Handling Scientific Experiments with HPC Clusters and Slurm

Antonio Emanuele Cinà
Assistant Professor @ University of Genoa, Italy
January 15, 2024

Seminar for the course **“Deep Learning and Computer Vision with PyTorch”**, University of Cagliari

Contents of this seminar

The Usenix Shell	What is the **Shell**? Basic Commands
Reproducible Python Environments	Creating reproducible environments with **Conda**
Working on HPC Clusters using SLURM	What is Slurm? **HPC** Introduction and **Slurm Scheduler**
Practical Insights	How can I put everything into **practice**?

The teaching material have been taken from:

Shell - Stanford CS Course | Conda Tutorials| HPC with Slurm - University of Cambridge

The Usenix Shell

The Shell

The **shell** is a program, alternative to the classical GUI, where users can type **commands**.

Using the shell will take some effort and some time to learn. You must learn a few commands.

Conversely, a **GUI** presents you with choices to select, automatically hiding commands.

The grammar of a shell allows you to combine existing tools into **powerful pipelines** and handle large volumes of data automatically.

Shell vs GUI

With a GUI, we give **instructions** by clicking a mouse and using menu-driven interactions.

While the visual aid of a GUI makes it intuitive to learn, this way of delivering instructions to a computer scales very poorly.

Imagine the following task: for a literature search, you have to copy the third line of one thousand text files in one thousand different directories and paste it into a single file.

Using a GUI, you would not only be clicking at your desk for **several hours**, but you could potentially also commit an error in the process of completing this repetitive task.

The shell allows such repetitive tasks to be done **automatically** and **fast**.

The Shell

Windows has two different CLIs installed by default, the Command Line Prompt (CMD) and Windows Powershell. Both are fine, but the power shell gives more of an shell feeling.

MacOs has by default Bash (MacOs Catalina has Zsh) accessible by using the Terminal application.

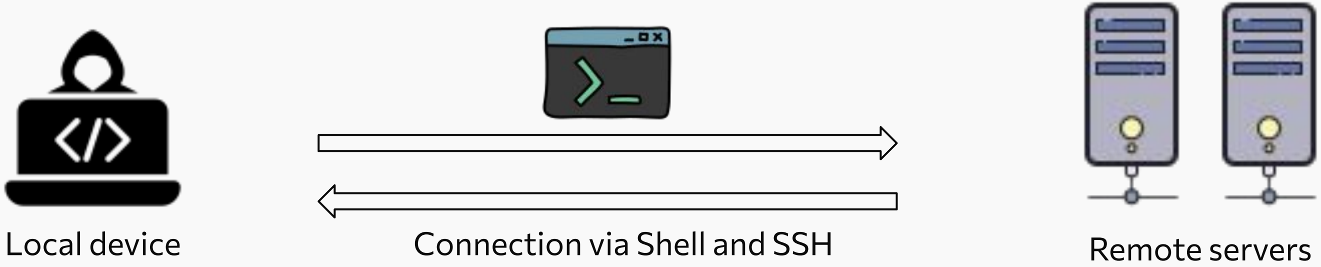
Linux users are probably already familiar with a shell. Which shell and terminal application is installed, depends on the installed distribution.

Shell for Scientific Experiments

The command line is often the easiest way to interact with **remote machines**.

Familiarity with the shell is near essential to run a variety of specialized tools and resources including **high-performance computing systems**.

As clusters and cloud computing systems become more popular for **scientific experiments**, being able to interact with the shell is becoming a necessary skill.



Local device

Connection via Shell and SSH

Remote servers

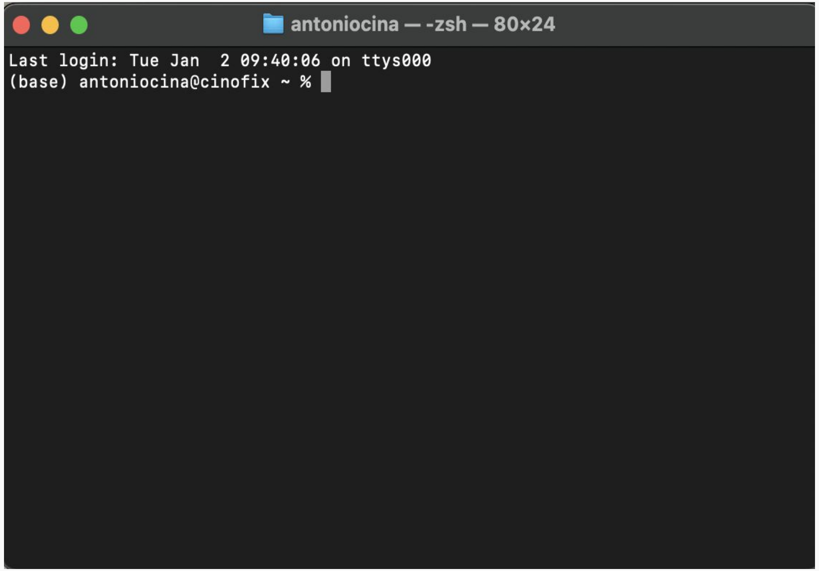
Meet the Unix Shell

The shell began with the UNIX OS in 1969.

Open the **Termina**l application; on macOS it's located in the Utilities folder of *Applications*, on Windows it's in your *start menu* (it might be called Ubuntu), and on Linux it'll be in your desktop environment's normal app launcher.

Bash is a Unix shell and command language that is the default login shell for most Linux and MacOS.

Interpreted, not compiled.



```
Last login: Tue Jan  2 09:40:06 on ttys000
(base) antoniocina@cinofix ~ %
```

Meet the Unix Shell

The shell is a text-based interface that takes **commands** instead of clicks

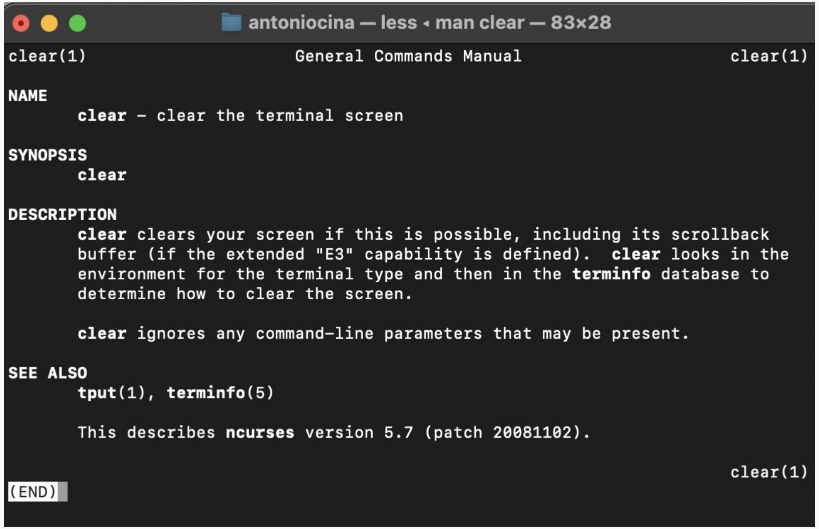
Commands are **pre-existing programs**:

<command name> <options> <input || output>

To know about an individual command use man:

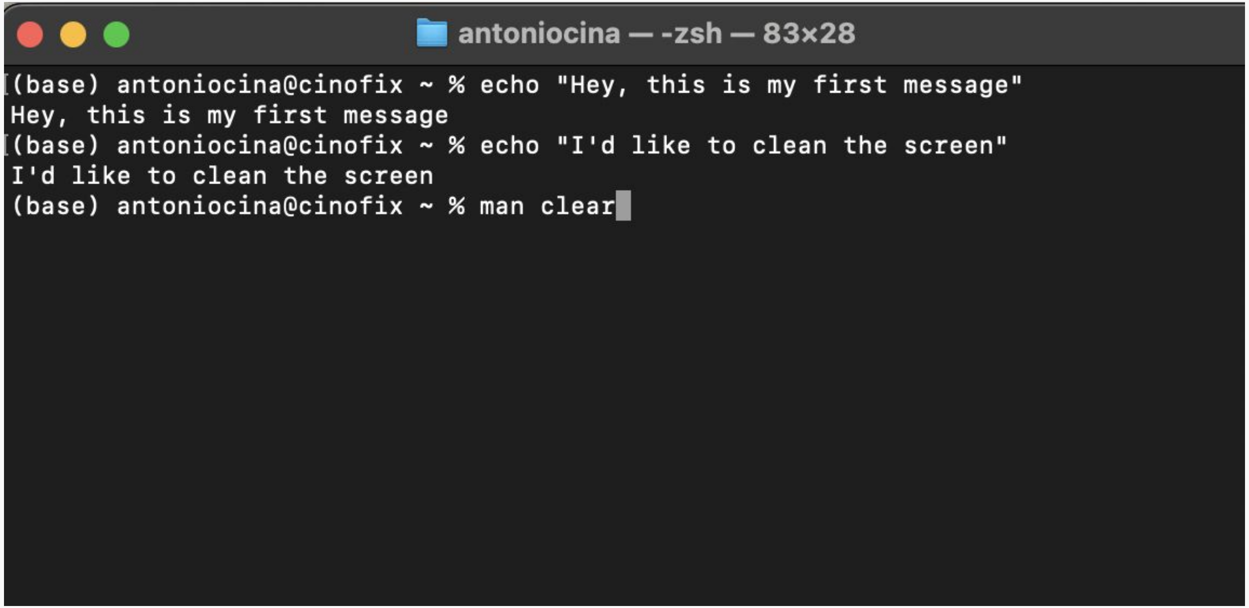
<command name> man

Short for manual page, or we can also use the --help option



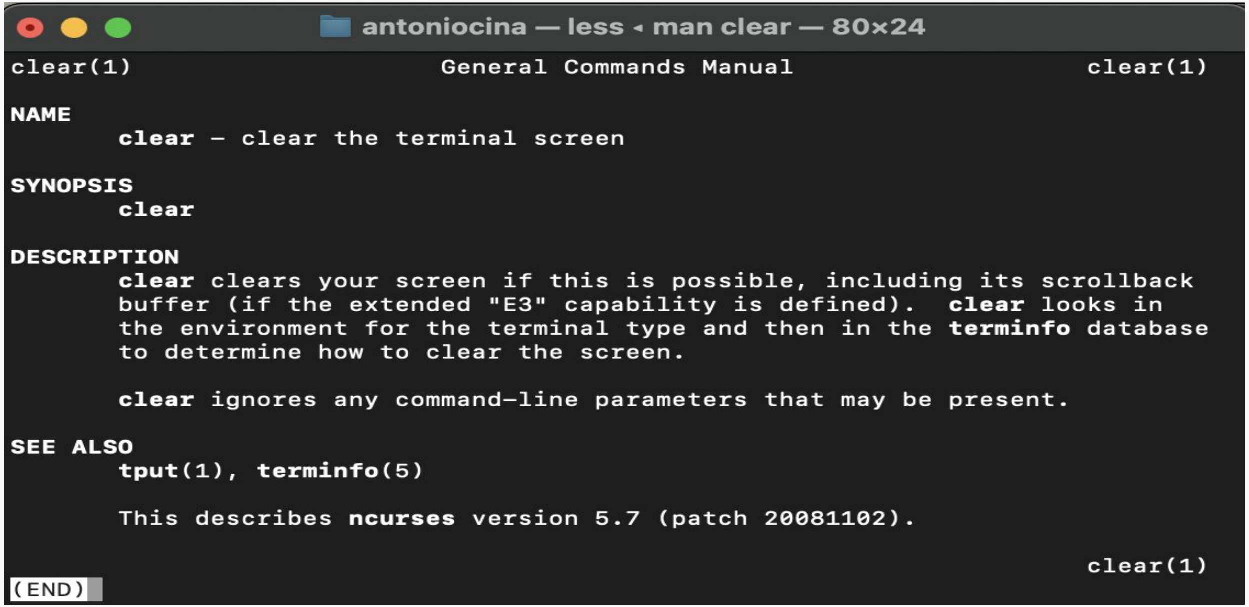
```
clear(1)                   General Commands Manual                  clear(1)

NAME
       clear - clear the terminal screen

SYNOPSIS
       clear

DESCRIPTION
       clear clears your screen if this is possible, including its scrollback
       buffer (if the extended "E3" capability is defined).  clear looks in the
       environment for the terminal type and then in the terminfo database to
       determine how to clear the screen.

       clear ignores any command-line parameters that may be present.

SEE ALSO
       tput(1), terminfo(5)

       This describes ncurses version 5.7 (patch 20081102).

                                                                    clear(1)
(END)
```

Meet the Unix Shell



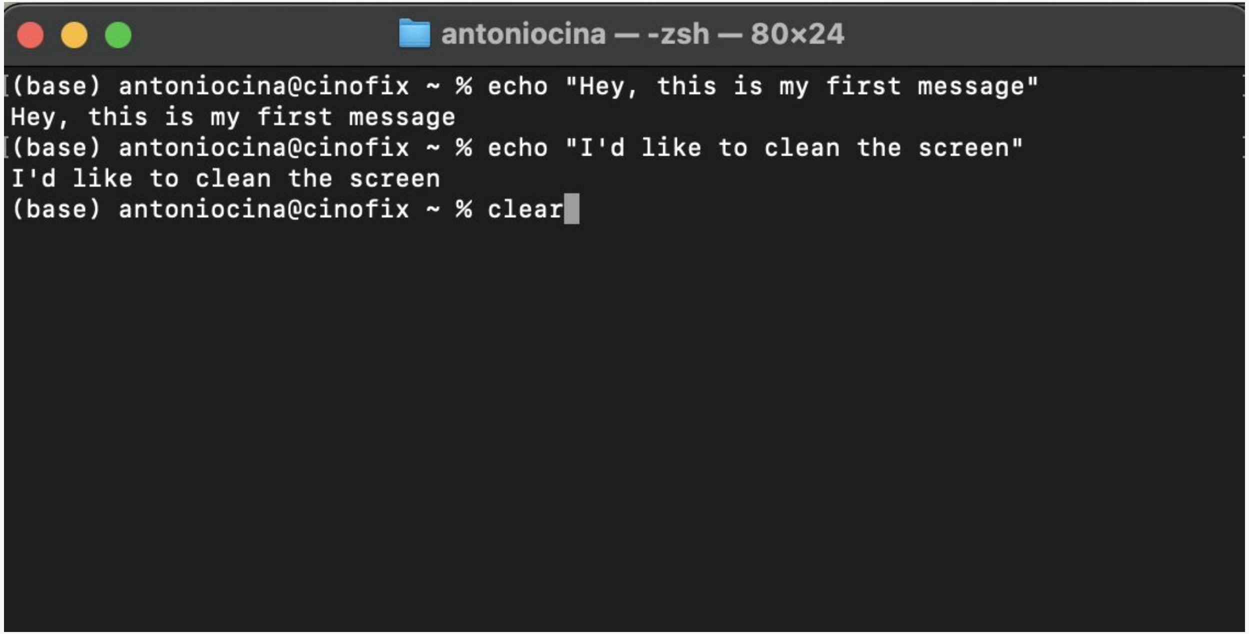
```
(base) antoniocina@cinofix ~ % echo "Hey, this is my first message"
Hey, this is my first message
(base) antoniocina@cinofix ~ % echo "I'd like to clean the screen"
I'd like to clean the screen
(base) antoniocina@cinofix ~ % man clear
```

Meet the Unix Shell



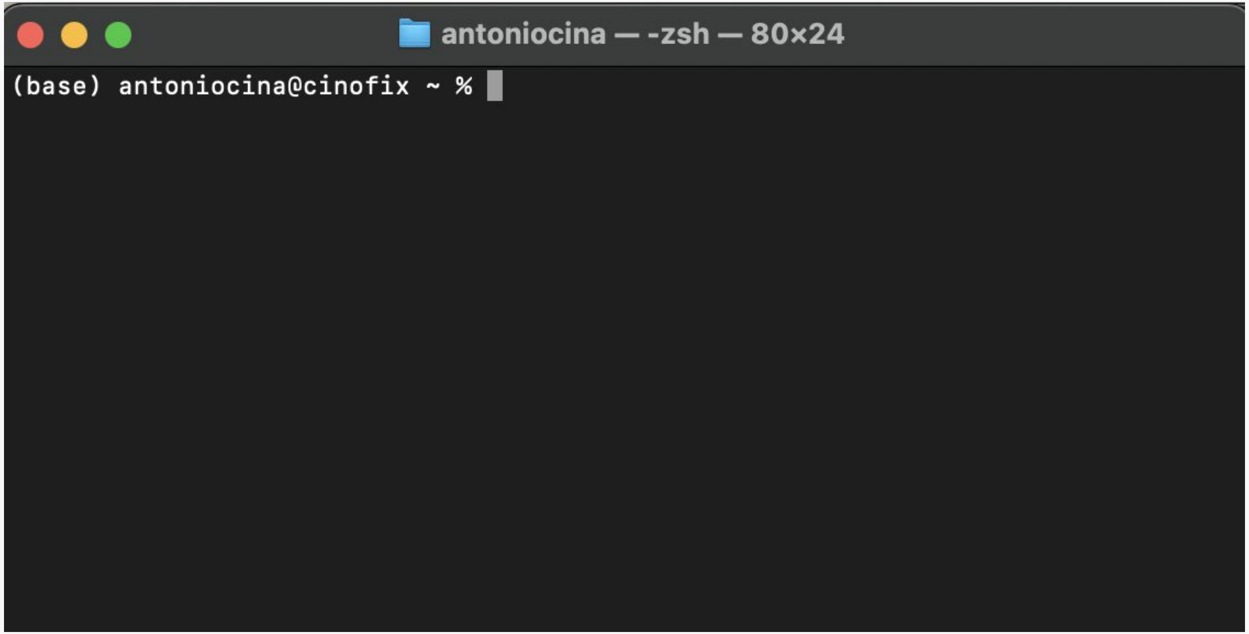
```
clear(1)                    General Commands Manual                   clear(1)

NAME
       clear - clear the terminal screen

SYNOPSIS
       clear

DESCRIPTION
       clear clears your screen if this is possible, including its scrollback
       buffer (if the extended "E3" capability is defined).  clear looks in
       the environment for the terminal type and then in the terminfo database
       to determine how to clear the screen.

       clear ignores any command-line parameters that may be present.

SEE ALSO
       tput(1), terminfo(5)

       This describes ncurses version 5.7 (patch 20081102).

                                                                      clear(1)
(END)
```

Meet the Unix Shell



```
(base) antoniocina@cinofix ~ % echo "Hey, this is my first message"
Hey, this is my first message
(base) antoniocina@cinofix ~ % echo "I'd like to clean the screen"
I'd like to clean the screen
(base) antoniocina@cinofix ~ % clear
```

Meet the Unix Shell

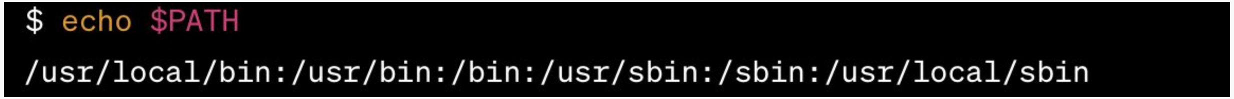


```
(base) antoniocina@cinofix ~ %
```

$PATH

An **environment variable** is a dynamic-named value that can affect the behavior of running processes.

It is part of the environment in which a process runs. The $PATH variable is a notable example, commonly used in Unix-like operating systems.



```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/sbin
```

In this example, $PATH is a colon-separated list of directories.

Users can customize the $PATH variable to include directories where their own executable files are located, ensuring easy access to their custom commands.

Running Programs

We can run a program by typing its **path** into the terminal.

When a command is entered in the shell, the system looks for the corresponding **executable** in these directories in the order specified. If a matching executable is found, it is executed.

To run a program in the current directory you need to give the path

- `$ ./local_program`

Some folders are globally visible, so you only need the program’s name.

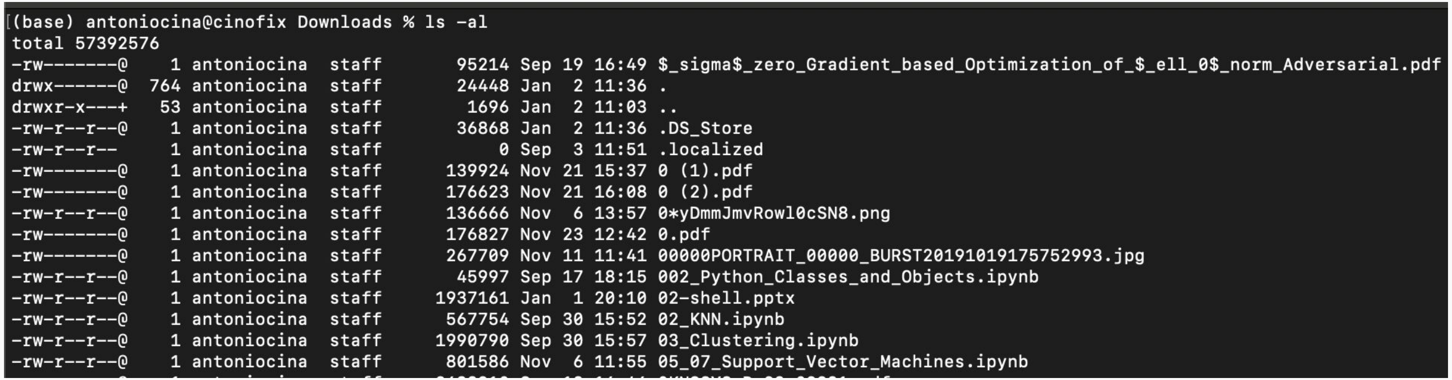
- /bin/ is globally visible because it is in the PATH shell variable
- This allows users to run commands without specifying the full path to the executable, making command execution more convenient and flexible.

All commands are **bash script** that are executed when you hit enter on a terminal line.

