BIO392 file formats and tools

Izaskun Mallona

Talk typesetting

- Commands/options are in `typewriter font`
- URLs are highlighted in blue

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?

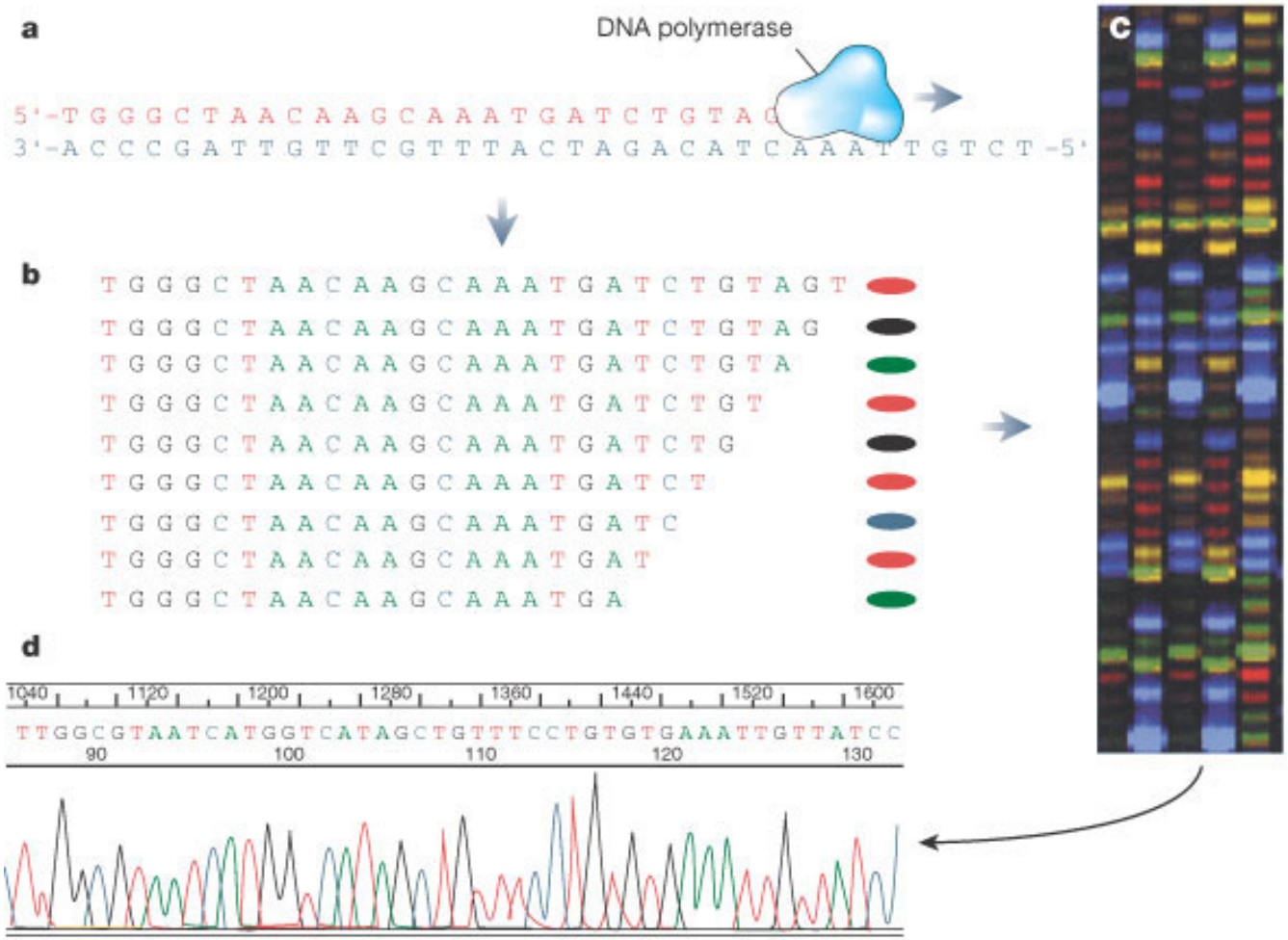
Commonly used formats

- 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)

