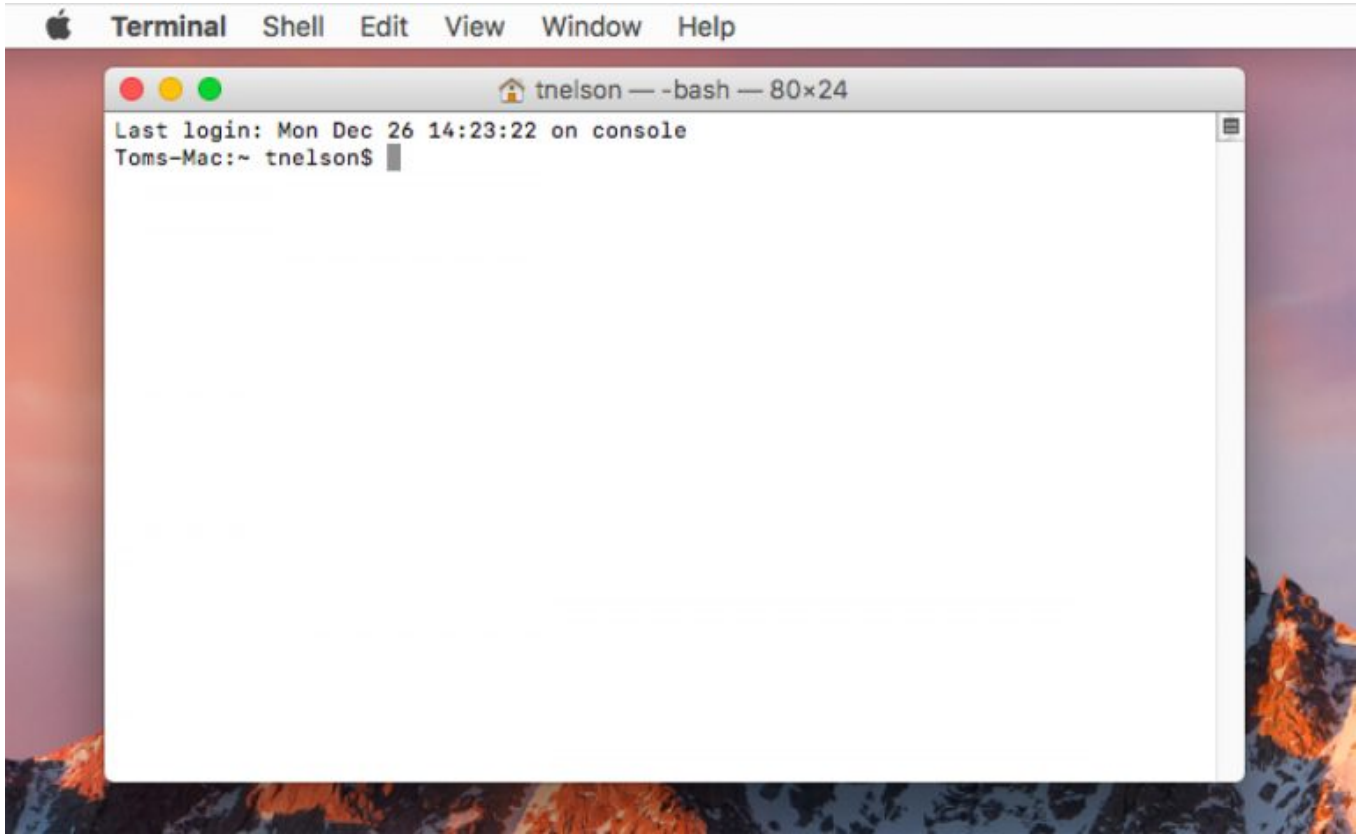
According to Peter H. Salus in A Quarter-Century of Unix (1994):

- Write programs that do one thing and do it well
- Write programs to work together
- Write programs to handle text streams, because that is a universal interface