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`

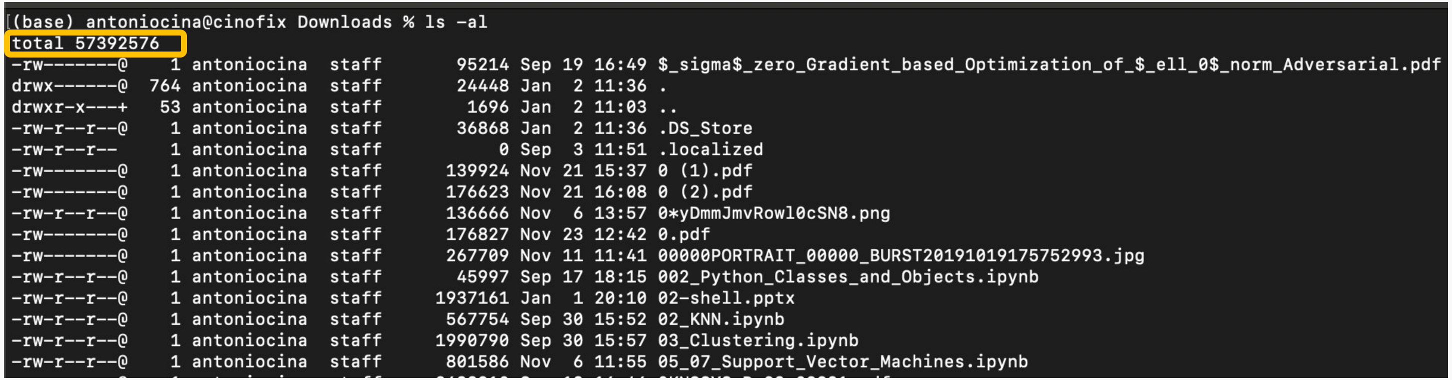


```
(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff       95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff       24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff        1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff       36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff           0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff      139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff      176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@    1 antoniocina  staff      136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@    1 antoniocina  staff      176827 Nov 23 12:42 0.pdf
-rw-------@    1 antoniocina  staff      267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@    1 antoniocina  staff       45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@    1 antoniocina  staff     1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@    1 antoniocina  staff      567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@    1 antoniocina  staff     1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@    1 antoniocina  staff      801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`



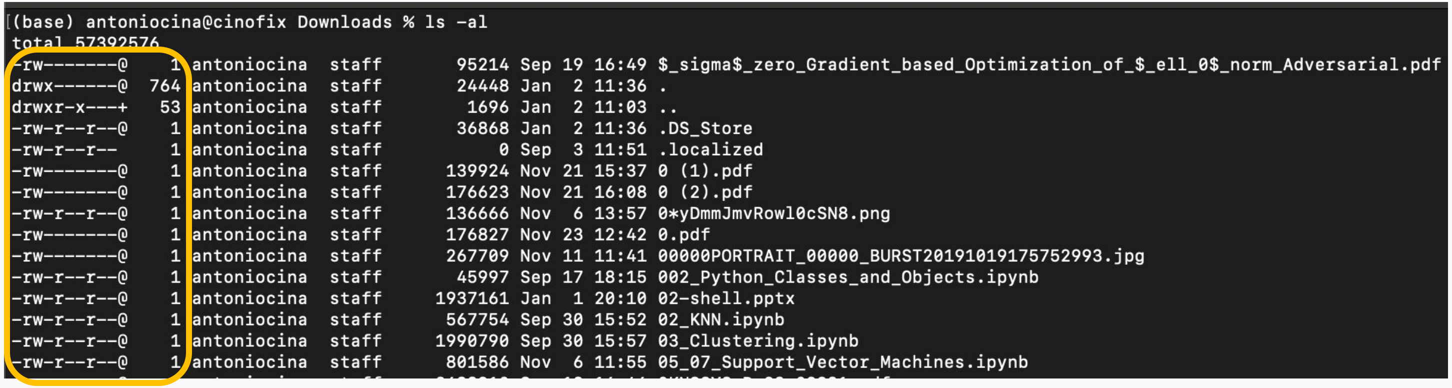
```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@   1 antoniocina  staff      95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@ 764 antoniocina  staff      24448 Jan  2 11:36 .
drwxr-x---+  53 antoniocina  staff       1696 Jan  2 11:03 ..
-rw-r--r--@   1 antoniocina  staff      36868 Jan  2 11:36 .DS_Store
-rw-r--r--    1 antoniocina  staff          0 Sep  3 11:51 .localized
-rw-------@   1 antoniocina  staff     139924 Nov 21 15:37 0 (1).pdf
-rw-------@   1 antoniocina  staff     176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@   1 antoniocina  staff     136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@   1 antoniocina  staff     176827 Nov 23 12:42 0.pdf
-rw-------@   1 antoniocina  staff     267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@   1 antoniocina  staff      45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@   1 antoniocina  staff    1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@   1 antoniocina  staff     567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@   1 antoniocina  staff    1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@   1 antoniocina  staff     801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

Total number of files

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`



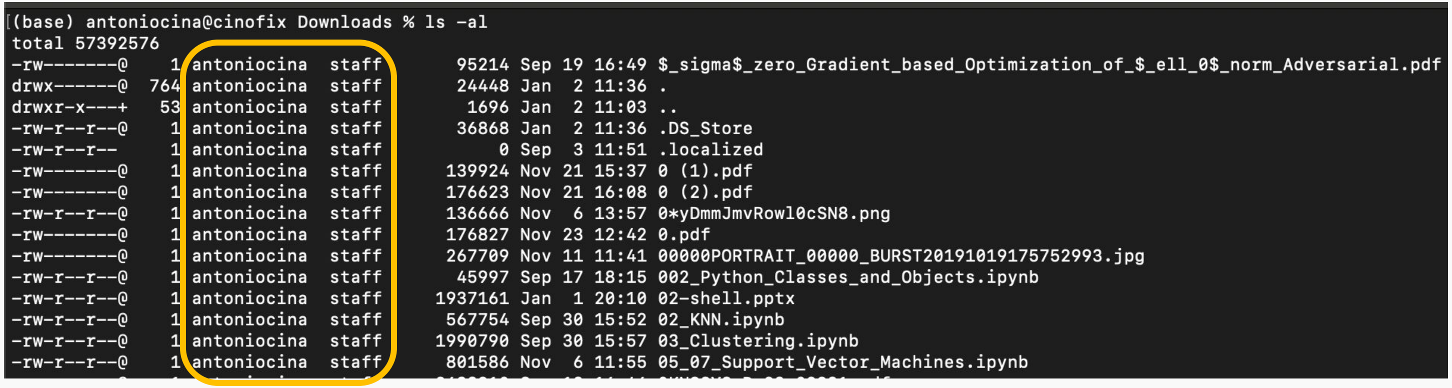
```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff       95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff       24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff        1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff       36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff           0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff      139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff      176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@    1 antoniocina  staff      136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@    1 antoniocina  staff      176827 Nov 23 12:42 0.pdf
-rw-------@    1 antoniocina  staff      267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@    1 antoniocina  staff       45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@    1 antoniocina  staff     1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@    1 antoniocina  staff      567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@    1 antoniocina  staff     1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@    1 antoniocina  staff      801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

Files permissions

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`



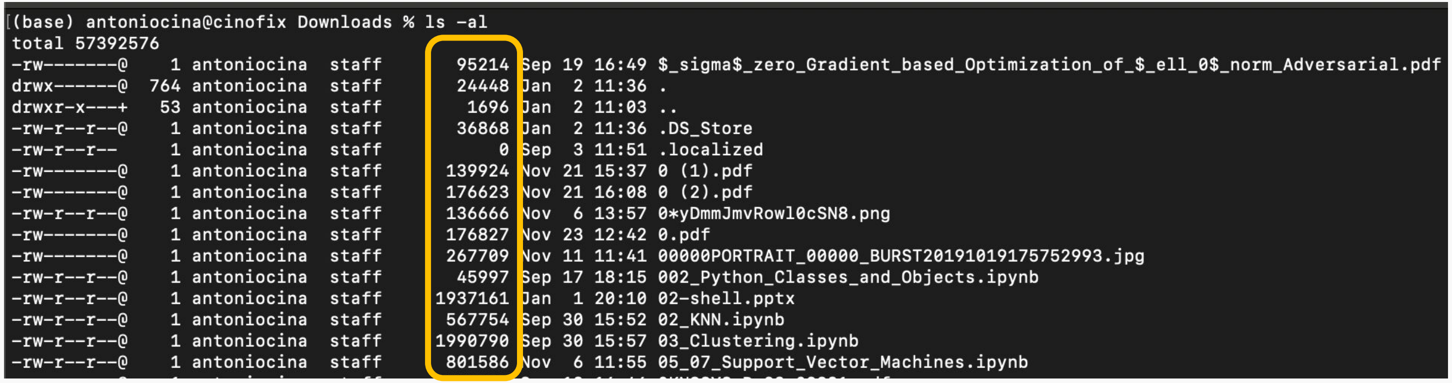
```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff     95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff     24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff      1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff     36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff         0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff    139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff    176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@    1 antoniocina  staff    136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@    1 antoniocina  staff    176827 Nov 23 12:42 0.pdf
-rw-------@    1 antoniocina  staff    267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@    1 antoniocina  staff     45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@    1 antoniocina  staff   1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@    1 antoniocina  staff    567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@    1 antoniocina  staff   1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@    1 antoniocina  staff    801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

File owner and File group

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`



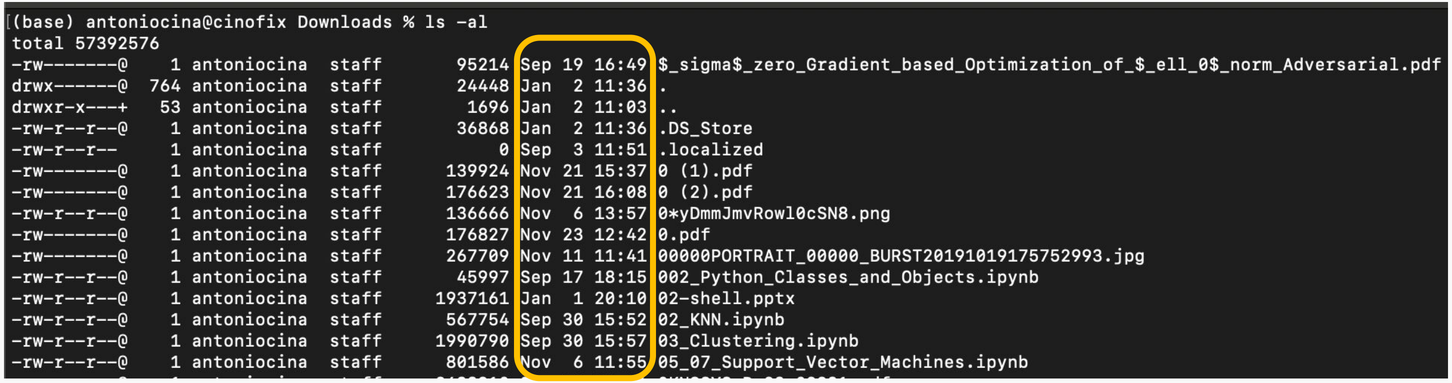
```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff     95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff     24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff      1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff     36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff         0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff    139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff    176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@    1 antoniocina  staff    136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@    1 antoniocina  staff    176827 Nov 23 12:42 0.pdf
-rw-------@    1 antoniocina  staff    267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@    1 antoniocina  staff     45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@    1 antoniocina  staff   1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@    1 antoniocina  staff    567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@    1 antoniocina  staff   1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@    1 antoniocina  staff    801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

File size

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`



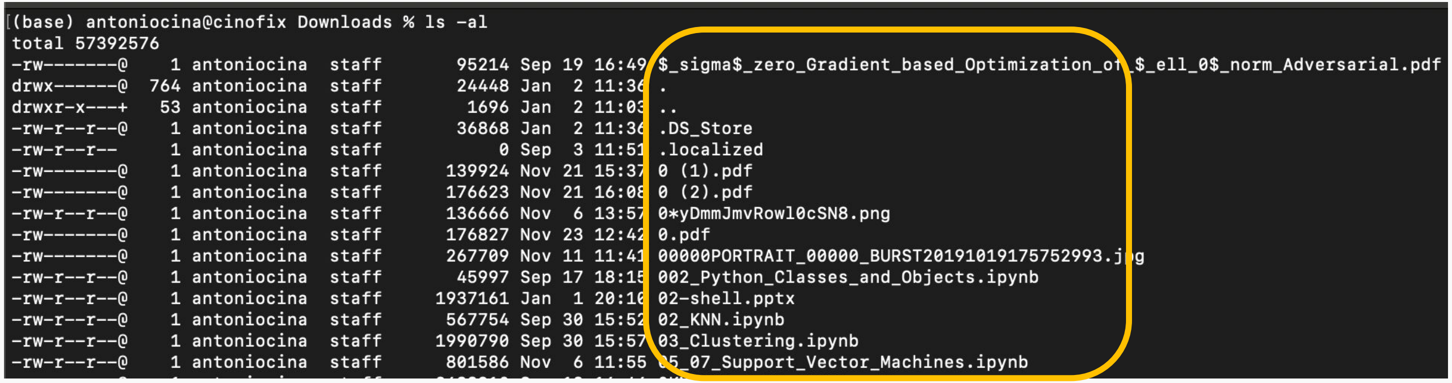
```
[(base) antoniocina@cinofix Downloads % ls -al
 total 57392576
 -rw-------@    1 antoniocina  staff       95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
 drwx------@  764 antoniocina  staff       24448 Jan  2 11:36 .
 drwxr-x---+   53 antoniocina  staff        1696 Jan  2 11:03 ..
 -rw-r--r--@    1 antoniocina  staff       36868 Jan  2 11:36 .DS_Store
 -rw-r--r--     1 antoniocina  staff           0 Sep  3 11:51 .localized
 -rw-------@    1 antoniocina  staff      139924 Nov 21 15:37 0 (1).pdf
 -rw-------@    1 antoniocina  staff      176623 Nov 21 16:08 0 (2).pdf
 -rw-r--r--@    1 antoniocina  staff      136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
 -rw-------@    1 antoniocina  staff      176827 Nov 23 12:42 0.pdf
 -rw-------@    1 antoniocina  staff      267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
 -rw-r--r--@    1 antoniocina  staff       45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
 -rw-r--r--@    1 antoniocina  staff     1937161 Jan  1 20:10 02-shell.pptx
 -rw-r--r--@    1 antoniocina  staff      567754 Sep 30 15:52 02_KNN.ipynb
 -rw-r--r--@    1 antoniocina  staff     1990790 Sep 30 15:57 03_Clustering.ipynb
 -rw-r--r--@    1 antoniocina  staff      801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

Last modification date

Files

Files are collections of data that are stored on a storage device for long-term storage. They can contain various types of information, such as text, images, audio, or program code.

Files can be listed with the command `$ ls -al`

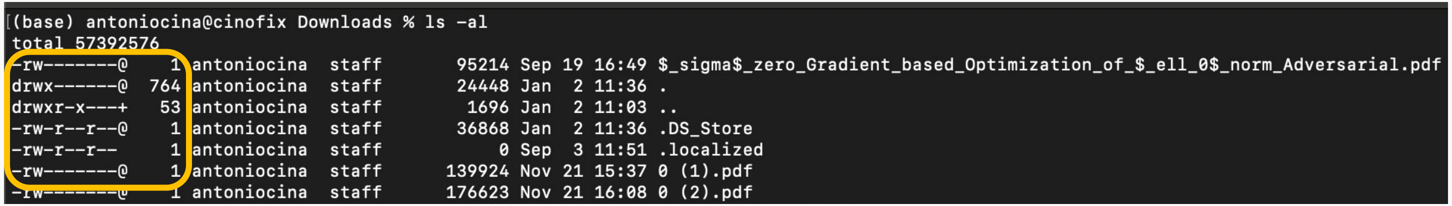


```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff       95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff       24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff        1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff       36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff           0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff      139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff      176623 Nov 21 16:08 0 (2).pdf
-rw-r--r--@    1 antoniocina  staff      136666 Nov  6 13:57 0*yDmmJmvRowl0cSN8.png
-rw-------@    1 antoniocina  staff      176827 Nov 23 12:42 0.pdf
-rw-------@    1 antoniocina  staff      267709 Nov 11 11:41 00000PORTRAIT_00000_BURST20191019175752993.jpg
-rw-r--r--@    1 antoniocina  staff       45997 Sep 17 18:15 002_Python_Classes_and_Objects.ipynb
-rw-r--r--@    1 antoniocina  staff     1937161 Jan  1 20:10 02-shell.pptx
-rw-r--r--@    1 antoniocina  staff      567754 Sep 30 15:52 02_KNN.ipynb
-rw-r--r--@    1 antoniocina  staff     1990790 Sep 30 15:57 03_Clustering.ipynb
-rw-r--r--@    1 antoniocina  staff      801586 Nov  6 11:55 05_07_Support_Vector_Machines.ipynb
```

File name

Files Permissions

The first set of permissions applies to the **owner** of the file.
The second set of permissions applies to the **user group** that owns the file.
The third set of permissions is generally referred to as **others**.



```
[(base) antoniocina@cinofix Downloads % ls -al
total 57392576
-rw-------@    1 antoniocina  staff      95214 Sep 19 16:49 $_sigma$_zero_Gradient_based_Optimization_of_$_ell_0$_norm_Adversarial.pdf
drwx------@  764 antoniocina  staff      24448 Jan  2 11:36 .
drwxr-x---+   53 antoniocina  staff       1696 Jan  2 11:03 ..
-rw-r--r--@    1 antoniocina  staff      36868 Jan  2 11:36 .DS_Store
-rw-r--r--     1 antoniocina  staff          0 Sep  3 11:51 .localized
-rw-------@    1 antoniocina  staff     139924 Nov 21 15:37 0 (1).pdf
-rw-------@    1 antoniocina  staff     176623 Nov 21 16:08 0 (2).pdf
```

Each character in the expression indicates whether a specific permission is granted or not.

- read (r) permission
- write (w) permission
- execute permission (x)

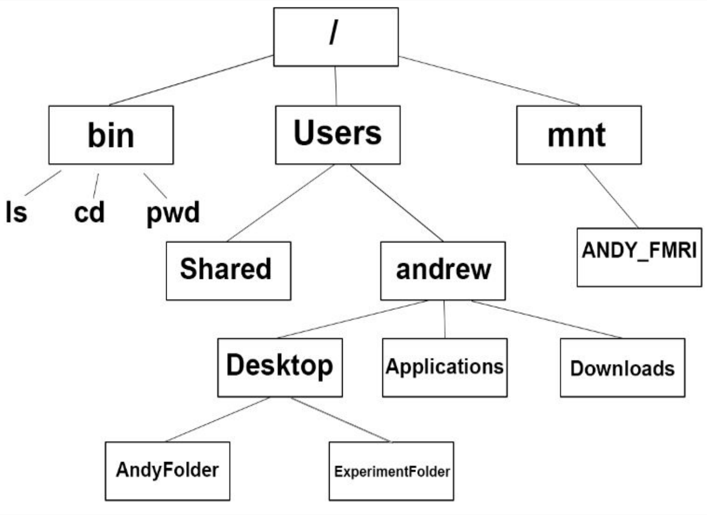
Files

Files contain other files, branching out from the root “/” forming a tree-like hierarchy.

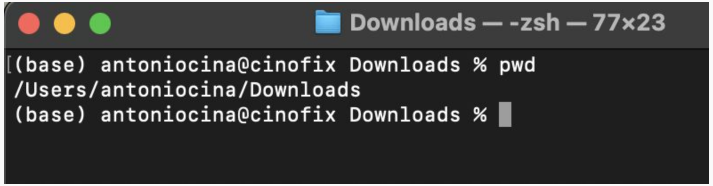
Files are located with a path of folders separated by “/” this is called the file path.

Paths starting with “/” are called absolute paths
- Start searching from the root of the file system

Paths that do NOT start with “/” are called relative paths
- Starts searching from current directory



The `$ pwd` command will print the current directory



```
(base) antoniocina@cinofix Downloads % pwd
/Users/antoniocina/Downloads
(base) antoniocina@cinofix Downloads %
```

Useful Commands

Command	Operation	Example
ls	See folder contents	ls -l
cd <folderName>	Move into given folder	cd Downloads
cp <source> <destination>	Make a copy of given file in given destination	cp file.txt myDir/
mv <oldName> <newname>	Rename or move given existing file to given name/destination	mv fil.txt file.txt
cat <fileName>	Print file contents to terminal window	cat file.txt
touch <filename>	Create empty file with given name	touch file.txt
echo <string>	Print given string to terminal window	echo “hello world”
pwd	Print working directory	pwd
mkdir <directoryName>	Create an empty directory at location specified	mkdir ~/newDir
exit	Exit the shell	exit

Useful Commands for Remote Working

Command	Operation	Example
wget <path_to_remote_file>	Downloads files from the web.	wget https://example.com/file.tar.gz
ssh <username>@<remote>	Establishes a secure shell connection to a remote server.	ssh acina@gpu1.unige
scp <localfile> <username>@<remote>:/<path>	Securely copies files between a local and a remote host.	scp main.py acina@gpu1.unige:/acina/project
tar *Compress:* tar -czvf <archive.tar.gz> <files> *Extract:* tar -xzvf <archive.tar.gz	Compresses or extracts files in a tarball archive	tar -czvf archive.tar.gz file1 tar -xzvf archive.tar.gz
zip <archive_name.zip> <files>	Compresses files into a zip archive	zip results.zip exp1.csv exp2.csv
unzip <archive_name.zip>	Extracts files from a zip archive.	unzip results.zip

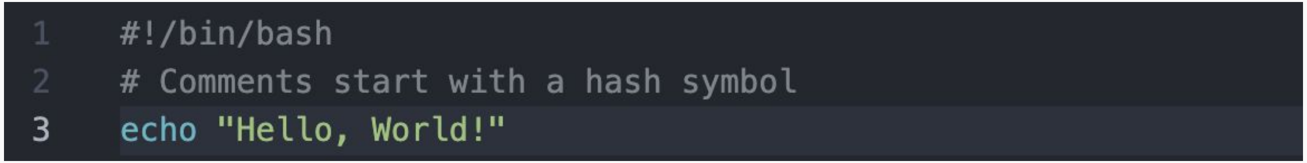
Bash Scripting

Bash is a command language and scripting shell integral to Unix-like operating systems.

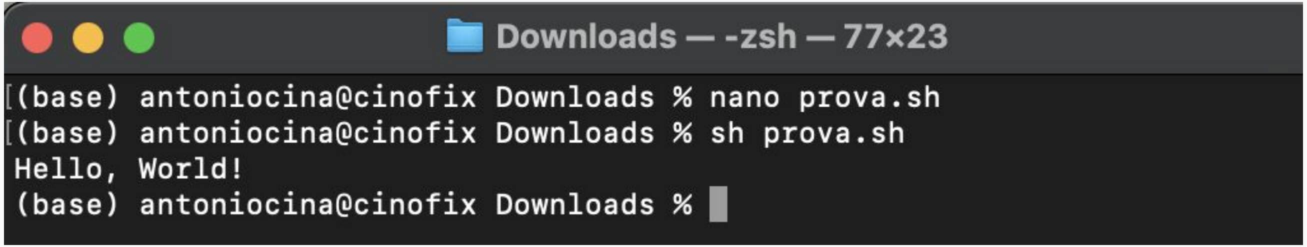
Why we want to use it?

Bash Scripting

Bash is a command language and scripting shell integral to Unix-like operating systems.



```bash
#!/bin/bash
# Comments start with a hash symbol
echo "Hello, World!"
```



```
[(base) antoniocina@cinofix Downloads % nano prova.sh
[(base) antoniocina@cinofix Downloads % sh prova.sh
Hello, World!
(base) antoniocina@cinofix Downloads %
```

Bash Scripting

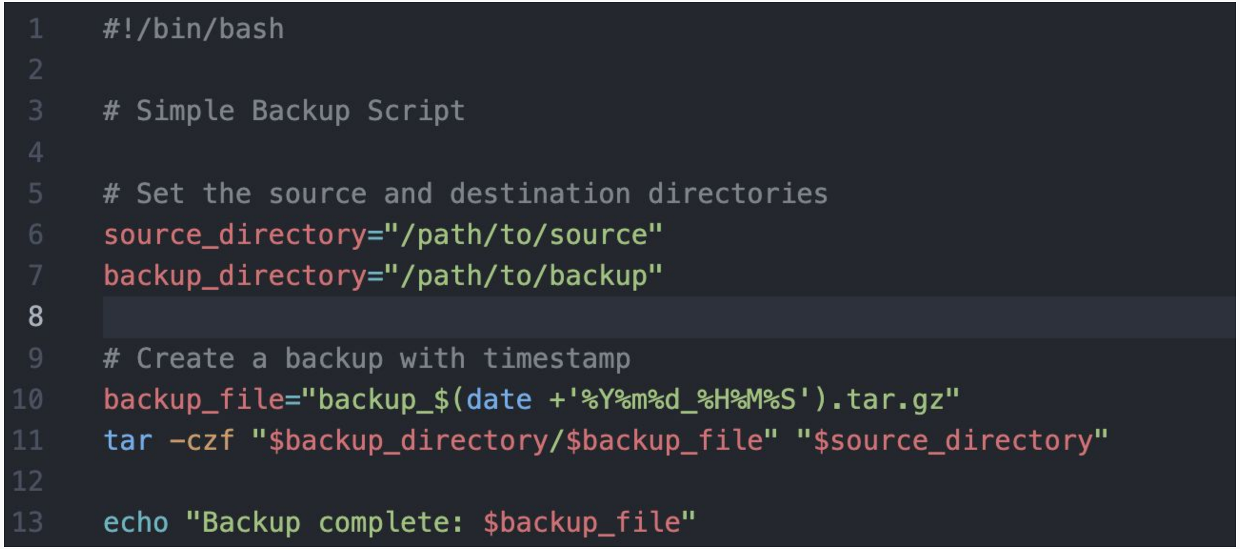
Bash is a command language and scripting shell integral to Unix-like operating systems.

Why we want to use it?

Automation of Repetitive Tasks
Bash scripts automate routine and repetitive tasks, reducing the manual effort required for activities such as file management, data processing, or system maintenance.

The ability to automate these tasks not only saves time but also minimizes the risk of human error, ensuring consistent and reliable execution.

