Transcriptome Data Analysis in Non-model Organisms

Jiratchaya Nuanpirom Prasert Yodsawat
Ponsit Sathapondecha

3/9/23

Table of contents

Preface

Welcome to the book of Transcriptome Data Analysis in Non-Model Organisms.

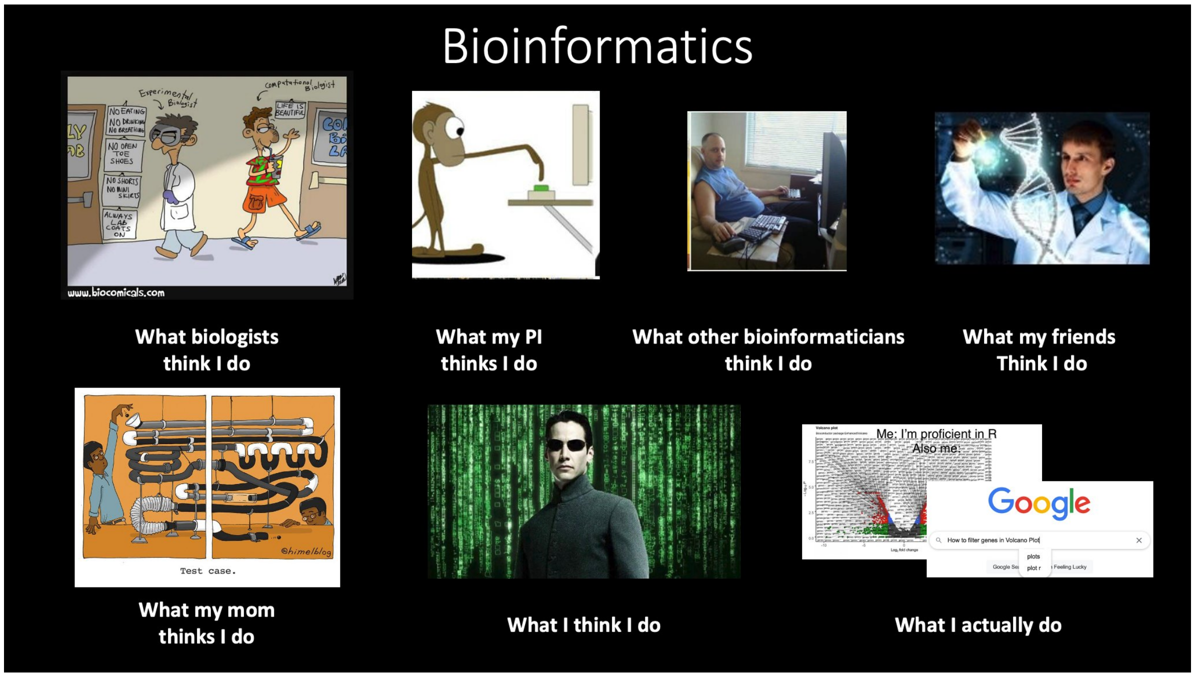


Figure 1: A confusion matrix. From twitter @ninadoak.

1 Introduction to MobaXterm, Terminal, and SSH

1.1 MobaXterm (for Windows)

MobaXterm is a toolbox for remote computing. In a single Windows application, it provides loads of functions that are tailored for programmers, webmasters, IT administrators and pretty much all users who need to handle their remote jobs in a more simple fashion. MobaXterm provides all the important remote network tools, such as SSH, X11, RDP, VNC, FTP, MOSH, and of course, Unix commands, and many more!

There are many advantages of having an All-In-One network application for your remote tasks, e.g. when you use SSH to connect to a remote server, a graphical SFTP browser will automatically pop up in order to directly edit your remote files.

Visit MobaXterm official website to see a demo: https://mobaxterm.mobatek.net/demo.html

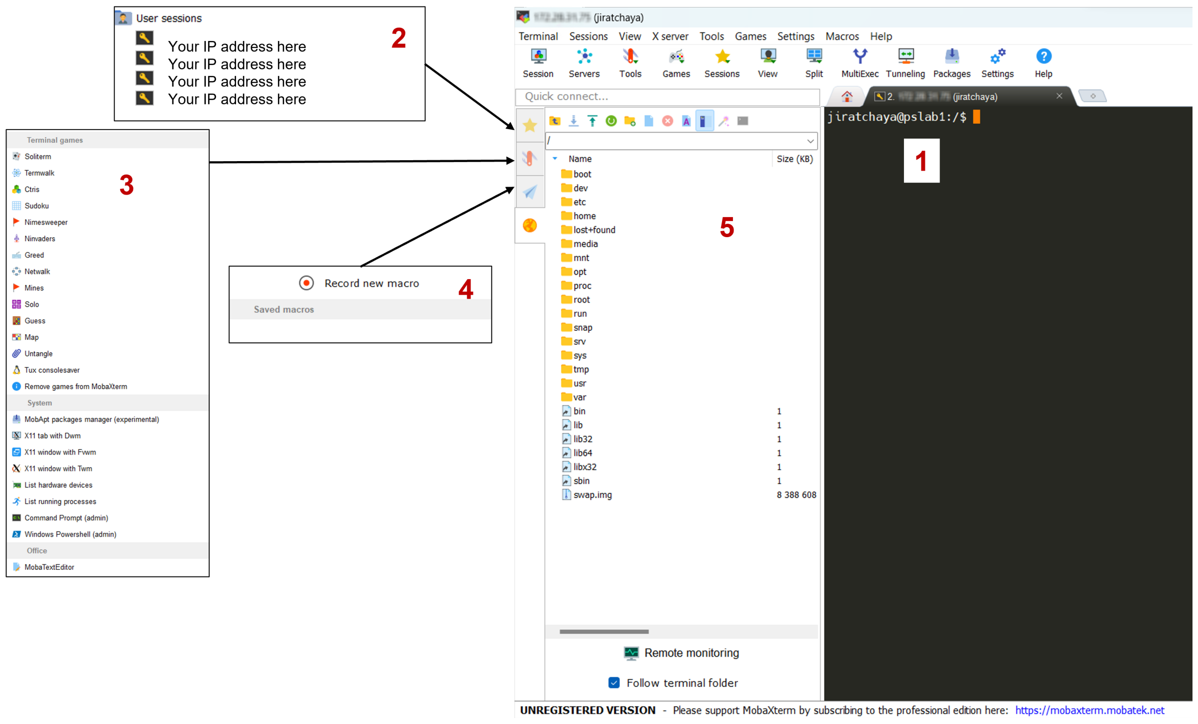
1.2 Terminal (for macOS)

Terminal provides a command-line interface to macOS. Each window in Terminal represents an instance of a shell process. The window contains a prompt that indicates you can enter a command. The prompt you see depends on your Terminal and shell settings, but it often includes the name of the host you're logged in to, your current working folder, your user name, and a prompt symbol. For example, if a user named michael is using the default zsh shell, the prompt appears as:

```
michael@MacBook-Pro ~ %
```

This indicates that the user named michael is logged in to a computer named MacBook-Pro, and the current folder is his home folder, indicated by the tilde (~).

MacOS features a built-in SSH client called Terminal which allows you to quickly and easily connect to a server. Starting from open the "terminal" app, and enter the standard SSH command:



```
jiratchaya@pslab1:/$
```

Name	Size (KB)
boot	
dev	
etc	
home	
lost+found	
media	
mnt	
opt	
proc	
root	
run	
snap	
srv	
sys	
tmp	
usr	
var	
bin	1
lib	1
lib32	1
lib64	1
libx32	1
sbin	1
swap.img	8 388 608

Figure 1.1: MobaXterm user interface. In the context of remote access through SSH and FTP, mobaXterm provides easy-to-access route as (1) a secure shell (SSH) terminal of the remote server, (2) a list of remote server you've accessed, (3) Utilities facilitating remote server access including entertainment, like Swiss army knife!, (4) If you want to reduce redundant typing, just set macro to it, and (5) a files available in the current working directory in the remote server, you can also transfer files from remote server to your local computer using this route!

```
ssh user@IPAddress
```

This will connect to the server via SSH with the username ‘user‘ and the default SSH port 22. The connection will look similar to the following:

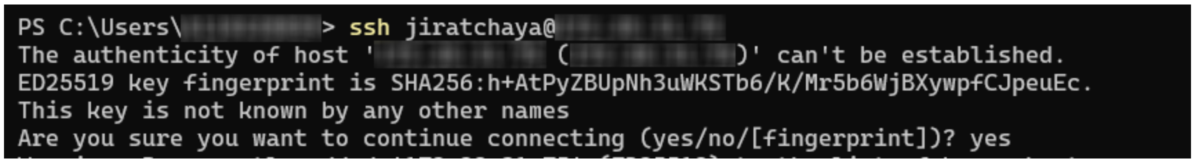
1.3 Connecting to Remote Server

Bioinformatics data processing tasks require more computing power than our laptops, so we need large servers or clusters. It’s likely you’ll work mostly over a network connection with remote machines on some projects. It can be frustrating for beginners to work with a remote machine. So, This part will introduce you to some commonly used bash commands. To make it easier for beginners to manage their remote machines, there are a range of different tools and technologies available, such as SSH, FTP, and terminal commands, which can be used to access and manage the environment of the machine. Additionally, there are a variety of bash commands which can be used to streamline the process of managing the machine Buffalo (2015).