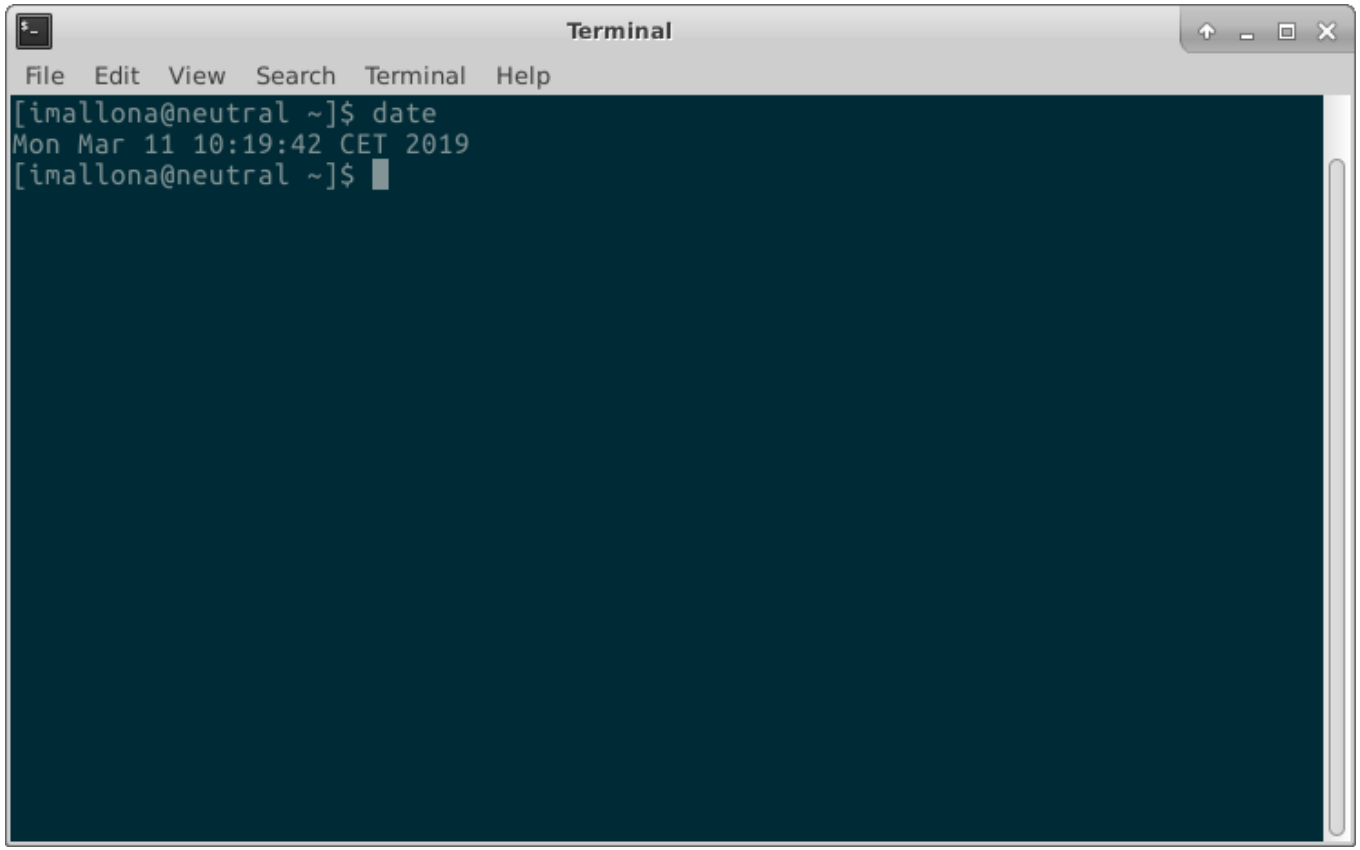
Why in bioinformatics?

- We interpret DNA, proteins as text; Unix is for text streams.
- Data are big (millions of lines of text, easily a couple of GB); spreadsheet software (Excel) cannot handle them.
- We need to keep track of our analysis for the sake of reproducibility: bash scripts.

Opening a terminal in MacOS



Opening a terminal in GNU/Linux



A quick reminder on computer files

- Files are data representations stored in computers as arrays of bytes
- File type is defined by its bytes and not by the filename extension
- Files contain metadata
- Importantly, plain text files are composed by bytes mapped directly to ASCII characters
- Text editors (notepad, gedit, vim...) allow editing plain text files
- (text files can be read without proprietary software)

Questions

- Sequences are often stored as FASTA, i.e.

  ```
  >hg_19_chr7_short_version
  tatatata
  ```

- How to save this to a file to be a fasta?
  1. Edit it with a text editor and save it as `test.fasta`
  2. Edit it with a text editor and save it as `test.fa`
  3. Edit it with a text editor and save it as `test.png`
  4. Edit it with a LibreOffice Writer and save it as an ODT file named `test.odt`
  5. Edit it with a LibreOffice Writer and save it as an ODT file named `test.fasta`
  6. Edit it with a R and `save()` it as an Rdata object `test.Rdata`

Questions

- Sequences are often stored as FASTA, i.e.

  ```
  >test
  acgt
  ```

- How to save this to a file to be a fasta?

