Session 2: **Working environment**

Introduction to Linux for Bioinformatics

Tunis | September 26th, 2024

Overview of Linux for Bioinformatics

Linux: An open-source (free) operating system widely used in bioinformatics due to its stability, security, and powerful command-line interface.

Why Linux in Bioinformatics?

- **Tool availability:** Most bioinformatics tools are developed and optimized for Linux.
- **Command line tools and scripting:** Linux's command-line interface and scripting languages (e.g., Bash) allow for the automation of repetitive tasks, streamlining workflows.
- **Efficiency:** Handles large datasets and complex computations effectively.

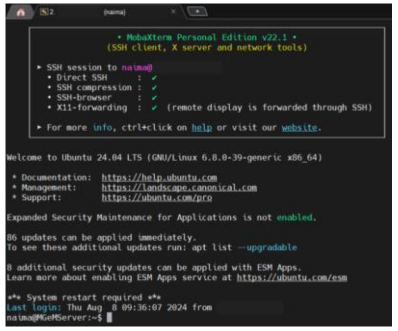
How to access a Linux-based system?

Local Desktop

VirtualBox

Remote connection: SSH Secure Shell protocol

MobaXterm


```
        • MobaXterm Personal Edition v22.1 •
       (SSH client, X server and network tools)

  ▸ SSH session to naima@[redacted]
    • Direct SSH      :  ✓
    • SSH compression :  ✓
    • SSH-browser     :  ✓
    • X11-forwarding  :  ✓  (remote display is forwarded through SSH)

  ▸ For more info, ctrl+click on help or visit our website.

Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

86 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

8 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: Thu Aug  8 09:36:07 2024 from [redacted]
naima@MGeMServer:~$ 
```

From Windows

Build in terminal

```
jumpcloud — -zsh — 80×24
Last login: Tue Jan 11 11:10:04 on console
jumpcloud@JumpCloud ~ % 
```

From MacOS

What is the command line?

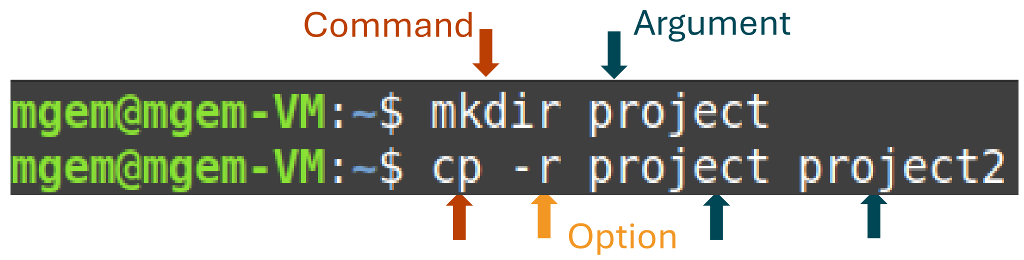
Command Line Interface* (CLI) is a text-based interface that allows users to interact with the computer operating system (OS) using the keyboard.

A *shell* refers to a program that is an intermediary between the user and the operating system (e.g. Bash, zsh, etc.).

How the CLI shell works?

1. **Command Prompt:** The CLI presents a prompt (symbol like $, >, or %) where the user types commands.

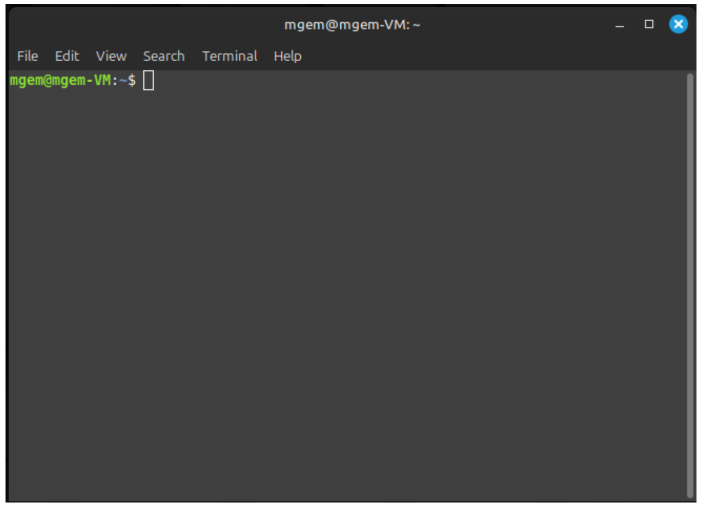
2. **Commands:** User's specific commands that the system interprets and executes such as file management (e.g., copying, moving, deleting files), running programs and scripts.



```
mgem@mgem-VM:~$ mkdir project
mgem@mgem-VM:~$ cp -r project project2
```

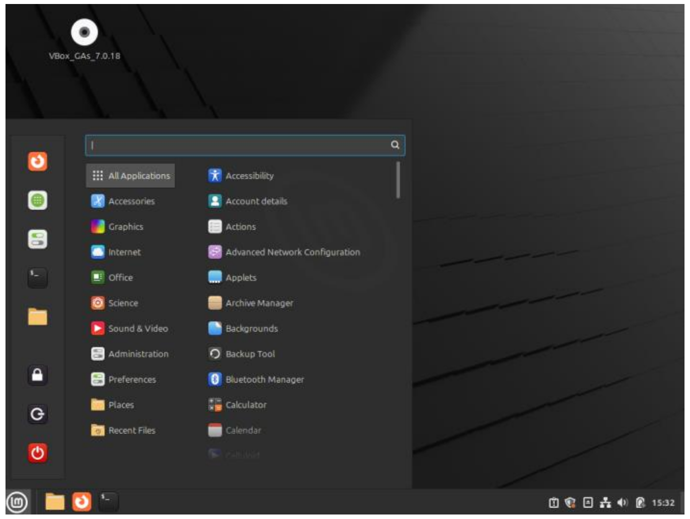
1. **Command execution:** The shell searches the system's PATH to locate and execute the command file associated with the user's input. The operating system then performs the requested actions.

