Using Python on Midway

Lecture's objectives

- Explain different types of "big problem"
- Compare and contrast different types of parallel computer architecture
- Identify the difference between node, socket, and core

Python Modules on Midway

There is a plethora of python modules on midway. How then to choose which to use?

```
[johnnyb@midway1]$ module avail python

------------------------ /software/modulefiles ------------------------
python/2.7                              python/2.7-2015q1
python/3.3                              python/2.7-2015q2(default)
python/3.4-2015q1                       python/2.7.12+gcc-4.7
python/3.5.2+gcc-4.8                    python/2.7.12+gcc-6.1
python/3.5.2+intel-16.0                 python/2.7.12+intel-16.0
--------------------------- /etc/modulefiles ------------------------
```

```
[johnnyb@midway1]$ module avail Anaconda2
   Anaconda2/4.1.1(default)

[johnnyb@midway1]$ module avail Anaconda3
Anaconda3/4.1.1(default)
```

Python Modules on Midway

There is a plethora of python modules on midway. How then to choose which to use?

- There are two main different python versions (python2.7 and python3.x) which are indicated in the module version.

- Some installs are built with different system compilers (gcc, intel) This is important if coupling your python code with other compiled software (want to have homogeneity in compilers/libs).

- Not all python modules have the same packages. You can check this with pip or conda (Anaconda distribution only).

- **Recommendation is to use the latest Anaconda3 module.**

Installing Python Packages

- People will commonly request that they have xyz python module installed.
- Users can do this themselves with pip in either Anaconda or python module.

```
[johnnyb@midway2]$ module load Anaconda3/2019.03

[johnnyb@midway2]$ pip install --user <package>
```

- Will install it by default to ~/.local unless you specify PYTHONUSERBASE
- If you pip –user install a package it will be available for that version of python.
- Its best to reuse the same python module when intending to use the package that you locally installed.

Creating Python Environments

- For workflows or use cases that require several packages and or specific versions of packages, an environment is the better choice.

```
[johnnyb@midway2]$ module load Anaconda3/2019.03

[johnnyb@midway2]$ conda create --name <env-name> python=<pyvers>

[johnnyb@midway2]$ source activate <env-name>
```

- Will create an environment in to ~/.conda/envs
- Users can then install with conda any package they like within their env.
- Note that conda and pip don't share information. If using conda env it is better to install packages with conda than with pip.

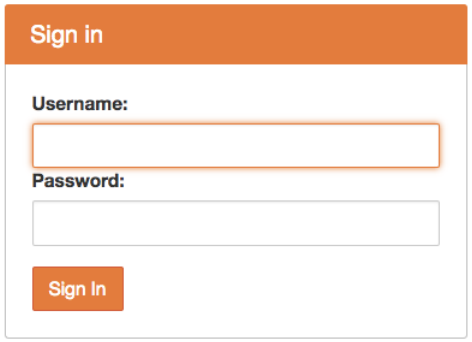
Running Jupyter Notebooks on Midway

1.) From the login nodes: (ssh **–Y** midway.rcc.uchicago.edu)

```
[johnnyb@midway1]$ module load Anaconda3/2019.03

[johnnyb@midway1]$ jupyter notebook
                       OR
[johnnyb@midway1]$ jupyter notebook <file.ipynb>
```

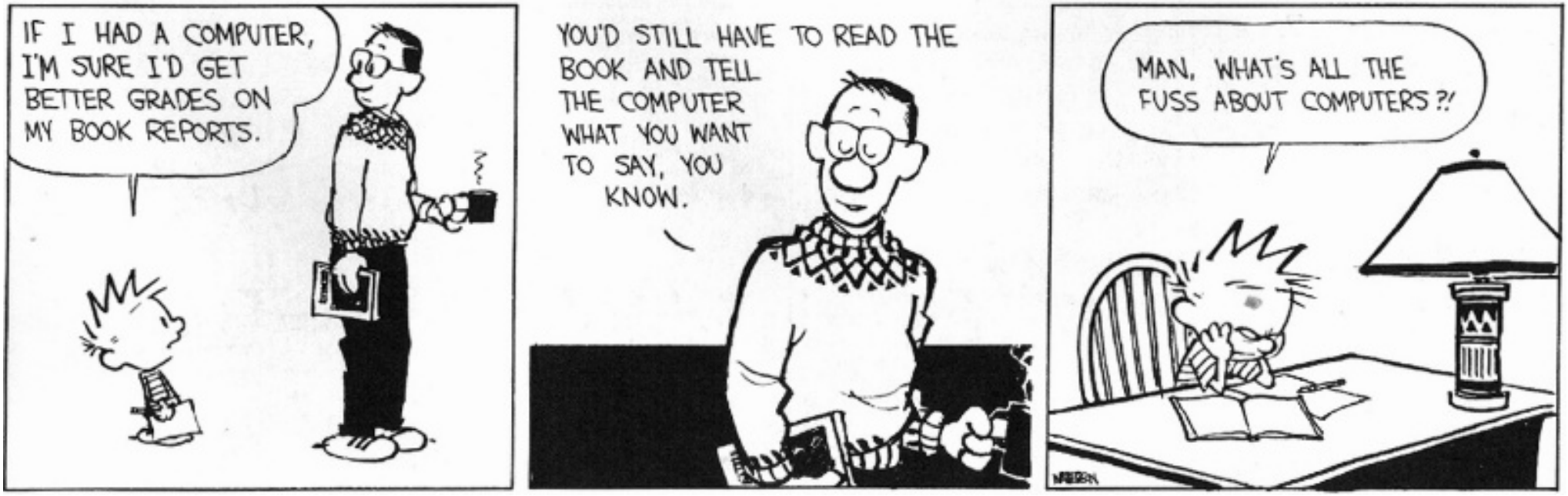
2.) Accessing the jupyterhub portal: https://jupyter.rcc.uchicago.edu



Username is your CNetID
Password is your CNetID password

You will land in /home/$USER directory upon login

RCC Help



- Email: help@rcc.uchicago.edu
- Web: rcc.uchicago.edu
- Phone: 773-795-2667
- Walk-in: Regenstein Library, suite 216