PIMA school

March 20, 2022

Contents

These are the notes for the PIMA school 2022.

Chapter 1

Lecture 0

1.1 Introduction to Shell terminal

The shell is a program where users can type commands. With the shell, its possible to invoke complicated programs like climate modeling software or simple commands that create an empty directory with only one line of code. The most popular Unix shell is Bash (the Bourne Again SHell so-called because its derived from a shell written by Stephen Bourne). Bash is the default shell on most modern implementations of Unix and in most packages that provide Unix-like tools for Windows.

Using the shell will take some effort and some time to learn. While a GUI presents you with choices to select, the command line does not. You must learn a few commands just like new vocabulary words in a language you're studying. However, unlike a spoken language, a small number of 'words' (i.e. commands) gets you a long way, and we will cover those essential few today.

The grammar of a shell allows you to combine existing tools into powerful pipelines and handle large volumes of data automatically. Sequences of commands can be written into a script, or a sequence of commands that can be run all at once, improving the reproducibility of workflows.

In addition, the command line is often the easiest way to interact with remote machines and supercomputers. 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 data crunching, being able to interact with the shell is becoming a necessary skill. We can build on the command-line skills covered here to tackle a wide range of scientific questions and computational challenges.

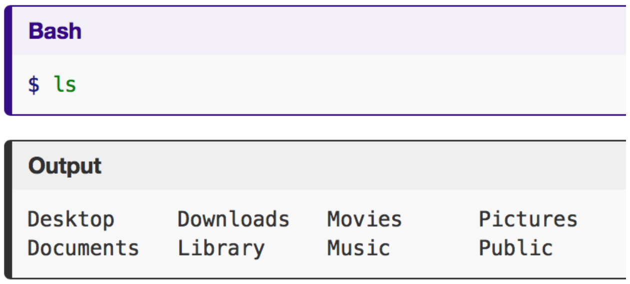
Let's get started.

When the shell is first opened, you are presented with a *prompt*, indicating that the shell is waiting for input.

The shell typically uses $ as the prompt, but may use a different symbol. In the examples for this lesson, well show the prompt as $. Most importantly: when typing commands, either from these lessons or from other sources, do not type the prompt, only the commands that follow it.

```
$
```

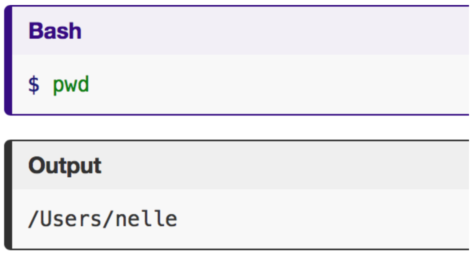
Figure 1.1: bash prompt



```
$ ls
```

```
Desktop     Downloads   Movies      Pictures
Documents   Library     Music       Public
```

Figure 1.2: ls command



```
$ pwd
```

```
/Users/nelle
```

Figure 1.3: pwd command

Also note that after you type a command, you have to press the Enter key to execute it.

The prompt is followed by a text cursor, a character that indicates the position where your typing will appear. The cursor is usually a flashing or solid block, but it can also be an underscore or a pipe. You may have seen it in a text editor program, for example.

So let's try our first command, 'ls' is short for listing. This command will list the contents of the current directory.

The part of the operating system responsible for managing files and directories is called the file system. It organizes our data into files, which hold information, and directories (also called 'folders'), which hold files or other directories.

Several commands are frequently used to create, inspect, rename, and delete files and directories. To start exploring them, we willl go to our open shell window.

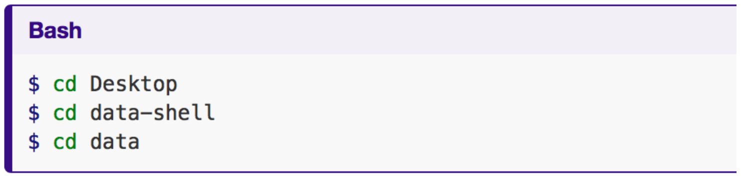
First let's find out where we are by running a command called *pwd* (which stands for 'print working directory'). Directories are like places - at any time while we are using the shell we are in exactly one place, called our current working directory. Commands mostly read and write files in the current working directory, i.e. 'here', so knowing where you are before running a command is important. pwd shows you where you are.

