Welcome to BST 281 Lab 1

31 Jan, 2018

Mike MacArthur

macarthur@g.harvard.edu

Office Hours: Fridays 930-1030a

FXB 2nd floor atrium

Homework

First homework due Monday 2/11 by 11:59p on Canvas

Extensions are allowed but must be approved ***in advance***
(The day before is not in advance!)

Homework can be downloaded and submitted from the Assignments section of Canvas
Canvas Assignments page

Lab Agenda

1. Set up programs
    i. Get Anaconda/Python installed
    ii. Set up Jupyter
    iii. Set up Atom text editor
2. Run terminal commands in Atom
    i. Python doctests
3. Cluster tutorial on O2/Odyssey
    i. Login basics/2-factor authenication
    ii. Running an interactive session
    iii. Running pre-installed software
4. Command line and Python practice

Python Setup

Intalling Python

1. Go to the Continuum downloads page and download Anaconda for your operating system.
2. After the download completes, check that Python was properly installed by running `python --version` in the terminal.

Jupyter Setup

1. After installing Anaconda open a terminal or anaconda prompt and run `jupyter notebook`
    - Jupyter will open in your browser, note that it will open in the directory that you're in in the command line/terminal

Atom Setup

Atom is a text editor that is simple but hackable and highly customizable. On a basic level it is similar to an app like Notepad, but with a few modifications available through downloadable packages, it can be used to develop complex programs. It is developed and maintained by the GitHub team, so there are a lot of resources on how to maximize your efficiency while using it. Searching "Atom editor" on Google or YouTube will bring you straight to many of these resources.

Installing Atom

1. Go to the Atom homepage and download atom

Customizing Atom

First download the platformio package which allows you to run a terminal within Atom

1. Open Atom
2. File > Settings
3. Select "install" on left menu
4. In "search packages" bar search for "terminal"
5. Install "platformio-ide-terminal"
6. Open a new terminal: packages > platformio > New Terminal

You can also check out the autocomplete-python package

Cluster Tutorial

There are a couple things to download to get running on the Cluster

For Windows Users:

Install X server/STFP client(s)

MobaXterm is an X server and SSH client that allows
you to connect to the cluster, facilitates 2 factor authenication (required on almost all clusters now)
and perform SFTP (transfer files from your local storage to cluster storage). You may have seen
Putty and Filezilla before… MobaXterm does both in one. But if you prefer Putty/Filezilla that will work as well.

For Mac Users:

Install Filezilla which is an SFTP client that will allow you to transfer files to/from the cluster.

Set up 2 factor authentication (Odyssey tutorial)

To access the cluster you'll need a way to execute two factor authentication. Odyssey encourages
DuoMobile, but Google Authenticator (and probably others) also work. Follow the step-by-step
guide linked above.

Accessing the Cluster

For Windows Users:

Start MobaXterm and click the 'Session' button in the top left. Select 'SSH' session type. In the remote host field type YOURUSERNAME@login.rc.fas.harvard.edu (replace YOURUSERNAME with the user name assigned to you by RC). Select OK and you should be prompted for your password and your two factor authentication code which you can get from DuoMobile or Google Authenticator.

For Mac Users:

Open a terminal session and type ssh YOURUSERNAME@login.rc.fas.harvard.edu (replace YOURUSERNAME with the user name assigned to you by RC). You should be prompted for your password and your two factor authentication code which you can get from DuoMobile or Google Authenticator.

Working in the Cluster

Now you're in the cluster! Odyssey uses the SLURM Workload Manager which is a commonly used job scheduler for cluster (it is also used by
the HMS O2 cluster). This is convenient because once you are familiar with SLURM will
be able to work on many different clusters.

When you first log in your are in the login node. This is like the lobby of a building, *do not work in the lobby*! There are many other "rooms" in the cluster that are specific for
doing work.

You move around the cluster using normal command line. For example, lets see what directory we're currently in and what's in that directory:

Run `pwd` then `ls`

Next lets make a new directory called 'testdir' and navigate into that directory:

run `mkdir testdir` then `cd testdir`

We're now in our new directory called 'testdir', if you run `ls` you should see that it is an empty directory.

If you want to go back up one directory run `cd ..`, which will take you back to the parent directory.

Transfering Files to the Cluster

What if we want to put a file from our local machine onto the cluster to work with it? To do that we need to use an SFTP client.

If you're on Windows and using MobaXterm you can do it by just dragging files from your local file explorer into your cluster folders that are displayed in the Scp tab on the left toolbar.

If you're on Mac then open up Filezilla and log in using your Odyssey credentials. See the RC documentation for a step-by-step on setting up Odyssey on Filezilla. Once you're logged in you will see both your local file system and the cluster file system in the Filezilla window. You can transfer files by just dragging and dropping between the two.

You should be able to see your 'testdir' directory that you made. Take the files 'fq1_1.fq.gz' and 'fq1_2.fq.gz' from the Lab1 module on Canvas and transfer them into your 'testdir' directory on the cluster.

Starting a Session on the Cluster

Now lets start working with these fastq files. Before we start working, remember that we're still in the login node. We need to submit a job to request a space on the cluster where there are resources allocated to perform more intensive operations.

We are going to start off by running an interactive session, which means that we will be given resources to run bigger commands, but it will still look like the bash prompt we see when we first log in.

Run `srun --mem 1000 -p test --pty bash`

The arguments we've provided tell the cluster specifics about what we're requesting, including memory, time, cores and partition (see more about partition in the Slurm Partitions section here)

The cluster will take a minute to figure out what resources to give you, then you'll notice the text next to your username change (ex mmacarthur@holylogin01 to mmacarthur@holy7c19…). You're now off the login node and free to work.

We can also submit jobs in a non-interactive format through scripts… more on that later.

Packages on the Cluster

One of the major advantages of the Odyssey cluster is that it has a lot of software already loaded onto it. There are two ways to search for modules: the first is to run the command `module-query SEARCHTERM` for example if I want to see if the fastqc software is on the cluster, I'll run `module-query fastqc`. This is good if you know exactly what your package is called. If you want to do a more general search, you can try `module spider SEARCHTERM`. This method takes a little longer, but returns more results.

Now that we've seen that fastqc exists on the cluster, let's load it so we can use. To load a module you just need to run the `module load` command.

Run `module load fastqc`

Working with fastq Files

Let's use fastqc to check the quality of the reads in our fastq files.

Make sure you're in 'testdir' and run `fastqc fq1_1.fq.gz`

This will generate an HTML file with the results. Take the file from the cluster into a local directory (like Documents) then open it. How many reads were there? How was the quality of the reads? You may notice some weird things about the reads. A lot of that is because these files are actually just toy examples containing 0.00005 of the original reads. When you have all of the reads the data looks much better. See the fastqc HTML file in the Lab1 folder on Canvas to see how the QC looks when all of the reads are included.