What you need to know for connecting to a remove server:

1. Your username and password in the remote server
2. IP address of the remote server, and which port to connect to server
3. You should know whether the remote server accessible via local network or a public IP address

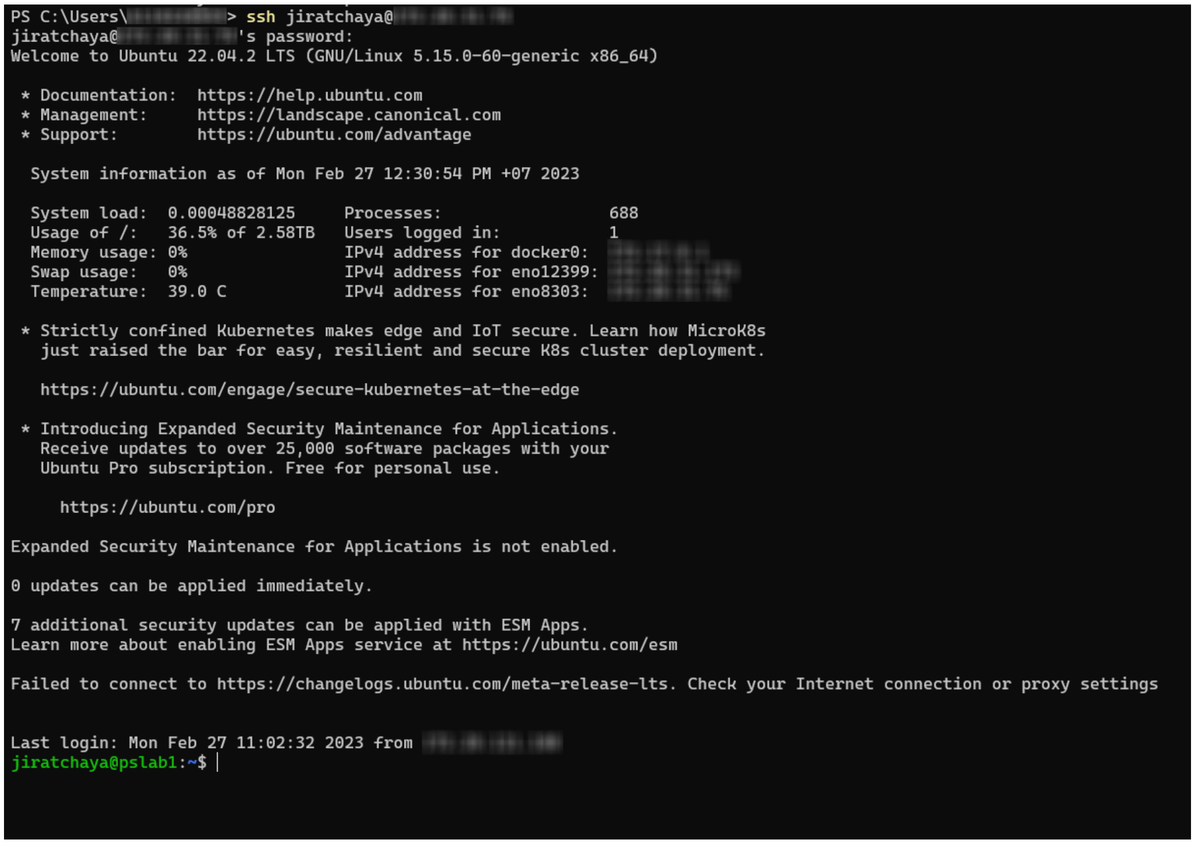
By default, SSH uses port 22 but it can be changed to a non-standard port. To securely connect the client to the remote server, SSH uses symmetric encryption, asymmetric encryption, and hashing. If you’re connecting for the first time, you’ll be asked to verify the server’s public key. Whenever you connect to the same server in the future, the client will reference this verified public key. During an SSH connection, the client and server negotiate a session key used to encrypt and decrypt data.



```
PS C:\Users\[redacted]> ssh jiratchaya@[redacted]
The authenticity of host '[redacted] ([redacted])' can't be established.
ED25519 key fingerprint is SHA256:h+AtPyZBUpNh3uWKSTb6/K/Mr5b6WjBXywpfCJpeuEc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Figure 1.2: In order to establish a connection, SSH needs to verify SHA keys once connected for the first time. Once authentication is complete, the SSH connection is secure and can be trusted for future access.

Upon connecting to the remote server, you’ll see a welcome message like this



```
PS C:\Users\[redacted]> ssh jiratchaya@[redacted]
jiratchaya@[redacted]'s password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-60-generic x86_64)

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

  System information as of Mon Feb 27 12:30:54 PM +07 2023

  System load:  0.00048828125     Processes:                688
  Usage of /:   36.5% of 2.58TB   Users logged in:          1
  Memory usage: 0%                IPv4 address for docker0: [redacted]
  Swap usage:   0%                IPv4 address for eno12399: [redacted]
  Temperature:  39.0 C            IPv4 address for eno8303: [redacted]