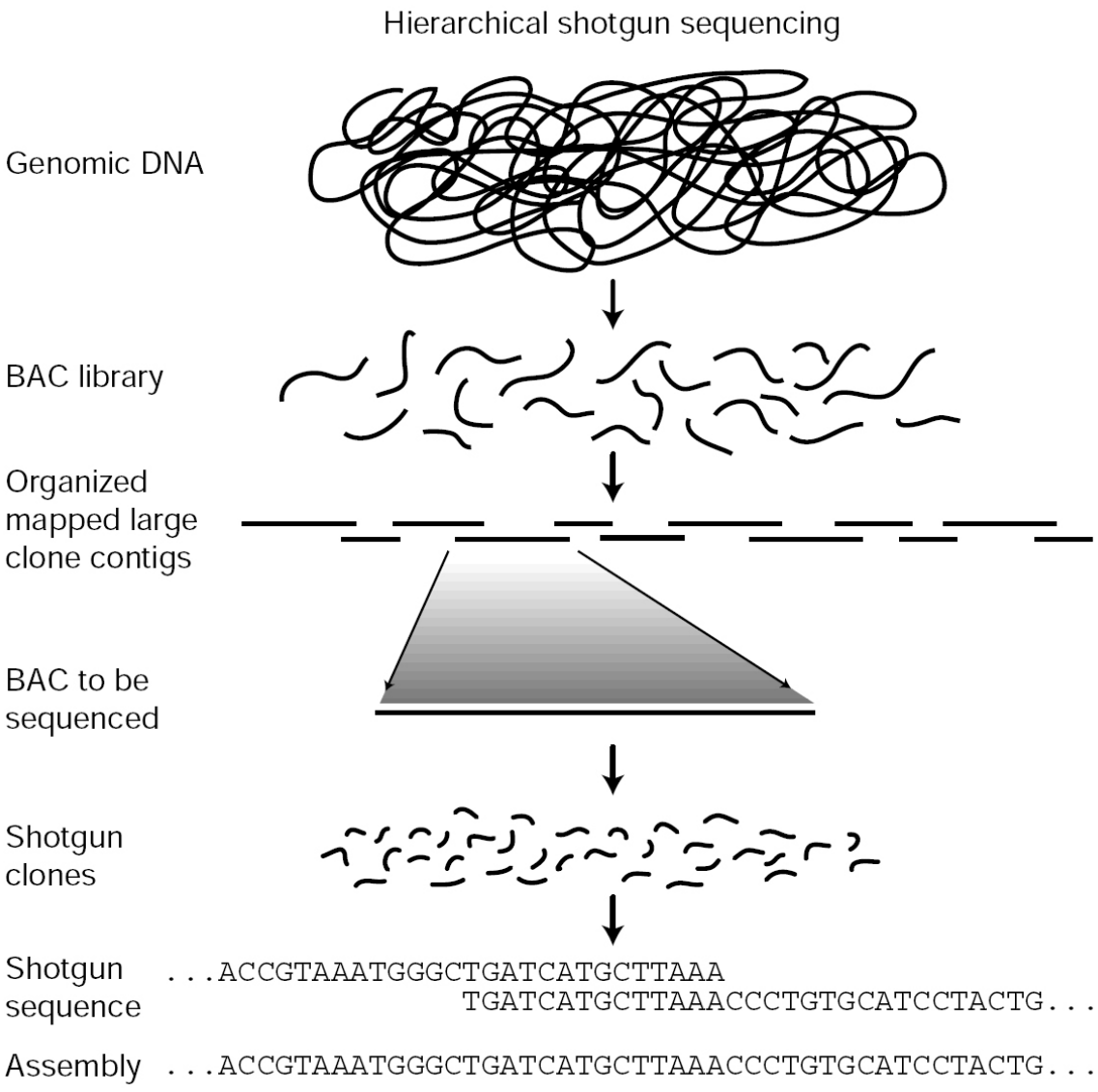
Reference genomes

- 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 mitochondrial genome
- (Tip: where (databases)? how? which format? does the human mitochondrial genome exist?)

