Mothur Helper

Ryan Nguyen

May 2018

1 Introduction

Mothur is a platform that is used for metagenomic analysis. While it is considered useful, there is one serious problem with it: it freaking sucks. In light of this information, I have developed a set of scripts to help with running your files through Mothur in the event you ever try to use it. If you have any questions, you can contact me at **rnguyen2018@berkeley.edu**. If it is after May 2018, then contact me at **ryan.nguyen@yale.edu**.

2 Download Instructions

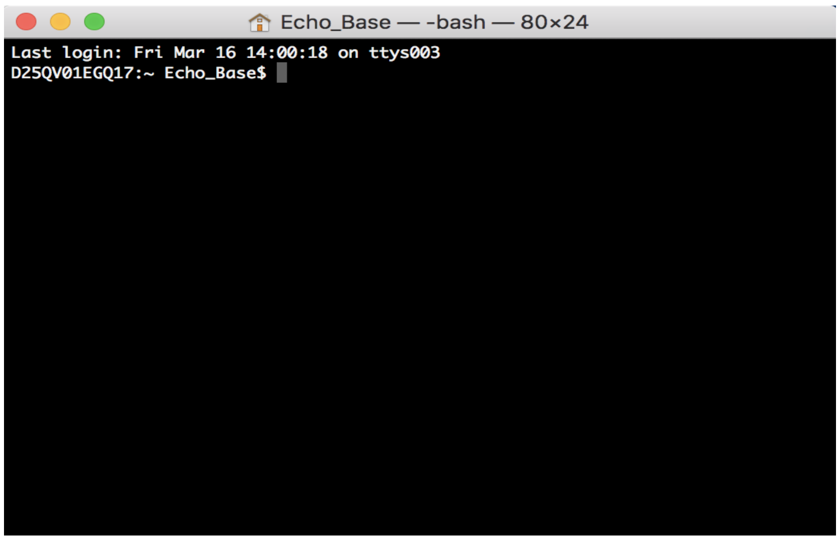
In order to use these scripts, you will need a couple of things (links to download are listed as well):

- Python 3: `https://www.python.org/downloads/release/python-360/`
- Git: `https://git-scm.com/downloads`
- R (latest version)
- mothur: `https://github.com/mothur/mothur/releases/`

Note: for mothur, make sure that it is added to your path directory. Will probably make a separate tutorial about this later. Pick the download that works for your system (MacOS or Windows). These are already downloaded onto the Mac in the microscope room. If these downloaded onto your computer successfully, you should be able to open them up on terminal (Mac) or Command Prompt (Windows).

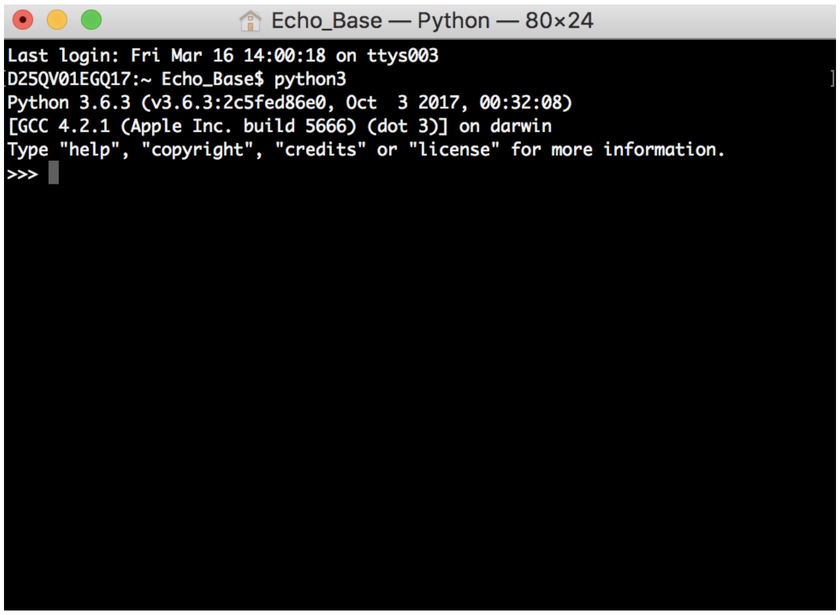
2.1 Making sure Python works on Mac

For Mac users, click on the magnifying glass in the upper right hand corner of your screen and type in "Terminal". Click on that. It should look something like this.



```
Last login: Fri Mar 16 14:00:18 on ttys003
D25QV01EGQ17:~ Echo_Base$
```

If you type in **python3**, you should get this on your screen.