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Mon Feb 27 11:02:32 2023 from [redacted]
jiratchaya@pslab1:~$
```

Figure 1.3: An example welcome message of server using Ubuntu, including general software and hardware status, information of the latest connection, as well as a prompt for user command.

2 Bash Command Language for Biologists

Shell scripts (or shell or UNIX) are widely used in bioinformatics because they're the interface for large bioinformatics programs. In this workshop, you'll learn how to use the necessary Bash command concepts. This will allow you to focus on the content of the commands in the following chapters rather than on understanding shell syntax. However, before we start learning bash, it's good to understand Linux file systems a little bit.

2.1 Linux File Systems

In Unix-like operating systems, the Linux file system defines the directory structure and contents. Even if they're located on different physical or virtual hard disks, all files and directories are located under the root directory.

Root (/) It is the root directory of the entire file system hierarchy and the primary hierarchy root. The root directory is where everything begins. This directory can be written only by root.

/bin Essential commands that must be available with all users, for example, `cat`, `ls`, `cp`, `cd`, `top`, `mkdir` and many more.

/dev Essential device files such as `/dev/null`, `/dev/shm`. This includes terminal devices, USB or other devices connected to the system.

/etc System-wide configuration files for the host, contain files that all programs need. Also included are startup and shutdown shell scripts for starting and stopping individual programs, such as `/etc/fstab` for permanently mounting external disks, `/etc/netplan` for configuring the network and IP address, and more.

/home Users' home directories, where they keep their saved files and settings. These directories are used to store all of a user's files and settings in one place so that they can easily access their data and keep it organized. For example `/home/ponsit`, `/home/jiratchaya`, `/home/prasert`.

/lib Contain essential libraries for the binaries in `/bin/` and `/sbin/`.

/media Mount points for removable media such as CD-ROMs (deprecated).

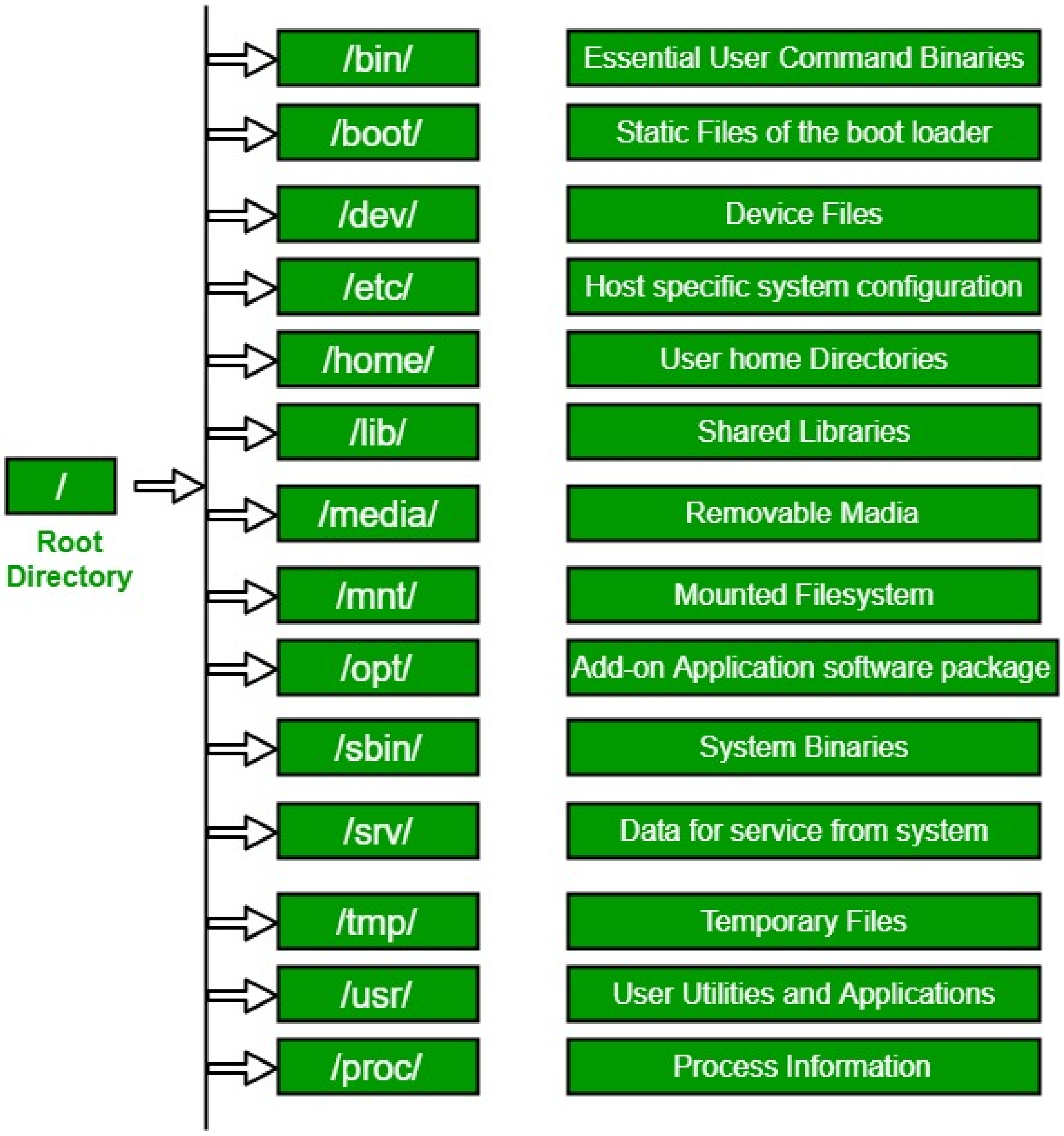


Figure 2.1: Schematic hierarchy of Linux file systems. The figure is adopted from https://www.geeksforgeeks.org/linux-file-hierarchy-structure.

/mnt Temporary mount directory where sysadmins can mount file systems, such as `/mnt/external_disk_1`, `/mnt/removable_drive_1`, etc.

/opt Optional application software packages, including add-on applications from individual vendors.

/sbin Essential system binaries, e.g., `fsck`, `init`, route.

/tmp Temporary files that aren't preserved between reboots and may be severely restricted.

/usr A secondary hierarchy for read-only user data. Most utilities and applications are located here.

2.2 Basic Bash Commands

Bash is a Unix shell that allows you to enter commands that are then interpreted and executed by the computer. Commands can be used to perform tasks such as creating a directory, running a program, or deleting a file. Bash is a type of interpreter that takes user input and converts it into a language that the computer can understand and execute. Commands usually consist of keywords, arguments, and flags that allow the user to control how the command is interpreted and executed by the computer.

Creating directories

Keeping all your files in a single directory makes things much easier for you and your collaborators, and makes it easier to reproduce. Suppose you're working on a transcriptome analysis of *Cyanophora paradoxa.* Your first step would be to choose a short, appropriate project name and create some basic directories.

> **Note:** In Linux file system, **directory** is exactly the same with **folder**.

To keep it short and clear, 'Cpa' is used as an alias article name for *C. paradoxa*, and as the name of the directory, followed by words describing your work, for example.

> **warning:** Avoid using spaces ( ) or special characters such as slashes (/), backslashes (\ ), accented characters ('), tilde (~), and many others. It is **recommended to use underscore (_)** or hyphen (-) instead of these special characters.

- Create a directory name 'Cpa_RNASeq' from current working directory

```
mkdir Cpa_RNASeq
```

This will create a directory named 'Cpa_RNASeq' in your current working directory. Let us create some subdirectories!

- Create subdirectory '01_Rawdata' under the 'Cpa_RNASeq' directory

```
mkdir Cpa_RNASeq/01_Rawdata
```

This will create a subdirectory name '01_Rawdata' in the directory 'Cpa_RNASeq'

- Create multiple directory at once

For example, if you want to create 2 directories named '02_QC' and '03_assembly' under the 'Cpa_RNASeq' directory, then simply type

```
mkdir Cpa_RNASeq/{02_QC,03_assembly}
```

Activity

A well-organized project directory can make life easier. Your project directory should be organized in a consistent and understandable way. A clear project organization makes it easier for both you and your collaborators to find out exactly where and what everything is located, which improves the reproducibility of research. It's also much easier to automate tasks when files are organized and clearly named.
In this workshop, we'll learn transcriptome data analysis in many steps from downloading reads to transcriptome annotation. Therefore, we'll divide each analysis step into subdirectories as follows. Let's assume that we have already created the directory `Cpa_RNASeq`.

```
.
  Cpa_RNASeq
      01_Rawdata
      02_QC
      03_assembly
      04_DE_analysis
      05_annotation
```

Could you generate bash command(s) to create these directories ?

Answer

```
mkdir Cpa_RNASeq/{01_Rawdata,02_QC,03_assembly,04_DE_analysis,05_annotation}
```

or

```
mkdir Cpa_RNASeq/01_Rawdata Cpa_RNASeq/02_QC Cpa_RNASeq/03_assembly Cpa_RNASeq/04_DE_ana
```

or

```
cd Cpa_RNASeq
mkdir 01_Rawdata 02_QC 03_assembly 04_DE_analysis 05_annotation
```

Navigating your file system

The file system manages the files and directories of the operating system. It organizes our data into files, which store information, and directories. When you see the prompt below on your terminal's screen, it means that your terminal has processed the command you entered and is ready for the next command.

```
jiratchaya@DESKTOP-P2DD13C:~$
```

`jiratchaya` is username using this terminal. The address @ symbol followed by `DESKTOP-P2DD13C` in a computer or server name. And, the dollar sign `$` is a prompt, which shows us that the shell is waiting for input. Your shell may use a different character as a prompt and may add information before the prompt.

If you want to find out where we are now, type

```
pwd
```

pwd stands for print working directory. Without explicit specification, the computer assumes that we want to execute commands in our current working directory. This can be a user's home directory (~).

If you want to change the directory, e.g. to the 'Cpa_RNASeq' directory we just created, just type the following

```
cd Cpa_RNASeq
```

`cd` stands for "change directory". You can change our working directory by typing `cd` followed by a directory name. In this case you change from the current directory to the directory named 'Cpa_RNASeq'.

Listing directories

We can see what files and subdirectories are in this directory by running ls, which stands for "listing":

```
ls
```

Expected result:

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls
01_Rawdata  02_QC  03_assembly
```

Let us look at the other way. This way is to list all the files and directories, including the users who own them, the permissions, and the file size in bytes.

```
ls -l
```

Expected result:

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls -l
total 12
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 01_Rawdata
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 02_QC
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 03_assembly
```

List files and folders, permissions and size in a human readable format.