2. **Output:** The system may return output/results directly in the terminal, such as informational messages, requested data, results of the operation, or an error message if something went wrong.



```
mgem@mgem-VM:~$
```

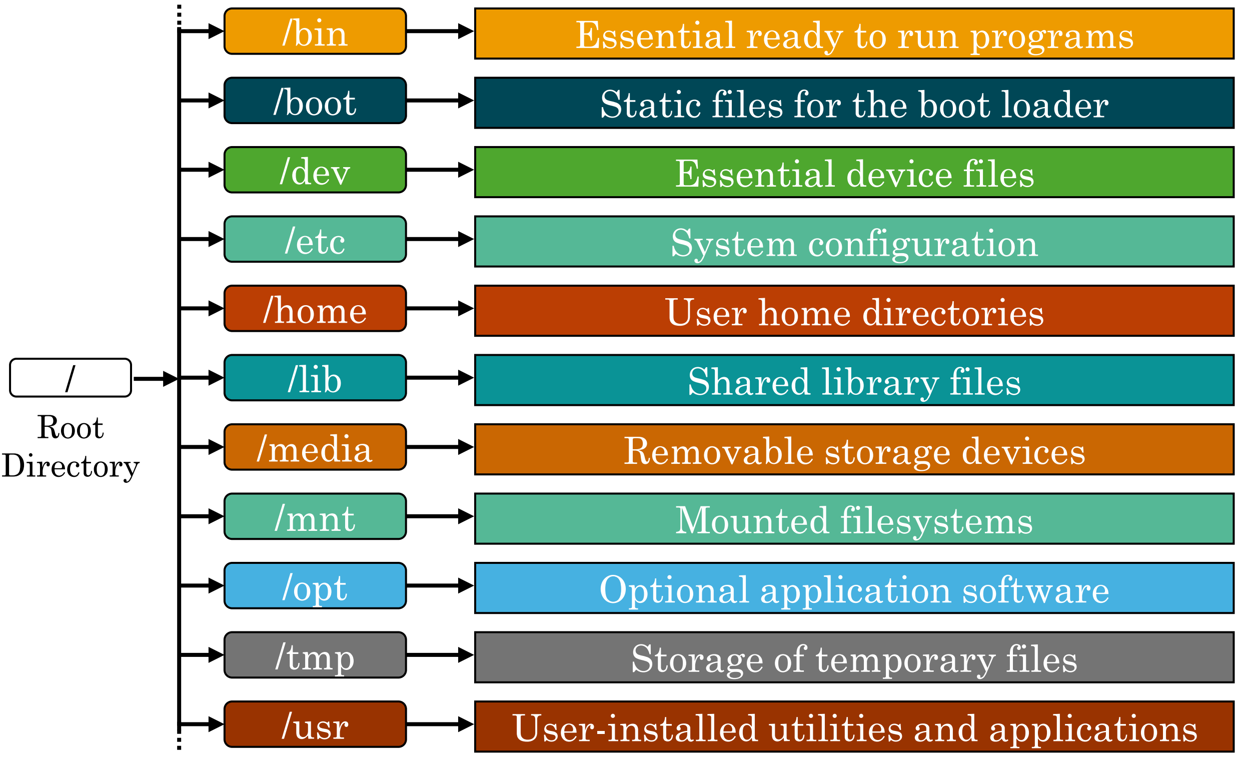
Command Line Interface (CLI)