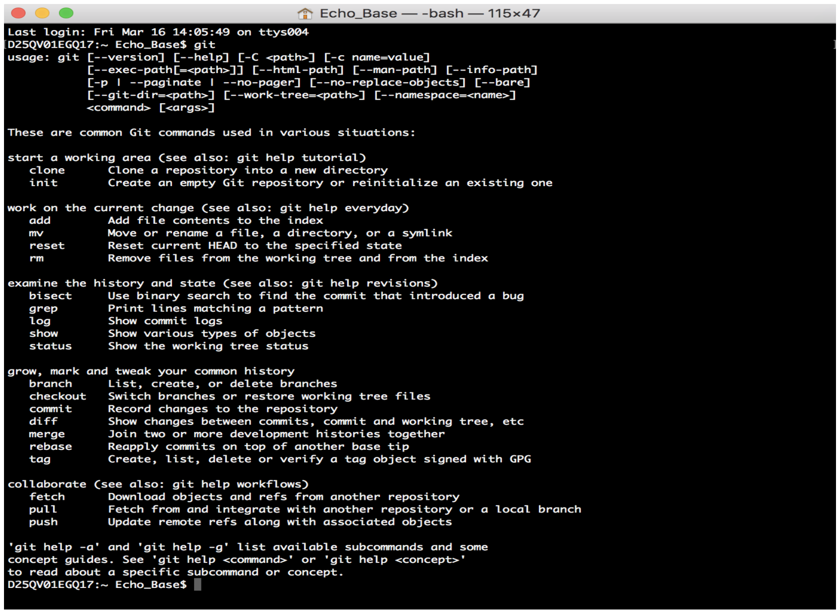


```
Last login: Fri Mar 16 14:00:18 on ttys003
D25QV01EGQ17:~ Echo_Base$ python3
Python 3.6.3 (v3.6.3:2c5fed86e0, Oct  3 2017, 00:32:08)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Hooray! It works. Now, press Ctrl-D to quit python.

2.2 Making Sure Git works on your Mac

To make sure that git works, type in **git** into your Terminal. You should see this on your screen:



```
Echo_Base — -bash — 115×47
Last login: Fri Mar 16 14:05:49 on ttys004
D25QV01EGQ17:~ Echo_Base$ git
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   reset      Reset current HEAD to the specified state
   rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect     Use binary search to find the commit that introduced a bug
   grep       Print lines matching a pattern
   log        Show commit logs
   show       Show various types of objects
   status     Show the working tree status

grow, mark and tweak your common history
   branch     List, create, or delete branches
   checkout   Switch branches or restore working tree files
   commit     Record changes to the repository
   diff       Show changes between commits, commit and working tree, etc
   merge      Join two or more development histories together
   rebase     Reapply commits on top of another base tip
   tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch      Download objects and refs from another repository
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
D25QV01EGQ17:~ Echo_Base$
```

Hooray! It works.

2.3 Downloading the scripts

So this step is to put a folder on your desktop where all the scripts will be stored. To do this, open up your Terminal and type in this line by line:

cd ~/Desktop (make sure there is a space between **cd** and the following statement)
mkdir mothur_helper
cd mothur_helper

DO NOT EXIT YOUR TERMINAL. On your desktop, there should now be a folder called **mothur_helper**. Now, type in this:

git init
git clone https://github.com/Traxlab/mothur_helper.git

If no errors popped up, then you have downloaded it correctly. Congrats!

3 Make_file.py

This script allows you to run multiple fastq files through the fastq.info (extract sequence and quality score data) command. In addition, it will make the stability.files file for further analysis.
In the mothur_helper folder, there should be a file called **make_file.py**.
Drag that into the folder where your fastq files are.

In the terminal, navigate to the folder of your fastq files and type this:
python3 make_file.py

Let that run for a little bit, and all those files should be made for you! You can then do your downstream analysis on these files.

4 Mothur_fuj.R

Okay, so once you've run through and made your various files from mothur, you can visualize your OTU's with mothur_fuj.R. First, move mothur_fuj.R to the folder of your .shared and .taxonomy files.
Next, change the ending of your .shared and .taxonomy files to csv files. So the ending of these files should now be .shared.csv and .taxonomy.csv, respectively.

Okay, here is where it gets a little tricky. Open up Mothur_fuj.R in a text editor. Find the place where it says:

otu_mat <- data.frame(read.csv("stability.trim.contigs...

This is the part in the script that reads in the shared.csv file. Replace the part in quotes with the name of your shared.csv file. Make sure to include the full name including the .csv part.

Next, find the place where it says

tax_mat <- data.frame(read.csv("stability.trim.contigs...

This is the part in the script that reads in the shared.csv file. Replace the part in quotes with the name of your taxonomy.csv file. Make sure to include the full name including the .csv part.

Once you have done that go into your terminal and navigate to the folder where all of these files are. Run this command:

Rscript mothur_fuj.R

This will generate a graph of the Top 40 OTU's from your OTU table generated from mother. It will be saved to the file TopOTU's.pdf