```
ls -lh
```

Expected result:

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls -l
total 12
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 01_Rawdata
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 02_QC
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 03_assembly
```

See all hidden files and directories

```
ls -la
```

Expected result:

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls -la
total 20
drwxr-xr-x 5 jiratchaya jiratchaya 4096 Mar  1 21:02 .
drwxr-x--- 3 jiratchaya jiratchaya 4096 Mar  1 21:02 ..
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 01_Rawdata
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 02_QC
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:02 03_assembly
```

Files and directories handling

Creating and editing files

When you work on the command line, you often need to create or edit text files. In this workshop, we recommend using `nano` as a text editor. Other Unix text editors you may have heard of include vi, vim, emacs, vscode, and many more.

We'll create the file `test.fasta`. To open an existing file or create a new file, type nano followed by the filename:

```
nano test.fasta
```

This will bring up the text editing screen on your terminal. Here you can type anything you want, but in this case we'll create a sequence like this.

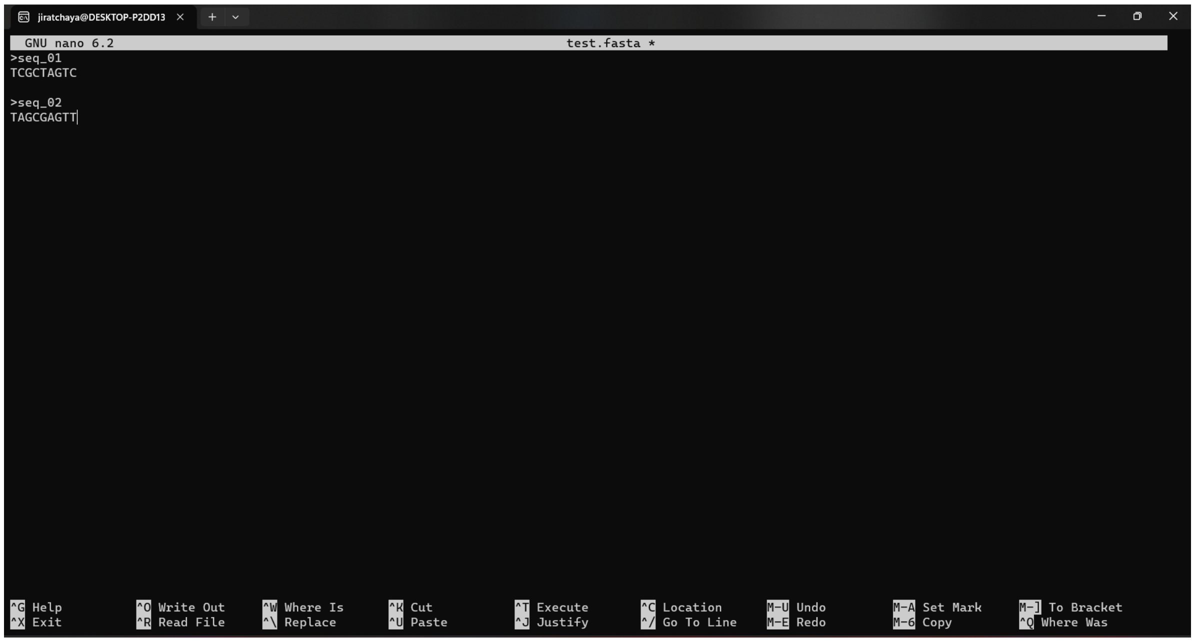
```
>seq_01
TCGCTAGTC

>seq_02
TAGCGAGTT
```

> **Note:**Always leave an enter in the last line. This is advantageous if this file is further used by many programmes.

To edit a file, you can use the navigation keys such as arrow keys, End, Home, PgUp or PgDn to control the cursor.

To save the changes you made to the file, press `Ctrl+o`. If the file doesn't exist yet, it'll be created after saving.

To exit nano, press `Ctrl+x`. If there are unsaved changes, you'll be asked if you want to save the changes. Nano will ask you 'Save modified buffer?', then type `y` to confirm the edit.



```
  GNU nano 6.2                                   test.fasta *
>seq_01
TCGCTAGTC

>seq_02
TAGCGAGTT
```

```
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was
```

Figure 2.2: The text editing screen is displayed once you have typed nano into some files. At the bottom of the window is a list of the most important keyboard shortcuts for the nano editor. All commands are preceded by either a `^` or an `M` character. The caret symbol (`^`) stands for the `Ctrl` key. For example, the commands `^J` mean that you press the `Ctrl` and `J` keys simultaneously. The letter `M` stands for the `Alt` key.

Copying files and directories

To copy files and directories the command `cp` can be used. `cp` stands for copy and is used to copy files and directories in Linux. An example: You copy the file `test.fasta` to `01_Rawdata` with the following syntax

```
cp [source file] [target_directory]/
```

For example

```
cp test.fasta 01_Rawdata/
```

Copy file to another file, using the syntax

```
cp [source_file] [new_file_name]
```

For example

```
cp test.fasta test_2.fasta
```

You can copy a file to a new file in the directory by using the following syntax

```
cp [source_file] [target_directory]/[new_file_name]
```

For example

```
cp test.fasta 01_Rawdata/another_test.fasta
```

To copying directory, use additional flag as follow

```
cp -r [source_directory] [new_directory_name]
```

The flag `-r` stands for recursive, i.e. all files and subdirectories in this directory are copied repeatedly. For example, `01_Rawdata` already contains `test.fasta`, which we copied before, and we want to duplicate this directory.

```
cp -r 01_Rawdata/ 01_Rawdata_new
```

Moving files and directories

To copy files and directories, the command `mv` can be used. `mv` stands for move and is used to move files and directories in Linux. For example, move the file `test_2.fasta` to the directory `01_Rawdata_new` with the following syntax

```
mv [file_to_move] [target_directory]
```

```
mv test_2.fasta 01_Rawdata_new/
```

Specifically, to move files and directories, no flags are required as with `cp`. So if we want to move `01_Rawdata_new` to a subdirectory of `01_Rawdata`, this can be done as follows

```
mv [source_file_or_dir] [target_file_or_dir]
```

```
mv 01_Rawdata_new/ 01_Rawdata
```

Moving file within the directory up to the current directory

```
mv [source_dir]/[source_file] .
```

The dot (.) stands for the current directory, which means you want to move something to the current directory. For example, we want to move the file `another_test.fasta`, which is in the directory `01_Rawdata`, to the current directory by typing

```
mv 01_Rawdata/another_test.fasta .
```

Deleting files and directories

Removing files and directories can be done with the command `rm`. `rm` stands for remove and is used to delete files and directories in Linux. It's simple and straightforward with the following syntax.

```
rm [file_to_delete]
```

For example, you are deleting file `another_test.fasta`

```
rm another_test.fasta
```

To delete directories, use additional flags

```
rm -rf [directory_to_delete]
```

The flag `-r` means that it does something recursive, which means that it deletes all files and subdirectories of the directory you want to delete. The flag `f` can help us delete some protected files and directories that you should think twice before deleting.

For example you want to delete `03_adapter_trimming` directory

```
rm -rf 03_adapter_trimming
```

Or delete subdirectory `01_Rawdata_new` by

```
rm -rf 01_Rawdata/01_Rawdata_new
```

Don't worry~ the `01_Rawdata` is still with us

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls -l
total 20
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 22:33 01_Rawdata
drwxr-xr-x 2 jiratchaya jiratchaya 4096 Mar  1 21:59 02_QC
-rw-r--r-- 1 jiratchaya jiratchaya   38 Mar  1 22:00 another_test.fasta
-rw-r--r-- 1 jiratchaya jiratchaya   38 Mar  1 21:59 test.fasta
-rw-r--r-- 1 jiratchaya jiratchaya   38 Mar  1 22:00 test_2.fasta
```

> **Danger zone:** Be sure to check the path of the location where you want to delete something with the command `rm -rf`, otherwise you'll unintentionally delete necessary files or directories.

Downloading file from URL

There are numerous ways to download a file from a URL via the command line on Linux, and two of the best tools for this task are `wget` and `curl`. Both tools have their advantages and disadvantages, depending on the download task at hand. However, in this workshop we'll mainly focus on downloading with `curl`.

For example, we want to download the latest (draft) genome assembly report of *Cyanophora paradoxa* from the NCBI genome database via `curl` as follows.