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 (in bioinformatics often R or 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?
- In data science: avoid manual steps of data analysis using scripts plus version control systems
- Spreadsheet editors used with sequences of mouse clicks are not reproducible

Reproducibility: reading

- Paper: https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970

How could we increase reproducibility?

- In data analysis: keeping track of all steps (using scripts)
- Using data standards
- What if we don't know how to program?
- Still, switching to command-line tools and keeping track of the commands used
- Using control version systems

The terminal

- Simple command line interface
- Present in MacOS and GNU/Linux computers
- Interprets the Unix shell language (commonly bash)

UNIX

- Efficient
- Scalable
- Portable
- Open

Unix philosophy

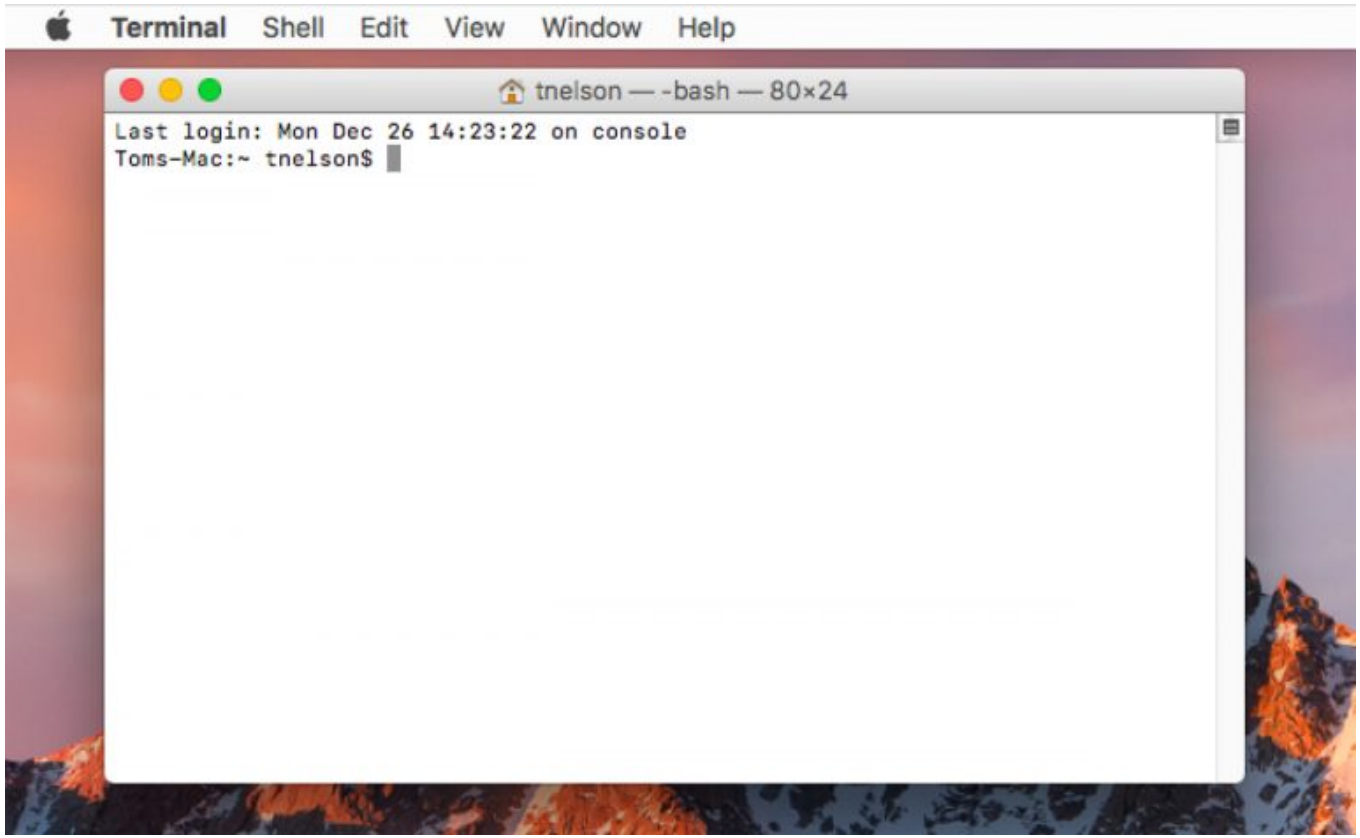
According to Peter H. Salus in A Quarter-Century of Unix (1994):