Here, the computer's response is */Users/nelle*, which is Nelle's home directory. The home directory path will look different on different operating systems. On Linux it may look like /home/nelle, and on Windows it will be similar to C:/Documents and Settings/nelle or C:/Users /nelle. (Note that it may look slightly different for different versions of Windows.) In future examples, we will use Mac output as the default - Linux and Windows output may differ slightly, but should be generally similar.

We will also assume that your pwd command returns your users home directory. If pwd returns something different you may need to navigate there using *cd* or some commands in this lesson will not work as written. See Exploring Other Directories for more details on the cd command.

Let's create a new directory called *pima* using the command (which has no output):

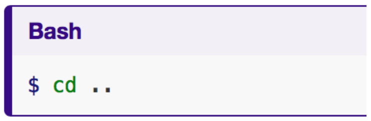
mkdir pima



```
Bash

$ cd Desktop
$ cd data-shell
$ cd data
```

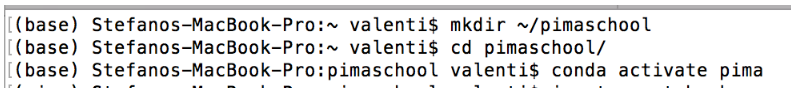
Figure 1.4: cd command



```
Bash

$ cd ..
```

Figure 1.5: command to go up one directory



```
[(base) Stefanos-MacBook-Pro:~ valenti$ mkdir ~/pimaschool
[(base) Stefanos-MacBook-Pro:~ valenti$ cd pimaschool/
[(base) Stefanos-MacBook-Pro:pimaschool valenti$ conda activate pima
```

Figure 1.6: make a new directory

1.2 Install miniconda and PIMA environment

Installing Miniconda3 under Windows

Download and run the appropriate installer from:
https://docs.conda.io/en/latest/miniconda.html
If prompted, you should choose to:
1) Accept the license / terms of use.
2) Install for just the current user, not all users.
Once installed, check that you can run the ”Anaconda Prompt”.
From the prompt, run the command:

conda –version

Video to install miniconda on Windows
https://www.youtube.com/watch?v=tXgPY4lc6fo

Installing Miniconda3 under Linux or macOS

Download the appropriate installer for your OS here: *https : //docs.conda.io/en/latest/miniconda.html*
For macOS, you can choose between a ”package” or ”bash” version.
I find it easier to follow the bash version, but the package version will work too. I recommend you make the following choices described below.
1) Accept the license / terms of use.
2) Do not install for all users, but just the current user.
3) You might be asked whether or not to allow the installer to issue ”conda init”.

Answering "no" will require you to activate conda yourself each time you open a new terminal, following the instructions that will be provided during installation. Make sure you record these instructions if you choose "no". I recommend answering "yes". This will set some environment variables persistently in your shell so that conda is setup every time you open a terminal. I recommend "yes", and the remaining instructions assume that you have done so. During the installation, take note of the installation location of miniconda in your log file. The default installation should put conda here: *YOUR_HOME_DIRECTORY/opt/miniconda*3/

Installing PIMA Conda Environment

Make sure your conda is fully up to date with:
conda update conda
then follow the prompts, e.g. selecting "y" as needed to update any out-of-date packages. We'll be using a conda environment specifically for PIMA school to avoid conflicts with any other projects on your computer, and to ensure that we all have the same software installed.
To create our environment:
conda create -n pima python=3.8 numpy scipy matplotlib ipython jupyter

---

Starting pima Conda Environment and Jupyter Notebook

Activate the pima python environment with
conda activate pima

now we need to install last package: sep
there are two way:

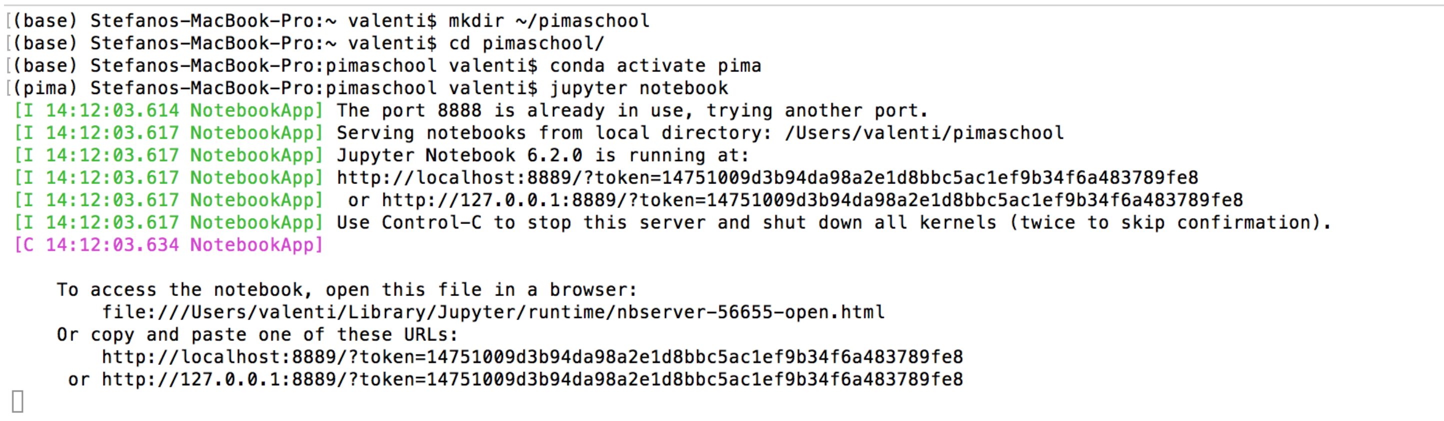
```
$  conda install -c conda-forge sep
```

or

```
$  pip install sep
```

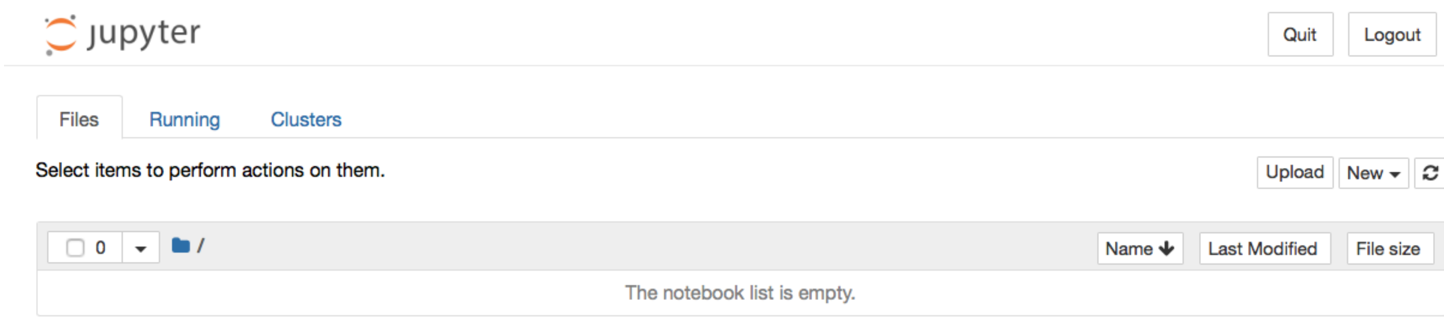
Then launch jupyter notebook with:
jupyter notebook
Note that the directory in which you run this command will be the rootdirectory in your jupyter notebook. Keep this in mind when you are reading or writing data files! When you are done with pima for the session you can deactivate the environment with:

```
$ conda deactivate
```



```
[(base) Stefanos-MacBook-Pro:~ valenti$ mkdir ~/pimaschool
[(base) Stefanos-MacBook-Pro:~ valenti$ cd pimaschool/
[(base) Stefanos-MacBook-Pro:pimaschool valenti$ conda activate pima
[(pima) Stefanos-MacBook-Pro:pimaschool valenti$ jupyter notebook
 [I 14:12:03.614 NotebookApp] The port 8888 is already in use, trying another port.
 [I 14:12:03.617 NotebookApp] Serving notebooks from local directory: /Users/valenti/pimaschool
 [I 14:12:03.617 NotebookApp] Jupyter Notebook 6.2.0 is running at:
 [I 14:12:03.617 NotebookApp] http://localhost:8889/?token=14751009d3b94da98a2e1d8bbc5ac1ef9b34f6a483789fe8
 [I 14:12:03.617 NotebookApp]  or http://127.0.0.1:8889/?token=14751009d3b94da98a2e1d8bbc5ac1ef9b34f6a483789fe8
 [I 14:12:03.617 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
 [C 14:12:03.634 NotebookApp]

     To access the notebook, open this file in a browser:
         file:///Users/valenti/Library/Jupyter/runtime/nbserver-56655-open.html
     Or copy and paste one of these URLs:
         http://localhost:8889/?token=14751009d3b94da98a2e1d8bbc5ac1ef9b34f6a483789fe8
      or http://127.0.0.1:8889/?token=14751009d3b94da98a2e1d8bbc5ac1ef9b34f6a483789fe8
```