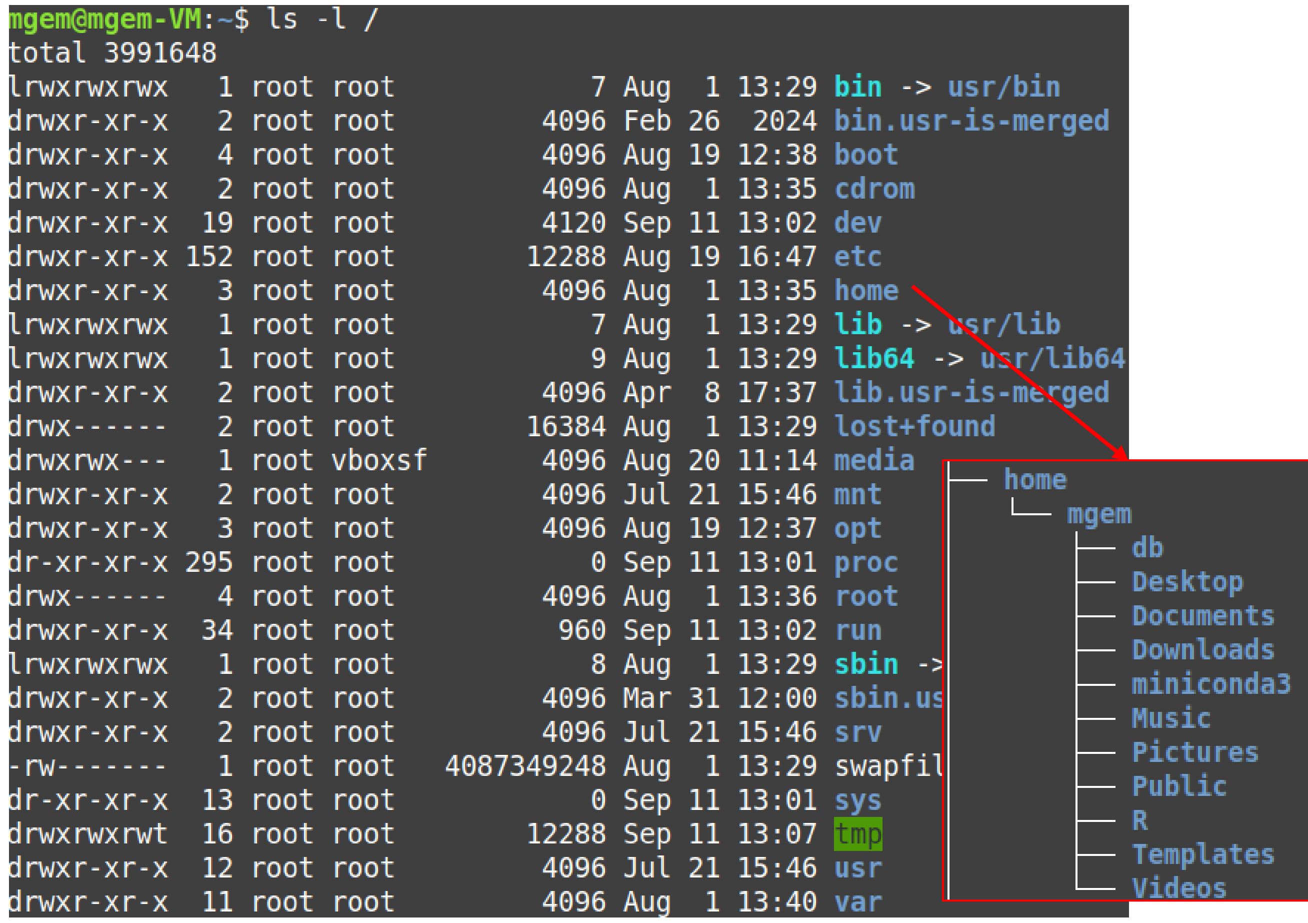


Graphical User Interface (GUI)

* Also known as shell, terminal, console, command prompts … etc.

Linux Directory Structure


/
Root Directory
/bin
Essential ready to run programs
/boot
Static files for the boot loader
/dev
Essential device files
/etc
System configuration
/home
User home directories
/lib
Shared library files
/media
Removable storage devices
/mnt
Mounted filesystems
/opt
Optional application software
/tmp
Storage of temporary files
/usr
User-installed utilities and applications



```
mgem@mgem-VM:~$ ls -l /
total 3991648
lrwxrwxrwx   1 root root          7 Aug  1 13:29 bin -> usr/bin
drwxr-xr-x   2 root root       4096 Feb 26  2024 bin.usr-is-merged
drwxr-xr-x   4 root root       4096 Aug 19 12:38 boot
drwxr-xr-x   2 root root       4096 Aug  1 13:35 cdrom
drwxr-xr-x  19 root root       4120 Sep 11 13:02 dev
drwxr-xr-x 152 root root      12288 Aug 19 16:47 etc
drwxr-xr-x   3 root root       4096 Aug  1 13:35 home
lrwxrwxrwx   1 root root          7 Aug  1 13:29 lib -> usr/lib
lrwxrwxrwx   1 root root          9 Aug  1 13:29 lib64 -> usr/lib64
drwxr-xr-x   2 root root       4096 Apr  8 17:37 lib.usr-is-merged
drwx------   2 root root      16384 Aug  1 13:29 lost+found
drwxrwx---   1 root vboxsf     4096 Aug 20 11:14 media
drwxr-xr-x   2 root root       4096 Jul 21 15:46 mnt
drwxr-xr-x   3 root root       4096 Aug 19 12:37 opt
dr-xr-xr-x 295 root root          0 Sep 11 13:01 proc
drwx------   4 root root       4096 Aug  1 13:36 root
drwxr-xr-x  34 root root        960 Sep 11 13:02 run
lrwxrwxrwx   1 root root          8 Aug  1 13:29 sbin ->
drwxr-xr-x   2 root root       4096 Mar 31 12:00 sbin.us
drwxr-xr-x   2 root root       4096 Jul 21 15:46 srv
-rw-------   1 root root 4087349248 Aug  1 13:29 swapfil
dr-xr-xr-x  13 root root          0 Sep 11 13:01 sys
drwxrwxrwt  16 root root      12288 Sep 11 13:07 tmp
drwxr-xr-x  12 root root       4096 Jul 21 15:46 usr
drwxr-xr-x  11 root root       4096 Aug  1 13:40 var
```

```
home
└── mgem
    ├── db
    ├── Desktop
    ├── Documents
    ├── Downloads
    ├── miniconda3
    ├── Music
    ├── Pictures
    ├── Public
    ├── R
    ├── Templates
    └── Videos
```

Absolute path: Complete route to a file or directory from the system root (/home/mgem/Documents)
Relative path: Location of a file or directory in relation to the current working directory (./Documents)

A single dot (.) refers to the current directory
Double dots (..) refers to the directory one level up

Basic CLI commands

Command	Description
whoami	Determine the current username (who am I?)
pwd	Present working directory (where am I?)
ls (-la)	List the content of a directory (with details)
cd *pathname*	Change directory (folder) in the file system
cd /	Move to the root folder of the file system
cd ..	Move one level up (one folder) in the file system
cp	Copy a file to another folder
mv	Move a file to another folder
mkdir	Creates a new directory (folder)
cat	Display a file
rm *filename*	Removes a file
rm -r (or rmdir)	Remove a directory
clear	Clears the CLI window
exit	Closes the CLI window
man *command*	Shows the manual for a given command

Default login path: /home/username ≡ ~
e.g., /home/username/Documents ≡ ~/Documents

Useful tricks to avoid excessive command typing

- Use copy/paste: ctrl+shift+c (copy) and ctrl+shirt+v (paste).
- Use Up/Down arrow keys: Cycle through recently executed commands.
- Use the TAB key: Autocomplete file/directory name.
- history command: list all recently used commands. Users can copy and paste a desired command to execute it again.
- The wildcard * symbol represents a string of any character of any length.