- 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



The shell (Unix shell)

- The Unix shell allows to save the sequential commands in a reproducible manner
- Activity for next day: run the tutorial by the Swiss Institute of Bioinformatics
  https://edu.sib.swiss/pluginfile.php/2878/mod_resource/content/4/couselab-html/content.html

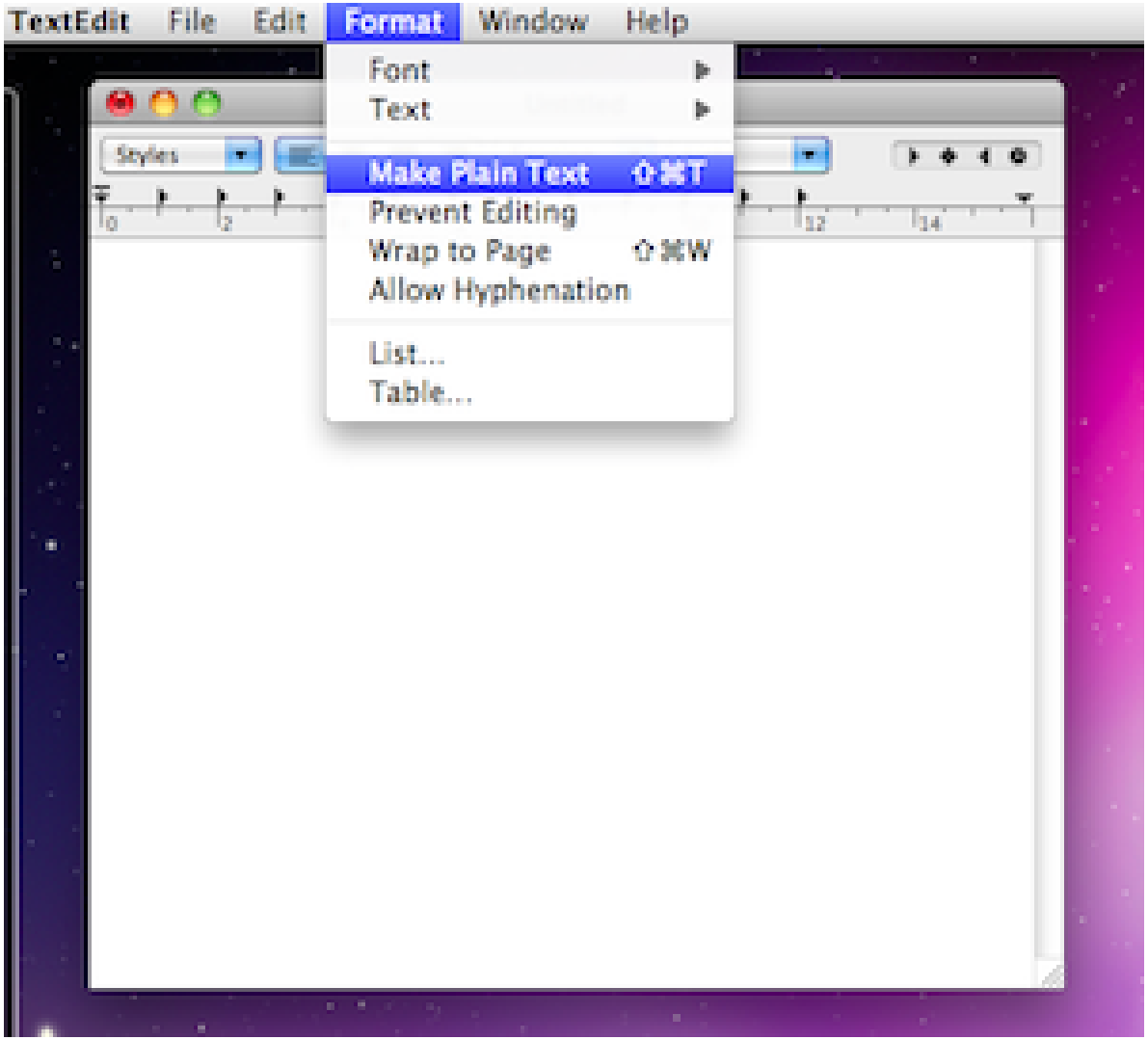
A quick reminder on computer files

- 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)