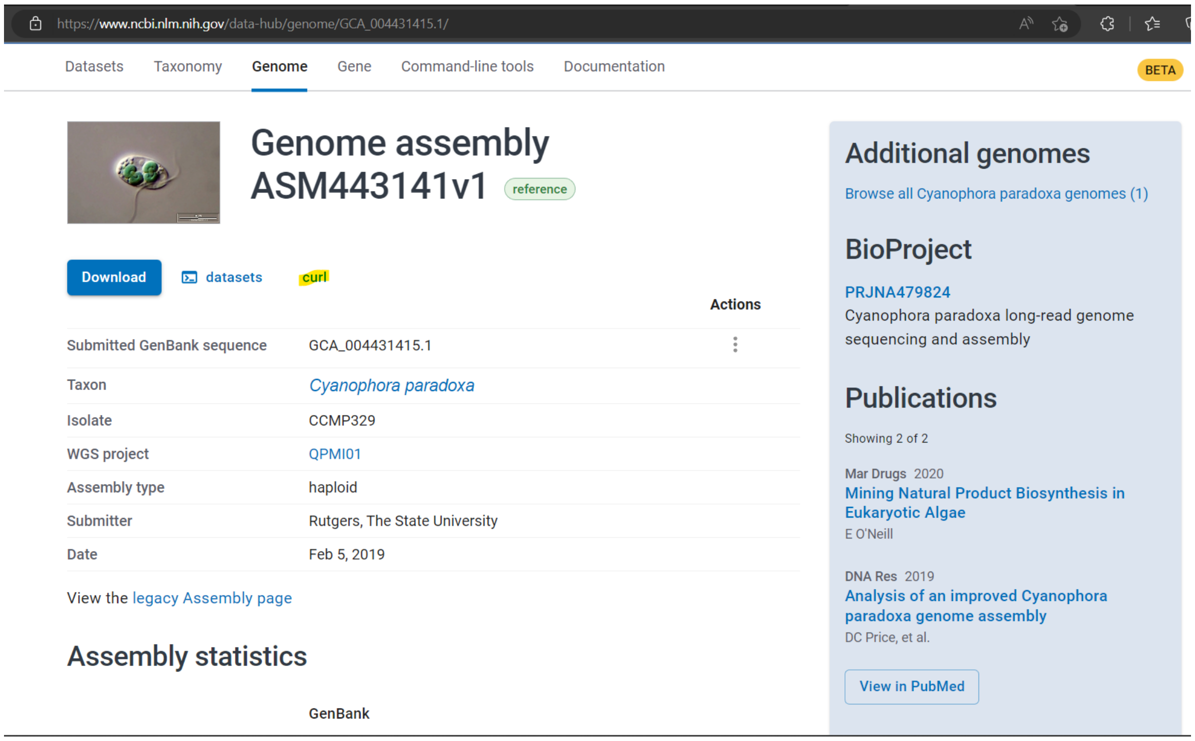
```
curl -O https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/431/415/GCA_004431415.1_ASM443141
```

Expected output

Figure 2.3: Remove everything with sudo privilege. From meme-arsenal.

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ curl -O https://ftp.ncbi.nlm.nih.gov/genomes/all/GC
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 61357  100 61357    0     0  21604      0  0:00:02  0:00:02 --:--:-- 21604
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ ls -l
total 80
drwxr-xr-x 2 jiratchaya jiratchaya  4096 Mar  1 22:33 01_Rawdata
drwxr-xr-x 2 jiratchaya jiratchaya  4096 Mar  1 21:59 02_QC
-rw-r--r-- 1 jiratchaya jiratchaya 61357 Mar  1 22:56 GCA_004431415.1_ASM443141v1_assembly_r
-rw-r--r-- 1 jiratchaya jiratchaya    38 Mar  1 22:00 another_test.fasta
-rw-r--r-- 1 jiratchaya jiratchaya    38 Mar  1 21:59 test.fasta
-rw-r--r-- 1 jiratchaya jiratchaya    38 Mar  1 22:00 test_2.fasta
```

Tips: The alternative way to retrieve genome information from NCBI, you can just go to NCBI Genome Data Hub and specify species name to get information. NCBI provides several routes to download files including `curl`!



Genome assembly ASM443141v1

Submitted GenBank sequence	GCA_004431415.1
Taxon	*Cyanophora paradoxa*
Isolate	CCMP329
WGS project	QPMI01
Assembly type	haploid
Submitter	Rutgers, The State University
Date	Feb 5, 2019

View the legacy Assembly page

Assembly statistics

GenBank

Additional genomes

Browse all Cyanophora paradoxa genomes (1)

BioProject

PRJNA479824
Cyanophora paradoxa long-read genome sequencing and assembly

Publications

Showing 2 of 2

Mar Drugs 2020
Mining Natural Product Biosynthesis in Eukaryotic Algae
E O'Neill

DNA Res 2019
Analysis of an improved Cyanophora paradoxa genome assembly
DC Price, et al.

View in PubMed

Figure 2.4: A genome assembly of *C. paradoxa* in NCBI genome data hub (Accessed: 1 March 2023)

Inspecting file

We'll inspect the assembly report file `GCA_004431415.1_ASM443141v1_assembly_report.txt` that we just downloaded from NCBI

- Count how many lines in that file

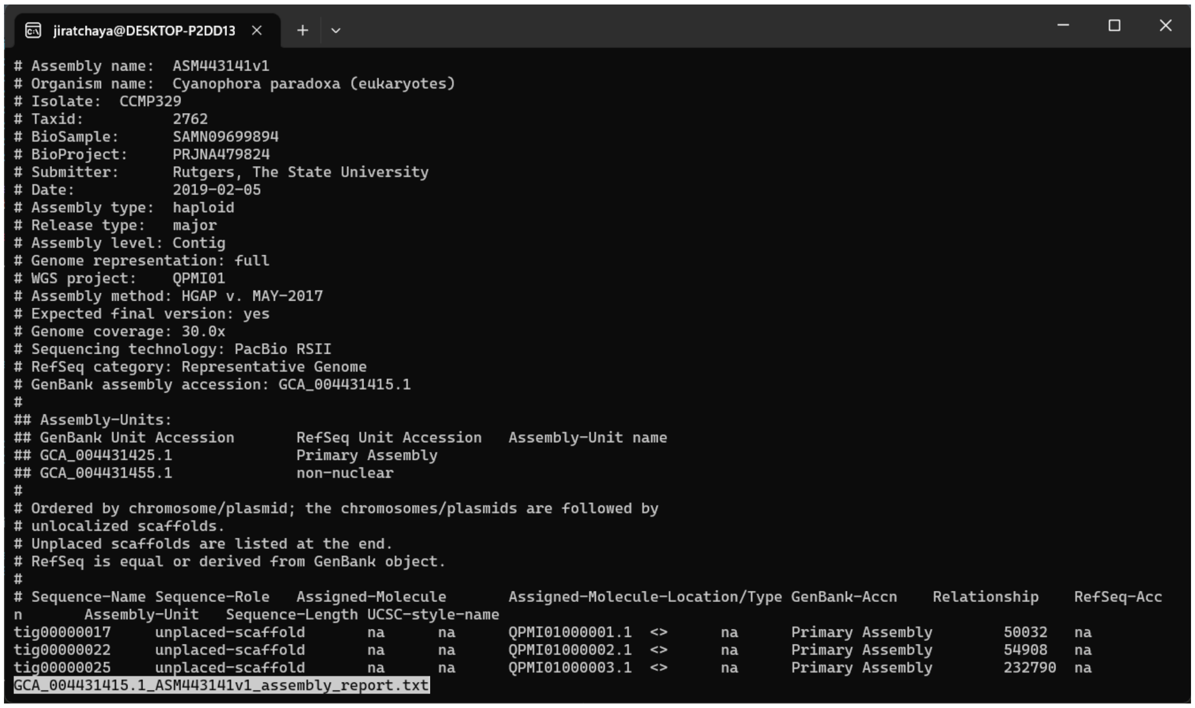
```
wc -l GCA_004431415.1_ASM443141v1_assembly_report.txt
```

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ wc -l GCA_004431415.1_ASM443141v1_assembly_report.t
743 GCA_004431415.1_ASM443141v1_assembly_report.txt
```

- Print some contents at a time.

Now you will see the number of lines that fit on your screen, and you can scroll `up` and `down` with the arrow keys. Then press `q` when you have checked your file.

```
less GCA_004431415.1_ASM443141v1_assembly_report.txt
```

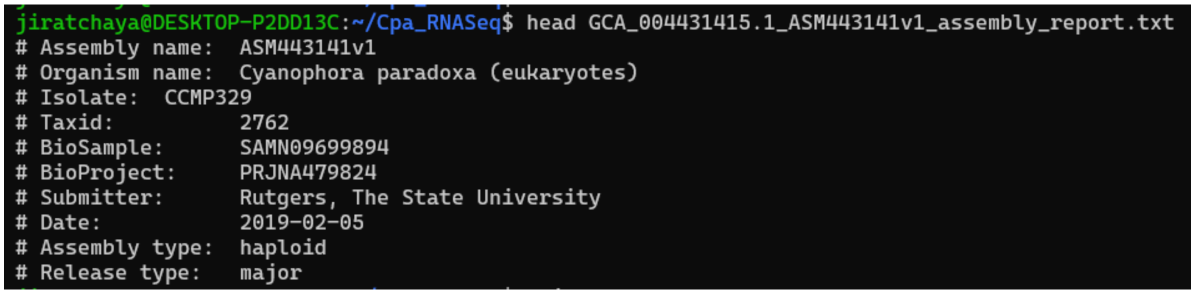


```
# Assembly name:  ASM443141v1
# Organism name:  Cyanophora paradoxa (eukaryotes)
# Isolate:  CCMP329
# Taxid:          2762
# BioSample:      SAMN09699894
# BioProject:     PRJNA479824
# Submitter:      Rutgers, The State University
# Date:           2019-02-05
# Assembly type:  haploid
# Release type:   major
# Assembly level: Contig
# Genome representation: full
# WGS project:    QPMI01
# Assembly method: HGAP v. MAY-2017
# Expected final version: yes
# Genome coverage: 30.0x
# Sequencing technology: PacBio RSII
# RefSeq category: Representative Genome
# GenBank assembly accession: GCA_004431415.1
#
## Assembly-Units:
## GenBank Unit Accession       RefSeq Unit Accession   Assembly-Unit name
## GCA_004431425.1              Primary Assembly
## GCA_004431455.1              non-nuclear
#
# Ordered by chromosome/plasmid; the chromosomes/plasmids are followed by
# unlocalized scaffolds.
# Unplaced scaffolds are listed at the end.
# RefSeq is equal or derived from GenBank object.
#
# Sequence-Name Sequence-Role   Assigned-Molecule       Assigned-Molecule-Location/Type GenBank-Accn    Relationship    RefSeq-Acc
n       Assembly-Unit   Sequence-Length UCSC-style-name
tig00000017     unplaced-scaffold       na      na      QPMI01000001.1  <>      na      Primary Assembly        50032   na
tig00000022     unplaced-scaffold       na      na      QPMI01000002.1  <>      na      Primary Assembly        54908   na
tig00000025     unplaced-scaffold       na      na      QPMI01000003.1  <>      na      Primary Assembly        232790  na
GCA_004431415.1_ASM443141v1_assembly_report.txt
```