Select items to perform actions on them.

The notebook list is empty.

Figure 1.7: empty notebook

1.3 Starting a Jupyter notebook

This course will make extensive use of the Jupyter Notebook interface to Scientific Python, whichis well suited to academic work (including independent research) because it combines code without-put in digestable chunks. Even when the end product is a polished piece of software, much of the development occurs in the sort of interactive sessions that Jupyter Notebooks provide.

After following the software installation instructions on the course website, activate the pima environment with:

```
$ conda activate pima
```

Then, navigate to a working directory for this session, and start the notebook with:

```
$ jupyter notebook
```

This should start the Jupyter Notebook server and open a client in your web browser. An example starting a Jupyter Notebook from Linux is shown in Fig. 1.1.

You should create one new Jupyter Notebook, by choosing the New (Python 3) option in your client. Change the name of your notebook to something that clearly identifies what we are doing. Start each notebook with comments (starting with # symbol) indicating the title of the notebook. See the first cell of Fig. xx for an example. This first cell is also a good place to issue the ipython magic function: % pylab inline which will setup the notebook for inline plots and load the numpy and matplotlib libraries for you.

Now that we have made our first notebook we can save it and it will have ipynb extension.

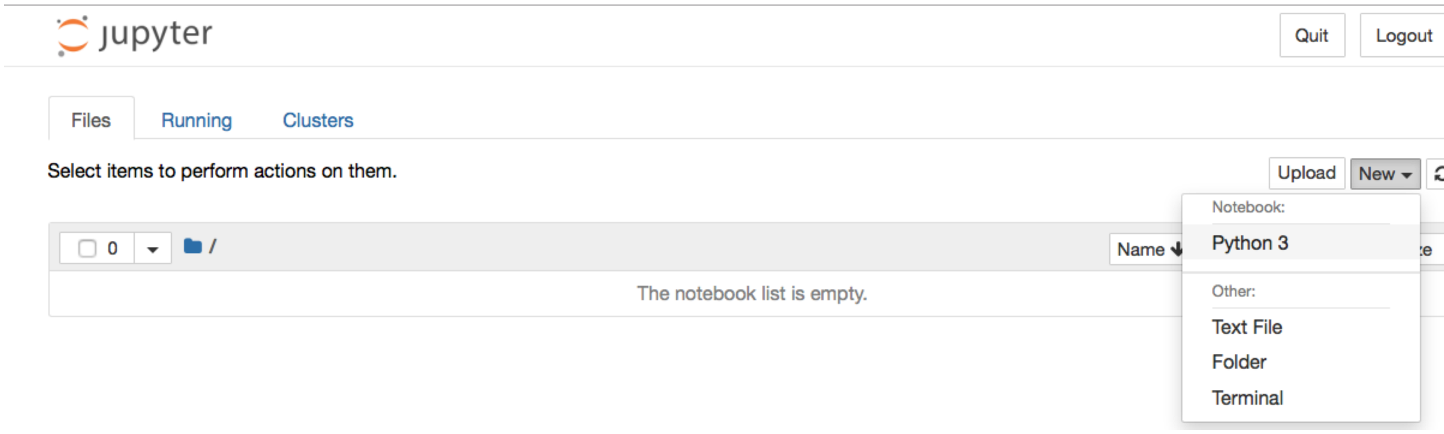
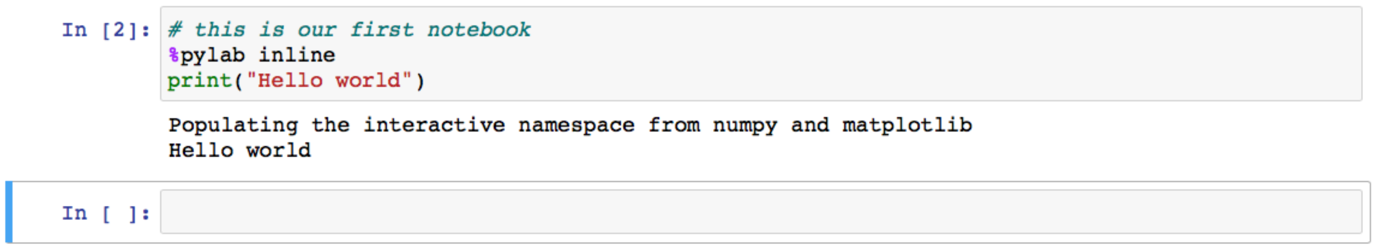


