Integrative Biology 200A *University of California, Berkeley*
Systematics *Spring 2010*
By Nick Matzke

Lab 13: Introduction to Scripting: NEXUS files, batch files, Mesquite, R, Python

In today's lab, we will learn about scripting. A "script" is a simple program that basically assembles a bunch of commands into a list and then executes them in series.

While everyone likes window-based programs for figuring out how to do an analysis the first time, what if you have to do the same analysis a dozen times, or hundreds of times with slightly differing parameters? What if you only do the analysis once a year, and have trouble remembering the sequence of commands? Scripting can be a huge advantage for a scientist in these situations.

A word about programming: if you're not going to take a bunch of computer science classes, the best way to learn programming is to have someone introduce you to the basics and get you over the initial "hump." The hardest part of learning a program from scratch is figuring out how to do the most basic things. Books can be useful, but tend to be very general and may not help you with your specific task. An introductory online example can be very useful. In general, the best strategy is to get something minimal working, and go from there.

In this lab, we will attempt to get one simple task working for each of these methods of scripting: NEXUS files, batch files, Mesquite, R, and Python.

R is a free, general statistics package which you will learn about more in IB200b. It is probably a necessary skill for any biologist, just for general statistical and data visualization purposes. We will just introduce a few of the phylogenetics applications here today.

Python is a free object-oriented programming language. It is extremely useful for manipulating text files and data, for example in DNA sequence files. It is also useful for constructing "pipelines" that schlep data, trees, etc., from one analysis program to another, grab and analyze the results, etc. The biogeography package LAGRANGE (used in a future lab) runs in Python.

We will see how far we can get; likely we will save the Python stuff for the biogeography lab, but I want you to attempt to download and install it in this lab.

Setup:

1. Download and install Clustalw
2. Download and install R
3. Macs and some PCs will have Python installed already (type "python –V" to see if you have it, and if so, what version). Attempt to download the free Enthought Python Distribution. This version of Python has many additional packages (for math etc.) and is required for LAGRANGE.

Assignment:

For each section, email me:

1. Describe (briefly) what each of the major commands in the script is doing.

2. Briefly answer any thought questions.

3. Send me comments on whether you got the script to work, what additional steps you had to take, etc. This will improve this lab (which is new) in the future. If you just can't get something to work (even after I try to help), that is fine, just describe where the problem occurred.

I. NEXUS files: Send commands to MrBayes

Many programs which read NEXUS files, such as MrBayes, can read in a series of commands via a NEXUS file. This can be extremely useful, much better than sitting around waiting for one analysis to finish and starting another one.

Just to show you how this works, I have set up a dummy example where one NEXUS file will run phylogenetic analyses of two cyanobacterial gene families. This is part of a project I am working on where we have hundreds of gene families, and want to look for congruence/incongruence between the gene families to see how much evidence of lateral gene transfer (LGT) there is, versus more prosaic sources of incongruence.

- Download the three NEXUS files in the MrBayes part of the lab downloads for today.
- Save them to your MrBayes directory.
- Open the files in a text editor, figure out which two have data blocks (containing ClustalW alignments of these proteins) and which one has a "mrbayes" commands block.
- Run MrBayes. Type "execute" then the name of the file with the mrbayes commands block.
- In email, briefly say what each of the commands does (just do each type of command once, not the repeats). You may have to look at the online MrBayes documentation.

The commands in the mrbayes commands block are:

```
#NEXUS
begin mrbayes;
      set autoclose=yes nowarn=yes;
      execute cyano_gf1.nexus;
      mcmcp nruns=2 printfreq=1000 samplefreq=1000 nchains=4 stopval=0.1;
      showmodel;
      mcmc;
      sumt burnin=501;
      sump burnin=501 Printtofile = Yes;
      showmodel;
      set autoclose=yes nowarn=yes;
```

```
        execute cyano_gf2.nexus;
        mcmcp nruns=2 printfreq=1000 samplefreq=1000 nchains=4 stopval=0.1;
        showmodel;
        mcmc;
        sumt burnin=501;
        sump burnin=501 Printtofile = Yes;
        showmodel;
end;
```

Thought questions:

A. In mcmcp, which parameter did I change to make these runs finish more quickly?
B. Once you had the output from these two gene families, how would you assess whether or not LGT was likely? (There is no single right answer to this one.)

II. Batch files

Kip has a batch file he uses to run ClustalW on a DNA fasta file in a bunch of different ways. Then he concatenates all of the slightly different alignments, giving each the same weight in the final analysis. Whether or not this is a good idea is somewhat controversial (see Lee (2001), "Unalignable sequences and molecular evolution", *TREE*, http://dx.doi.org/10.1016/S0169-5347(01)02313-8), but it shows the usefulness of scripting. Imagine doing this by hand.

1. Make sure you have command-line clustalw working (if you can't get it to work, work with someone else).
2. Download the appropriate script for PC or Mac.
3. Save it and the data file (ray_seqs_all.fasta) to your clustalw directory (or maybe your home directory, or maybe any directory; it will depend on your clustalw settings; the clustalw directory is most likely to work).
4. Look at it in a text editor. Note that "REM" means comments (which are ignored) on a PC, "#" indicates this on a mac.
5. On Mac, you need to make the file executable by users.
    a. type "ls -l" at the Terminal
    b. to make file executable, type "chmod u+x clustalwscrip_mac.bat"
    c. type "ls -l" again to see that the permissions have changed for user (who can now rwx, i.e. read/write/execute), but not for groups (g) or others (o). Note that chmod a+x will change to executable for all users/groups/others.
6. On Mac, type "./clustalwscrip_mac.bat" to run
7. On PC, just type "clustalwscrip_PC.bat" (I think, I don't have a PC handy)

Thought question: What parameters is Kip varying in his various runs of ClustalW?

III. Mesquite scripts

Any windows commands you make in Mesquite can also be scripted. See http://mesquiteproject.org/mesquite_folder/docs/mesquite/scripting.html#examples for much more information.

1. Download dq_v3_xmult_l10_keep.nex
2. Open in Mesquite
3. Go to the Tree Window
4. Go to Window→Scripting→Send Script
5. Paste in the below (or paste from mes_script.txt)

```
String.resultsFile 'results.txt';
saveMessageToFile *String.resultsFile 'RESULTS with different trees';
appendReturnToFile *String.resultsFile;
getWindow;
tell It;
      getNumTrees;
      Integer.numReps *It;
      ifNotCombinable *Integer.numReps;
            Integer.numReps 10; [in case indefinite number of trees]
      endIf;
endTell;

Integer.count 0;
for *Integer.numReps;
      increment.count;
      getWindow;
      tell It;
            setTreeNumber *Integer.count;
      endTell;
      getEmployee #mesquite.ancstates.RecAncestralStates.RecAncestralStates;
      tell It;
            getLastResult;
            String.result *It;
            appendMessageToFile *String.resultsFile *String.result;
            appendReturnToFile *String.resultsFile;
      endTell;
endFor;
```

Thought question: Look at the results file. What did this script do?

Manipulating Trees in R

Here, we will learn just a smidgen of R. Take a look at the PDF on APE I sent around. Read pages 6, 28, and 65-78 and try out the basic commands, using tree files you already have.

Python

We will save this for next lab, but try to get it installed.