Molecular Clocks and Tree Dating

Today we are going to use several different methods of testing the molecular clock and estimating node times. We will use a couple of likelihood ratio tests to test the molecular clock against a totally unconstrained tree and a tree with a few branches allowed to vary independently. We will also use several rate smoothing methods to infer divergence times. We will not deal with several commonly used methods. In particular we will not use any relative rate tests to test the molecular clock. This is a very active field and there are constantly new methods and new programs being developed.

Testing for Global Molecular Clock

Under the null hypothesis, the phylogeny is rooted and the branch lengths are constrained such that all of the tips can be drawn at a single time plane. Under the alternative hypothesis, each branch is allowed to vary independently. The alternative hypothesis invokes s - 2 additional parameters, where s is the number of sequences. The likelihood ratio test statistic is -2logL = 2(logL0 - logL1), where L0 and L1 are the likelihoods under the null and alternative hypotheses, respectively. The significance of the likelihood ratio test statistic can be approximated using a chi-square distribution (with s - 2 degrees of freedom).

The following example shows how to perform the likelihood ratio test of the molecular clock using PAUP*.

1. Execute the file Cephalopod.nex (available on the IB 200A website).

This file contains molecular data, and it also contains one tree. For this exercise, we have accepted this tree as our working phylogenetic hypothesis and we are now going to test whether it obeys a molecular clock. You can look at the trees if you want using "showtrees." First, we will calculate the likelihood of this tree without enforcing a molecular clock. For speed, we'll use the Hasegawa, Kishino, and Yano (1985) (HKY85) model of DNA substitution with among site rate variation described using a gamma distribution. In PAUP, this model is set the variant=HKY under the likelihood settings (lset).

2. Estimate model parameters for the Ts:tv ratio and the gamma distribution shape parameter, use these commands:
   ```
   lset tratio=estimate variant=HKY shape=estimate;
   lscores;
   ```
3. Record the –lnL score. This is the likelihood score for the alternative hypothesis, which allows branches to vary independently.
4. Now, we will change the likelihood settings to enforce a molecular clock:
   ```
   lset tratio=estimate variant=HKY shape=estimate clock=yes;
   ```
5. Recalculate the likelihood score under this null model:
   ```
   lscores;
   ```

Conduct a likelihood ratio test in Excel to determine if you can reject the null model. As you know, the likelihood ratio test compares a simple model to a more complex one, to see if adding the extra parameters offers a significant improvement to the model. This is necessary since adding parameters will always improve the model, at least a little bit. Since a molecular clock only allows a single rate, it

can be considered a simpler version of the HKY85. In testing a molecular clock, the degrees of freedom are the number of taxa - 2 (Felsenstein 1981).

6. Open an Excel file.
7. The likelihood ratio (LR) can be calculated as

   LR = 2 ((HKY85 + clock –lnL) – (HKY85 -lnL))

   (I believe this is because subtracting natural logs is the same as dividing…)
8. The degrees of freedom (DF) can be calculated as:

   DF = number of taxa – 2

   The cephalopod matrix has 15 taxa, so there are 13 degrees of freedom.
9. Use the *chidist* function in Excel to get a p-value:

   =chidist(LR,DF)

   If the p-values is less than 0.05, you can reject the simpler model (in this case, the global molecular clock.)

The null hypothesis, that the rate of evolution is homogeneous among all branches in the phylogeny, is rejected. Rates of substitution significantly vary among branches and a molecular clock is inappropriate. Why is the likelihood score of the alternative model higher than the null model?

Testing for a Local Molecular Clock

In the previous example we tested whether the entire tree fit a clock as opposed to every branch on the tree having an independent rate. We could also test whether a clade has a different rate from the rest of the tree. We can not do this in *PAUP*,* because *PAUP** does not allow us to specify different rates on different branches. Instead we will use *BASEML,* a program from the *PAML* package of phylogeny programs by Ziheng Yang. This program does ML anaysis of DNA sequences, and allows us to specify a tree and different distributions of rates on the tree. All these programs can be found at http://abacus.gene.ucl.ac.uk/software/paml.html.

This program is entirely controlled by the input files. You will need to download these from the web.

10. Go to the syllabus page of the IB 200A website. Download three files:

CephTree.trees, BaseML.ctl, CephSeq.nuc

The first file is *CephTree.trees* – open it with a text editor. As you can see this tree contains the same tree in Newick format as we used in the previous example. You will also see a '$1' after the clade containing Joubiniteuthis and Moroteuthis. This specifies that all the branches in this clade will have a different rate than the other branches in the tree. Open the file *BaseML.ctl* with a text editor. This is the control file for the BaseML program. When BASEML.exe is run, it automatically opens the control file, which must be in the same folder as it. The first line of the file specifies the file with the DNA sequences. The second line specifies the tree file. There are many other options in this file, but the only one that we are concerned with here is the **clock** option at the bottom of the tree. Here you can specify how the rates on the branches are grouped. 0 allows the rates on all the branches to vary independently; 1 enforces a molecular clock; and 2 enforces separate molecular clocks on each set of specified branches. We'll start with 0.

11. To run BaseML, just double-click on the BaseML program (If you ever want to use BaseML from a windows computer, it is slightly better to run it from the command prompt, but double-clicking will also work if there are no errors.)

12. When it is done, record the likelihood score. (It will be at the bottom of the screen, after –
    `lnL = )`
13. Open up the *BaseML.ctl* file in a text editor. Change the **clock** setting to 1. Run *BASEML* and record the likelihood score. Repeat the process for a **clock** setting of 2.
14. Use the likelihood ratio test to compare these models. Which model has the highest likelihoods? Why? Which model is the best?

Estimating Divergence Times Using r8s

Now we will use r8s (that's a pun pronounced "rates") to estimate divergence times. R8s uses a tree with branch lengths derived from another program, and then tries to estimate the node times by some measure of the rate differences between these branches. It is by Mike Sanderson and is freely available at http://loco.biosci.arizona.edu/r8s/. r8s uses normal nexus files as input files but you need to make a few additional commands.. In particular you need to specify the timing of the nodes which we can locate in time. As we learned in lecture, it is very difficult to locate a node in time.

Open the *CEPHr8s.nex* file in a text editor. The tree block is the same as you would find in any nexus file. Branch lengths are included after the colons. Some of the internal nodes are also named, after the closed parentheses, but before the columns. It is necessary to name nodes so that dates can be assigned to them. The r8s block has several commands.

lengths=persite means that the branch length is in changes per site not total number of changes

ultrametric=no means that the input tree is not ultrametric

fixage taxon=Clade1 age=150 sets the age of node Clade1 to 150

constrain taxon=node2 min_age=200 max_age=300 forces node two to be between 200 and 300. These times are measured backwards from the present, so that min_age=200 means that this divergence happened at least 200 (million?) years ago. I believe the units are relative, depending on what you input.

divtime method=LF starts the fitting algorithm using the Langley-Fitch method which deduces node times using maximum likelihood of the branch lengths assuming a constant rate of substitution

15. Download the file *CEPHr8s.nex* from the 200A website
16. Open the command terminal, type `cd` space, drag the folder labeled r8s into the terminal and hit return. Type
    `./r8s -v`
    hit return and then type
    `execute CEPHr8s.nex`

    This will execute the file and the commands that we inputted in the r8s block. According to the instructions this should be easier to do but I couldn't get it to work the other way.
17. When the program is done running type

`showage`

to output a table of node ages.

18. Now, let's try a different divergence time method, Non-Parametiric Rate Smoothing. Open the *CEPHr8s.nex* file in a text editor again, and change it to **method = NPRS**. When this option is set, the program minimizes the squares of the differences between adjacent branch lengths
19. Run r8s again. How do the node estimates for the two methods compare?