Lab 1: Introduction to Unix/Python

Nearly all analysis of astronomical data takes place on computers. While there are many different programs, operating systems, and programming languages that are used, unix and python are particularly common. This lab is designed to introduce you to the basics of unix and python.

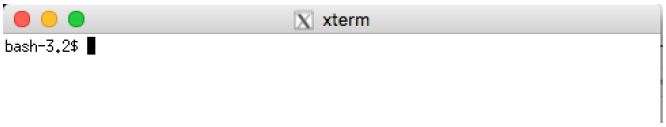
Unix

Introduction

Accessing the command line, and associated unix commands, is possible on a Mac using either the terminal or XQuartz app. Terminal gives you access to the command line. X11 also also gives you access to the command line, with the addition of providing a GUI frontend. Terminal is built in to mac, while XQuartz is an add on. For most of what we do, either would work.

> Open the XQuartz app.

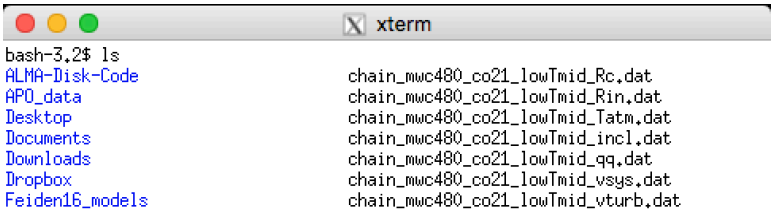
Once you open XQuartz, you will presented a window that looks like:



From here you can enter commands.

> Type **`ls`** and hit enter.

The **`ls`** command lists the contents of the current directory. You will get an output that looks something like:



> Type **`ls -a`** and hit enter.

The **`-a`** is a flag that you have sent to the **`ls`** command. This tells the **`ls`** command to list all hidden files (e.g. those whose file name begins with a period).

```
bash-3.2$ ls -a
.
..
.CFUserTextEncoding
.DS_Store
.Trash
.Xauthority
.almaot
.anyconnect
.astropy
.atom
```

Multiple flags can be used by including the individual letters in a string after the dash, e.g. **`ls -al`**

You can also tell the **`ls`** command to look in a different directory.

```
bash-3.2$ ls APO_data/
Q1NA01 tcam
bash-3.2$ ls APO_data/Q1NA01/
UT200106
bash-3.2$ ls APO_data/Q1NA01/UT200106/
ALMA123.0001.fits ALMA18.0023.fits  ALMA5.0019.fits   HD37887.0027.fits
ALMA123.0002.fits ALMA18.0024.fits  ALMA5.0020.fits   HD37887.0028.fits
ALMA123.0003.fits ALMA18.0025.fits  ALMA5.0021.fits   HD37887.0029.fits
```

You can also only list files that follow a particular naming convention using the `*`. This is a wildcard and selects any file whose name includes any possible value in the `*`

```
bash-3.2$ ls APO_data/Q1NA01/UT200106/HD37887*fits
APO_data/Q1NA01/UT200106/HD37887.0001.fits
APO_data/Q1NA01/UT200106/HD37887.0002.fits
APO_data/Q1NA01/UT200106/HD37887.0003.fits
APO_data/Q1NA01/UT200106/HD37887.0004.fits
APO_data/Q1NA01/UT200106/HD37887.0005.fits
```

If you want to go up a directory, use **`..`**

```
bash-3.2$ ls ..
Shared          firstboot       kevinflahertyastro macadmin
bash-3.2$ ls ../..
Applications            bin                     opt
Library                 cores                   private
```

To determine the current directory that you are in, use the **`pwd`** command

```
bash-3.2$ pwd
/Users/kevinflahertyastro
```

To move from one directory to another directory, use the **`cd`** command

```
bash-3.2$ cd APO_data/
bash-3.2$ pwd
/Users/kevinflahertyastro/APO_data
bash-3.2$ ls
Q1NA01 tcam
```

Each unix command has an associated manual. To open the manual type **man ls**, where **ls** can be replaced by any command. To move around in the manual: **j**/**k** move you down/up one line at a time, while **[spacebar]**/**b** move you down/up one page at a time.