Basic CLI commands: File manipulation

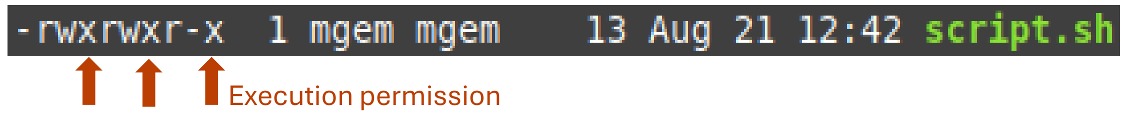
> or >> : Redirecting output
| : Piping output

Text files: Human-readable, can be viewed and modified using a text editor

- Text documents (e.g., README files)
- Data in text format (e.g., FASTA, FASTQ, VCF, ...)
- Scripts:
  - Bash scripts (*.sh or *.csh)
  - Python scripts (*.py)
  - R scripts (*.R)

Bash script
A bash script is a file containing a sequence of commands that are executed by the bash program line by line. It allows you to perform a series of actions, such as navigating to a specific directory, creating a folder, and launching a process using the command line.

Script manipulation	
#! /bin/bash	Bash script header
chmod +x script.sh	Give execution permission to a *bash* script
./script.sh	Run a *bash* script



```
-rwxrwxr-x  1 mgem mgem    13 Aug 21 12:42 script.sh
```

Create and print file	
nano first.txt	Create a text file (.txt) named *first*
echo "Hello world!" > first.txt	Insert “Hollo word” into *first.txt* file
cat first.txt	Print content of *first.txt* file
cat file1.txt file2.txt > file3.txt	Concatenate the contents of the file1/2 into *file3.txt*
echo “RandomText“ >> first.txt	Append the content of *first.txt* file
grep “o” first.txt	Search for pattern “o” matches
cat first.txt | grep "o"	Piping the output of cat into grep command
wc -l first.txt	Count the number of lines in *first* file
grep "o" first.txt | wc -l	Count number of lines with pattern “o”
head first.txt	Display top line of a file
tail first.txt	Display bottom of a file

Compression and Archives	
gzip first.txt	Compress *first* file into *first.txt.gz* file
gunzip first.txt.gz	Uncompress *first* file
zip -r folder.zip folder/	Compress *folder* to *folder.zip* file
unzip folder.zip	Uncompress folder *folder.zip*
tar -cvzf folder.tar.gz folder/	Compress *folder* into *folder.tar.gz*
tar -xvzf folder.tar.gz	Extract *folder.tar.gz*

Packages and Environments Management

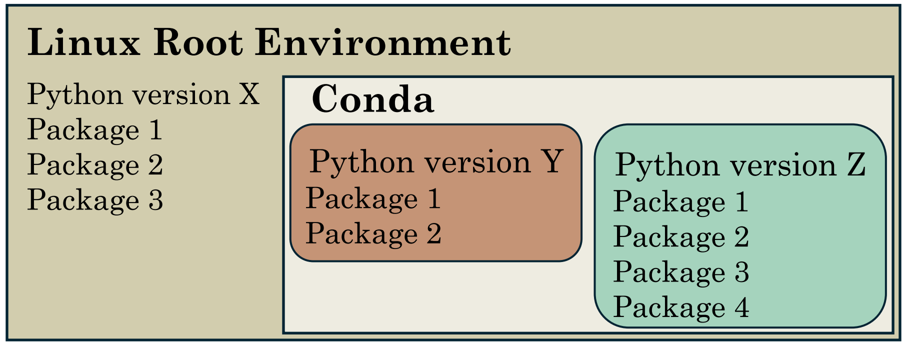
In bioinformatics, managing software dependencies and environments is vital for:

- Reproducibility: Ensuring that analyses can be replicated reliably.
- Consistency: Maintaining stable environments across different computational tasks.
- Efficiency: Streamlining workflows and reducing conflicts.

Challenges

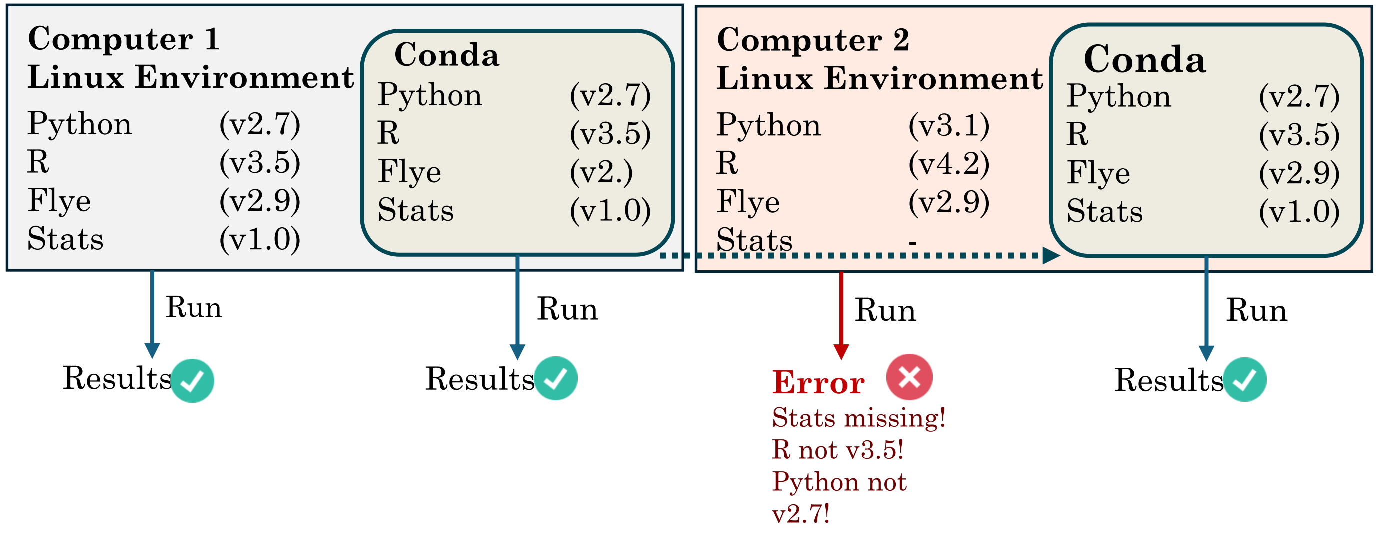
- Bioinformatics tools are developed in various programming languages.
- Specific library versions are often required for different tools.
- Manual management of these dependencies can be complex, especially in Linux environments.