Bash Scripting



```bash
#!/bin/bash

# Simple Backup Script

# Set the source and destination directories
source_directory="/path/to/source"
backup_directory="/path/to/backup"

# Create a backup with timestamp
backup_file="backup_$(date +'%Y%m%d_%H%M%S').tar.gz"
tar -czf "$backup_directory/$backup_file" "$source_directory"

echo "Backup complete: $backup_file"
```

Bash Scripting

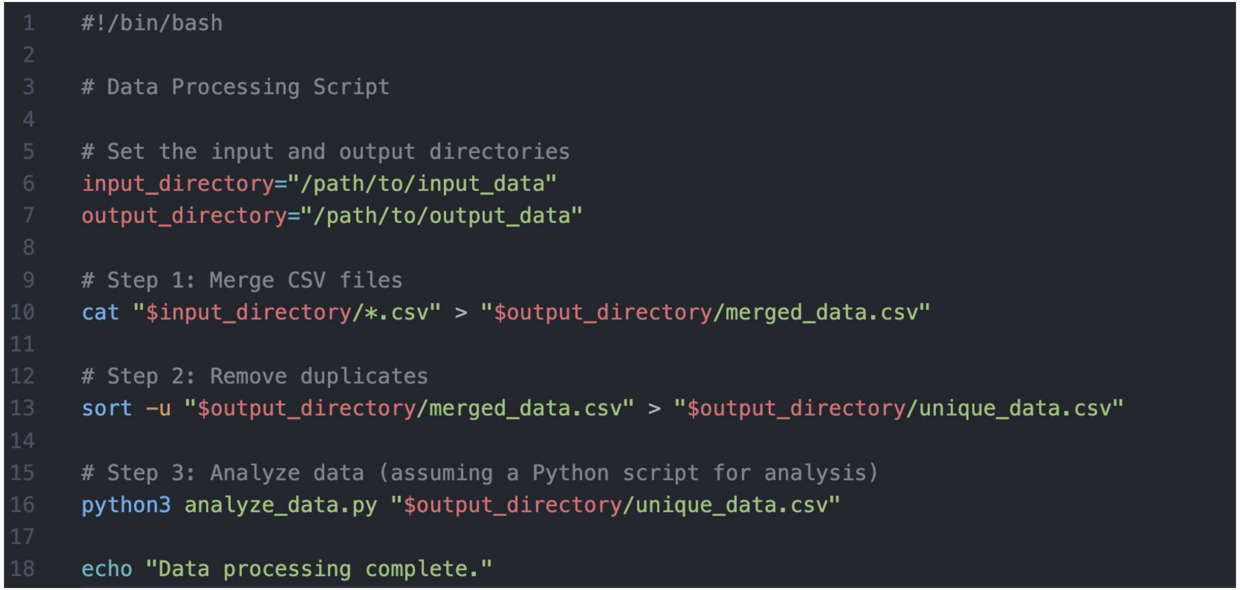
Bash is a command language and scripting shell integral to Unix-like operating systems.

Why we want to use it?

Efficient Command-Line Operations
Bash scripts provide a means to encapsulate and execute complex command-line operations with a single script, simplifying intricate processes.

Users can create custom scripts to encapsulate sequences of commands, making it easier to handle and manage a series of operations without the need to remember or type them individually.

Bash Scripting



```bash
#!/bin/bash

# Data Processing Script

# Set the input and output directories
input_directory="/path/to/input_data"
output_directory="/path/to/output_data"

# Step 1: Merge CSV files
cat "$input_directory/*.csv" > "$output_directory/merged_data.csv"

# Step 2: Remove duplicates
sort -u "$output_directory/merged_data.csv" > "$output_directory/unique_data.csv"

# Step 3: Analyze data (assuming a Python script for analysis)
python3 analyze_data.py "$output_directory/unique_data.csv"

echo "Data processing complete."
```

Bash Scripting

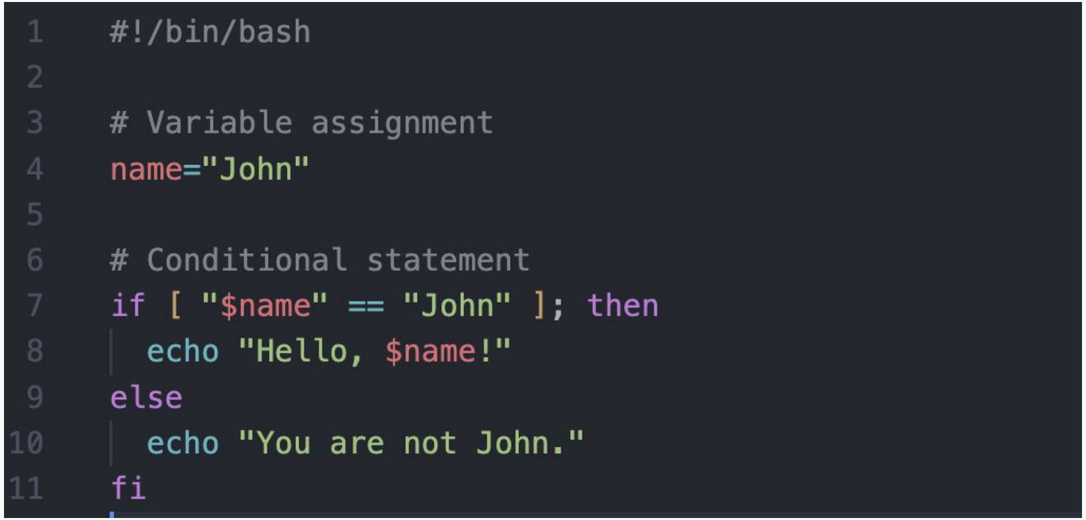
Bash is a command language and scripting shell integral to Unix-like operating systems.

Why we want to use it?

Task Scheduling and System Automation
Bash scripting facilitates the scheduling of tasks through cron jobs or other scheduling mechanisms, enabling the automatic execution of scripts at predefined intervals.

System administrators often leverage Bash scripts to automate system-related tasks, ensuring timely execution of maintenance routines and updates.

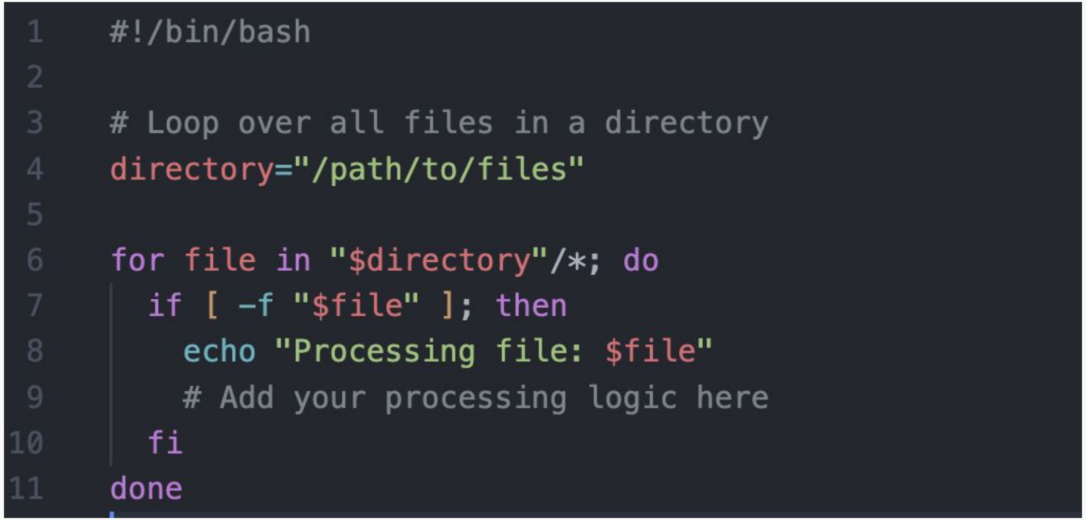
Variables and Control Flow



```bash
#!/bin/bash

# Variable assignment
name="John"

# Conditional statement
if [ "$name" == "John" ]; then
  echo "Hello, $name!"
else
  echo "You are not John."
fi
```

This script assigns a value to a variable and uses a conditional statement to print a message based on the variable's value.

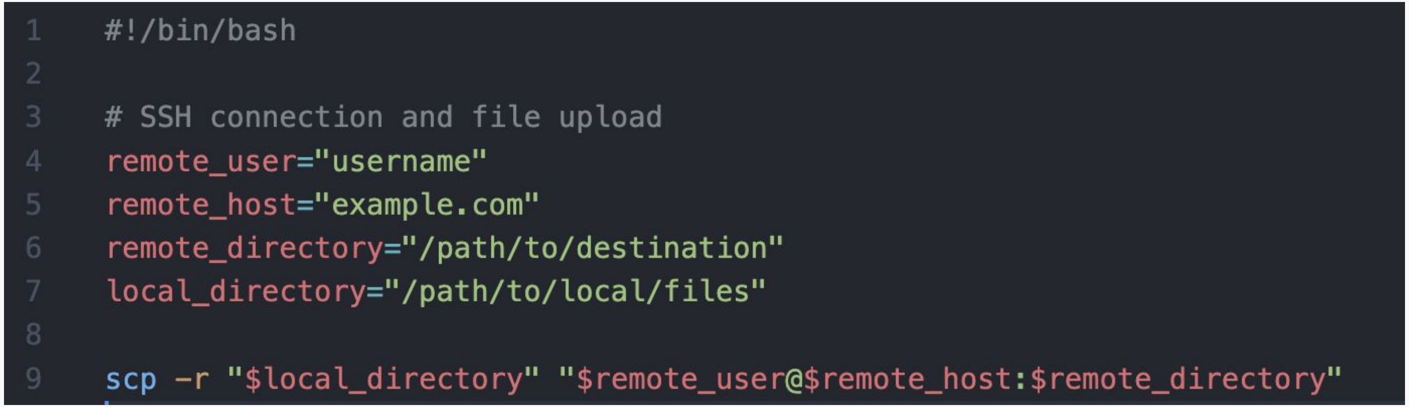
Looping Over All Files



```bash
#!/bin/bash

# Loop over all files in a directory
directory="/path/to/files"

for file in "$directory"/*; do
  if [ -f "$file" ]; then
    echo "Processing file: $file"
    # Add your processing logic here
  fi
done
```

This script uses a for loop to iterate through all files in a specified directory, checking if each item is a regular file before processing.

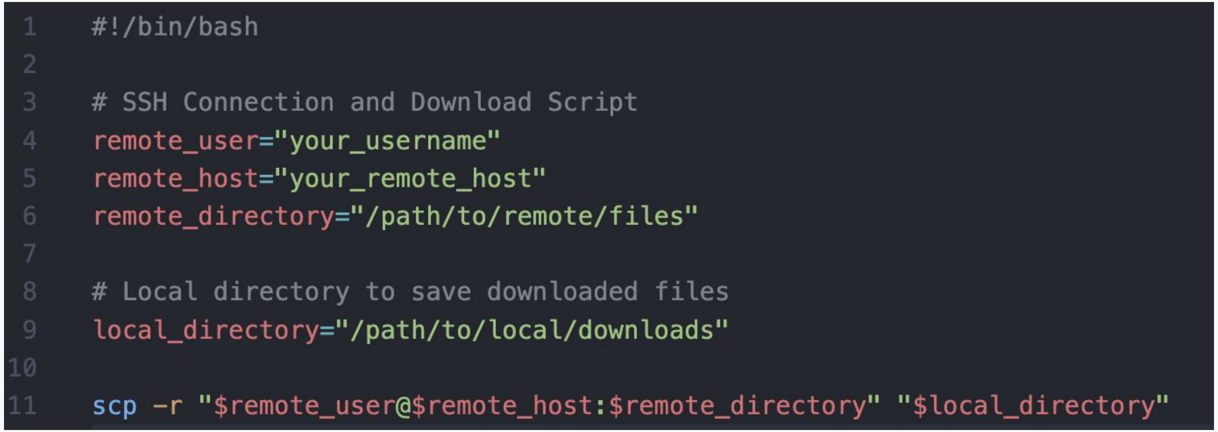
SSH Connection and Upload All Files



```bash
#!/bin/bash

# SSH connection and file upload
remote_user="username"
remote_host="example.com"
remote_directory="/path/to/destination"
local_directory="/path/to/local/files"

scp -r "$local_directory" "$remote_user@$remote_host:$remote_directory"
```

In this script, scp securely copies the local files to a remote server using SSH.

SSH Connection and Download All Files



```bash
#!/bin/bash

# SSH Connection and Download Script
remote_user="your_username"
remote_host="your_remote_host"
remote_directory="/path/to/remote/files"

# Local directory to save downloaded files
local_directory="/path/to/local/downloads"

scp -r "$remote_user@$remote_host:$remote_directory" "$local_directory"
```

In this script, scp securely download the remote files to the local machine using SSH.

Exercise 1: Generating 100 Empty CSVs

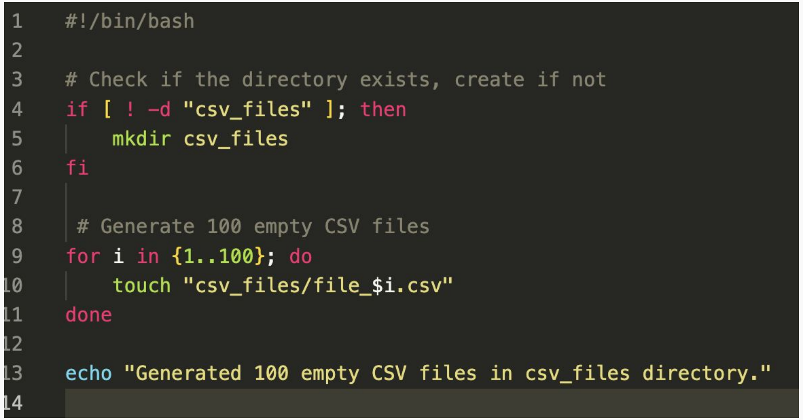
Problem description: Create a Bash script to generate 100 empty CSV files named "file_i," where "i" represents the index of the file.

The script should:

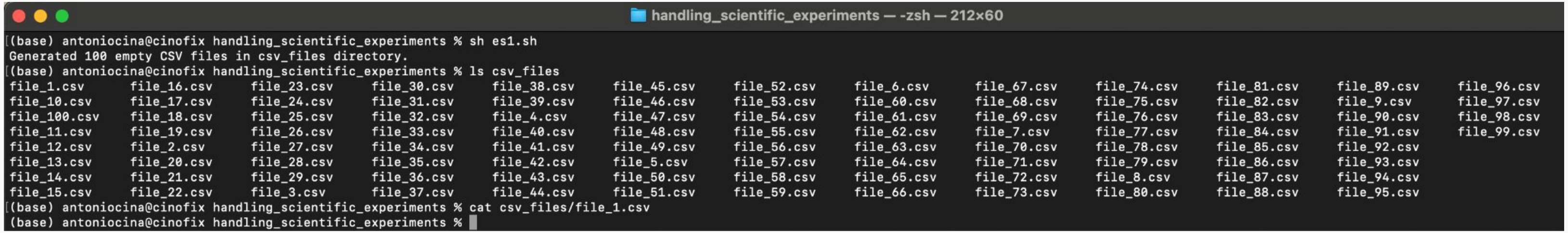
- Check if a directory named "csv_files" exists. If not, create it.
- Generate 100 empty CSV files within the "csv_files" directory, naming them "file_1.csv" to "file_100.csv."

Exercise 1: Generating 100 Empty CSVs

The touch command rename or move given existing file to given name/destination.



```
#!/bin/bash

# Check if the directory exists, create if not
if [ ! -d "csv_files" ]; then
    mkdir csv_files
fi

 # Generate 100 empty CSV files
for i in {1..100}; do
    touch "csv_files/file_$i.csv"
done

echo "Generated 100 empty CSV files in csv_files directory."
```



```
[(base) antoniocina@cinofix handling_scientific_experiments % sh es1.sh
 Generated 100 empty CSV files in csv_files directory.
[(base) antoniocina@cinofix handling_scientific_experiments % ls csv_files
 file_1.csv      file_16.csv     file_23.csv     file_30.csv     file_38.csv     file_45.csv     file_52.csv     file_6.csv      file_67.csv     file_74.csv     file_81.csv     file_89.csv     file_96.csv
 file_10.csv     file_17.csv     file_24.csv     file_31.csv     file_39.csv     file_46.csv     file_53.csv     file_60.csv     file_68.csv     file_75.csv     file_82.csv     file_9.csv      file_97.csv
 file_100.csv    file_18.csv     file_25.csv     file_32.csv     file_4.csv      file_47.csv     file_54.csv     file_61.csv     file_69.csv     file_76.csv     file_83.csv     file_90.csv     file_98.csv
 file_11.csv     file_19.csv     file_26.csv     file_33.csv     file_40.csv     file_48.csv     file_55.csv     file_62.csv     file_7.csv      file_77.csv     file_84.csv     file_91.csv     file_99.csv
 file_12.csv     file_2.csv      file_27.csv     file_34.csv     file_41.csv     file_49.csv     file_56.csv     file_63.csv     file_70.csv     file_78.csv     file_85.csv     file_92.csv
 file_13.csv     file_20.csv     file_28.csv     file_35.csv     file_42.csv     file_5.csv      file_57.csv     file_64.csv     file_71.csv     file_79.csv     file_86.csv     file_93.csv
 file_14.csv     file_21.csv     file_29.csv     file_36.csv     file_43.csv     file_50.csv     file_58.csv     file_65.csv     file_72.csv     file_8.csv      file_87.csv     file_94.csv
 file_15.csv     file_22.csv     file_3.csv      file_37.csv     file_44.csv     file_51.csv     file_59.csv     file_66.csv     file_73.csv     file_80.csv     file_88.csv     file_95.csv
[(base) antoniocina@cinofix handling_scientific_experiments % cat csv_files/file_1.csv
 (base) antoniocina@cinofix handling_scientific_experiments %
```

Exercise 2: Move CSV Files

Problem description: Write a Bash script that moves all CSV files from one directory to another.

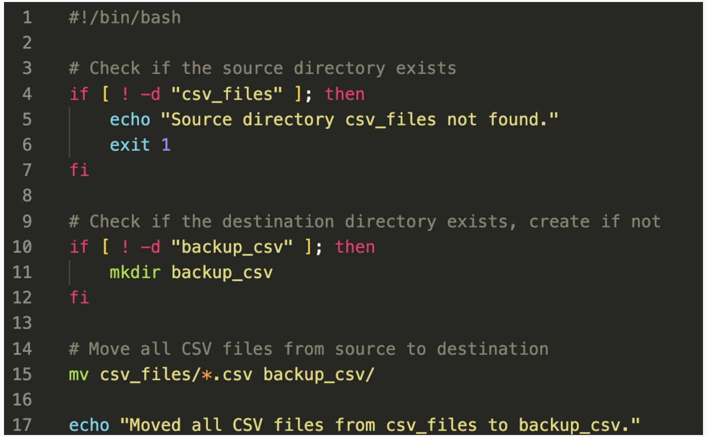
The script should:

- Check if the source directory "csv_files" exists. If not, display an error message and exit.
- Check if the destination directory "backup_csv" exists. If not, create it.
- Move all CSV files from "csv_files" to "backup_csv."
- Display a message indicating the number of files moved.

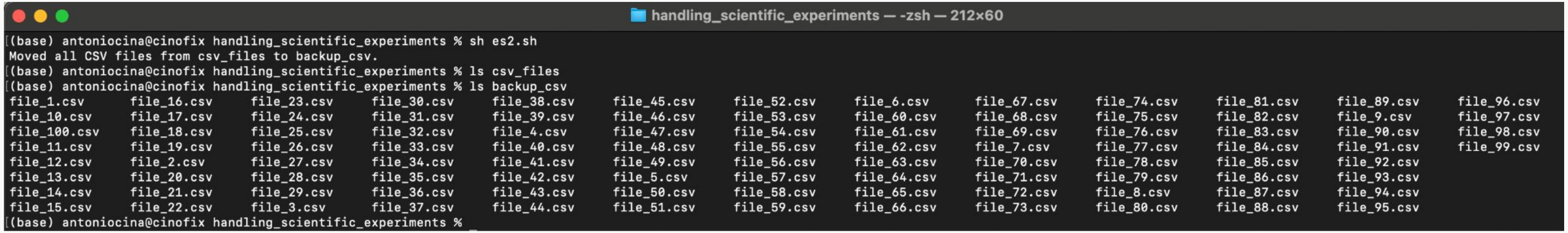
Exercise 2: Move CSV Files

The -d flag tests whether the provided name exists and is a directory.

The mv command creates an empty file.



```bash
#!/bin/bash

# Check if the source directory exists
if [ ! -d "csv_files" ]; then
    echo "Source directory csv_files not found."
    exit 1
fi

# Check if the destination directory exists, create if not
if [ ! -d "backup_csv" ]; then
    mkdir backup_csv
fi

# Move all CSV files from source to destination
mv csv_files/*.csv backup_csv/

echo "Moved all CSV files from csv_files to backup_csv."
```



```
[(base) antoniocina@cinofix handling_scientific_experiments % sh es2.sh
Moved all CSV files from csv_files to backup_csv.
[(base) antoniocina@cinofix handling_scientific_experiments % ls csv_files
[(base) antoniocina@cinofix handling_scientific_experiments % ls backup_csv
file_1.csv      file_16.csv     file_23.csv     file_30.csv     file_38.csv     file_45.csv     file_52.csv     file_6.csv      file_67.csv     file_74.csv     file_81.csv     file_89.csv     file_96.csv
file_10.csv     file_17.csv     file_24.csv     file_31.csv     file_39.csv     file_46.csv     file_53.csv     file_60.csv     file_68.csv     file_75.csv     file_82.csv     file_9.csv      file_97.csv
file_100.csv    file_18.csv     file_25.csv     file_32.csv     file_4.csv      file_47.csv     file_54.csv     file_61.csv     file_69.csv     file_76.csv     file_83.csv     file_90.csv     file_98.csv
file_11.csv     file_19.csv     file_26.csv     file_33.csv     file_40.csv     file_48.csv     file_55.csv     file_62.csv     file_7.csv      file_77.csv     file_84.csv     file_91.csv     file_99.csv
file_12.csv     file_2.csv      file_27.csv     file_34.csv     file_41.csv     file_49.csv     file_56.csv     file_63.csv     file_70.csv     file_78.csv     file_85.csv     file_92.csv
file_13.csv     file_20.csv     file_28.csv     file_35.csv     file_42.csv     file_5.csv      file_57.csv     file_64.csv     file_71.csv     file_79.csv     file_86.csv     file_93.csv
file_14.csv     file_21.csv     file_29.csv     file_36.csv     file_43.csv     file_50.csv     file_58.csv     file_65.csv     file_72.csv     file_8.csv      file_87.csv     file_94.csv
file_15.csv     file_22.csv     file_3.csv      file_37.csv     file_44.csv     file_51.csv     file_59.csv     file_66.csv     file_73.csv     file_80.csv     file_88.csv     file_95.csv
[(base) antoniocina@cinofix handling_scientific_experiments % _
```

Reproducible Python Environments

Managing Software

Python can very rapidly translate your ideas into readable code solutions.

- Write the data to an hdf5 file format? Import h5py!
- Plot some figure, xkcd style? Import matplotlib!
- Need Machine Learning? Keras, Pytorch, ScikitLearn!

Unfortunately, the packages are **updated**, **restructured**, **improved**, or just **rewritten**, just because the authors came up with a better way to solve their problem.

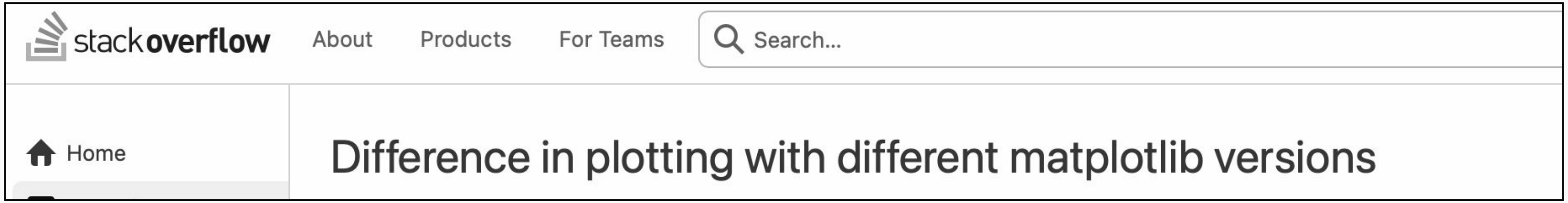
These changes can be **breaking changes** for the code you have written.