Manipulating files

The **mv** command will move a file from one directory to another location.

```
bash-3.2$ mv helper.py APO_data/
bash-3.2$ ls APO_data/
Q1NA01    helper.py tcam
```

Move the helper.py file into the APO_data directory

It can also be used to change the name of a file

```
bash-3.2$ mv APO_data/helper.py new_helper.py
bash-3.2$ ls new*py
new_helper.py
```

Move the helper.py file back into the current directory, and change its name

The **cp** command is similar but it copies the file instead of just moving it.

The **rm** commands deletes (removes) a file. Note that there is no trash-can in Unix. Once you delete a file it is gone!

Directories can be deleted with the **rmdir** command. To remove a directory and all of its contents, use the **-r** flag on **rmdir**.

You can create a new directory using the **mkdir** command.

Additional commands

gzip/gunzip: Used when unpacking a compressed file (with the .gz suffix)

tar: Unpacking multiple that have been packaged together. The two you need are:

tar -xvf [filename] Extract (**-x**) a tar'ed file, be verbose (**-v**), and apply to the file whose name comes after this command (**-f**)

tar -xvzf [filename] Same as above, but for a file that has been compressed.

tar -cvf [filename] [list of files] Take the list of files and package them into a tar file with the name filename.

chmod: Change permissions for a file.

clear: Clear the contents of the window

`more [filename]`: See what is in the contents of a file. Navigate in the same way as when looking at the manual for a command.

`less [filename]`: Same as above.

`alias aliasname = 'commandyouwanttoalias'` : Create a shortcut for a particular command.

`lpr [filename]`: Print out a file

`ls APO_Data > datalist`: The use of **>** redirects (or pipes) the output of the **`ls`** command so that instead of being shown on the screen it is placed in the **`datalist`** file.

`ls APO_Data >> datalist`: The use of **>>** appends the output of the **`ls`** command to the file, rather than overwriting the previous contents.

Python (and Jupyter notebooks):

Python is quickly becoming the main coding framework within astronomy, although C++ is still often used for computationally heavy projects (e.g. large numerical simulations). This guide will introduce you to the very basics of python, while providing resources for digging (much, much) deeper.

There are a number of ways to start python. The three which we will focus on all occur on the terminal.

1. On the computers in TPL 301, the first thing you need to do is set up the path to the python installation. To do this, as soon as you open terminal/Xquartz type **`source .bash_profile`** and hit enter.

2. Next you need to do is switch to python 3. These computers are set up with multiple versions of python, but we will mostly be using python 3. Python 2.7 is another frequent version, and is the default version on these computers. To make python 3 available, enter **`source activate python3`** on the command line and hit enter.

3. Starting python.

There are multiple methods for starting python from the command line.

A. *python*: Standard command-line version of python.

Type **`python`** at a terminal and hit enter. Type **`exit()`** and hit enter to exit this python terminal.

B. *ipython*: Command-line version of python with additional functionality, including loading some useful packages. In particular, it loads numpy (as np) and matplotlib.pyplot (as plt).

> Type **`ipython --pylab`** (there are two dashes in front of pylab) at a terminal and hit enter. Type **`exit`** and hit enter to exit this python terminal.

C. *Jupyter notebooks*

Jupyter notebooks are a tool for collecting code, along with text, all in one place. Much of the work that you do for this course will be within jupyter notebooks, and we will focus on them here.

> Type **`jupyter notebook`** into the terminal and hit enter. This will start a jupyter notebook server and open a browser window with a list of files in the current directory.

> To create a new file, click **New** in the upper right and select **Python 3**.
This will open a new window in which you can enter commands. It will start with a single cell that is set to accept code.

> Within this cell type **`print("Hello World")`** and hit **shift-enter**.
This will execute the cell (enter will create a new line within this cell instead of executing the code). The above code should have printed to the screen, and then created a new cell. Code cells are one type of cell. You can also convert the cell type to Markdown using the dropdown menu. Markdown is a markup language, similar to HTML, that allows you to enter text. This allows you to enter notes interspersed with your code.

The name of the notebook can be changed at the top, by clicking on Untitled Notebook and enter new text.

A notebook can be saved using the save button in the upper left.

When you are done with a notebook, save the results and close the notebook window. Even though the notebook has been closed, it is still running (notebooks that are running are colored green). To shutdown a notebook, click the checkbox next to its name and click the Shutdown button that appears.