Conda is a package manager allows specific versions of programs to be installed, alongside their dependencies. Different sets of programs can be installed in different virtual environments.

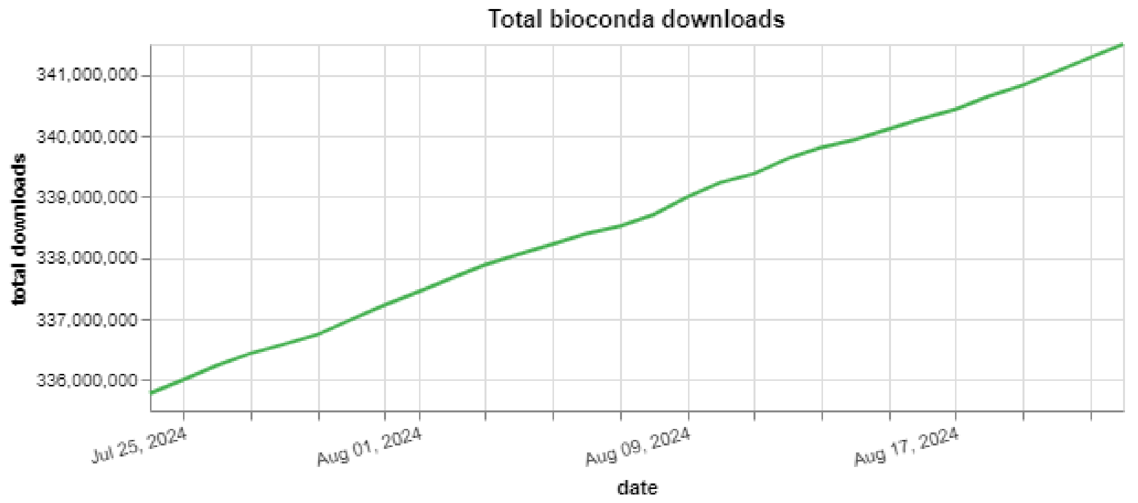


Packages and Environments Management

- **Simplified package management:** Installation, management, and updating of software packages and their dependencies across different programming languages.
- **Environment Management:** Allows to create isolated environments for different projects avoiding conflicts between different software requirements.
- **Extensive Package Ecosystem:** Provides access to a large collection of pre-built packages for various domains.
- **Conda Channels:** Serve as repositories for hosting and managing packages. (e.g., **Bioconda**, conda-forge and general community channels)



Bioconda is a community-enabled repository of 3,000+ bioinformatics packages, installable via the conda package manager.

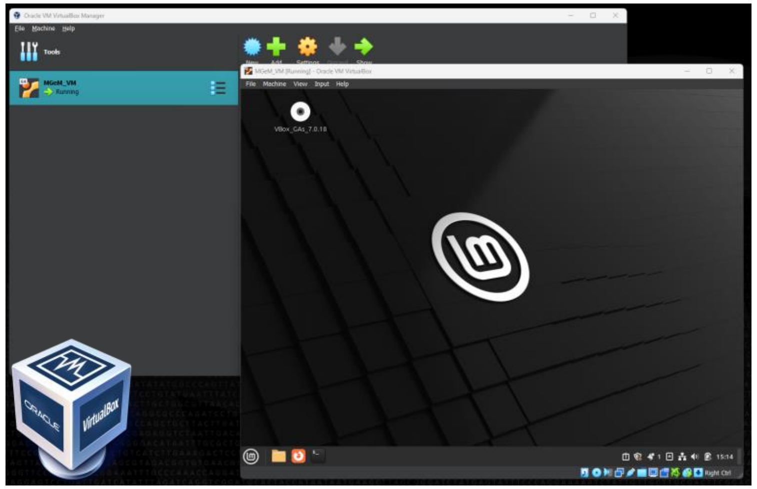


Bioconda is not available for windows systems

Good news !

Most of the tools required for the analysis pipelines in our hands-on sessions are **pre-installed** within a Conda environment.

The Conda environments can be easily extracted from the virtual machine and transferred to a more powerful computing resource.



HTS File format

Breakdown of the common file formats

POD5

Raw signal data

FASTQ

Sequence + Quality

FASTA

Sequence

SAM/ BAM

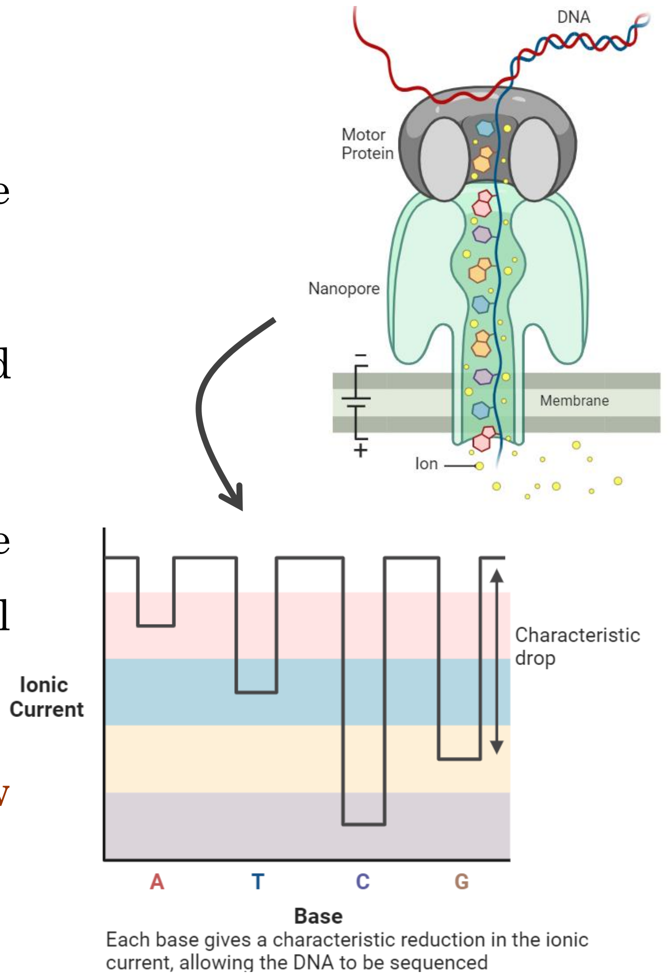
Sequence alignment

HTS File format

Oxford Nanopore Files

What is POD5?