Managing Software

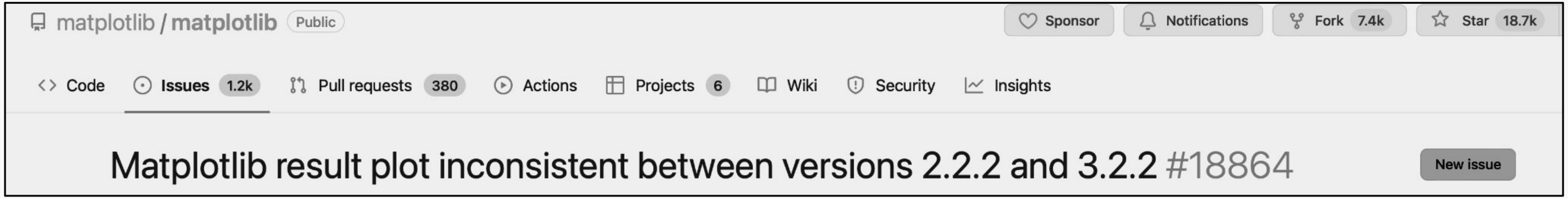
“ Popular packages, such as Numpy, Matplotlib, or Pytorch are very reliable! ”

Managing Software

“ Popular packages, such as Numpy, Matplotlib, or Pytorch are very reliable! ”



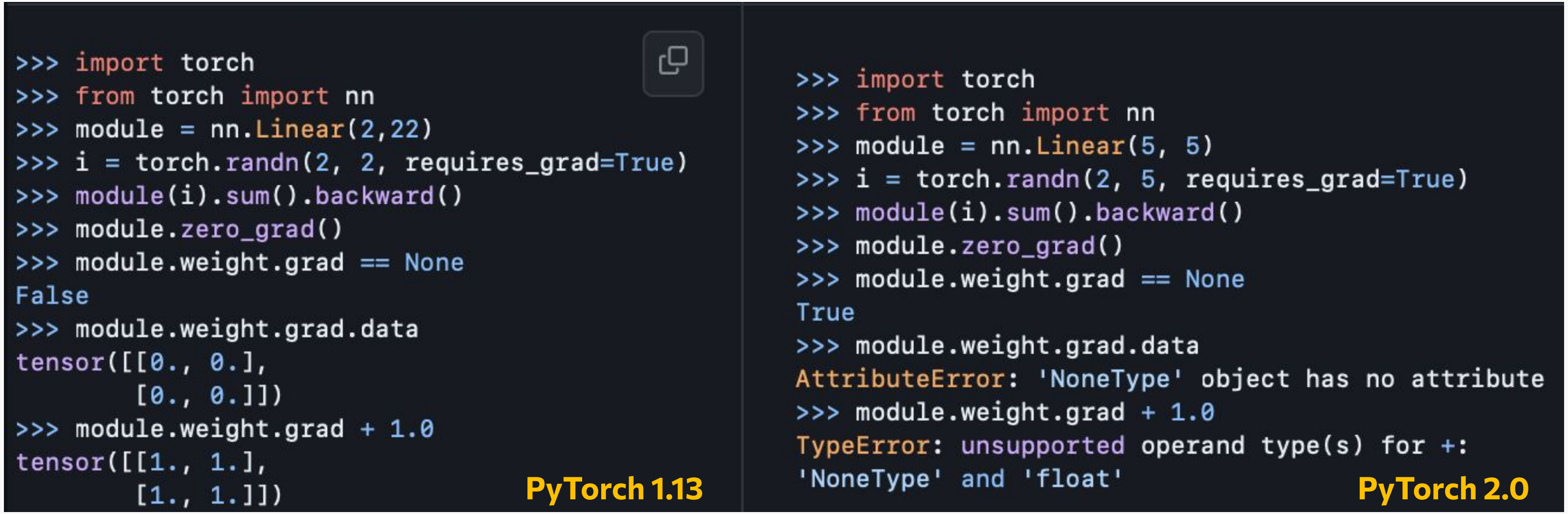
Difference in plotting with different matplotlib versions



Matplotlib result plot inconsistent between versions 2.2.2 and 3.2.2 #18864

However, using packages that are not as popular, breaking changing can happen more often, especially when upgrading the package or Python itself.

Pytorch Inconsistencies



```
>>> import torch
>>> from torch import nn
>>> module = nn.Linear(2,22)
>>> i = torch.randn(2, 2, requires_grad=True)
>>> module(i).sum().backward()
>>> module.zero_grad()
>>> module.weight.grad == None
False
>>> module.weight.grad.data
tensor([[0., 0.],
        [0., 0.]])
>>> module.weight.grad + 1.0
tensor([[1., 1.],
        [1., 1.]])
```

PyTorch 1.13

```
>>> import torch
>>> from torch import nn
>>> module = nn.Linear(5, 5)
>>> i = torch.randn(2, 5, requires_grad=True)
>>> module(i).sum().backward()
>>> module.zero_grad()
>>> module.weight.grad == None
True
>>> module.weight.grad.data
AttributeError: 'NoneType' object has no attribute
>>> module.weight.grad + 1.0
TypeError: unsupported operand type(s) for +:
'NoneType' and 'float'
```

PyTorch 2.0

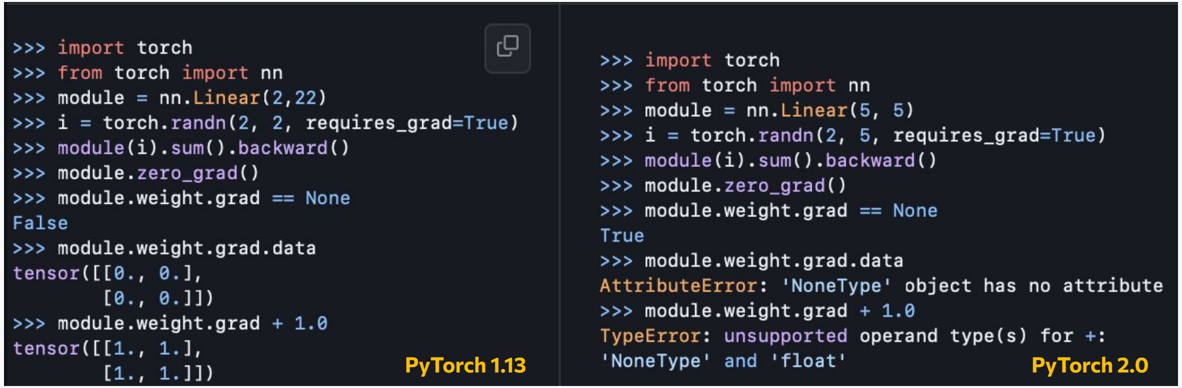
Taken from Pytorch official release notes: https://github.com/pytorch/pytorch/releases

Gradients from Pytorch 2.0 are set to None instead of zeros by default in torch.optim.*.zero_grad() and torch.nn.Module.zero_grad()

Pytorch Inconsistencies

> “
>
> *In other words, the **set_to_none kwarg is now True by default instead of False**. Setting grads to None reduces peak memory usage and increases performance. This will break code that directly accesses data or does computation on the grads after calling zero_grad() as they will now be None. To revert to the old behavior, pass in zero_grad(set_to_none=False).*
>
> ”

– Official Version Note



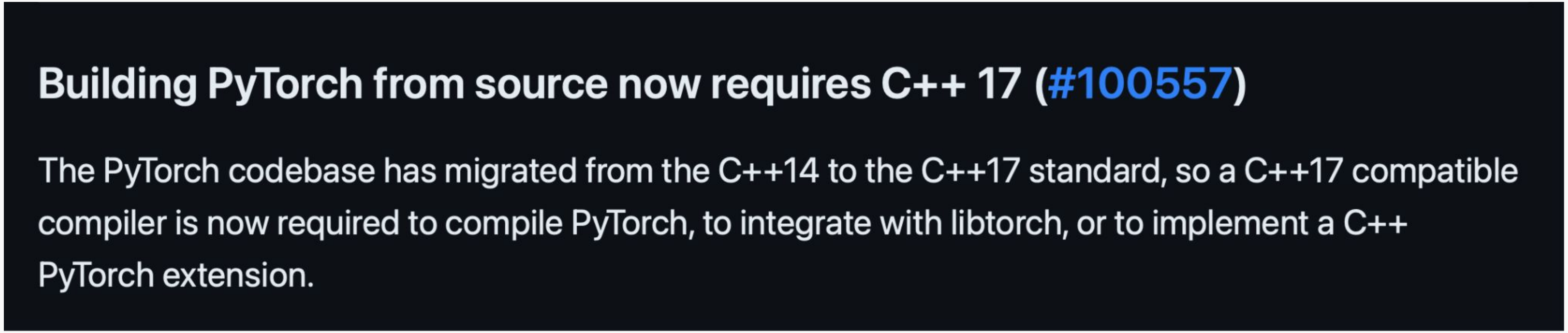
```
>>> import torch
>>> from torch import nn
>>> module = nn.Linear(2,22)
>>> i = torch.randn(2, 2, requires_grad=True)
>>> module(i).sum().backward()
>>> module.zero_grad()
>>> module.weight.grad == None
False
>>> module.weight.grad.data
tensor([[0., 0.],
        [0., 0.]])
>>> module.weight.grad + 1.0
tensor([[1., 1.],
        [1., 1.]])
```

PyTorch 1.13

```
>>> import torch
>>> from torch import nn
>>> module = nn.Linear(5, 5)
>>> i = torch.randn(2, 5, requires_grad=True)
>>> module(i).sum().backward()
>>> module.zero_grad()
>>> module.weight.grad == None
True
>>> module.weight.grad.data
AttributeError: 'NoneType' object has no attribute
>>> module.weight.grad + 1.0
TypeError: unsupported operand type(s) for +:
'NoneType' and 'float'
```

PyTorch 2.0

Backend Incompatible Changes



Building PyTorch from source now requires C++ 17 (#100557)

The PyTorch codebase has migrated from the C++14 to the C++17 standard, so a C++17 compatible compiler is now required to compile PyTorch, to integrate with libtorch, or to implement a C++ PyTorch extension.

The migration of the PyTorch codebase from the C++14 to the C++17 standard implies several changes in the code and build process. While this migration brings new features and improvements to the codebase, it can potentially introduce compatibility issues and errors, especially when interacting with other dependencies or projects that may not fully support C++17.

Managing Software

Managing software tools involve maintaining an organized environment for software dependencies, which is important for ensuring the **repeatability**, and **reproducibility** of our experiments.

Documenting the exact versions of software packages and dependencies used in an experiment enables researchers to reproduce results **consistently**, or to avoid **incompatibilities** and pitfalls .

Solutions:

pip　Pipenv　Poetry　venv　**conda**

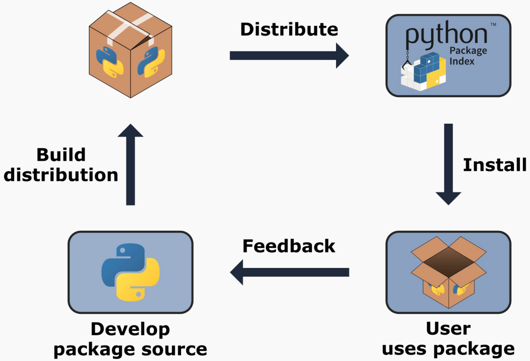
Python venv: https://docs.python.org/3/library/venv.html
pip: https://pypi.org/project/pip/
Pipenv: https://pipenv.pypa.io/en/latest/
Poetry: https://python-poetry.org/

CONDA

Conda is an open-source package management and **environment** management system that runs on Windows, macOS, and Linux.

It works across multiple programming languages, especially for Python.

It simplifies the process of package installation and ensures reproducibility by capturing dependencies and their versions.

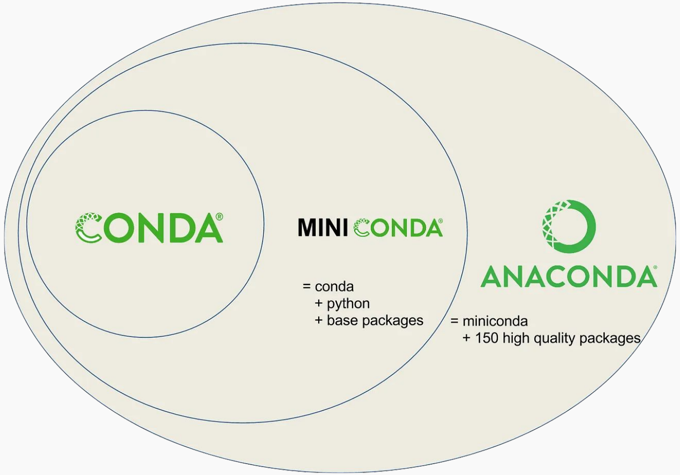


Conda - Miniconda - Anaconda

Conda is a package and environment management system that works across multiple programming languages.

Miniconda is a minimalistic distribution that includes only Conda, its dependencies, and a minimal Python interpreter.

Anaconda is a full distribution that includes Conda, along with a comprehensive collection of pre-installed packages for data science, machine learning, and scientific computing. It aims to provide an all-in-one solution for users in these domains.



Conda environments

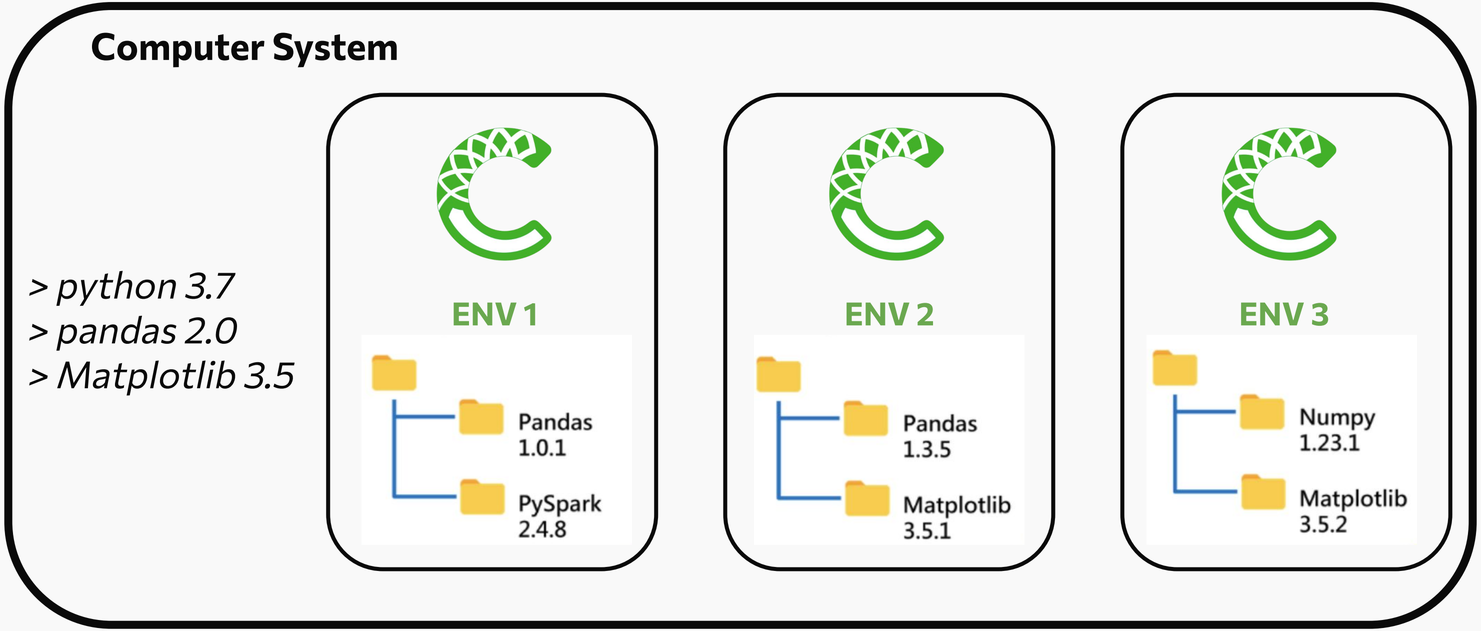
Conda enables users to create **isolated environments** with specific package versions, making it easier to ensure reproducibility in data scientific computing.

It is good practice to have a unique environment for **each project**. For example, you may have one environment with PyTorch 1.7 and its dependencies, and another environment with PyTorch 2.0.

This ensures that dependencies of one project will not create breaking changes for another.

Effective software management tools facilitate collaboration among researchers. Make the projects self-contained and reproducible by capturing all package dependencies in a single requirements file.

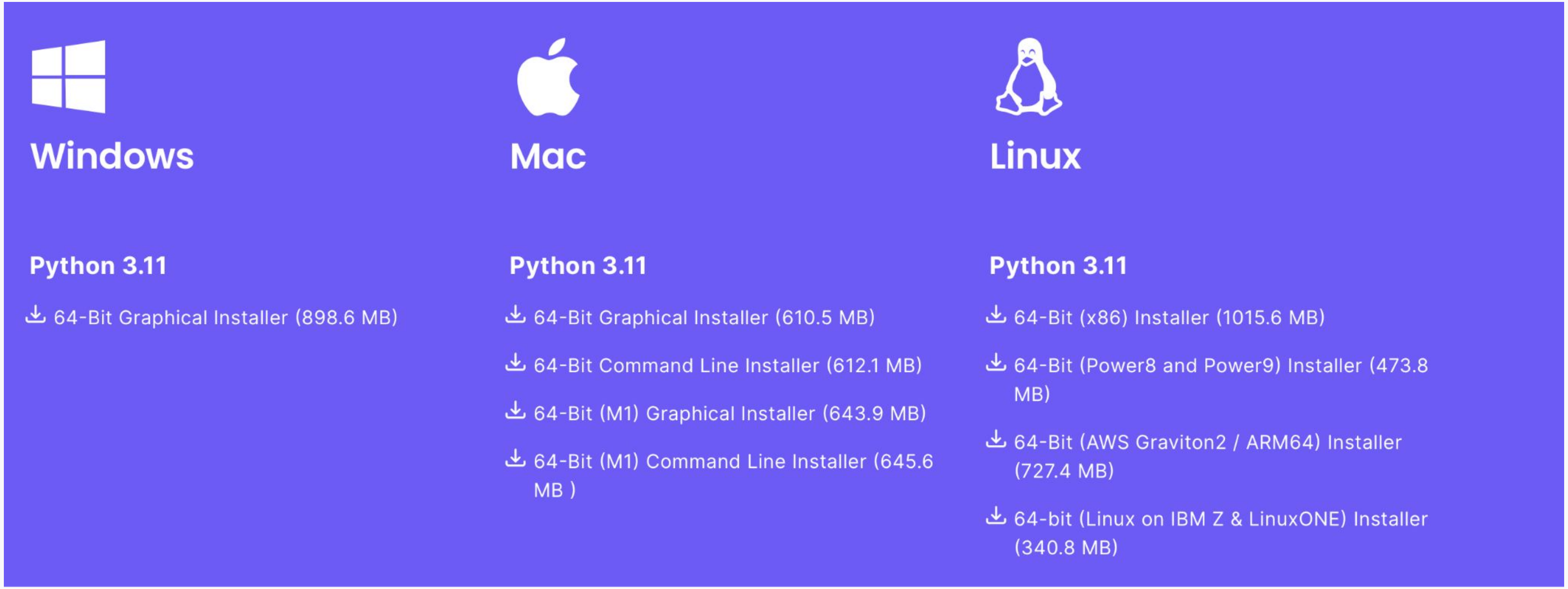
Conda environments



Computer System

> python 3.7
> pandas 2.0
> Matplotlib 3.5

Conda installation



Windows

Python 3.11

- ⤓ 64-Bit Graphical Installer (898.6 MB)

Mac

Python 3.11

- ⤓ 64-Bit Graphical Installer (610.5 MB)
- ⤓ 64-Bit Command Line Installer (612.1 MB)
- ⤓ 64-Bit (M1) Graphical Installer (643.9 MB)
- ⤓ 64-Bit (M1) Command Line Installer (645.6 MB)

Linux

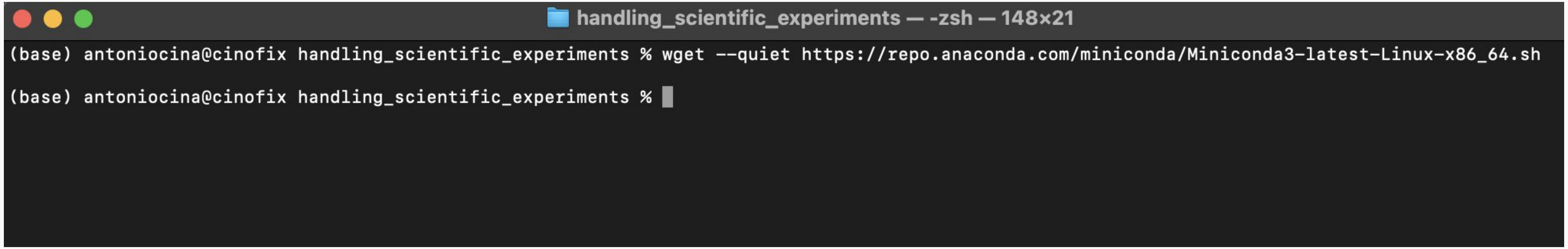
Python 3.11

- ⤓ 64-Bit (x86) Installer (1015.6 MB)
- ⤓ 64-Bit (Power8 and Power9) Installer (473.8 MB)
- ⤓ 64-Bit (AWS Graviton2 / ARM64) Installer (727.4 MB)
- ⤓ 64-bit (Linux on IBM Z & LinuxONE) Installer (340.8 MB)

Conda installation

1. Download the latest version from miniconda

```
$ wget --quiet https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

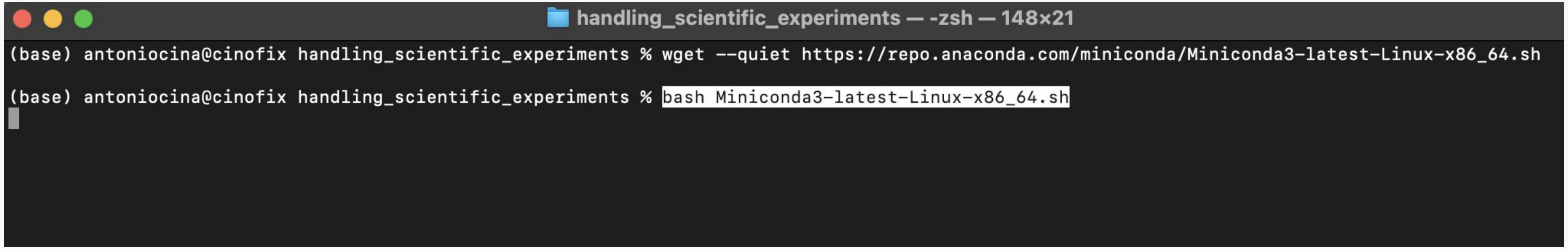


```
(base) antoniocina@cinofix handling_scientific_experiments % wget --quiet https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

(base) antoniocina@cinofix handling_scientific_experiments % 
```

Conda installation

2. Install the miniconda distribution by running the bash script

```
$ bash Miniconda3-latest-Linux-x86_64.sh
```



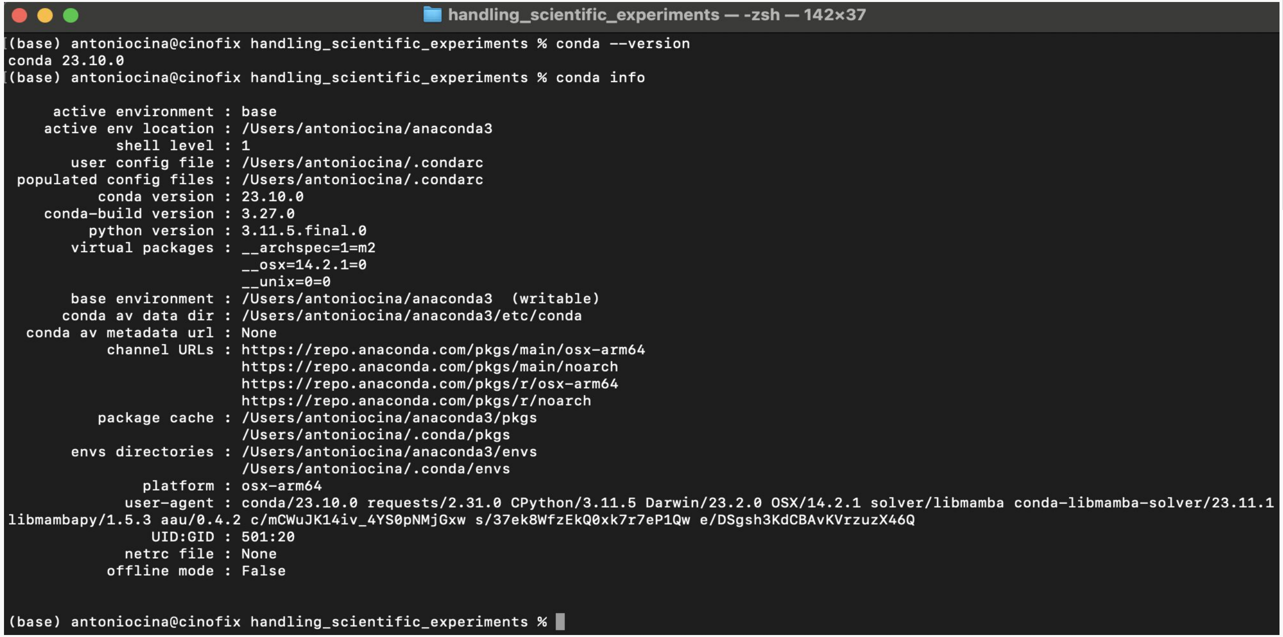
```
(base) antoniocina@cinofix handling_scientific_experiments % wget --quiet https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

(base) antoniocina@cinofix handling_scientific_experiments % bash Miniconda3-latest-Linux-x86_64.sh
```

Conda installation

3. Check conda has been correctly installed

```
$ conda –version
```

```
$ conda  info
```

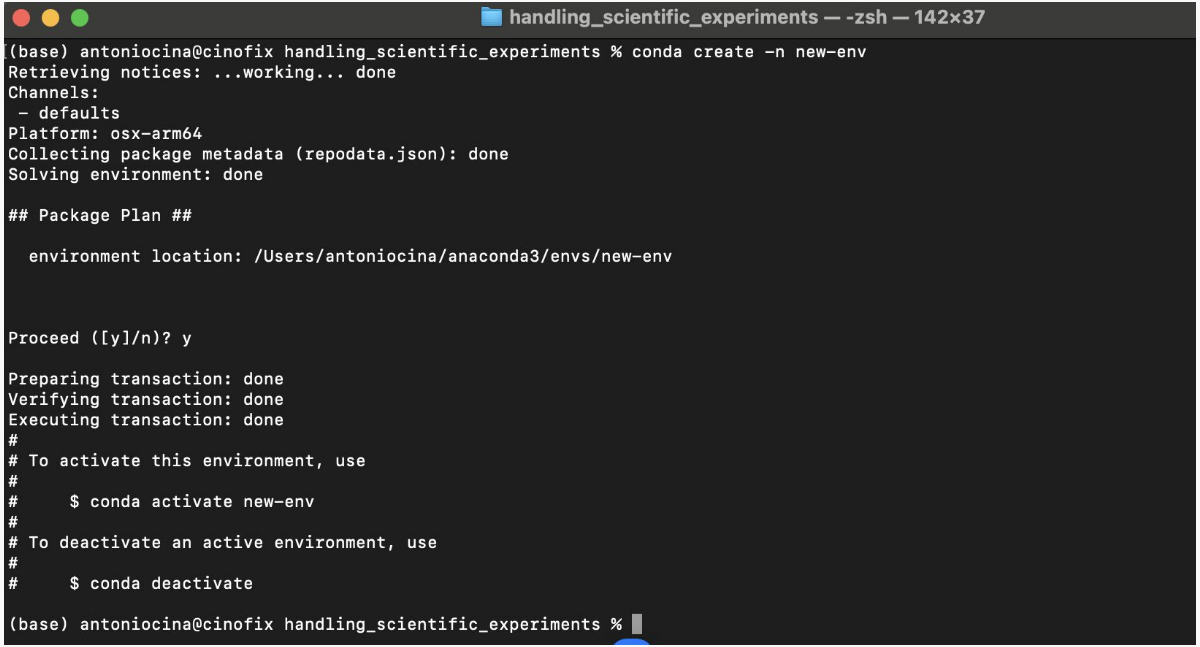


```
(base) antoniocina@cinofix handling_scientific_experiments % conda --version
conda 23.10.0
(base) antoniocina@cinofix handling_scientific_experiments % conda info

     active environment : base
    active env location : /Users/antoniocina/anaconda3
            shell level : 1
       user config file : /Users/antoniocina/.condarc
 populated config files : /Users/antoniocina/.condarc
          conda version : 23.10.0
    conda-build version : 3.27.0
         python version : 3.11.5.final.0
       virtual packages : __archspec=1=m2
                          __osx=14.2.1=0
                          __unix=0=0
       base environment : /Users/antoniocina/anaconda3  (writable)
      conda av data dir : /Users/antoniocina/anaconda3/etc/conda
  conda av metadata url : None
           channel URLs : https://repo.anaconda.com/pkgs/main/osx-arm64
                          https://repo.anaconda.com/pkgs/main/noarch
                          https://repo.anaconda.com/pkgs/r/osx-arm64
                          https://repo.anaconda.com/pkgs/r/noarch
          package cache : /Users/antoniocina/anaconda3/pkgs
                          /Users/antoniocina/.conda/pkgs
       envs directories : /Users/antoniocina/anaconda3/envs
                          /Users/antoniocina/.conda/envs
               platform : osx-arm64
             user-agent : conda/23.10.0 requests/2.31.0 CPython/3.11.5 Darwin/23.2.0 OSX/14.2.1 solver/libmamba conda-libmamba-solver/23.11.1
libmambapy/1.5.3 aau/0.4.2 c/mCWuJK14iv_4YS0pNMjGxw s/37ek8WfzEkQ0xk7r7eP1Qw e/DSgsh3KdCBAvKVrzuzX46Q
                UID:GID : 501:20
             netrc file : None
           offline mode : False


(base) antoniocina@cinofix handling_scientific_experiments % 
```

Creating a new environment

```
$ conda create -n new-env
```

The base command is conda create, and the flag -n specify the name new environment (“new-env”).

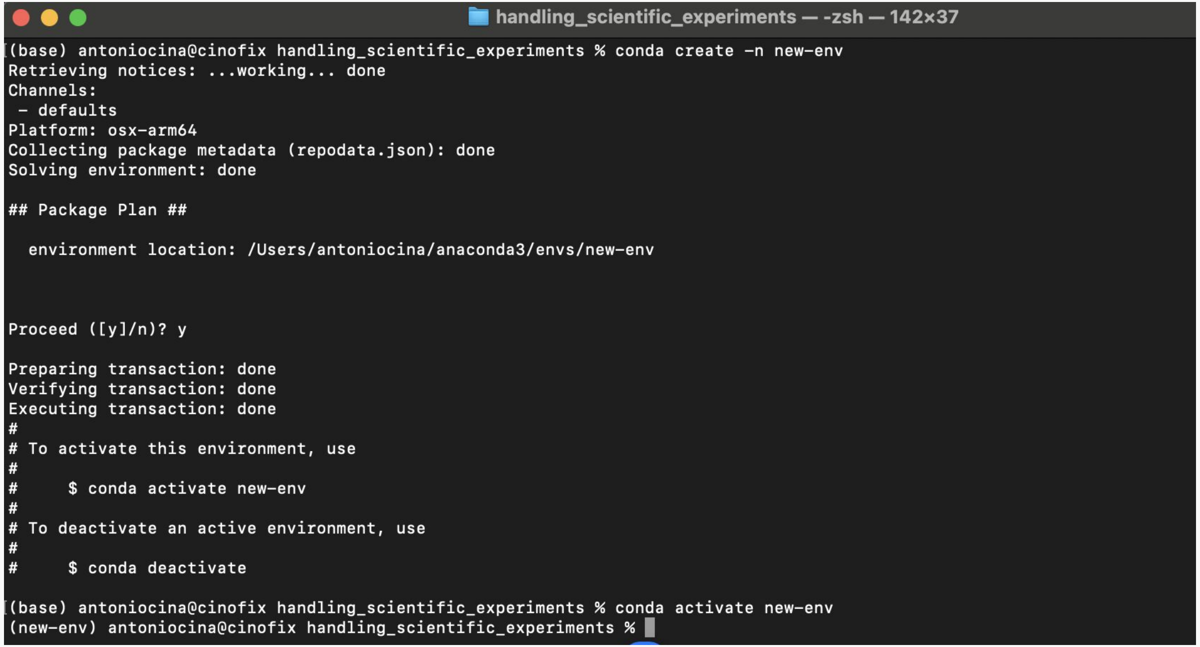


```
(base) antoniocina@cinofix handling_scientific_experiments % conda create -n new-env
Retrieving notices: ...working... done
Channels:
 - defaults
Platform: osx-arm64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /Users/antoniocina/anaconda3/envs/new-env



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate new-env
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) antoniocina@cinofix handling_scientific_experiments %
```

Creating a new environment

```
$ conda activate new-env
```

Now we can see our prompt has changed to include new-env at the front.

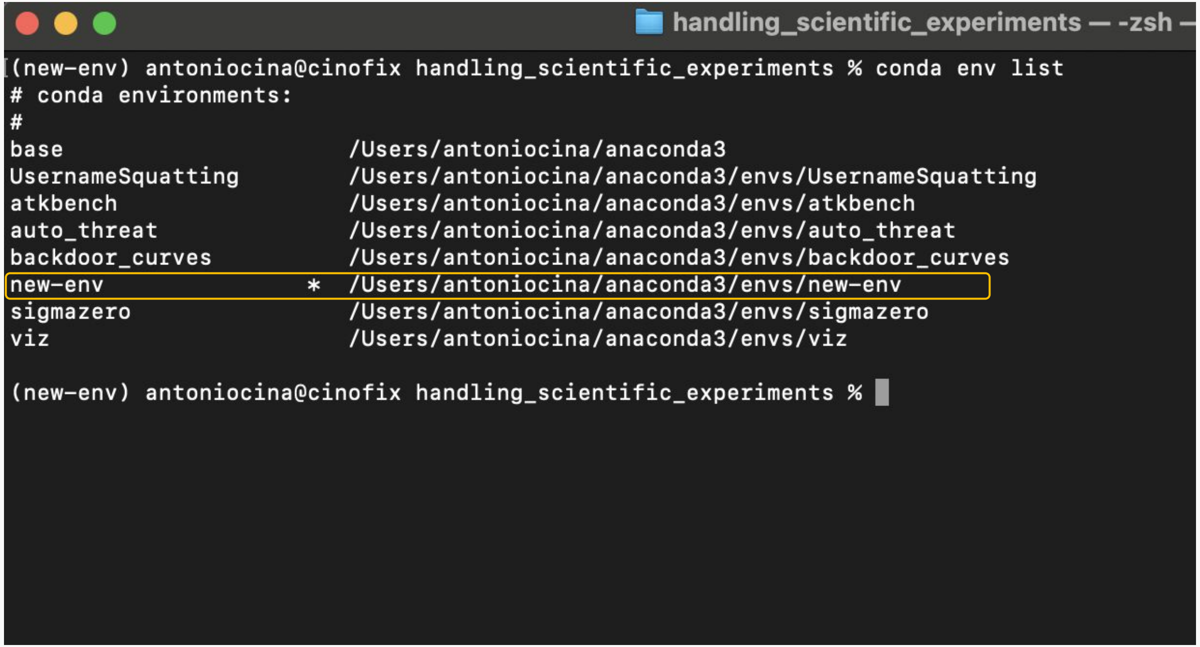


```
(base) antoniocina@cinofix handling_scientific_experiments % conda create -n new-env
Retrieving notices: ...working... done
Channels:
 - defaults
Platform: osx-arm64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /Users/antoniocina/anaconda3/envs/new-env



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate new-env
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) antoniocina@cinofix handling_scientific_experiments % conda activate new-env
(new-env) antoniocina@cinofix handling_scientific_experiments %
```

Creating a new environment

`$ conda env list`

Will print out all of the available conda environments.

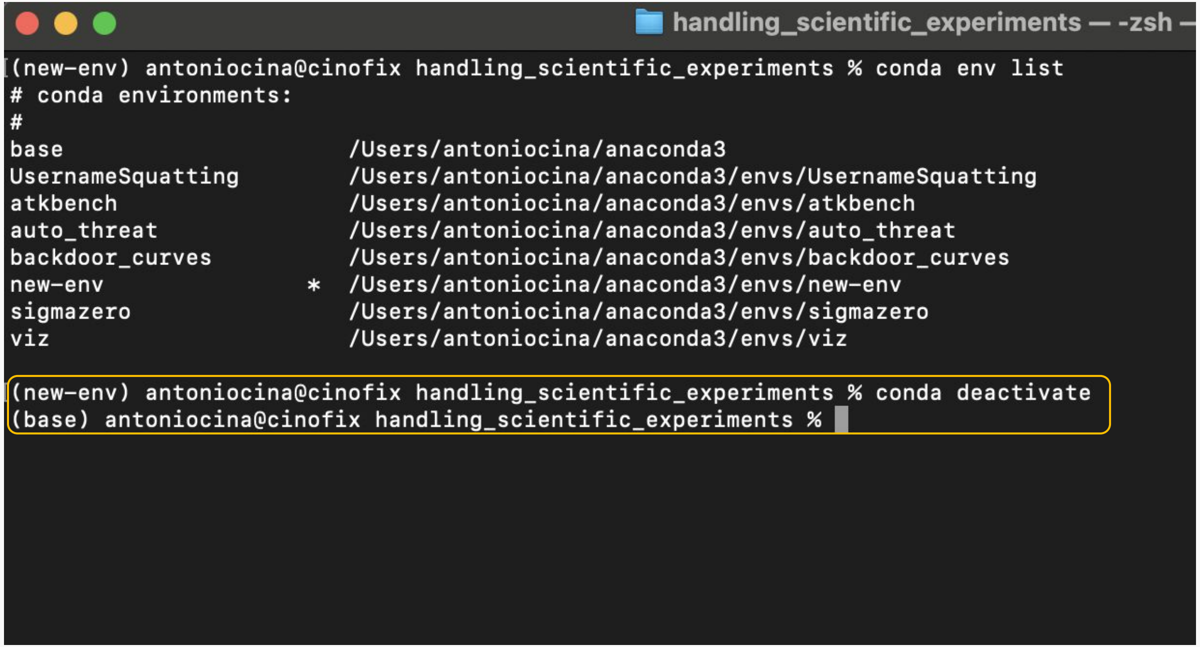


```
(new-env) antoniocina@cinofix handling_scientific_experiments % conda env list
# conda environments:
#
base                     /Users/antoniocina/anaconda3
UsernameSquatting        /Users/antoniocina/anaconda3/envs/UsernameSquatting
atkbench                 /Users/antoniocina/anaconda3/envs/atkbench
auto_threat              /Users/antoniocina/anaconda3/envs/auto_threat
backdoor_curves          /Users/antoniocina/anaconda3/envs/backdoor_curves
new-env               *  /Users/antoniocina/anaconda3/envs/new-env
sigmazero                /Users/antoniocina/anaconda3/envs/sigmazero
viz                      /Users/antoniocina/anaconda3/envs/viz

(new-env) antoniocina@cinofix handling_scientific_experiments %
```

Creating a new environment

`$ conda deactivate`

Deactivate the current environment and returns to the base.

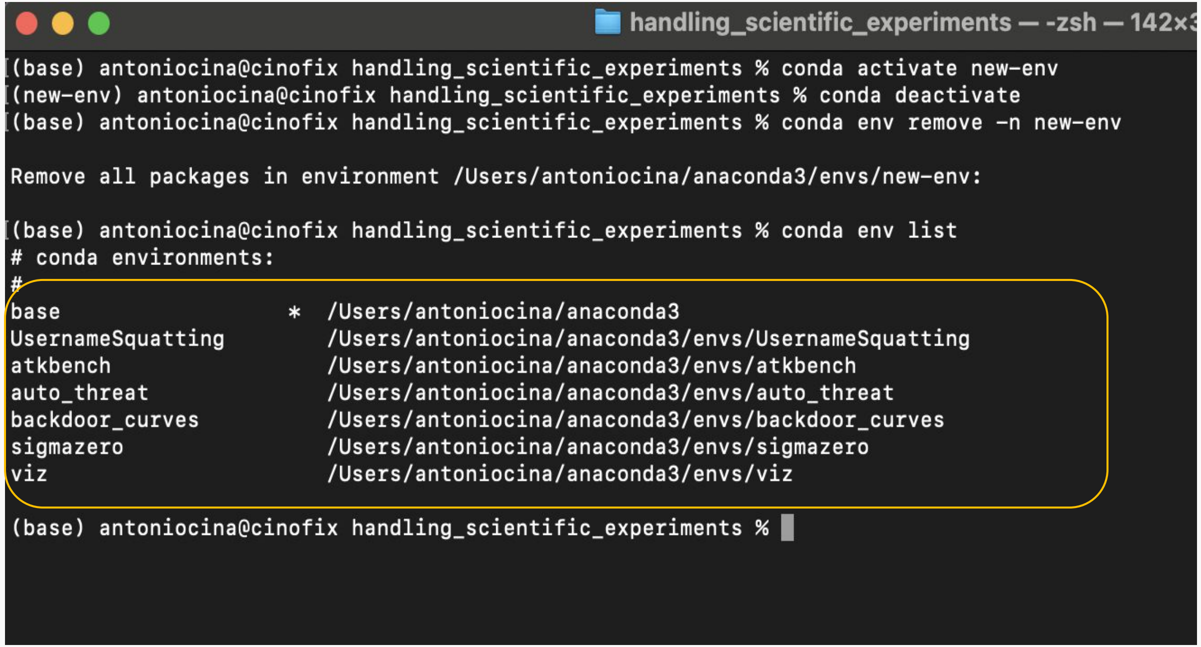


```
(new-env) antoniocina@cinofix handling_scientific_experiments % conda env list
# conda environments:
#
base                     /Users/antoniocina/anaconda3
UsernameSquatting        /Users/antoniocina/anaconda3/envs/UsernameSquatting
atkbench                 /Users/antoniocina/anaconda3/envs/atkbench
auto_threat              /Users/antoniocina/anaconda3/envs/auto_threat
backdoor_curves          /Users/antoniocina/anaconda3/envs/backdoor_curves
new-env               *  /Users/antoniocina/anaconda3/envs/new-env
sigmazero                /Users/antoniocina/anaconda3/envs/sigmazero
viz                      /Users/antoniocina/anaconda3/envs/viz

(new-env) antoniocina@cinofix handling_scientific_experiments % conda deactivate
(base) antoniocina@cinofix handling_scientific_experiments %
```

Creating a new environment

```
$ conda env remove -n new-env
```

Once deactivated, we can also remove an environment.



```
(base) antoniocina@cinofix handling_scientific_experiments % conda activate new-env
(new-env) antoniocina@cinofix handling_scientific_experiments % conda deactivate
(base) antoniocina@cinofix handling_scientific_experiments % conda env remove -n new-env

Remove all packages in environment /Users/antoniocina/anaconda3/envs/new-env:

(base) antoniocina@cinofix handling_scientific_experiments % conda env list
# conda environments:
#
base                  *  /Users/antoniocina/anaconda3
UsernameSquatting        /Users/antoniocina/anaconda3/envs/UsernameSquatting
atkbench                 /Users/antoniocina/anaconda3/envs/atkbench
auto_threat              /Users/antoniocina/anaconda3/envs/auto_threat
backdoor_curves          /Users/antoniocina/anaconda3/envs/backdoor_curves
sigmazero                /Users/antoniocina/anaconda3/envs/sigmazero
viz                      /Users/antoniocina/anaconda3/envs/viz

(base) antoniocina@cinofix handling_scientific_experiments %
```

Creating a new environment

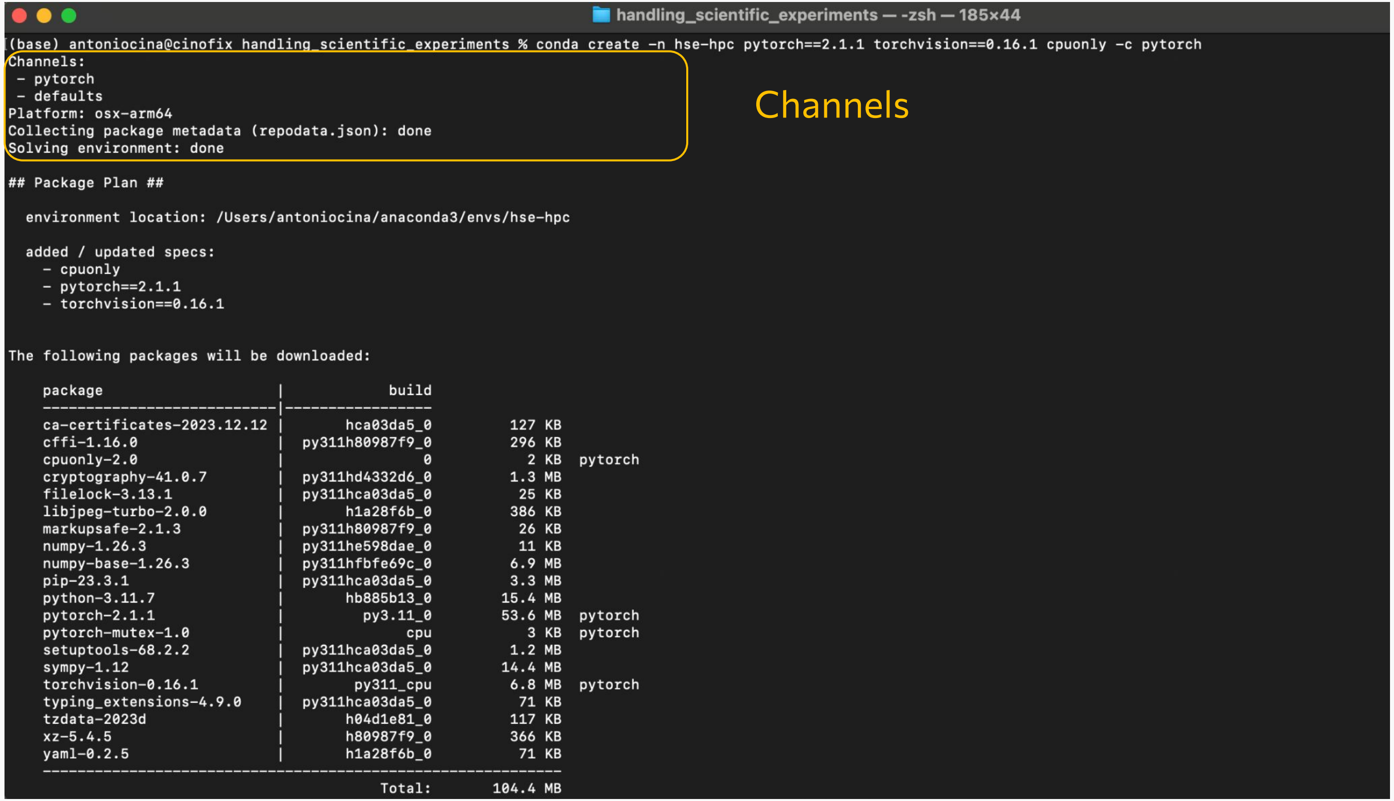
```
$ conda create -n hse-hpc pytorch==2.1.1 torchvision==0.16.1 cpuonly -c pytorch
```

Create a new environment with name hse-hpc.

The -c flag defines the channel.

The == specification defines the package version to install.

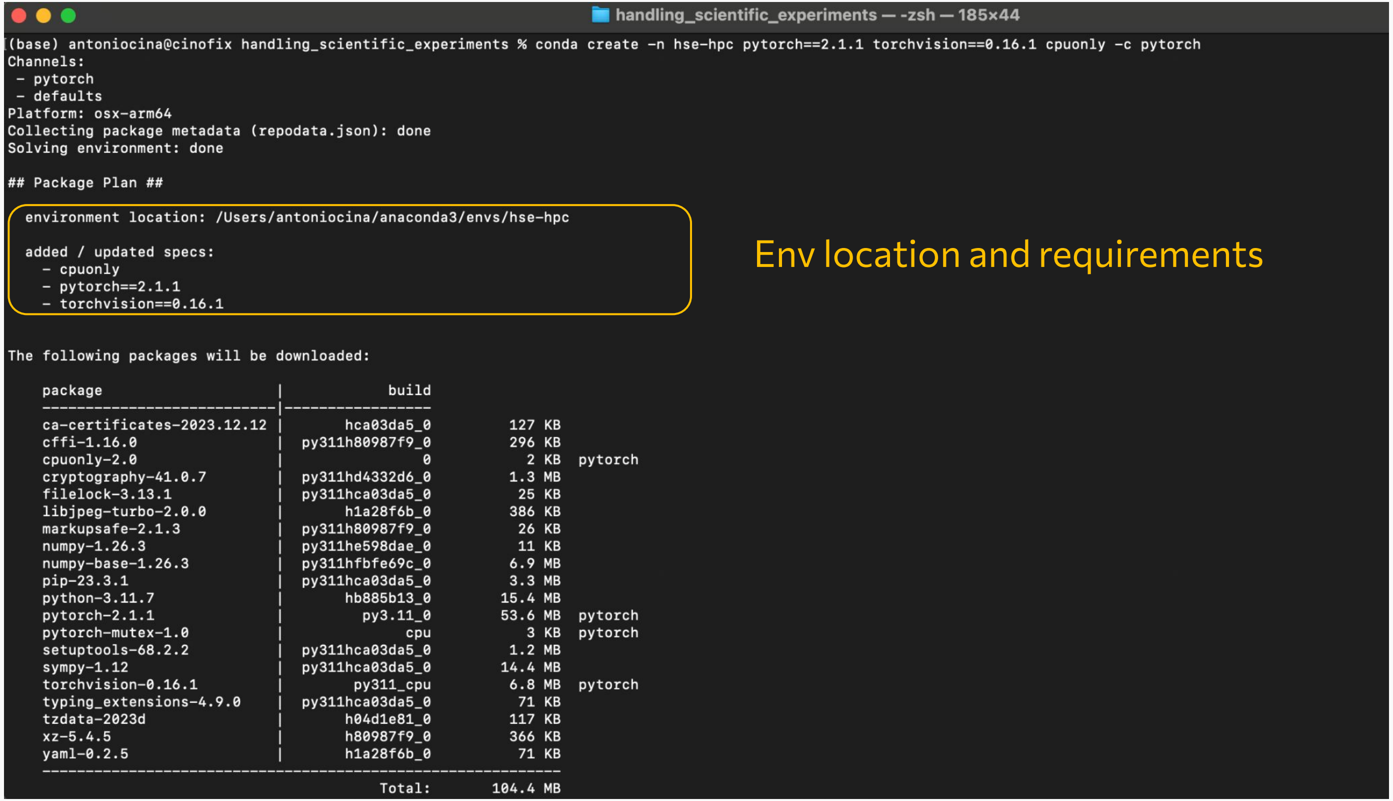
Creating a new environment