Figure 2.5: Example of inspecting a file with the `less` command. Users can scroll up and down with the arrow keys and exit by pressing `q`.

- Print top 10 lines of file

```
head GCA_004431415.1_ASM443141v1_assembly_report.txt
```

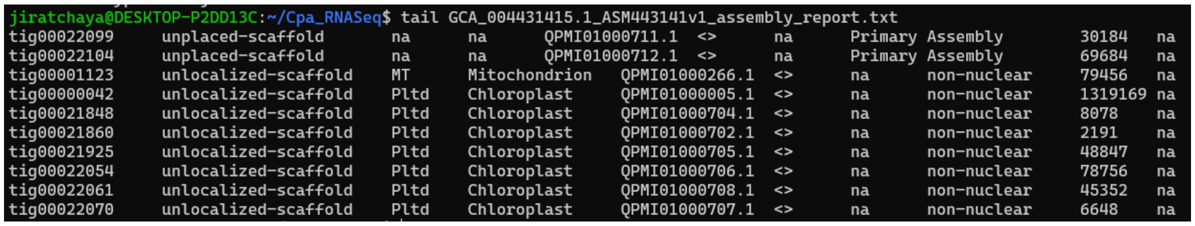


```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ head GCA_004431415.1_ASM443141v1_assembly_report.txt
# Assembly name:  ASM443141v1
# Organism name:  Cyanophora paradoxa (eukaryotes)
# Isolate:  CCMP329
# Taxid:          2762
# BioSample:      SAMN09699894
# BioProject:     PRJNA479824
# Submitter:      Rutgers, The State University
# Date:           2019-02-05
# Assembly type:  haploid
# Release type:   major
```

Figure 2.6: The first 10 lines of *C. paradoxa* assembly report file

- Print bottom 10 lines of file

```
tail GCA_004431415.1_ASM443141v1_assembly_report.txt
```



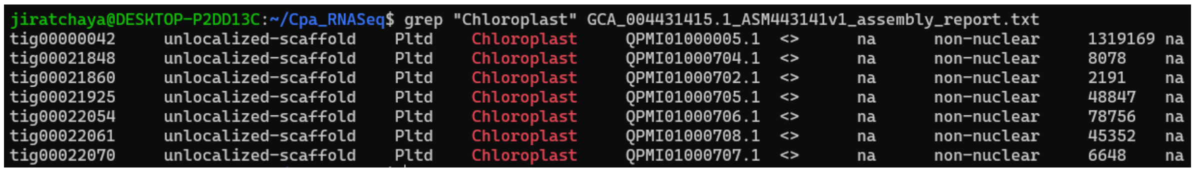
```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ tail GCA_004431415.1_ASM443141v1_assembly_report.txt
tig00022099     unplaced-scaffold       na      na      QPMI01000711.1  <>      na      Primary Assembly        30184   na
tig00022104     unplaced-scaffold       na      na      QPMI01000712.1  <>      na      Primary Assembly        69684   na
tig00001123     unlocalized-scaffold    MT      Mitochondrion   QPMI01000266.1  <>      na      non-nuclear     79456   na
tig00000042     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000005.1  <>      na      non-nuclear     1319169 na
tig00021848     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000704.1  <>      na      non-nuclear     8078    na
tig00021860     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000702.1  <>      na      non-nuclear     2191    na
tig00021925     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000705.1  <>      na      non-nuclear     48847   na
tig00022054     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000706.1  <>      na      non-nuclear     78756   na
tig00022061     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000708.1  <>      na      non-nuclear     45352   na
tig00022070     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000707.1  <>      na      non-nuclear     6648    na
```

Figure 2.7: The last 10 lines of *C. paradoxa* assembly report file.

- Print only lines with a specific pattern of word.

For example, we'll print only the lines contain the word "Chloroplast"

```
grep "Chloroplast" GCA_004431415.1_ASM443141v1_assembly_report.txt
```



```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ grep "Chloroplast" GCA_004431415.1_ASM443141v1_assembly_report.txt
tig00000042     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000005.1  <>      na      non-nuclear     1319169 na
tig00021848     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000704.1  <>      na      non-nuclear     8078    na
tig00021860     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000702.1  <>      na      non-nuclear     2191    na
tig00021925     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000705.1  <>      na      non-nuclear     48847   na
tig00022054     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000706.1  <>      na      non-nuclear     78756   na
tig00022061     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000708.1  <>      na      non-nuclear     45352   na
tig00022070     unlocalized-scaffold    Pltd    Chloroplast     QPMI01000707.1  <>      na      non-nuclear     6648    na
```

Figure 2.8: Extracted lines with a specific word "Chloroplast" in assembly report file.

Show latest commands we used

You can simply press arrow keys `up` or `down` to see your latest commands that you typed in the terminal.

Another way to see the latest command by typing below in the terminal

```
history
```

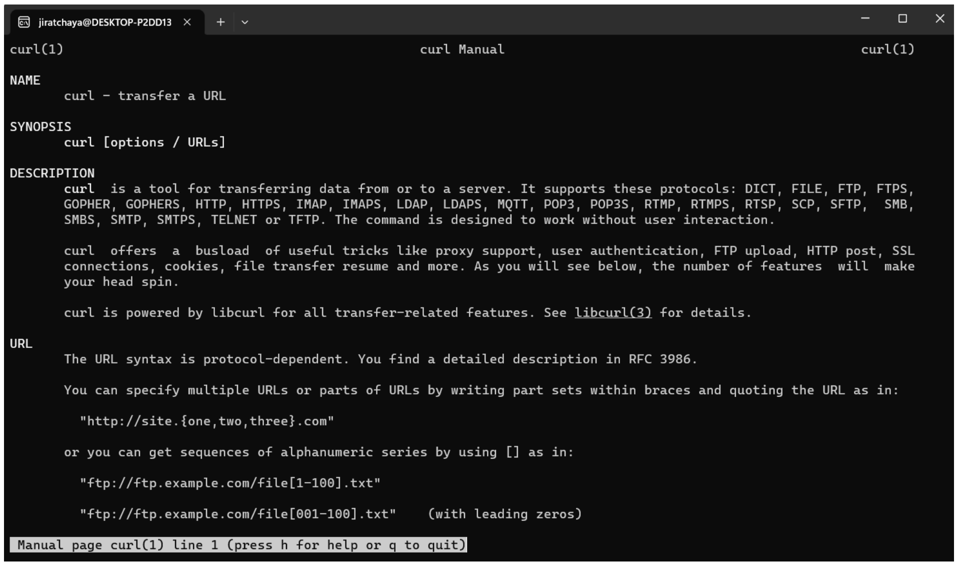
History is able to keep track of the command lines you use, associate any data with each line, and use information from previous lines when writing new lines.

Shortcut: Tab Completion

When typing file or directory names, it's easy to mistype. Instead, we can use 'tab' to complete what we want to type. The shell will try to fill in the rest of a directory or file name if you press tab after typing.