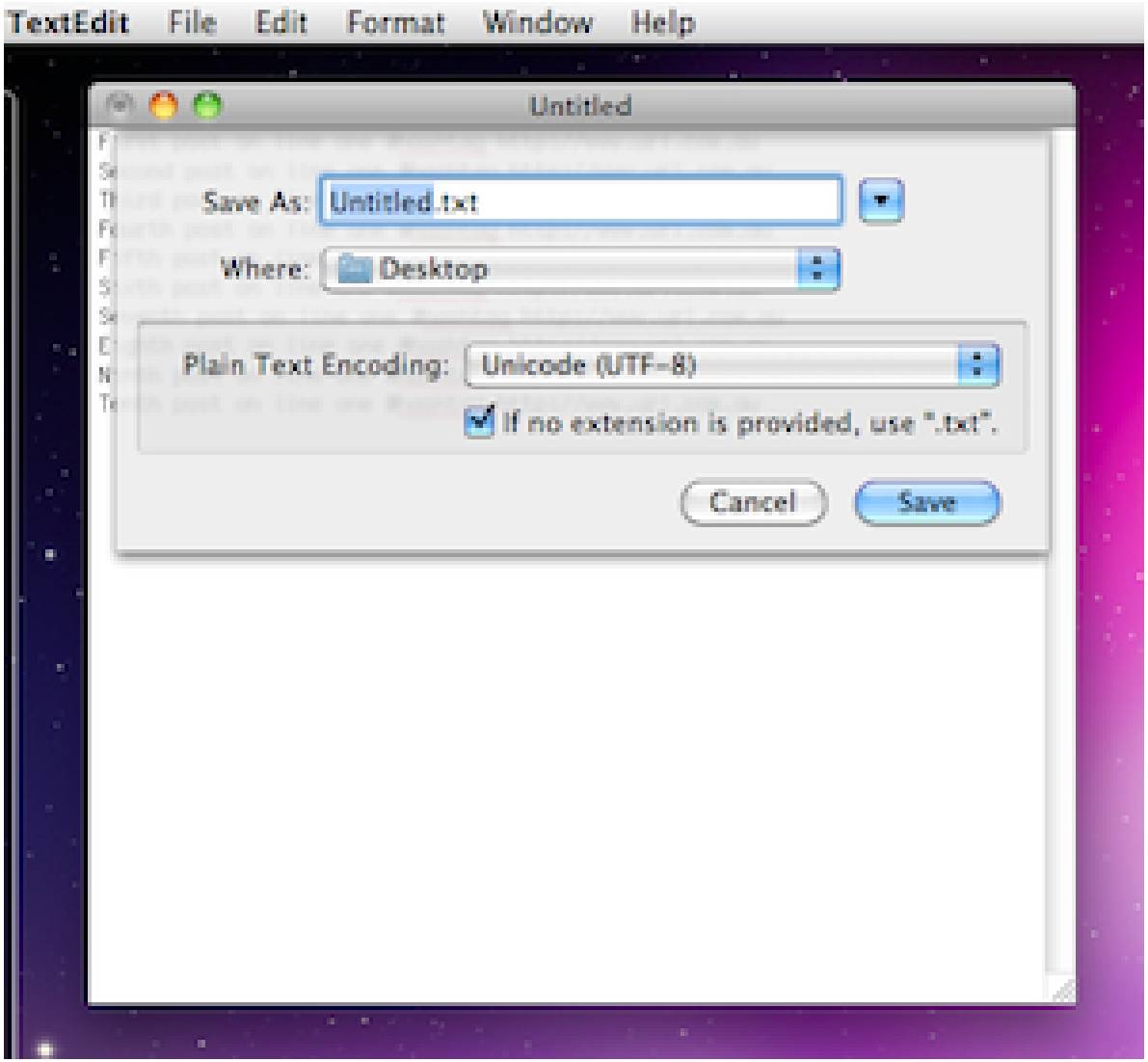
Setting up the Mac text editor to save text