```
(base) antoniocina@cinofix handling_scientific_experiments % conda create -n hse-hpc pytorch==2.1.1 torchvision==0.16.1 cpuonly -c pytorch
Channels:
 - pytorch
 - defaults
Platform: osx-arm64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /Users/antoniocina/anaconda3/envs/hse-hpc

  added / updated specs:
    - cpuonly
    - pytorch==2.1.1
    - torchvision==0.16.1


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2023.12.12 |       hca03da5_0         127 KB
    cffi-1.16.0                |  py311h80987f9_0         296 KB
    cpuonly-2.0                |                0           2 KB  pytorch
    cryptography-41.0.7        |  py311hd4332d6_0         1.3 MB
    filelock-3.13.1            |  py311hca03da5_0          25 KB
    libjpeg-turbo-2.0.0        |       h1a28f6b_0         386 KB
    markupsafe-2.1.3           |  py311h80987f9_0          26 KB
    numpy-1.26.3               |  py311he598dae_0          11 KB
    numpy-base-1.26.3          |  py311hfbfe69c_0         6.9 MB
    pip-23.3.1                 |  py311hca03da5_0         3.3 MB
    python-3.11.7              |       hb885b13_0        15.4 MB
    pytorch-2.1.1              |         py3.11_0        53.6 MB  pytorch
    pytorch-mutex-1.0          |              cpu           3 KB  pytorch
    setuptools-68.2.2          |  py311hca03da5_0         1.2 MB
    sympy-1.12                 |  py311hca03da5_0        14.4 MB
    torchvision-0.16.1         |        py311_cpu         6.8 MB  pytorch
    typing_extensions-4.9.0    |  py311hca03da5_0          71 KB
    tzdata-2023d               |       h04d1e81_0         117 KB
    xz-5.4.5                   |       h80987f9_0         366 KB
    yaml-0.2.5                 |       h1a28f6b_0          71 KB
    ------------------------------------------------------------
                                           Total:       104.4 MB
```

Channels

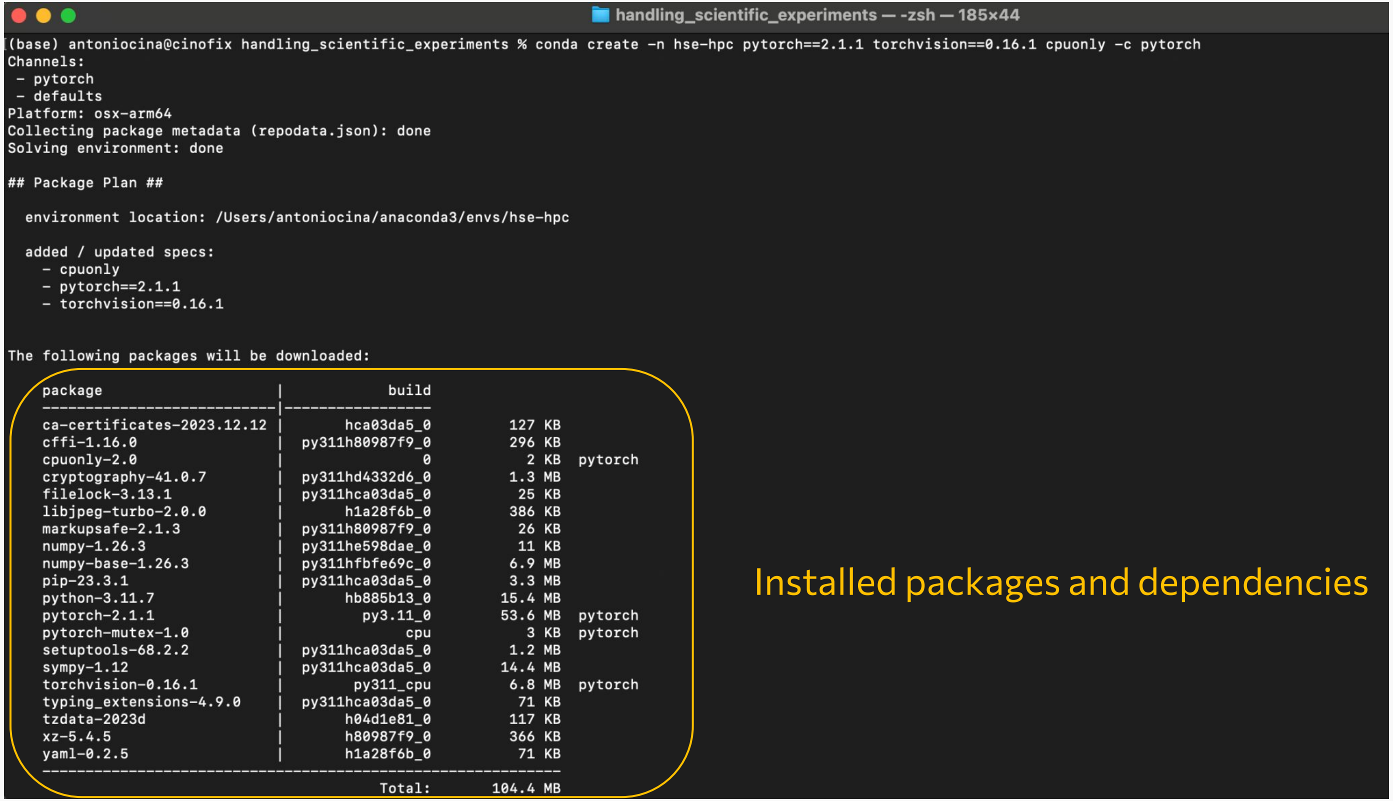
Creating a new environment



```
[(base) antoniocina@cinofix handling_scientific_experiments % conda create -n hse-hpc pytorch==2.1.1 torchvision==0.16.1 cpuonly -c pytorch
Channels:
 - pytorch
 - defaults
Platform: osx-arm64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /Users/antoniocina/anaconda3/envs/hse-hpc

  added / updated specs:
    - cpuonly
    - pytorch==2.1.1
    - torchvision==0.16.1


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2023.12.12 |       hca03da5_0         127 KB
    cffi-1.16.0                |  py311h80987f9_0         296 KB
    cpuonly-2.0                |                0           2 KB  pytorch
    cryptography-41.0.7        |  py311hd4332d6_0         1.3 MB
    filelock-3.13.1            |  py311hca03da5_0          25 KB
    libjpeg-turbo-2.0.0        |       h1a28f6b_0         386 KB
    markupsafe-2.1.3           |  py311h80987f9_0          26 KB
    numpy-1.26.3               |  py311he598dae_0          11 KB
    numpy-base-1.26.3          |  py311hfbfe69c_0         6.9 MB
    pip-23.3.1                 |  py311hca03da5_0         3.3 MB
    python-3.11.7              |       hb885b13_0        15.4 MB
    pytorch-2.1.1              |          py3.11_0        53.6 MB  pytorch
    pytorch-mutex-1.0          |              cpu           3 KB  pytorch
    setuptools-68.2.2          |  py311hca03da5_0         1.2 MB
    sympy-1.12                 |  py311hca03da5_0        14.4 MB
    torchvision-0.16.1         |        py311_cpu         6.8 MB  pytorch
    typing_extensions-4.9.0    |  py311hca03da5_0          71 KB
    tzdata-2023d               |       h04d1e81_0         117 KB
    xz-5.4.5                   |       h80987f9_0         366 KB
    yaml-0.2.5                 |       h1a28f6b_0          71 KB
    ------------------------------------------------------------
                                           Total:       104.4 MB
```

Env location and requirements

Creating a new environment



```
[(base) antoniocina@cinofix handling_scientific_experiments % conda create -n hse-hpc pytorch==2.1.1 torchvision==0.16.1 cpuonly -c pytorch
Channels:
 - pytorch
 - defaults
Platform: osx-arm64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /Users/antoniocina/anaconda3/envs/hse-hpc

  added / updated specs:
    - cpuonly
    - pytorch==2.1.1
    - torchvision==0.16.1


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2023.12.12 |       hca03da5_0         127 KB
    cffi-1.16.0                |  py311h80987f9_0         296 KB
    cpuonly-2.0                |                0           2 KB  pytorch
    cryptography-41.0.7        |  py311hd4332d6_0         1.3 MB
    filelock-3.13.1            |  py311hca03da5_0          25 KB
    libjpeg-turbo-2.0.0        |       h1a28f6b_0         386 KB
    markupsafe-2.1.3           |  py311h80987f9_0          26 KB
    numpy-1.26.3               |  py311he598dae_0          11 KB
    numpy-base-1.26.3          |  py311hfbfe69c_0         6.9 MB
    pip-23.3.1                 |  py311hca03da5_0         3.3 MB
    python-3.11.7              |       hb885b13_0        15.4 MB
    pytorch-2.1.1              |         py3.11_0        53.6 MB  pytorch
    pytorch-mutex-1.0          |              cpu           3 KB  pytorch
    setuptools-68.2.2          |  py311hca03da5_0         1.2 MB
    sympy-1.12                 |  py311hca03da5_0        14.4 MB
    torchvision-0.16.1         |        py311_cpu         6.8 MB  pytorch
    typing_extensions-4.9.0    |  py311hca03da5_0          71 KB
    tzdata-2023d               |       h04d1e81_0         117 KB
    xz-5.4.5                   |       h80987f9_0         366 KB
    yaml-0.2.5                 |       h1a28f6b_0          71 KB
    ------------------------------------------------------------
                                           Total:       104.4 MB
```

Installed packages and dependencies

Export environment

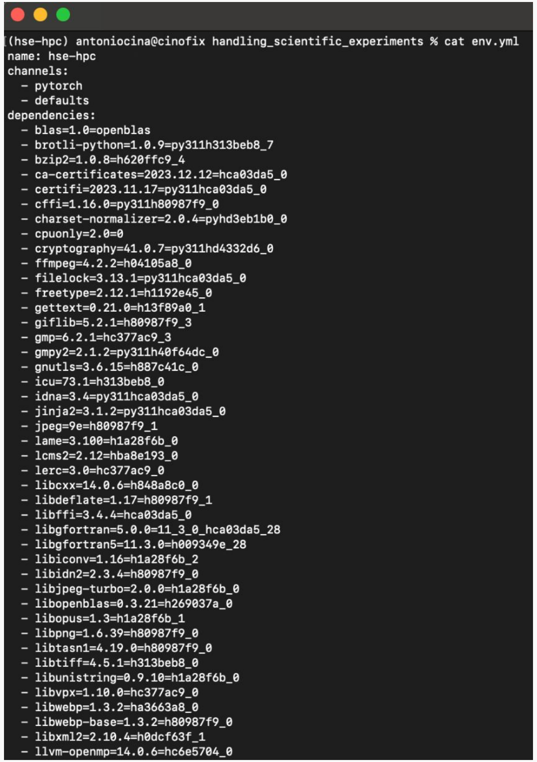
```
$ conda env export > env.yml
```

Export the active environment to a new yml file.

```
$ conda env create -f env.yml
```

Create a new environment from a yml file.

The -f flag serves to specify the file describing env dependencies.



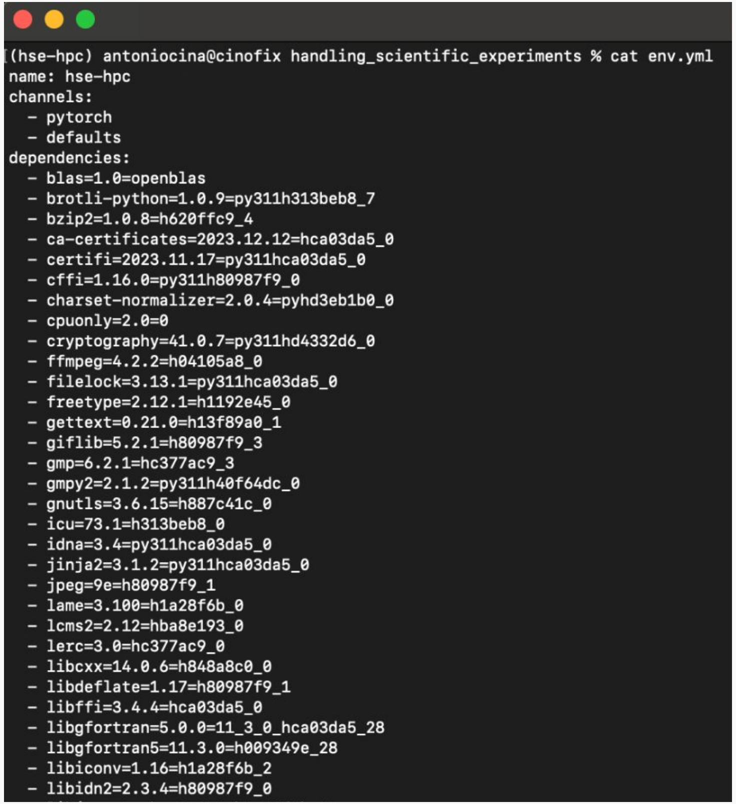
```
(hse-hpc) antoniocina@cinofix handling_scientific_experiments % cat env.yml
name: hse-hpc
channels:
  - pytorch
  - defaults
dependencies:
  - blas=1.0=openblas
  - brotli-python=1.0.9=py311h313beb8_7
  - bzip2=1.0.8=h620ffc9_4
  - ca-certificates=2023.12.12=hca03da5_0
  - certifi=2023.11.17=py311hca03da5_0
  - cffi=1.16.0=py311h80987f9_0
  - charset-normalizer=2.0.4=pyhd3eb1b0_0
  - cpuonly=2.0=0
  - cryptography=41.0.7=py311hd4332d6_0
  - ffmpeg=4.2.2=h04105a8_0
  - filelock=3.13.1=py311hca03da5_0
  - freetype=2.12.1=h1192e45_0
  - gettext=0.21.0=h13f89a0_1
  - giflib=5.2.1=h80987f9_3
  - gmp=6.2.1=hc377ac9_3
  - gmpy2=2.1.2=py311h40f64dc_0
  - gnutls=3.6.15=h887c41c_0
  - icu=73.1=h313beb8_0
  - idna=3.4=py311hca03da5_0
  - jinja2=3.1.2=py311hca03da5_0
  - jpeg=9e=h80987f9_1
  - lame=3.100=h1a28f6b_0
  - lcms2=2.12=hba8e193_0
  - lerc=3.0=hc377ac9_0
  - libcxx=14.0.6=h848a8c0_0
  - libdeflate=1.17=h80987f9_1
  - libffi=3.4.4=hca03da5_0
  - libgfortran=5.0.0=11_3_0_hca03da5_28
  - libgfortran5=11.3.0=h009349e_28
  - libiconv=1.16=h1a28f6b_2
  - libidn2=2.3.4=h80987f9_0
  - libjpeg-turbo=2.0.0=h1a28f6b_0
  - libopenblas=0.3.21=h269037a_0
  - libopus=1.3=h1a28f6b_1
  - libpng=1.6.39=h80987f9_0
  - libtasn1=4.19.0=h80987f9_0
  - libtiff=4.5.1=h313beb8_0
  - libunistring=0.9.10=h1a28f6b_0
  - libvpx=1.10.0=hc377ac9_0
  - libwebp=1.3.2=ha3663a8_0
  - libwebp-base=1.3.2=h80987f9_0
  - libxml2=2.10.4=h0dcf63f_1
  - llvm-openmp=14.0.6=hc6e5704_0
```

Personal tips: Export environment

```
$ conda env export --no-build | grep -v "^prefix: " > env.yml
```

–no-build removes the build information, which sometimes creates conflicts.

The grep -v "^prefix: " filters out the last row, describing the environment path. This is not useful for other developers!



```
(hse-hpc) antoniocina@cinofix handling_scientific_experiments % cat env.yml
name: hse-hpc
channels:
  - pytorch
  - defaults
dependencies:
  - blas=1.0=openblas
  - brotli-python=1.0.9=py311h313beb8_7
  - bzip2=1.0.8=h620ffc9_4
  - ca-certificates=2023.12.12=hca03da5_0
  - certifi=2023.11.17=py311hca03da5_0
  - cffi=1.16.0=py311h80987f9_0
  - charset-normalizer=2.0.4=pyhd3eb1b0_0
  - cpuonly=2.0=0
  - cryptography=41.0.7=py311hd4332d6_0
  - ffmpeg=4.2.2=h04105a8_0
  - filelock=3.13.1=py311hca03da5_0
  - freetype=2.12.1=h1192e45_0
  - gettext=0.21.0=h13f89a0_1
  - giflib=5.2.1=h80987f9_3
  - gmp=6.2.1=hc377ac9_3
  - gmpy2=2.1.2=py311h40f64dc_0
  - gnutls=3.6.15=h887c41c_0
  - icu=73.1=h313beb8_0
  - idna=3.4=py311hca03da5_0
  - jinja2=3.1.2=py311hca03da5_0
  - jpeg=9e=h80987f9_1
  - lame=3.100=h1a28f6b_0
  - lcms2=2.12=hba8e193_0
  - lerc=3.0=hc377ac9_0
  - libcxx=14.0.6=h848a8c0_0
  - libdeflate=1.17=h80987f9_1
  - libffi=3.4.4=hca03da5_0
  - libgfortran=5.0.0=11_3_0_hca03da5_28
  - libgfortran5=11.3.0=h009349e_28
  - libiconv=1.16=h1a28f6b_2
  - libidn2=2.3.4=h80987f9_0
```

Personal tips: LIBMAMBA Solver

mamba is a replacement for the conda solver that works to improve certain aspects of the conda infrastructure.

It is able to perform much faster installations (helping loads with 'environment resolution' steps).

We install mamba with conda:

```
$ conda install -n base conda-libmamba-solver
```

You can always use `$ --solver=classic` when creating the environment to re-enable the classic solver temporarily for specific operations.

High Performance Computing

Underlying Problem

Research problems involve **extensive computations** that surpass the capabilities of laptop computers.

Insufficient memory, limited CPU cores, and inadequate disk space can hinder the execution of complex tasks.

Resource constraints become evident when computations require **parallel processing**, or **GPUs** acceleration.

Underlying Problem

Continuous and resource-intensive computations may lead to higher **energy consumption**, impacting the overall operational costs associated with experimentation.

Intensive computations pose a considerable **risk of damaging** computer hardware.

Local setups are susceptible to **voltage drops**, introducing the risk of data loss and system instability. Unstable power conditions can result in unexpected shutdowns, causing data corruption or loss, and potential damage to hardware components.

High Performance Computing

High Performance Computing most generally refers to the practice of **aggregating computing power** in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation.

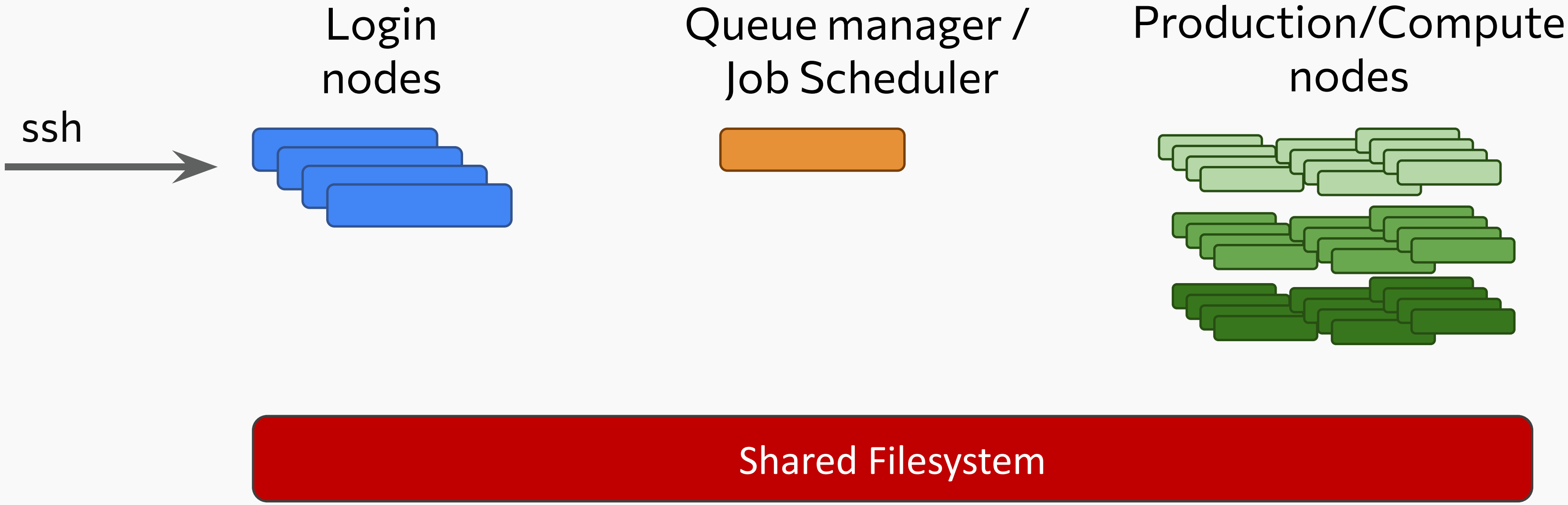
A HPC cluster is a large computer composed of a collection of **many separate servers** which are called nodes.

There may be different types of nodes for different types of tasks.
For example,

- Node 1, 2 equipped with 2 x NVIDIA A100 each;
- Node 3, 4 equipped with 8 x NVIDIA Quadro RTX 8000;

Nodes are typically connected to one another with a fast interconnect.

High Performance Computing



ssh

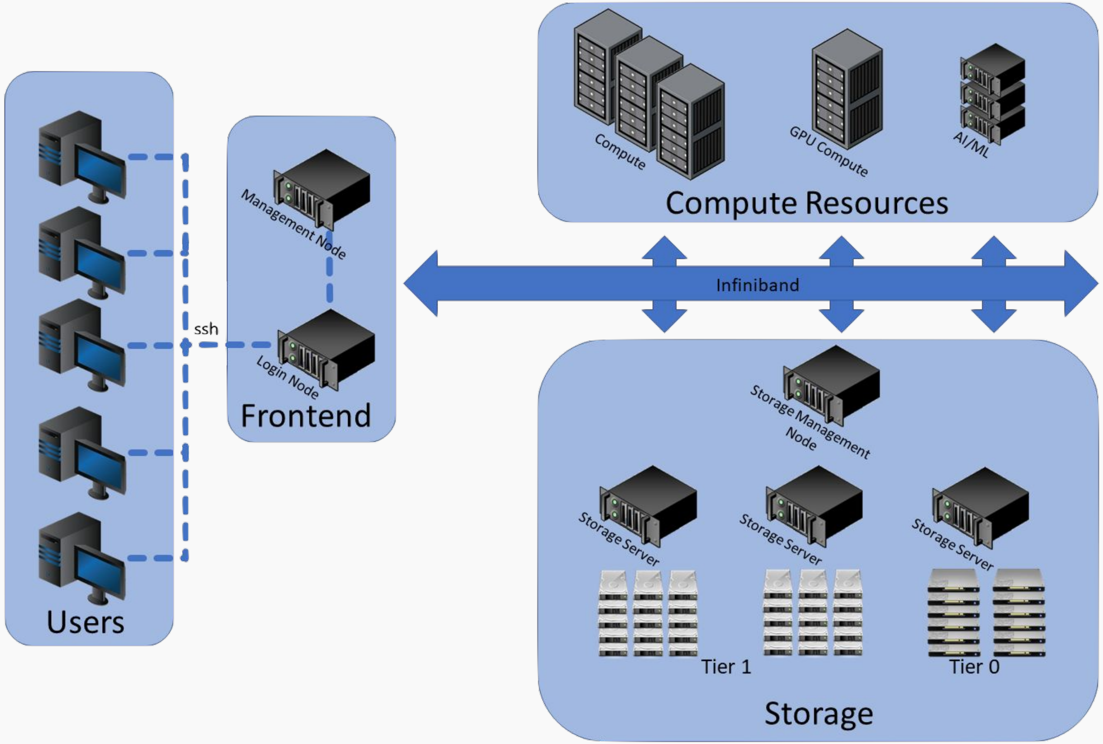
Login nodes

Queue manager / Job Scheduler

Production/Compute nodes

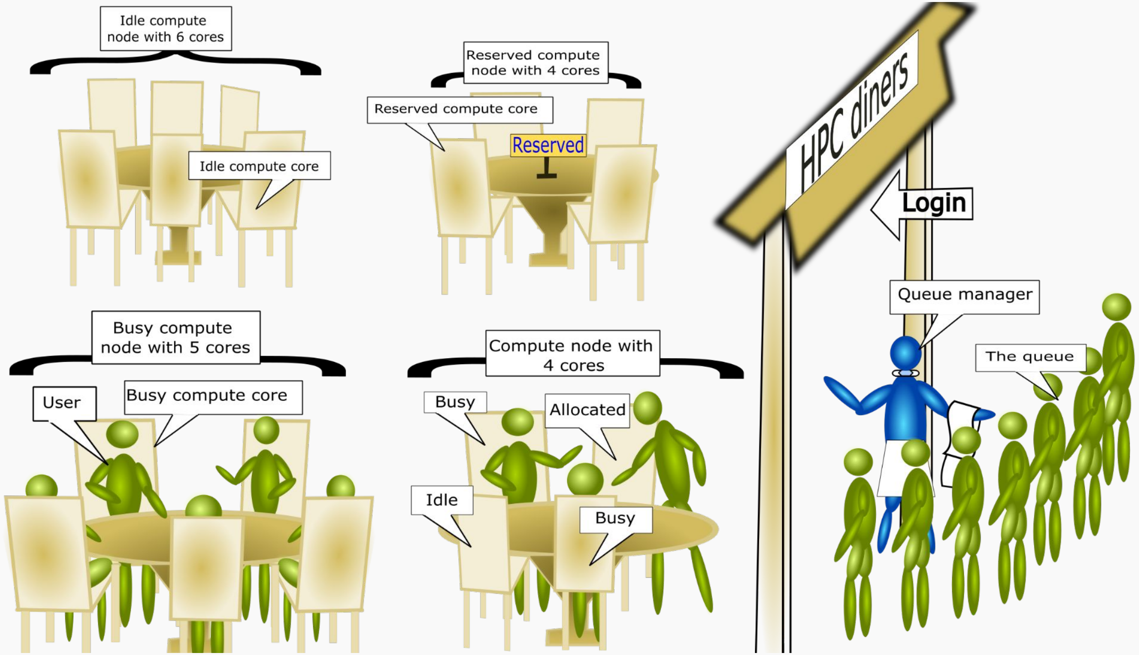
Shared Filesystem

High Performance Computing



Queue manager / Job Scheduler

An HPC system might have thousands of nodes and thousands of users. How do we decide who gets what and when? How do we ensure that a task is run with the resources it needs?



Slurm

Slurm

Slurm

ENJOY
Slurm
SODA
IT'S HIGHLY ADD
TIVE!
VOTED #1 SOFT DRINK OF THE 3
WHIMMY WHAM WHAM WOZZLE!
YOU WIN!
SLURMS McKENZIE
Slurm

Slurm

Slurm, short for "Simple Linux Utility for Resource Management," is an open-source **job scheduler** and **resource management system**.

Slurm can start multiple jobs on a single node, or a single job on multiple nodes.

Slurm coordinates and optimizes the allocation of resources such as CPUs, GPUs, and memory to users' jobs.

It ensures fair usage, prevents resource conflicts, and optimizes the utilization of available computing power.

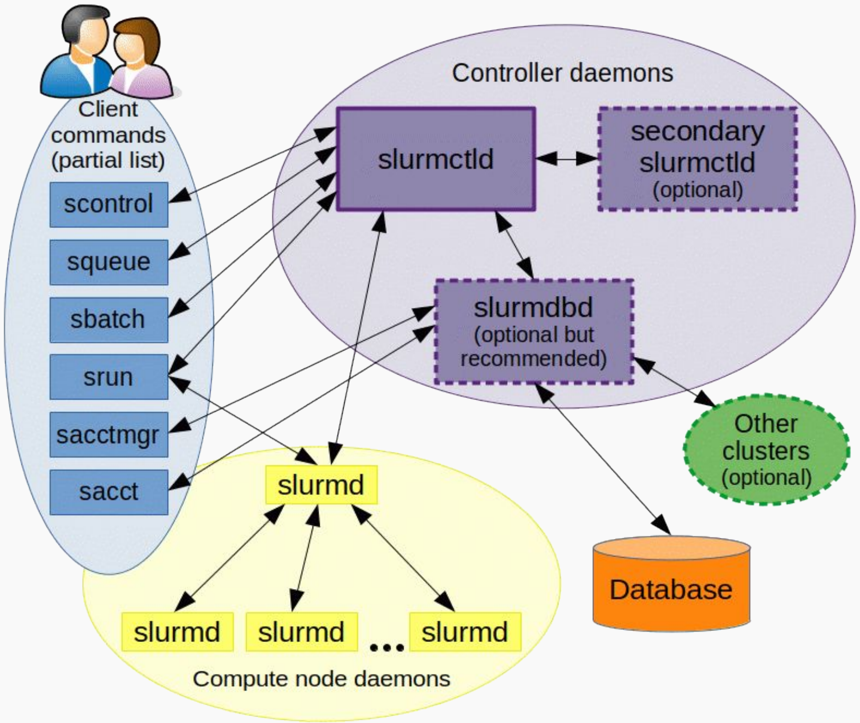
Slurm technicalities

slurmd is the Slurm daemon responsible for managing and executing tasks on the compute nodes.

The daemon monitors the resource utilization on its assigned node, ensuring that jobs are allocated resources within the specified limits

slurmctld is the Slurm controller that is responsible for job scheduling and allocation.

It decides how to distribute jobs across the compute nodes based on the specified policies, resource availability, and job priorities.



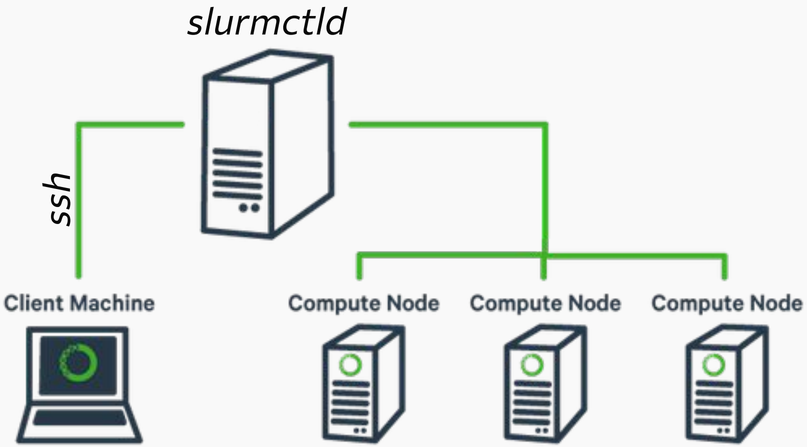
Client commands (partial list)
scontrol
squeue
sbatch
srun
sacctmgr
sacct
Controller daemons
slurmctld
secondary slurmctld (optional)
slurmdbd (optional but recommended)
Other clusters (optional)
Database
slurmd
slurmd
slurmd
...
slurmd
Compute node daemons

Slurm + CONDA

Conda within the HPC cluster ensures consistent environments across all compute nodes.

Multiple users may have different software requirements, Conda makes it easy to set up and share environments with the necessary dependencies.

Test and debug locally, experiment on the cluster!!



Slurm: The UniVE cluster experience

We will delve into the practical aspects of utilizing Slurm on the UniVE workstations.

The UniVE cluster boast:

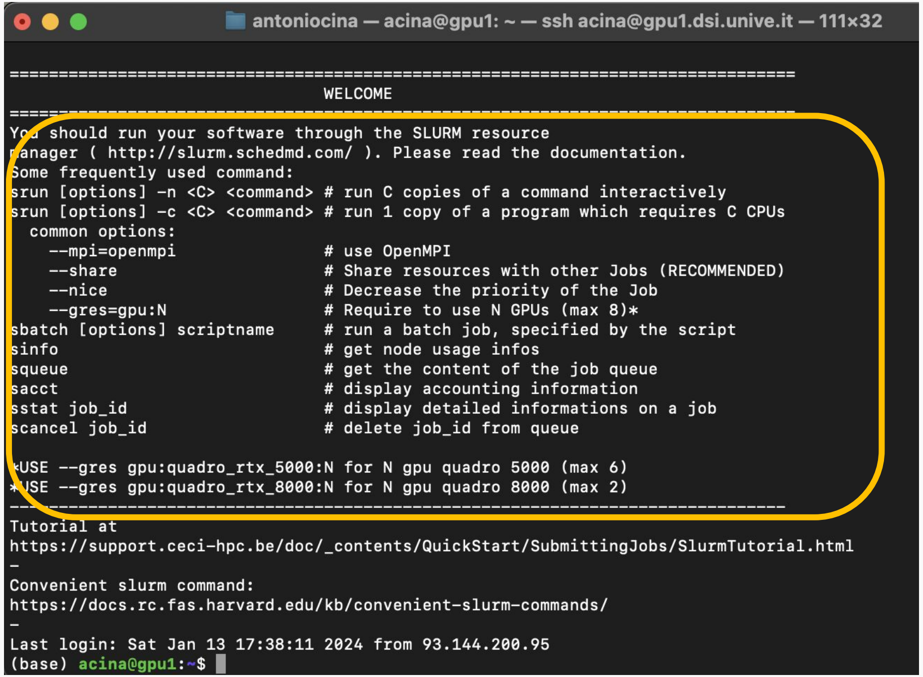
- 2 NVIDIA GTX 8000 GPUs with 48GB of memory each, suitable for memory-intensive workloads.
- 6 NVIDIA GTX 5000 GPUs, each with 16GB of memory, provide a robust solution for various GPU-accelerated computations.

Slurm: The UniVE cluster experience

List of available commands on slurm.

Tutorial at
https://support.ceci-hpc.be/doc/_contents/QuickStart/SubmittingJobs/SlurmTutorial.html

Slurm command:
https://docs.rc.fas.harvard.edu/kb/convenient-slurm-commands/

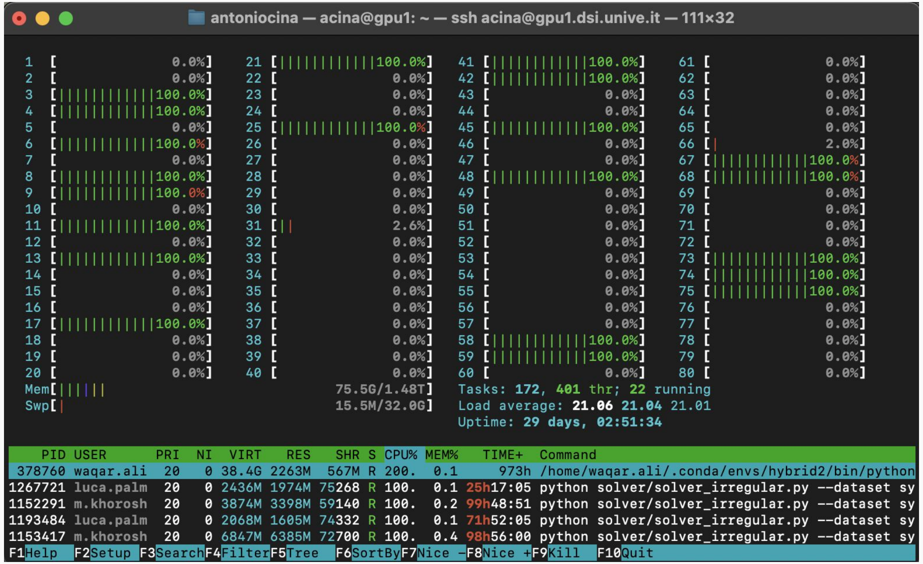

antoniocina — acina@gpu1: ~ — ssh acina@gpu1.dsi.unive.it — 111×32

```
================================================================================
                                WELCOME
================================================================================
You should run your software through the SLURM resource
manager ( http://slurm.schedmd.com/ ). Please read the documentation.
Some frequently used command:
srun [options] -n <C> <command> # run C copies of a command interactively
srun [options] -c <C> <command> # run 1 copy of a program which requires C CPUs
  common options:
    --mpi=openmpi               # use OpenMPI
    --share                     # Share resources with other Jobs (RECOMMENDED)
    --nice                      # Decrease the priority of the Job
    --gres=gpu:N                # Require to use N GPUs (max 8)*
sbatch [options] scriptname     # run a batch job, specified by the script
sinfo                           # get node usage infos
squeue                          # get the content of the job queue
sacct                           # display accounting information
sstat job_id                    # display detailed informations on a job
scancel job_id                  # delete job_id from queue

*USE --gres gpu:quadro_rtx_5000:N for N gpu quadro 5000 (max 6)
*USE --gres gpu:quadro_rtx_8000:N for N gpu quadro 8000 (max 2)
-------------------------------------------------------------------------------
Tutorial at
https://support.ceci-hpc.be/doc/_contents/QuickStart/SubmittingJobs/SlurmTutorial.html
-
Convenient slurm command:
https://docs.rc.fas.harvard.edu/kb/convenient-slurm-commands/
-
Last login: Sat Jan 13 17:38:11 2024 from 93.144.200.95
(base) acina@gpu1:~$ 
```

Visualize hardware usage

Before launching any calculation task, it is advisable to **check the state of the hardware** to ensure correct and efficient operations.

`$ htop` is a powerful command-line utility that provides an interactive and real-time overview of system processes, memories utilization, and CPUs status.

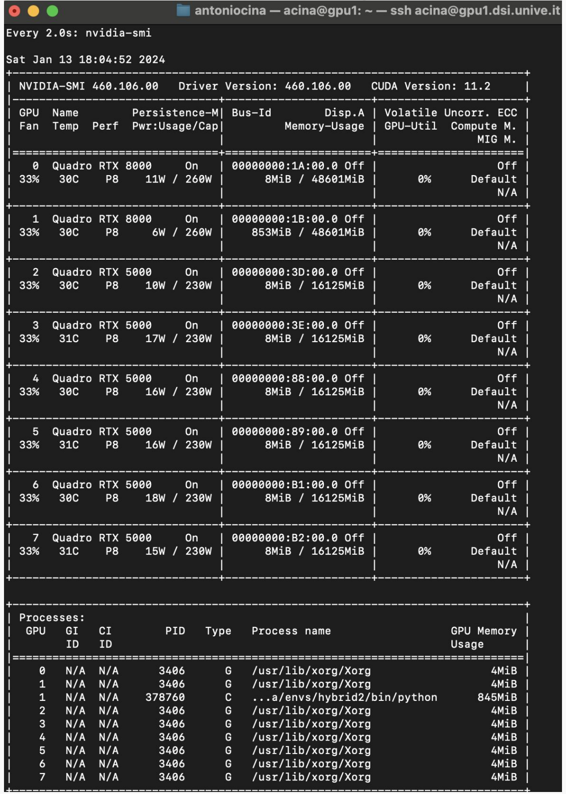

antoniocina — acina@gpu1: ~ — ssh acina@gpu1.dsi.unive.it — 111×32

```
  1  [              0.0%]   21 [||||||||||||100.0%]   41 [||||||||||||100.0%]   61 [              0.0%]
  2  [              0.0%]   22 [              0.0%]   42 [||||||||||||100.0%]   62 [              0.0%]
  3  [||||||||||||100.0%]   23 [              0.0%]   43 [              0.0%]   63 [              0.0%]
  4  [||||||||||||100.0%]   24 [              0.0%]   44 [              0.0%]   64 [              0.0%]
  5  [              0.0%]   25 [||||||||||||100.0%]   45 [||||||||||||100.0%]   65 [              0.0%]
  6  [||||||||||||100.0%]   26 [              0.0%]   46 [              0.0%]   66 [|             2.0%]
  7  [              0.0%]   27 [              0.0%]   47 [              0.0%]   67 [||||||||||||100.0%]
  8  [||||||||||||100.0%]   28 [              0.0%]   48 [||||||||||||100.0%]   68 [||||||||||||100.0%]
  9  [||||||||||||100.0%]   29 [              0.0%]   49 [              0.0%]   69 [              0.0%]
 10  [              0.0%]   30 [              0.0%]   50 [              0.0%]   70 [              0.0%]
 11  [||||||||||||100.0%]   31 [||            2.6%]   51 [              0.0%]   71 [              0.0%]
 12  [              0.0%]   32 [              0.0%]   52 [              0.0%]   72 [              0.0%]
 13  [||||||||||||100.0%]   33 [              0.0%]   53 [              0.0%]   73 [||||||||||||100.0%]
 14  [              0.0%]   34 [              0.0%]   54 [              0.0%]   74 [||||||||||||100.0%]
 15  [              0.0%]   35 [              0.0%]   55 [              0.0%]   75 [||||||||||||100.0%]
 16  [              0.0%]   36 [              0.0%]   56 [              0.0%]   76 [              0.0%]
 17  [||||||||||||100.0%]   37 [              0.0%]   57 [              0.0%]   77 [              0.0%]
 18  [              0.0%]   38 [              0.0%]   58 [||||||||||||100.0%]   78 [              0.0%]
 19  [              0.0%]   39 [              0.0%]   59 [||||||||||||100.0%]   79 [              0.0%]
 20  [              0.0%]   40 [              0.0%]   60 [              0.0%]   80 [              0.0%]
 Mem[||||||                          75.5G/1.48T]   Tasks: 172, 401 thr; 22 running
 Swp[|                               15.5M/32.0G]   Load average: 21.06 21.04 21.01
                                                    Uptime: 29 days, 02:51:34

    PID USER       PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 378760 waqar.ali   20   0 38.4G 2263M  567M R 200.  0.1     973h /home/waqar.ali/.conda/envs/hybrid2/bin/python
1267721 luca.palm   20   0 2436M 1974M 75268 R 100.  0.1 25h17:05 python solver/solver_irregular.py --dataset sy
1152291 m.khorosh   20   0 3874M 3398M 59140 R 100.  0.2 99h48:51 python solver/solver_irregular.py --dataset sy
1193484 luca.palm   20   0 2068M 1605M 74332 R 100.  0.1 71h52:05 python solver/solver_irregular.py --dataset sy
1153417 m.khorosh   20   0 6847M 6385M 72700 R 100.  0.4 98h56:00 python solver/solver_irregular.py --dataset sy
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Visualize hardware usage

`$ nvidia-smi` is a command-line tool provided by NVIDIA for monitoring and managing GPU devices.

`$ watch nvidia-smi` provides detailed real-time information on GPU utilization, memory usage, temperature, and processes using the GPUs.

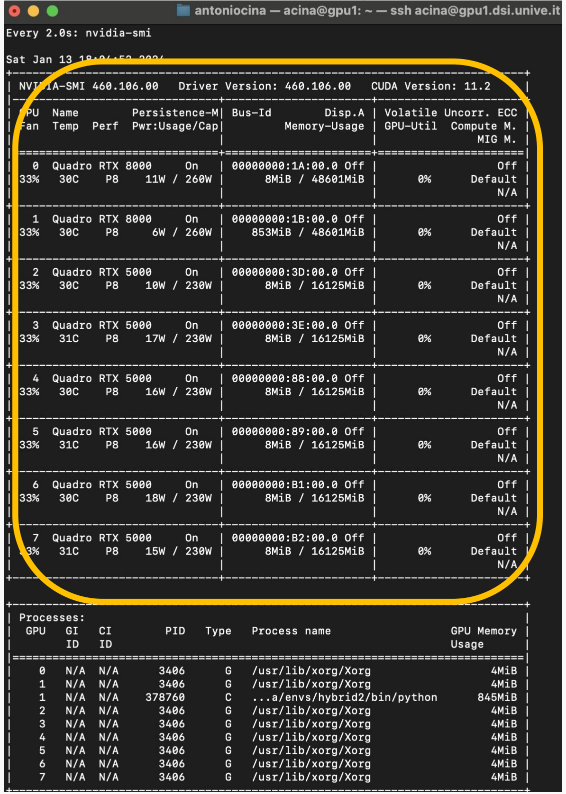


```
Every 2.0s: nvidia-smi

Sat Jan 13 18:04:52 2024
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.106.00   Driver Version: 460.106.00   CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Quadro RTX 8000     On   | 00000000:1A:00.0 Off |                  Off |
| 33%   30C    P8    11W / 260W |      8MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Quadro RTX 8000     On   | 00000000:1B:00.0 Off |                  Off |
| 33%   30C    P8     6W / 260W |    853MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  Quadro RTX 5000     On   | 00000000:3D:00.0 Off |                  Off |
| 33%   30C    P8    10W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  Quadro RTX 5000     On   | 00000000:3E:00.0 Off |                  Off |
| 33%   31C    P8    17W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   4  Quadro RTX 5000     On   | 00000000:88:00.0 Off |                  Off |
| 33%   30C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   5  Quadro RTX 5000     On   | 00000000:89:00.0 Off |                  Off |
| 33%   31C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   6  Quadro RTX 5000     On   | 00000000:B1:00.0 Off |                  Off |
| 33%   30C    P8    18W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   7  Quadro RTX 5000     On   | 00000000:B2:00.0 Off |                  Off |
| 33%   31C    P8    15W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+


+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A    378760      C   ...a/envs/hybrid2/bin/python      845MiB |
|    2   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    3   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    4   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    5   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    6   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    7   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
+-----------------------------------------------------------------------------+
```

Visualize hardware usage

The first block shows the GPUs status, i.e.:

- their name;
- temperature;
- energy consumption;
- memory usage;
- percentage of utility.