- **POD5** is a newer format developed by ONT to store raw signal data from nanopore sequencing runs, replacing the previous FAST5 format
- ONT introduced POD5 to enhance performance in terms of both storage efficiency and access speed
- Contain raw electrical signal data (changes in ionic current) from the nanopore sensor, as well as metadata related to sequencing, like channel and the flow cell Information
- Used by ONT's basecalling software (e.g., Guppy, Dorado), which **translates** the raw signal data into nucleotide sequences (A, T, C, G)



Each base gives a characteristic reduction in the ionic current, allowing the DNA to be sequenced

HTS File format

FASTQ format

- Text-based format used for storing both nucleotide sequences and their corresponding quality score
- Common file extensions: .fastq, and .fq
- Generated directly from sequencing machines (Oxford Nanopore: Fast5 format, PacBio: HDF5 format)

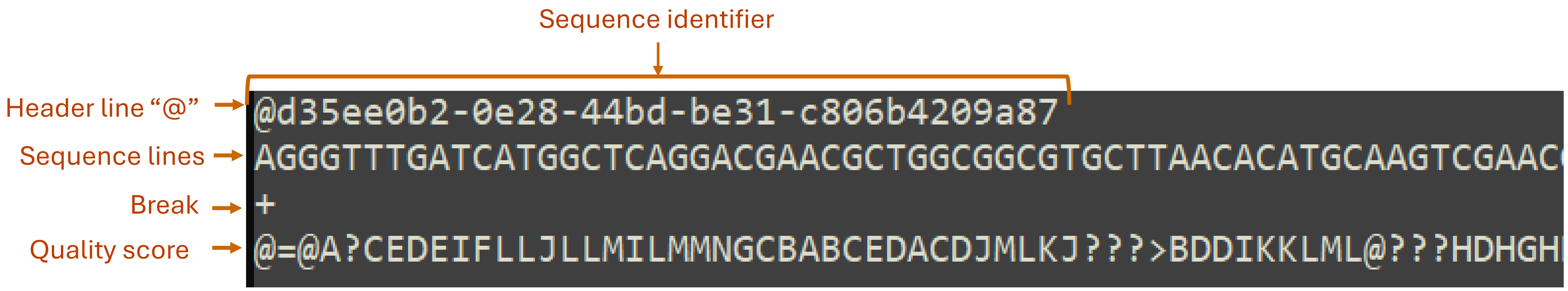
FASTQ file structure

1. Header line: Begins with a @ symbol, followed by a UNIQUE sequence identifier

2. Sequence data: The actual nucleotide (A, T, G, C, N) on 1 line

3. A separator: Simply as a + sign, sometimes as +SequenceName

4. Quality Score: ASCII characters to represent the numerical quality score (Phred +33) for each base



```
@d35ee0b2-0e28-44bd-be31-c806b4209a87
AGGGTTTGATCATGGCTCAGGACGAACGCTGGCGGCGTGCTTAACACATGCAAGTCGAAC
+
@=@A?CEDEIFLLJLLMILMMNGCBABCEDACDJMLKJ???>BDDIKKLML@???HDHGH
```