2.2.1 Have no idea what this command can do

Basically, the built-in Linux system commands store usage in their command manual. You can call `man` followed by a command you want to learn more about. for example `man curl`.



```
curl(1)                                          curl Manual                                         curl(1)

NAME
       curl - transfer a URL

SYNOPSIS
       curl [options / URLs]

DESCRIPTION
       curl  is a tool for transferring data from or to a server. It supports these protocols: DICT, FILE, FTP, FTPS,
       GOPHER, GOPHERS, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, MQTT, POP3, POP3S, RTMP, RTMPS, RTSP, SCP, SFTP,  SMB,
       SMBS, SMTP, SMTPS, TELNET or TFTP. The command is designed to work without user interaction.

       curl  offers  a  busload  of useful tricks like proxy support, user authentication, FTP upload, HTTP post, SSL
       connections, cookies, file transfer resume and more. As you will see below, the number of features  will  make
       your head spin.

       curl is powered by libcurl for all transfer-related features. See libcurl(3) for details.

URL
       The URL syntax is protocol-dependent. You find a detailed description in RFC 3986.

       You can specify multiple URLs or parts of URLs by writing part sets within braces and quoting the URL as in:

         "http://site.{one,two,three}.com"

       or you can get sequences of alphanumeric series by using [] as in:

         "ftp://ftp.example.com/file[1-100].txt"

         "ftp://ftp.example.com/file[001-100].txt"    (with leading zeros)

 Manual page curl(1) line 1 (press h for help or q to quit)
```

Figure 2.9: A manual page of `curl`. Users can scroll using arrow keys up and down, and quit reading by press `q`.

2.3 Maintaining Long-Running Jobs with tmux

When we run programs through the Unix shell, they run until they terminate successfully or are terminated with an error. Multiple processes running simultaneously on your computer, such as system files, web browser, email application, bioinformatics programs, and so on. In bioinformatics, we often work with processes that run for a long period of time. Therefore, it's important that we know how to work with processes and manage them using the Unix shell. In this section, we'll learn the basics of dealing with processes.

In addition, processes are also terminated if the connection to the servers is interrupted, the network connection drops immediately, or the power fails. Since we're constantly working with remote computers in our daily work in bioinformatics, we need a way to prevent the accidental termination of long-running applications. Leaving the local terminal's connection to a remote computer open while a program is running is an unsafe solution, even the most reliable networks can experience short outages.

Some software offers the user the possibility to run their work as a background process, e.g. Nohup, Screen and Tmux. In this workshop, we propose **Terminal Multiplexer (Tmux)**, which allows you to create a session with multiple windows, each of which can run its own processes. The Tmux sessions are persistent, which means that all the windows and their processes can be easily restored by reattaching the session.

When Tmux is running on a remote machine, you can maintain a persistent session that isn't lost when the connection drops or you close your terminal window to go home (or even exit your terminal programme). Rather, all Tmux sessions can be reattached to the terminal you're currently at - simply log back into the remote machine via SSH and reattach the Tmux session. All windows remain undisturbed and all processes continue to run.

A simple usage of Tmux

Open a terminal and use the following command

```
tmux
```

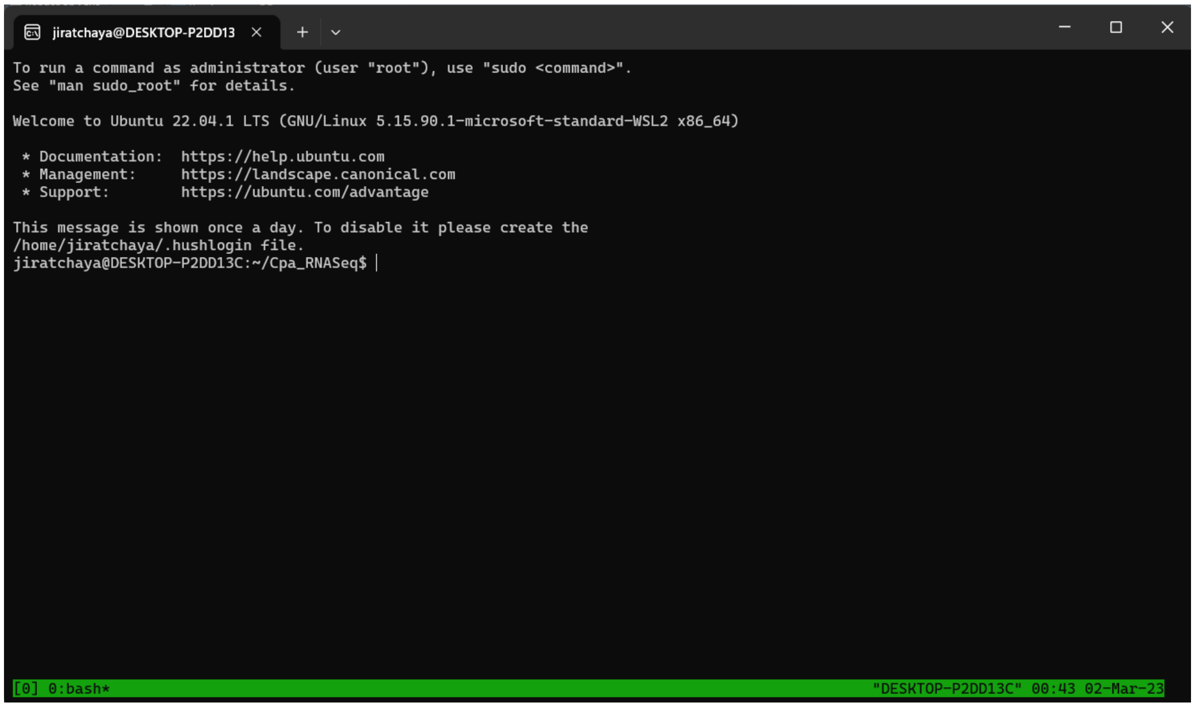
You see a command prompt as usual, but you now see a taskbar-style menu at the bottom of the terminal that contains something like bash 0 *. The asterisk indicates that this is your active window.

Detach a session

This allows you to leave the tmux session, but it continues to run in the background. Just press the key

This is Jimmy. Jimmy makes web applications. He does a good job, but he's a little socially awkward. You see, he's always fumbling around for his mouse, constantly switching between multiple terminal windows and his clunky GUI-based IDE.
This is Billy. Billy also makes web applications. But Billy uses tmux, the terminal multiplexer, which allows him to manage his code and his consoles all in the same window. He's able to move quickly and effortlessly through his projects, never taking his hands off the keyboard.
Because he's so productive at what he does, he has plenty of extra time to play golf... attend social events... and of course, take Mary-Anne out to the malt shoppe for date night.
Poor Jimmy doesn't have time for date night. He's still looking for his mouse.
Get Productive. Get tmux.

Figure 2.10: How tmux increase you pruductivity :/ (From Billy uses tmux in Reddit)



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.90.1-microsoft-standard-WSL2 x86_64)

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

This message is shown once a day. To disable it please create the
/home/jiratchaya/.hushlogin file.
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ |

[0] 0:bash*                                                        "DESKTOP-P2DD13C" 00:43 02-Mar-23
```

Figure 2.11: Tmux windows

```
[ctrl + b] + d
```

Your terminal will print

```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ tmux
[detached (from session 0)]
```

This should take you back to a standard prompt. Remember that the Tmux session continues in the background, and you can recall it at any time.

Name the Tmux session

You may find it helpful to name your sessions with meaningful titles to keep things organized. Let's try naming your first session with Tmux.

You can name it anything that we want, but in this case I will name it 'process2'. Enter the following command:

```
tmux new -s process2
```

You should now have a new Tmux session running. If you look in the lower left area of the window, you will see the name of your session rather than the generic 'bash'.

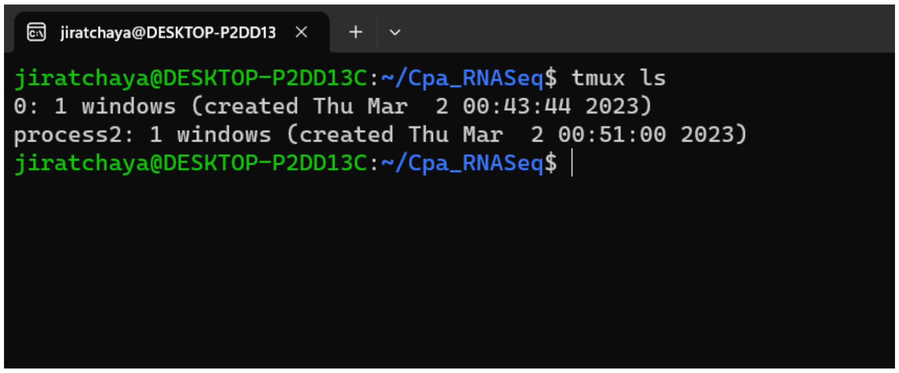
List tmux sessions

What happened to your session? It is still running in the background. You can reopen the session by name or number ID, but what if you forgot the session name?

There is a list function built into tmux:

```
tmux ls
```

This lists all your current tmux sessions. When you run it, you get output like this:



```
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$ tmux ls
0: 1 windows (created Thu Mar  2 00:43:44 2023)
process2: 1 windows (created Thu Mar  2 00:51:00 2023)
jiratchaya@DESKTOP-P2DD13C:~/Cpa_RNASeq$
```

Figure 2.12: List of running tmux sessions.

Reenter (aka reattach) a session in Tmux

To reopen your tmux session, you can use the tmux command with the attach or attach-session option as follows:

```
tmux a -t [session_name]
```

For example, we'll reenter to the process2 session.