Figure 1.8: make new python 3 file in notebook



```
In [2]: # this is our first notebook
        %pylab inline
        print("Hello world")

        Populating the interactive namespace from numpy and matplotlib
        Hello world

In [ ]:
```

Figure 1.9: notebook "hello world"

1.3.1 notebook fast command

Here we report some short command for notebook:

Control vs Edit modes

- esc for control mode
- a : insert a new cell above current cell
- b : insert a new cell below current cell
- d+d : delete current cell
- m : convert cell to markdown
- y : convert cell to code
- enter for edit mode
- shift+enter : execute cell

Code vs Markdown Cells

- Markdown:
  bullets
  headings

equations

links

very simiar to HTML

Navigating the notebook

- plus : add a new cell
- File - Save and Checkpoint or picture of floppy drive: save
- Close browser window and type ctl+c twice in command line : Exit

Chapter 2

Lecture 1

2.1 Introcuction to scipy, numpy and definition

Introduction to arrays, variable, string, list, loop and function. Use the material in the notebook: $http://localhost:8888/notebooks/lecture1/PIMA_lecture1.ipynb$

and extra material form scipy lecture.
Plot a spectrum at the end of the lecture, More on plotting in the next lecture.

Chapter 3

Lecture 2

3.1 Plot and histograms

Chapter 4

Lecture 3

4.1 fits images, plot the image, identify object, wcs

Chapter 5

Lecture 4

5.1 Magnitudes, flux,measure flux, distance modulus, convert to magnitude calibrate photometry

Chapter 6

Lecture 5

6.1 Discover a planet planets

Chapter 7

Lecture 6

7.1 Measure the distance of a Supernova

Chapter 8

Lecture 7

8.1 Color image