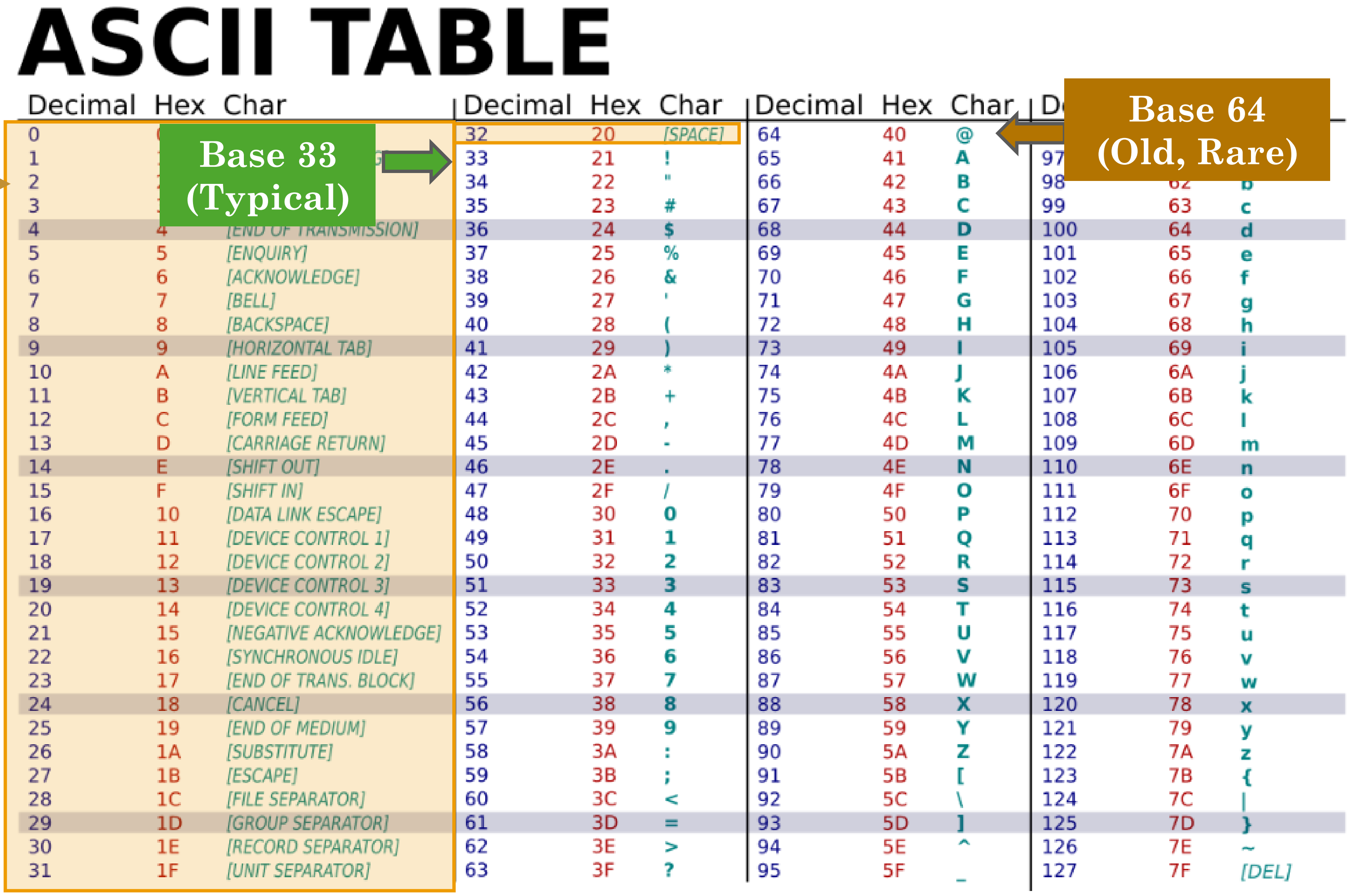
HTS File format

FASTQ format: Understanding Quality String

These characters don't appear in print

Base 33 (Typical)

Base 64 (Old, Rare)



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[illegible]	32	20	[SPACE]	64	40	@	[illegible]	[illegible]	[illegible]
1	[illegible]	[illegible]	33	21	!	65	41	A	97	[illegible]	[illegible]
2	[illegible]	[illegible]	34	22	"	66	42	B	98	62	b
3	[illegible]	[illegible]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

HTS File format

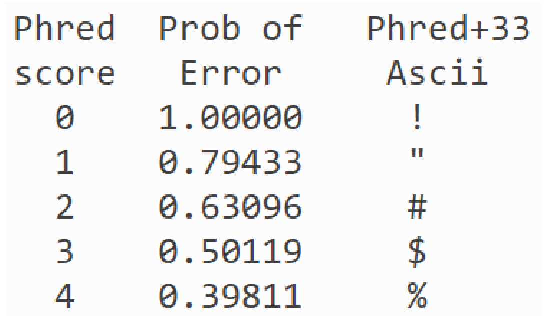
FASTQ format: Understanding Quality String

- Quality strings use ASCII values to encode a two-digit integer using a single character
- Keeps files smaller
- Keeps scores in alignment with bases
- Confidence > Phred > Base value adjustment > Number > Character

$$Q = -10 \times \log_{10}(p)$$

p = probability call is not correct

Quality Score (Q)	Probability (incorrect base call)	Base Call Accuracy
0	1/1	0%
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.90%
40	1 in 10,000	99.99%



Phred score	Prob of Error	Phred+33 Ascii
0	1.00000	!
1	0.79433	"
2	0.63096	#
3	0.50119	$
4	0.39811	%

- The quality line is always the same length as the sequence line
- A single FASTQ file can contain multiple sequences, each with its own header (millions)
- Sequences can be different length

HTS File format

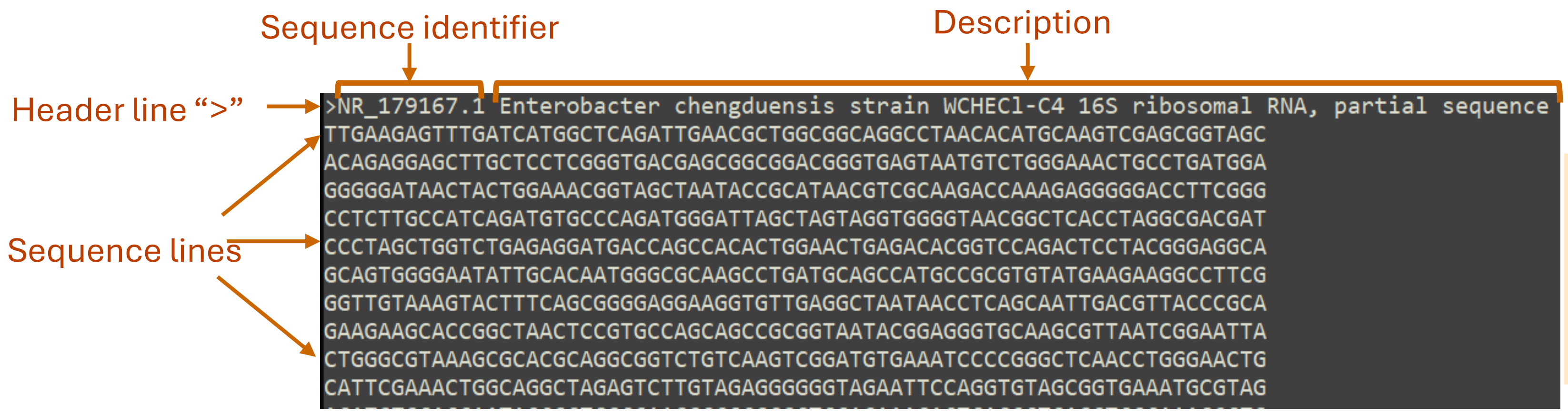
FASTA format

- Text-based format used for representing nucleotide or protein sequences
- Typical file extensions: .fasta, .fa, .fas, .fna, .faa, fsa
- Commonly used for storing and sharing of sequence data
- Compatible with a broad range of bioinformatics tools and software

FASTA file structure

1. **Header line**: Begins with a **>** symbol, followed by an identifier and optional description
2. **Sequence data**: The actual nucleotide (A, T, G, C) or amino acid sequence, presented in a plain text format.