- 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: plain text



Avoiding RTF: plain text



Turning off autospell check

- UNIX commands are case sensitive
- By default, TextEdit capitalizes/spell checks contents
- Exercise: disable this feature
- http://osxdaily.com/2014/05/06/turn-off-autocorrect-pages-textedit-mac/

Organizing a shell script

- Write on top the shebang
- Write the date and what's the script about, your name and date
- Tip: comment lines start with #
- Tip: multiline chunks are split by a \
- Introduce the commands (one line each)

Scripting

- Save the file (script) somewhere with a back-up system
- You could 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

Intro to the terminal

- Use the arrow keys to move back/forth
- Use the ‘tab‘ key to autocomplete
- Lines starting with # are comments - not evaluated
- Text can be written after the prompt (dollar sign **$**)

Basic Unix commands: **man**ual

```
man man # prints the manual for the manual command
man sed # commonly used editing program

# exit pressing ‘q‘
```

Basic Unix commands: **p**rint **w**orking **d**irectory

```
$ pwd
# prints my path, e.g. /home/guest
```

Basic Unix commands: **c**hange **d**irectory

```
$ cd .. # goes to parent
$ cd /tmp # goes to a folder named ‘/tmp‘
```

Basic Unix commands: **l**ist **d**irectory

```
$ ls # list files in a given path
$ ls -l # lists files and their attributes
```

Basic Unix commands: **mk**e **dir**ectory

```
$ mkdir newdir # creates the directory newdir
```

Basic Unix commands: **rm**ove **dir**ectory

```
# removes the directory newdir (created before)
$ rmdir newdir
```

Basic Unix commands: **co**p**y** file

```
$ cp source-file destination-file

# let's create a file using echo
#   and redirecting the echo to a file
$ echo 'lorem ipsum' > test_file

# cp test_file test_file_another
$ ls
```

Basic Unix commands: **less** printing files one screen at a time

```
$ less test_file_another

# exit pressing q
```

Basic Unix commands: check the **head** of a file

```
$ head test_file_another
```

Other common commands

```
cat - Printing Files Onto the Screen
mv - Moving and Renaming Files
rm - Removing Files and Directories
chmod - Changing Access Permissions
diff - Finding the Differences Between Two Files
grep - Searching for Strings in Files
wc - Counting Words
sed - Stream editing files
awk - A full language to process texts
```

Software installs - compiling bedtools (for next week)

- Some tools we use are not standard Unix programs, but domain-specific software
- BEDtools is one of the most used tools for handling coordinate-based file formats
- If interested, read paper https://academic.oup.com/bioinformatics/article/26/6/841/244688

Compiling software in Unix

- We will download the open source code of bedtools and then generate the executables
- That is **compiling** the source, and generates binaries fit to our hardware and OS
- Also, makes the same code runnable in many platforms
- That's transparent for us: the developer wrote a Makefile that automates this
- Extra reading: https://www.wired.com/2010/02/Compile_Software_From_Source_Code/

Compiling bedtools

(The code is available at the exercises file)

```
cd # to your home directory or wherever you decide
cd soft # the folder was created before (for kent utils)

curl -L https://github.com/arq5x/bedtools2/releases/download/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
alias bedtools='./bin/bedtools'
```

Activities

- (on site) Download some data/install bedtools (https://github.com/compbiozurich/UZH-BIO392/blob/imallona/course-material/2020/imallona/3_exercises.md, section **Set up**)
- (on site) Run exercises 1-4 (same url above)
- (online/afternoon) Run the introductory course on Unix at https://edu.sib.swiss/pluginfile.php/2878/mod_resource/content/4/couselab-html/content.html