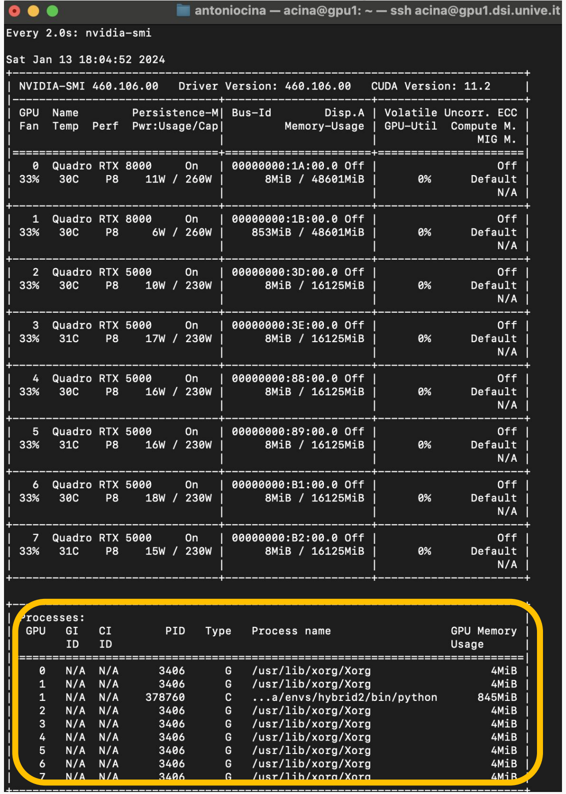
antoniocina — acina@gpu1: ~ — ssh acina@gpu1.dsi.unive.it

```
Every 2.0s: nvidia-smi

Sat Jan 13 18:04:52 2024
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.106.00   Driver Version: 460.106.00   CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Quadro RTX 8000     On   | 00000000:1A:00.0 Off |                  Off |
| 33%   30C    P8    11W / 260W |      8MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Quadro RTX 8000     On   | 00000000:1B:00.0 Off |                  Off |
| 33%   30C    P8     6W / 260W |    853MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  Quadro RTX 5000     On   | 00000000:3D:00.0 Off |                  Off |
| 33%   30C    P8    10W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  Quadro RTX 5000     On   | 00000000:3E:00.0 Off |                  Off |
| 33%   31C    P8    17W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   4  Quadro RTX 5000     On   | 00000000:88:00.0 Off |                  Off |
| 33%   30C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   5  Quadro RTX 5000     On   | 00000000:89:00.0 Off |                  Off |
| 33%   31C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   6  Quadro RTX 5000     On   | 00000000:B1:00.0 Off |                  Off |
| 33%   30C    P8    18W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   7  Quadro RTX 5000     On   | 00000000:B2:00.0 Off |                  Off |
| 33%   31C    P8    15W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A    378760      C   ...a/envs/hybrid2/bin/python      845MiB |
|    2   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    3   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    4   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    5   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    6   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    7   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
+-----------------------------------------------------------------------------+
```

Visualize hardware usage

The second block shows:

- the GPU id;
- processor identifier using the GPU;
- the type of processes such as “C” (Compute), “G” (Graphics), and “C+G” (Compute and Graphics context).
- process name;
- GPU memory usage;



```
Every 2.0s: nvidia-smi

Sat Jan 13 18:04:52 2024
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.106.00   Driver Version: 460.106.00   CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Quadro RTX 8000     On   | 00000000:1A:00.0 Off |                  Off |
| 33%   30C    P8    11W / 260W |      8MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Quadro RTX 8000     On   | 00000000:1B:00.0 Off |                  Off |
| 33%   30C    P8     6W / 260W |    853MiB / 48601MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  Quadro RTX 5000     On   | 00000000:3D:00.0 Off |                  Off |
| 33%   30C    P8    10W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  Quadro RTX 5000     On   | 00000000:3E:00.0 Off |                  Off |
| 33%   31C    P8    17W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   4  Quadro RTX 5000     On   | 00000000:88:00.0 Off |                  Off |
| 33%   30C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   5  Quadro RTX 5000     On   | 00000000:89:00.0 Off |                  Off |
| 33%   31C    P8    16W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   6  Quadro RTX 5000     On   | 00000000:B1:00.0 Off |                  Off |
| 33%   30C    P8    18W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   7  Quadro RTX 5000     On   | 00000000:B2:00.0 Off |                  Off |
| 33%   31C    P8    15W / 230W |      8MiB / 16125MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    1   N/A  N/A    378760      C   ...a/envs/hybrid2/bin/python      845MiB |
|    2   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    3   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    4   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    5   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    6   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
|    7   N/A  N/A      3406      G   /usr/lib/xorg/Xorg                  4MiB |
+-----------------------------------------------------------------------------+
```

Running experiments with Slurm

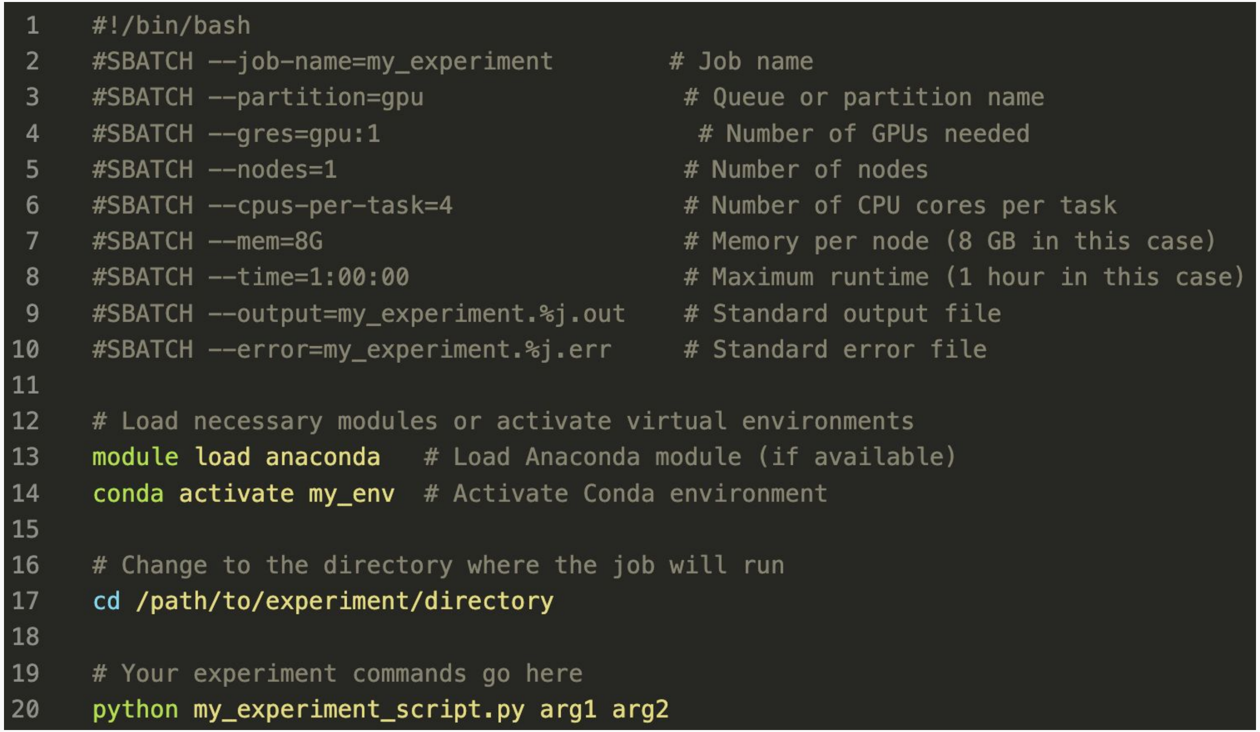
Slurm batch script defines the job parameters, resource requirements, and the commands to be executed.

SBATCH **Directives** specify various job parameters:

- --job-name: A user-defined name for the job.
- --partition: The queue or partition on which the job should run.
- --gres: The type and quantity of resources (GPUs in this case).
- --nodes: The number of nodes requested.
- --cpus-per-task: The number of CPU cores requested per task.
- --mem: The memory allocated per node.
- --time: The maximum runtime for the job.
- --output and --error: File paths for standard output and standard error.

Running experiments with Slurm

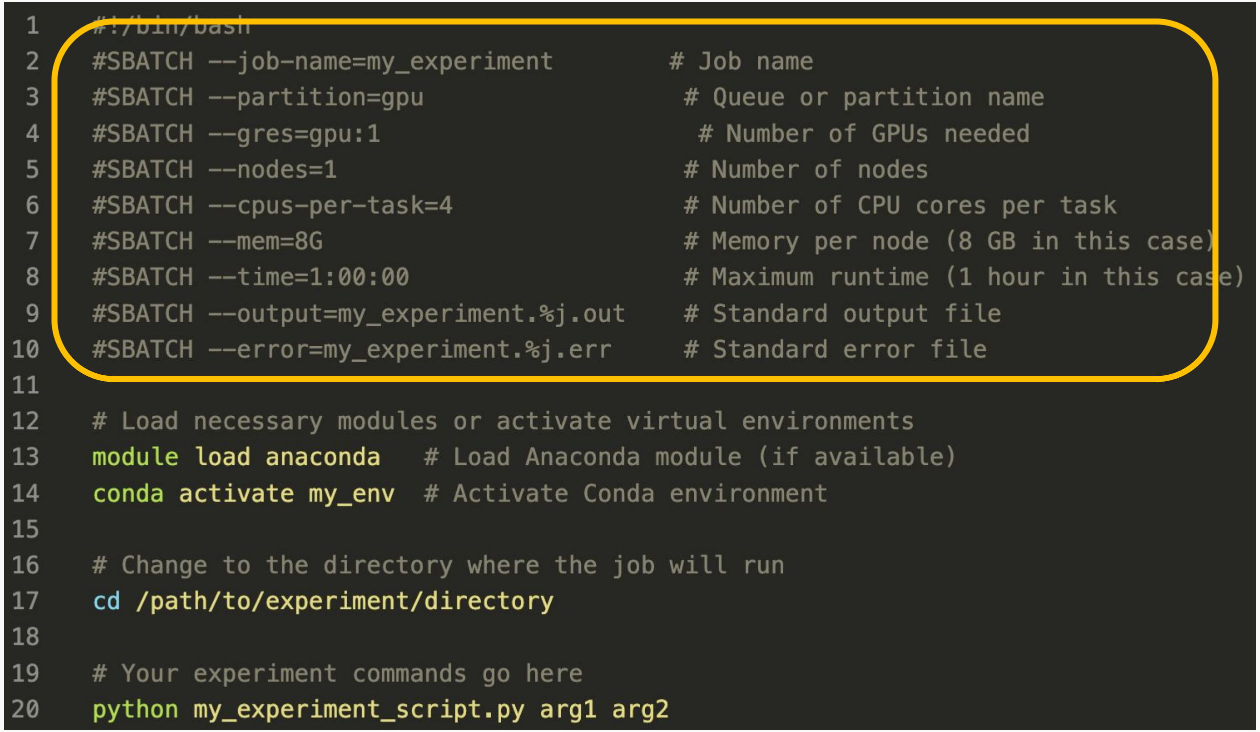
run_example.slurm



```
#!/bin/bash
#SBATCH --job-name=my_experiment        # Job name
#SBATCH --partition=gpu                  # Queue or partition name
#SBATCH --gres=gpu:1                      # Number of GPUs needed
#SBATCH --nodes=1                        # Number of nodes
#SBATCH --cpus-per-task=4                # Number of CPU cores per task
#SBATCH --mem=8G                         # Memory per node (8 GB in this case)
#SBATCH --time=1:00:00                   # Maximum runtime (1 hour in this case)
#SBATCH --output=my_experiment.%j.out    # Standard output file
#SBATCH --error=my_experiment.%j.err     # Standard error file

# Load necessary modules or activate virtual environments
module load anaconda   # Load Anaconda module (if available)
conda activate my_env  # Activate Conda environment

# Change to the directory where the job will run
cd /path/to/experiment/directory

# Your experiment commands go here
python my_experiment_script.py arg1 arg2
```

Running experiments with Slurm

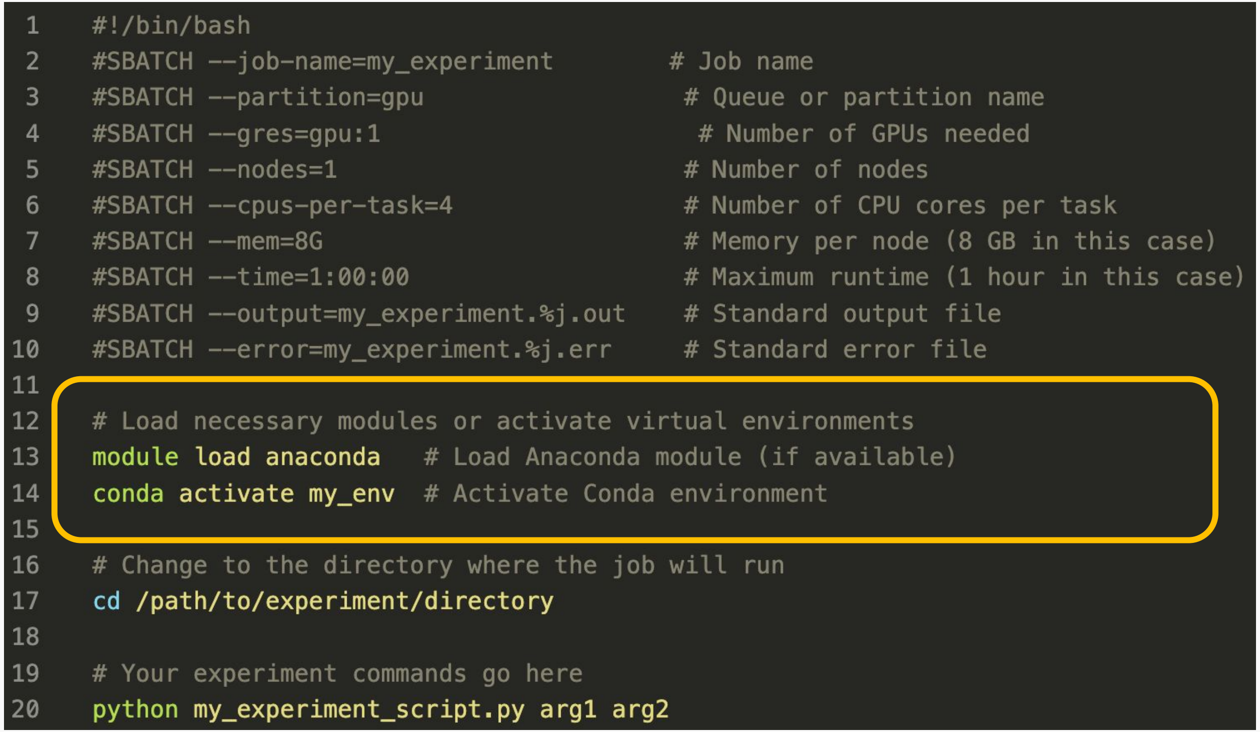
Slurm directives for GPU, CPU and memory allocation.



```
#!/bin/bash
#SBATCH --job-name=my_experiment        # Job name
#SBATCH --partition=gpu                  # Queue or partition name
#SBATCH --gres=gpu:1                      # Number of GPUs needed
#SBATCH --nodes=1                        # Number of nodes
#SBATCH --cpus-per-task=4                # Number of CPU cores per task
#SBATCH --mem=8G                         # Memory per node (8 GB in this case)
#SBATCH --time=1:00:00                   # Maximum runtime (1 hour in this case)
#SBATCH --output=my_experiment.%j.out    # Standard output file
#SBATCH --error=my_experiment.%j.err     # Standard error file

# Load necessary modules or activate virtual environments
module load anaconda   # Load Anaconda module (if available)
conda activate my_env  # Activate Conda environment

# Change to the directory where the job will run
cd /path/to/experiment/directory

# Your experiment commands go here
python my_experiment_script.py arg1 arg2
```

Running experiments with Slurm

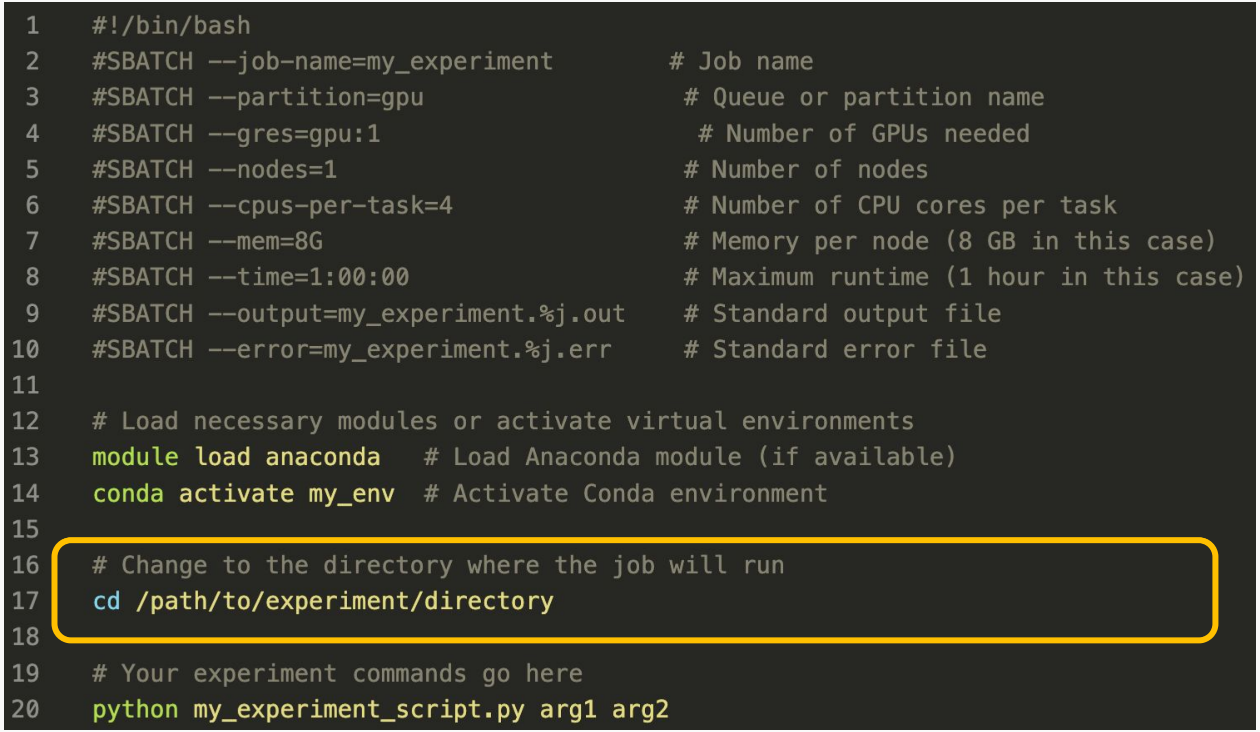
Load module for running on conda environment.



```bash
#!/bin/bash
#SBATCH --job-name=my_experiment        # Job name
#SBATCH --partition=gpu                  # Queue or partition name
#SBATCH --gres=gpu:1                      # Number of GPUs needed
#SBATCH --nodes=1                        # Number of nodes
#SBATCH --cpus-per-task=4                # Number of CPU cores per task
#SBATCH --mem=8G                         # Memory per node (8 GB in this case)
#SBATCH --time=1:00:00                   # Maximum runtime (1 hour in this case)
#SBATCH --output=my_experiment.%j.out    # Standard output file
#SBATCH --error=my_experiment.%j.err     # Standard error file

# Load necessary modules or activate virtual environments
module load anaconda   # Load Anaconda module (if available)
conda activate my_env  # Activate Conda environment

# Change to the directory where the job will run
cd /path/to/experiment/directory

# Your experiment commands go here
python my_experiment_script.py arg1 arg2
```

Running experiments with Slurm

Set working directory.
Not always necessary.

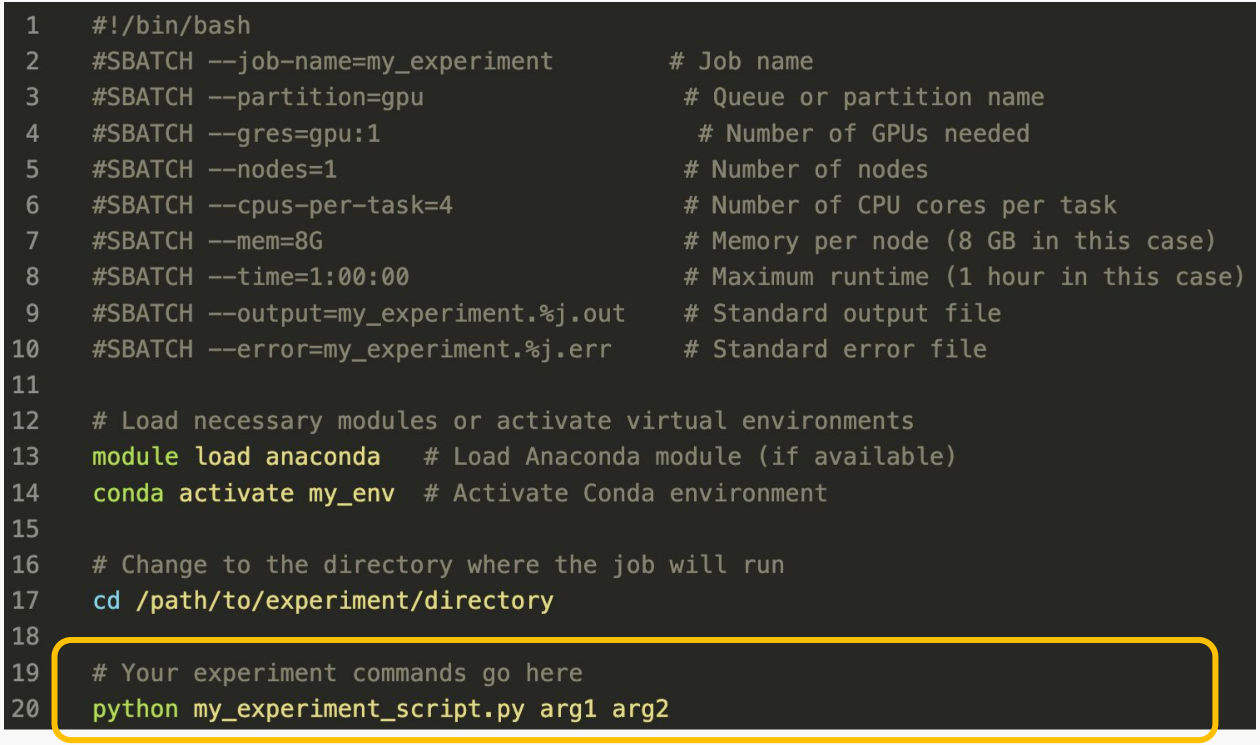


```
#!/bin/bash
#SBATCH --job-name=my_experiment        # Job name
#SBATCH --partition=gpu                  # Queue or partition name
#SBATCH --gres=gpu:1                      # Number of GPUs needed
#SBATCH --nodes=1                        # Number of nodes
#SBATCH --cpus-per-task=4                # Number of CPU cores per task
#SBATCH --mem=8G                         # Memory per node (8 GB in this case)
#SBATCH --time=1:00:00                   # Maximum runtime (1 hour in this case)
#SBATCH --output=my_experiment.%j.out    # Standard output file
#SBATCH --error=my_experiment.%j.err     # Standard error file

# Load necessary modules or activate virtual environments
module load anaconda   # Load Anaconda module (if available)
conda activate my_env  # Activate Conda environment

# Change to the directory where the job will run
cd /path/to/experiment/directory

# Your experiment commands go here
python my_experiment_script.py arg1 arg2
```

Running experiments with Slurm

Running python command as usual.

Tip: run with -u flag.



```
#!/bin/bash
#SBATCH --job-name=my_experiment        # Job name
#SBATCH --partition=gpu                  # Queue or partition name
#SBATCH --gres=gpu:1                      # Number of GPUs needed
#SBATCH --nodes=1                        # Number of nodes
#SBATCH --cpus-per-task=4                # Number of CPU cores per task
#SBATCH --mem=8G                         # Memory per node (8 GB in this case)
#SBATCH --time=1:00:00                   # Maximum runtime (1 hour in this case)
#SBATCH --output=my_experiment.%j.out    # Standard output file
#SBATCH --error=my_experiment.%j.err     # Standard error file

# Load necessary modules or activate virtual environments
module load anaconda   # Load Anaconda module (if available)
conda activate my_env  # Activate Conda environment

# Change to the directory where the job will run
cd /path/to/experiment/directory

# Your experiment commands go here
python my_experiment_script.py arg1 arg2
```

Running experiments with Slurm

The sbatch command is used in Slurm to submit batch scripts for execution.

Syntax: `$ sbatch example.slurm`

To monitor running jobs in Slurm, we can use the squeue command.
This command provides information about jobs currently in the queue, including their status, job ID, name, partition, and more.

Syntax: `$ watch squeue`

Useful Commands for Remote Working

Command	Operation
sbatch	Submits a batch script to SLURM. The batch script may be given to sbatch through a file name on the command line, or if no filename is specified, sbatch will read in a script from standard input.
squeue	Used to view job and job step information for jobs managed by SLURM.
scancel	Used to signal or cancel jobs, job arrays or job steps.
sinfo	Used to view partition and node information for a system running SLURM.

MNIST training example

Create conda environment with pytorch dependencies.

```
$ conda create -n mnist pytorch==2.1.2 torchvision==0.16.2 cudatoolkit -c  pytorch
```

Create the slurm file, use a template or use the Slurm builder.

Insert the python execution command: `python train.py --batch_size 128 --epochs 20 --device cuda`

The sbatch command is used in Slurm to submit batch scripts for execution. `$ sbatch run_mnist.slurm`

Monitor jobs currently in the queue:

```
$ watch squeue --format="%.18i %.9P %.30j %.8u %.8T %.10M %.9l %.6D %R"
```

Monitor log with `$ tail -f log_filename`

Cancel jobs currently in the queue:

`$ scancel job_id` or `$ scancel -u username`

Use the HPC ...

Ethically

- Do not use the login node for production runs.

Smartly

- Optimise your jobs for CPU, GPUs, Memory, and time usage;
- Create universal and software-specific submission scripts (but never sample specific);
- Reduce the number of CPU cores or GPUs if it doesn’t have a very significant effect to go in production earlier;
- Check production node usage;

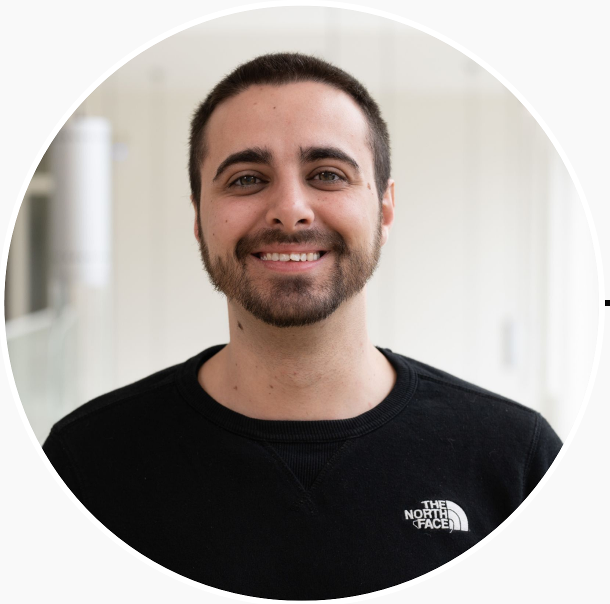
Efficiently

- Run multiple samples in parallel;
- Build up dependency managers;
- Test locally ... run in production :-)

Contact

Antonio Emanuele Cinà
Assistant Professor @ University of Genoa

antonio.cina@unige.it



If you have any questions, don't hesitate to contact me.