Typically, Sequence data can be spread across multiple lengths of length 80, 70, or 60 bases



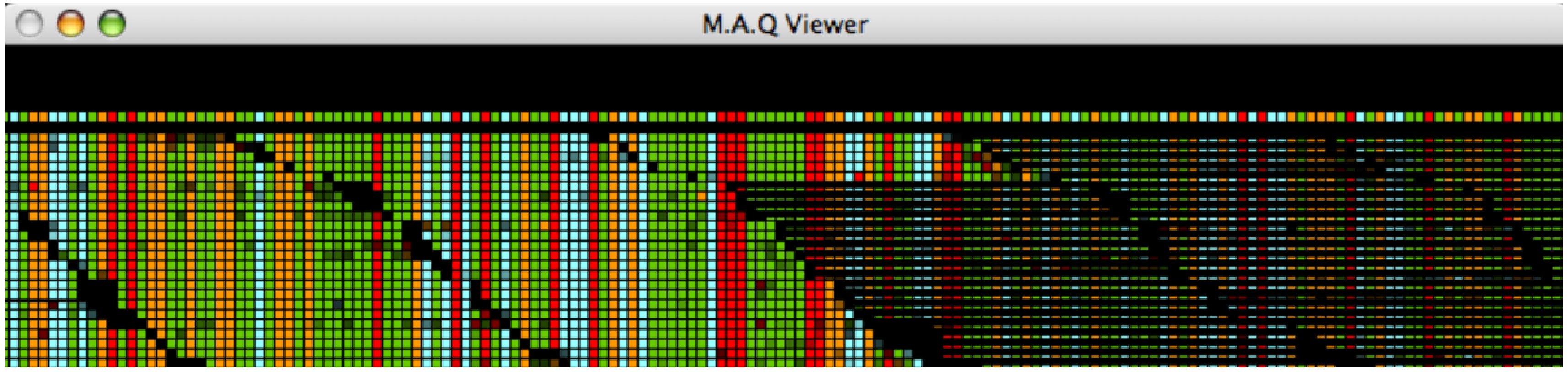
```
>NR_179167.1 Enterobacter chengduensis strain WCHECl-C4 16S ribosomal RNA, partial sequence
TTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCCTAACACATGCAAGTCGAGCGGTAGC
ACAGAGGAGCTTGCTCCTCGGGTGACGAGCGGCGGACGGGTGAGTAATGTCTGGGAAACTGCCTGATGGA
GGGGGATAACTACTGGAAACGGTAGCTAATACCGCATAACGTCGCAAGACCAAAGAGGGGGACCTTCGGG
CCTCTTGCCATCAGATGTGCCCAGATGGGATTAGCTAGTAGGTGGGGTAACGGCTCACCTAGGCGACGAT
CCCTAGCTGGTCTGAGAGGATGACCAGCCACACTGGAACTGAGACACGGTCCAGACTCCTACGGGAGGCA
GCAGTGGGGAATATTGCACAATGGGCGCAAGCCTGATGCAGCCATGCCGCGTGTATGAAGAAGGCCTTCG
GGTTGTAAAGTACTTTCAGCGGGGAGGAAGGTGTTGAGGCTAATAACCTCAGCAATTGACGTTACCCGCA
GAAGAAGCACCGGCTAACTCCGTGCCAGCAGCCGCGGTAATACGGAGGGTGCAAGCGTTAATCGGAATTA
CTGGGCGTAAAGCGCACGCAGGCGGTCTGTCAAGTCGGATGTGAAATCCCCGGGCTCAACCTGGGAACTG
CATTCGAAACTGGCAGGCTAGAGTCTTGTAGAGGGGGGTAGAATTCCAGGTGTAGCGGTGAAATGCGTAG
```

- A single FASTA file can contain multiple sequences, each with its own header
- Sequences do not need to have the same length

HTS File format

SAM/BAM format

- **SAM** (Sequence Alignment/Map) files are a standard format used to store aligned sequence data
- **BAM** (Binary Alignment/Map) is just the binary version of a SAM file, suitable for fast processing
- Hold aligned sequence reads along with metadata, including:
  - The sequence of nucleotides.
  - Quality scores for each base.
  - Alignment data, including reference genome coordinates.
  - Auxiliary tags such as mapping quality, gaps/mismatches, etc.



HTS Data Source

Where to get HTS data?

- Your own Experiment
- **NCBI Sequence Read Archive (SRA):** A comprehensive repository for raw sequence data from various projects.

  (https://www.ncbi.nlm.nih.gov/sra)
- **NCBI Genome:** A comprehensive repository for genome sequences of various organisms.

  (https://www.ncbi.nlm.nih.gov/genome)
- **European Nucleotide Archive (ENA):** Provides access to raw reads, assembled sequences, and functional information.

  (https://www.ebi.ac.uk/ena/browser/home)
- **DNA Data Bank of Japan (DDBJ):** Offers sequence data submission and retrieval services, including HTS data.

  (https://www.ddbj.nig.ac.jp/index-e)

HTS Data Source

How to retrieve HTS Data?

1. Find relevant datasets by keywords, organism, study or accession number
2. Access the datasets through the database's web interface, or, using command line tools such as sra-tools (NCBI SRA) or ascp (ENA)
3. Before downloading, review associated metadata to ensure the dataset's suitability for the analysis (e.g., sample type, sequencing platform).
4. Download raw sequence data in FASTQ format using commands such as prefetch, fastq-dump (for SRA)
5. Download genome assemblies in FASTA format using commands such as ncbi-genome-download (for NCBI)

Example: `ncbi-genome-download -s refseq --formats fasta -A accessions.txt -o output --metadata-table metadata.tsv  bacteria`

What's Next?

➡ Prepare for the Hands-On Session

- Set up the Linux Virtual Machine
- Ensure all necessary tools and environments are ready
- Ensure you understand the directory structure where you will be working
- Get familiar with the command line interface (CLI)