YES Edit it with a text editor and save it as `test.fasta`

YES Edit it with a text editor and save it as `test.fa`

YES WTF Edit it with a text editor and save it as `test.png`

NO Edit it with a LibreOffice Writer and save it as an ODT file named `test.odt`

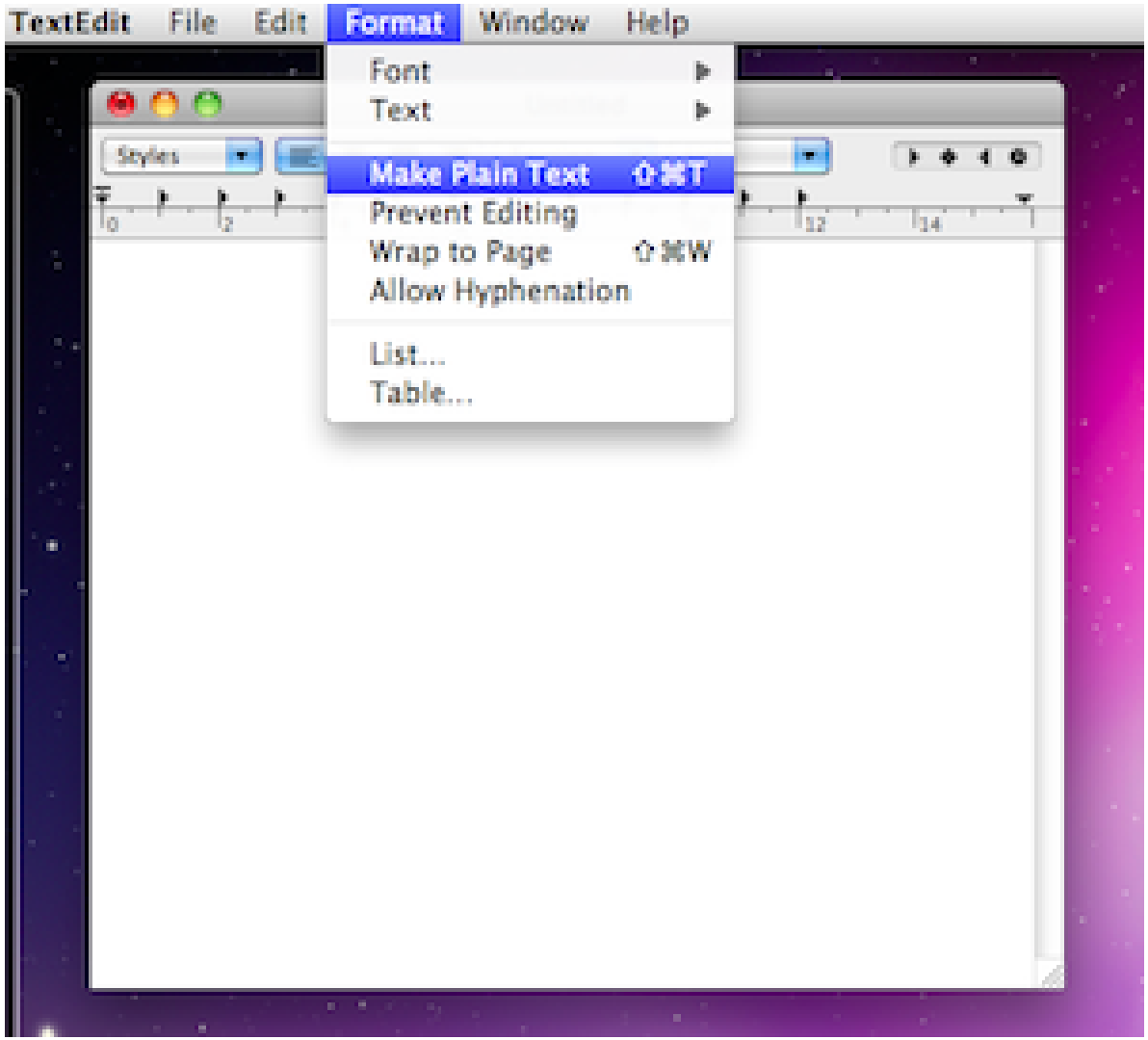
NO Edit it with a LibreOffice Writer and save it as an ODT file named `test.fasta`