To close the server, either click the Quit button in the upper right of the server window and close the server window, or type `ctrl-C` twice in the terminal window where the jupyter server is running and then close the server window.

Jupyter notebook quick start

- Open terminal/XQuartz
- Type **`source .bash_profile`** and hit **enter**. This will provide access to python.
- Type **`source activate python3`** and hit **enter**. This will switch to an environment with python3 instead of python 2.7.
- **`cd`** into the desired directory.
- Type **`jupyter notebook`** and hit **enter**.

Python Tutorial:

Python is an extensive programming language with many features, and today we can only scratch the surface. You will start with a python bootcamp taken from https://prappleizer.github.io, which has a number of other tutorials, and also includes a detailed textbook about the basics of using python. The python notebooks are available on the astro server:

> Mount the astro server, using the directions below.
> From within the Astro211_S20 folder, copy the Lab1_UnixPython folder onto your desktop.
> Open XQuartz and cd into the Lab1_UnixPython folder.
> Start up jupyter notebook under python 3 and open the notebook titled first_day_tutorial_part_1.ipynb
> Complete the tutorial, executing all of the code cells, and input answers to the questions. When you are done, save your results. Shutdown the notebook.
> Add you name to the name of the notebook and copy it into the Place_Your_Results_Here folder on the server.
> Start part 2.

If you have no experience with python then your goal is simply to complete the bootcamp notebooks (either here, or later).

If you do have experience with python, consider tackling the following challenge:
Open a new Jupiter notebook and create a function that plots a blackbody of a given temperature. Label the axes. Print out the peak wavelength of the blackbody. Set up the function so that it can be called multiple times to plot blackbodies of different temperatures on the same graph. Raise an error when the temperature is negative. Include a docstring documenting your code.

Accessing the server

All of our image data are stored on the astronomy Mac server at a mountable location named Shared. Within Shared are the following folders:

CCD_Data	0.6m main camera
Spectro_Data	0.6m spectrograph
WF_Data	widefield imager
PtGrey_Data	planetary imager

All approved astronomy users have full read/write access to the folders mentioned here, so please be careful not to disturb anyone else's data. Do not do your work inside Shared, but grab copies of your files and work on them elsewhere. Anything placed on the desktop of your astronomy network account will be accessible from any computer.

To access these folders from a Mac:
1. Select
   Go → Connect to Server…
2. In the Server address field enter
   afp://astro-server2.williams.edu
   and click **Connect**
3. Select **Registered User**, and enter your astronomy username/password, and click **Connect**
4. Select **Shared**, and click **OK**. You will then see the **Shared** volume on the desktop

To access these folders from a Windows 7 PC:
1. Double-click the Computer icon on the desktop
2. Click the **Map network drive**
3. In the **Folder**: field enter
   \\astro-server2.williams.edu/Shared
4. Uncheck **Reconnect at logon**, check **Connect using different credentials**
5. Click **Finish**
6. Enter your astronomy username/password, and click **OK**. **Shared** will then be available as a mounted volume.

Additional Resources

Python for Astronomers (https://prappleizer.github.io): The resource from which the bootcamp notebooks were drawn from. Includes additional tutorials on e.g., data I/O, linear regression as well as an extensive textbook introducing the basics of python.

Python Lecture Notebooks (https://github.com/Morisset/Python-lectures-Notebooks): Github repository of Python lectures given at the Instituto de Astronomia - UNAM

Practical Python for Astronomers (https://python4astronomers.github.io): A detailed guide to python, with a focus on astronomical tasks (e.g., reading and writing fits files, fitting and modeling 1-D and 2-D data).

Learn Astropy (http://learn.astropy.org): Tutorials and examples specific to the Astropy package.

Python Data Science Handbook (https://jakevdp.github.io/PythonDataScienceHandbook/): Detailed introduction to Python. Includes good notes on the Pandas package, as well as a chapter on machine learning in python.

Unix guide (https://prappleizer.github.io/unix_full.pdf): Unix basics, beyond what is described above.

Jupyter Basics (https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/Notebook/Notebook%20Basics.ipynb): Guide from the creators of Jupyter notebooks. Click the Notebook link at the top (part of the directory structure) to access other notebooks.