```
tmux a -t process2
```

Exit tmux when finish running

Quitting tmux is exactly the same as quitting the standard terminal by pressing the keys `Ctrl+d` or by entering

```
exit
```

2.4 Resources

- Buffalo, V. (2015). *Bioinformatics data skills: Reproducible and robust research with open source tools*. " O'Reilly Media, Inc.".
- Introduction to the command line interface by Harvard Chan Bioinformatics Core (Accessed on 27 Feb 2023).
- Introducing the Shell, from the course Introduction to the Command Line for Genomics in bioinformatics-core-shared-training (Accessed on 28 Feb 2023)
- Bash cheat sheet from RehanSaeed GitHub repository (Accessed on 1 March 2023).
- Getting Started with Tmux [Beginner's Guide]. By linuxhandbook.com (Accessed on 2 March 2023)

3 Data Retrieval with NCBI SRA Toolkit

NCBI (Natianal Center for Biotechnology Information) is a major source of biological databases related to life and health sciences research, as well as a major source of bioinformatics tools and services. NCBI hosts various types of biological data submitted by researchers from around the world, such as GenBank for nucleotide sequence submissions, Sequence Read Archive (SRA) for raw sequence data, Genome for submitting full or draft genomes, Gene Expression Omnibus (GEO) for quantitative gene expression data sets, and many more.

NCBI SRA toolkit is a set of utilities for downloading, viewing, and searching large amounts of high-throughput sequencing data from the NCBI SRA database.

SRA toolkit can

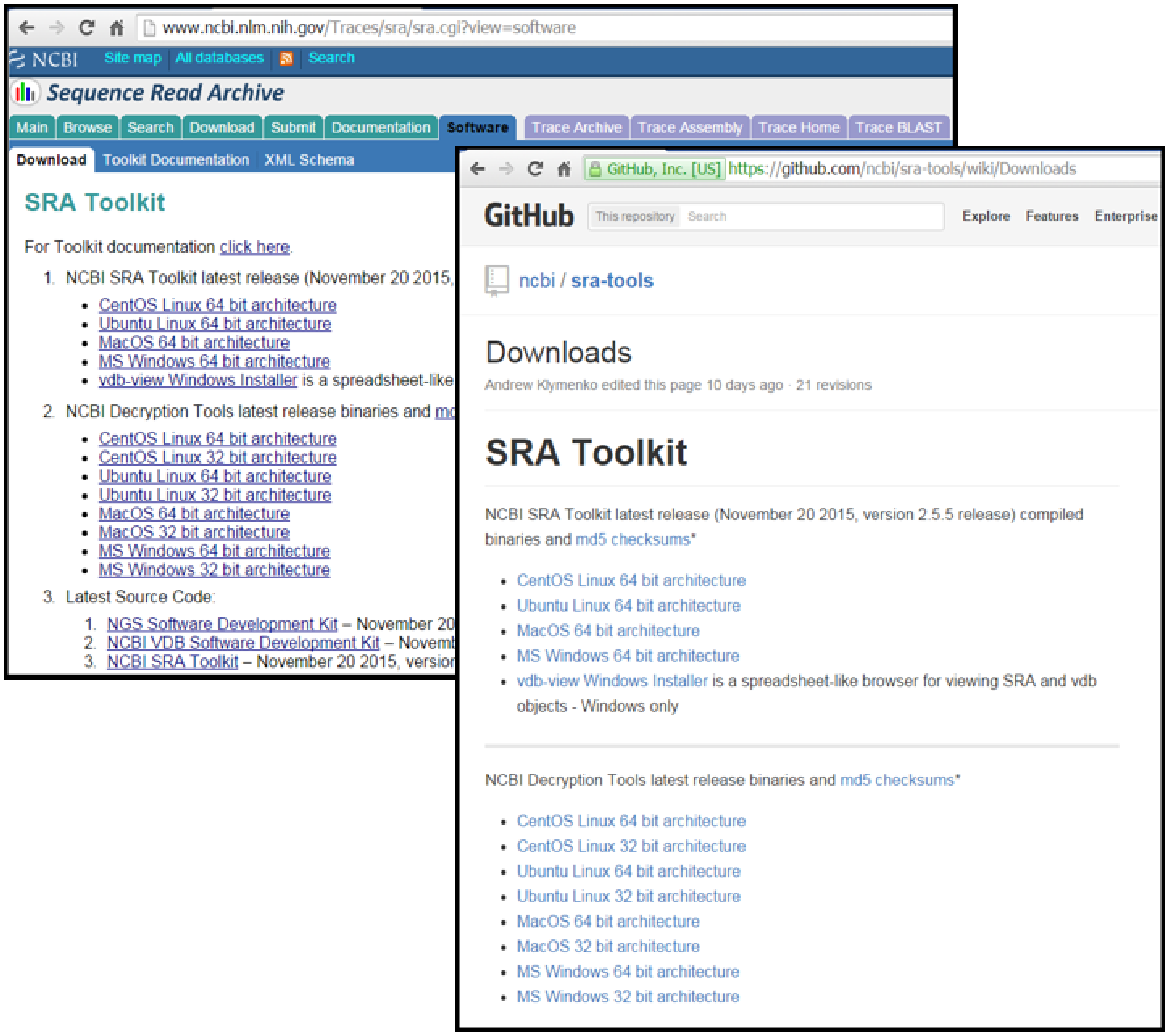
- Effectively download the large volume of high-throughput sequencing data
- Convert SRA file into other biological file format
- Retrieve a small subset of large files
- Search within SRA files and fetch specific sequences

3.1 What is Sequence Read Archives (SRA)

The Sequence Read Archive (SRA) is the largest publicly accessible repository for high-throughput sequencing data. SRA accepts data from all areas of sequencing projects as well as metagenomics and environmental studies. Sequencing data may be isolated from a single species or from multiple species as in metagnomics studies.

SRA also refers in the file description to the format defined by NCBI for NGS data in the SRA database. All data submitted to NCBI must be stored in SRA format and can be converted back to a FASTQ, FASTA, or BAM file depending on the original submission by the researchers. Here, the SRA Toolkit provides tools for downloading data, converting various data formats to SRA format and vice versa, and extracting SRA data to other formats.

Researchers often use SRA data to make discoveries and conduct reproducible research. Data sets can be compared using the SRA web interface. However, if you want to download files for local use on your computer, you should use a command line interface, and the SRA Toolkit is highly recommended by NCBI.



SRA Toolkit

For Toolkit documentation click here.

1. NCBI SRA Toolkit latest release (November 20 2015,
   - CentOS Linux 64 bit architecture
   - Ubuntu Linux 64 bit architecture
   - MacOS 64 bit architecture
   - MS Windows 64 bit architecture
   - vdb-view Windows Installer is a spreadsheet-like
2. NCBI Decryption Tools latest release binaries and mo
   - CentOS Linux 64 bit architecture
   - CentOS Linux 32 bit architecture
   - Ubuntu Linux 64 bit architecture
   - Ubuntu Linux 32 bit architecture
   - MacOS 64 bit architecture
   - MacOS 32 bit architecture
   - MS Windows 64 bit architecture
   - MS Windows 32 bit architecture
3. Latest Source Code:
   1. NGS Software Development Kit – November 20
   2. NCBI VDB Software Development Kit – Novemb
   3. NCBI SRA Toolkit – November 20 2015, versior

Downloads

Andrew Klymenko edited this page 10 days ago · 21 revisions

SRA Toolkit

NCBI SRA Toolkit latest release (November 20 2015, version 2.5.5 release) compiled binaries and md5 checksums*

- CentOS Linux 64 bit architecture
- Ubuntu Linux 64 bit architecture
- MacOS 64 bit architecture
- MS Windows 64 bit architecture
- vdb-view Windows Installer is a spreadsheet-like browser for viewing SRA and vdb objects - Windows only

NCBI Decryption Tools latest release binaries and md5 checksums*

- CentOS Linux 64 bit architecture
- CentOS Linux 32 bit architecture
- Ubuntu Linux 64 bit architecture
- Ubuntu Linux 32 bit architecture
- MacOS 64 bit architecture
- MacOS 32 bit architecture
- MS Windows 64 bit architecture
- MS Windows 32 bit architecture

Figure 3.1: Screenshots of NCBI Sequence Read Archives