NO Edit it with a R and `save()` it as an Rdata object `test.Rdata`

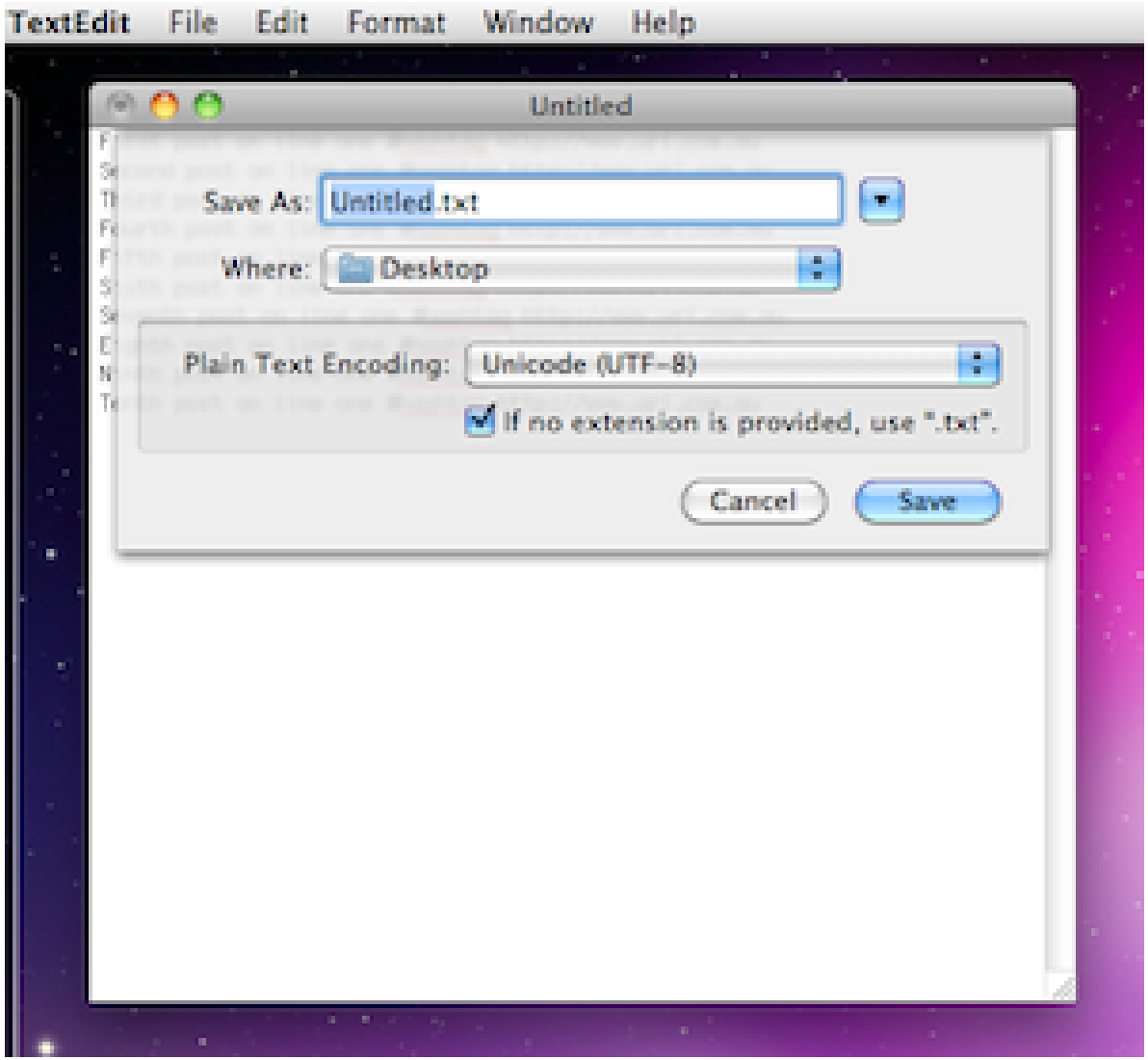
Setting up the Mac text editor to save text

- Create new file
- Go to `Format` and select `Make Plain Text`
- For saving, go to `File`, `Save As` and `Plain Text Encoding` setting: `Unicode (UTF-8)`.

Avoiding RTF in Mac: plain text



Avoiding RTF in Mac: plain text



Shell scripting

- Write on top the shebang
- Write the date and what's the script about, your name and date
- Tip: comment lines start with #
- Introduce the commands (one line each)
- Save it as a text file!
- Commit the changes/backup the file
- Run the script to run the commands in batch typing `bash name_of_the_script.sh`

Reproducibility for software

- UNIX solves the reproducibility, scalability and openness for data (text) streams, but extra software might be needed
- The importance of software versioning for reproducibility: keeping track of the software installed
- Using open source software (no blackboxes!)
- Installs can be run command-line, so specific versions can be stored and included into the analysis script

Compiling software - bedtools

(The code is available at the exercises file) Please note this is a bash script and specifies the exact software version to install

```
#!/bin/bash

cd # to your home directory
mkdir -p soft # creates a folder
cd soft # goes there

# the url is chopped into two pieces for readability
url_base=https://github.com/arq5x/bedtools2/releases/download/
curl -L "$url_base"/v2.25.0/bedtools-2.25.0.tar.gz \
  > bedtools-2.25.0.tar.gz

tar zxvf bedtools-2.25.0.tar.gz
cd bedtools2
make
```

Exercises

- Run the exercises till number 4

Next...

- Fasta (Reference genomes)
- (Multi)fasta and FastQ (Unaligned sequences)
- SAM/BAM (Alignments)
- BED (Genomic ranges)
- GFF/GTF (Gene annotation)
- Wiggle files, BEDgraphs and BigWigs (Genomic scores).
- Indexed BEDgraphs/Wiggles
- VCFs (variants)