Integrative Biology 200B
"Systematics"

University of California, Berkeley
Spring 2011
by Nick Matzke

Lab 15: Maximum Likelihood Estimation of Biogeographic History on Phylogenies using `DIVA` and `Lagrange`

Setup

The main goal for today's lab is just to get these programs working. This can be a bit of a challenge – but part of the skill set you need in order to do evolutionary analyses is the ability to figure out how to get scientific software to function.

Summary of lab

If you have done DIVA and Python-Lagrange in IB200a, you can skip those parts and focus on getting the C++ version of Lagrange to work (and, help others get DIVA/Lagrange to work, work through the example files with them, etc.).

If you can't get some version of Lagrange to work (or if e.g. you are doing the long slow download of the Enthought Python Distribution for Python Lagrange), work with someone who does have it working.

The lab files (and some, but not all executables) are downloadable most simply here:
http://ib.berkeley.edu/courses/ib200b/labs/lab15/lab15.zip

What to turn in

Read the questions in the lab and answer them for yourself, but all I want you to turn in is:

- an email saying what you did/didn't get to work. Probably everyone should be able to get DIVA to work. Lagrange will be harder. If you end up looking on someone else's computer when they run Lagrange, say that.

- Ignore the other instructions to email answers to questions

- I just want to get a sense of how feasible people in the class found these programs to be.

Background on why academic software is sometimes hard to use

Scientists are not trained software developers, they usually have no staff who can work on usability and graphical user interfaces and the like, and they often don't have the ability/time to keep software up to date with continually changing operating systems at the like. For all of that, the software often does amazing things – just don't expect it to be as easy as Microsoft Word and their thousands of software engineers, user trials, and billions of dollars.

The main point of saying the above is to encourage patience and tolerance when you are trying to use academic software.

Some useful guidelines on attempting to install academic software
(derived from long, hard experience)

- The easiest option is always to find an "executable" or "binary" version of a program that is appropriate for your operating system (OS). These have been compiled for your OS ("precompiled binaries") by someone else. Typically all you have to do with these is download them, put them in a desired directory, and run them from the command line.
- If you don't have an executable, you can try to compile the software yourself from the source code. For well-established packages that are used by hundreds of thousands of people, this often works (you have to have the right compilers installed on your machine). For specialist software like phylogenetics software, in my experience it's a 50-50 thing.
- If you are stuck trying to compile something from source, sometimes you will discover that you have to compile another package, which itself requires another package, etc. This can get *extremely* frustrating. Even after putting a lot of time in, and make an attempt to install all dependencies, in my experience it's still only maybe a 75% chance that some bit of software will successfully compile and work.
- Some software packages are available via things like *fink* and *port*, which attempt to find and download all dependencies for you. This can be useful for common software libraries which might be dependencies for your specialist program.
- When you get error messages, the first thing to do is **google the error message** and see what people say about it.
- It is possible to waste *days* messing around with getting software to work, so an important rule is: KNOW WHEN TO GIVE UP. Give it an hour or two, and if you're not making progress, GIVE UP. Try emailing colleagues that use the software, and/or the authors of the software.
- Sometimes the best option is just to have someone else run your data on their computer, which has the ability to run the software. This is one reason collaborations happen.
- Once you have a compiled executable, ALWAYS START WITH THE RUNNING THE EXAMPLE FILES. This will show you how the formatting etc.

works, and will also double-check whether or not the program works, before you invest a lot of time in reformatting your data.

Setup for DIVA – Dispersal-Vicariance Analysis

DIVA was written by Fredrik Ronquist (coauthor with John Huelsenbeck of MrBayes). For a long time only a Windows executable was available. Recently, Jonathan Nylander compiled a Mac version.

The Windows version used to be here:
http://www.ebc.uu.se/systzoo/research/diva/diva.html

…but that website seems to have disappeared. Therefore, I am putting both versions, along with the original manual, etc., on our website here:

http://ib.berkeley.edu/courses/ib200b/labs/lab15/

DOWNLOAD EVERYTHING INTO THE SAME DIRECTORY, e.g.:
something/something/lab15/

This is mostly simply done by downloading the zipfile:
http://ib.berkeley.edu/courses/ib200b/labs/lab15/lab15.zip

Lagrange: general background

Lagrange was written by Stephen Smith and Rick Ree.

- The original version was written in Python and will work on any system (after you have successfully installed the Enthought Python Distribution)

- The new version, currently in Beta form, was written in C++ and is much faster (probably 100s of times faster). However, compiled binaries only currently exist for Mac 10.4 and 10.6 (the 10.6 might work on 10.5, we'll see). If you are a computer jock you might try compiling the source code for Windows, but it took me a whole weekend of plinking to successfully compile it for 10.4, so be advised.
- The C++ version is totally rewritten from the Python version. As far as I know, they use different input files etc.

Lagrange C++ BETA version

Basically, if you have a Mac, you should see if one of these compiled binaries will work: