

## **Lab 8: Molecular Evolution**

There are many different features of genes and genomes that can be explored using phylogenetic methods. Today we're going to do a likelihood test for different rates of evolution in different parts of a DNA sequence. This is in general an important part of studying gene evolution. Knowing if different parts of genes evolve at different rates allows us to: use the appropriate model of sequence evolution when deducing gene phylogenies; detect the affects of natural selection on genes; and better understand the patterns and processes involved in the evolution of genes and genomes.

We will attempt to detect different rates of evolution between introns and exons within a gene. Our data set consists of five paralogous Protein Tyrosine Kinase genes from the *Caenorhabditis elegans* genome. As they are paralogous genes, their phylogeny is not well known and can only be inferred from the sequences themselves. Although we will only test for differences between these two broad regions in just a few genes, the same general principles can be applied to all likelihood tests for rate variation.

Finally, we will also use this test to illustrate the differences between Maximum Likelihood and Bayesian model testing. We will do the same test comparing the same two models using each method. We will explore the difference between joint estimation and marginal estimation of both the models and the nuisance parameters, and learn how to interpret the outcome from a Metropolis Hastings MCMC.

Note: The specific models employed here are used for DNA, but once you understand how a probabilistic substitution model plus a tree confers likelihood on a DNA alignment, you are prepared to understand how similar models can be devised for amino acids, morphological characters, etc.

### **Lab Prep**

1. Download the appropriate RAxML executable from this page <http://icwww.epfl.ch/~stamatak/index-Dateien/Page443.htm> . The executables are about a third of the way down the page. Most Mac people probably have Macs with Intel chips (iMAC, I think). You can try the pthreads version if you like -- threading allows RAxML to use multiple processors in parallel, which can speed up your jobs -- but it is probably simpler to not use those for now. Unpack it and put it in a folder named RAxML.
2. Download MrBayes from this page <http://mrbayes.csit.fsu.edu/download.php> . Unpack it, run it, and put it in a folder named MrBayes.

- Download these two files from our website and put them in the RaxML folder:  
PTK\_Nem\_phyl and Parts
- Download this file and put it in the MrBayes folder: PTK\_Nem\_small.nex

## The Model

### *Nucleotide substitution models*

There are multiple models for describing the probability of one nucleotide turning into another along one branch of a phylogeny. All rely on establishing a rate of change between every pair of nucleotides; these rates can be described in a **transition matrix**. For example under the Jukes-Cantor model every nucleotide has an equal chance of changing into every other nucleotide; such a model can be described with this matrix:

		To			
		A	C	G	T
From	A	–	$\alpha$	$\alpha$	$\alpha$
	C	$\alpha$	–	$\alpha$	$\alpha$
	G	$\alpha$	$\alpha$	–	$\alpha$
	T	$\alpha$	$\alpha$	$\alpha$	–

The transition matrix and the branch lengths can be used to calculate the probability of going from one nucleotide to another along any branch. A zero branch length will result in no changes, so that every site has a 100% chance of being in the same state that it began. On the other hand, after a branch of infinite length the probability of any site having a particular nucleotide will be that nucleotide's **equilibrium frequency**, regardless of what nucleotide it started out as. The equilibrium frequency depends only on the transition matrix, and not on the starting state. Under the Jukes-Cantor model every base has an equilibrium frequency of 0.25, so after an infinite amount of time every site an equal chance of being any base. For intermediate branch lengths between zero and infinity the probability of going from one state to another will depend on both the starting state and the transition matrix.

Another common model is *Kimura's two-parameter* in which there is a different rate of change for transitions (among pyrimidines, C and T, or among purines, A and G) and transversions (between pyrimidines and purines).

		To			
		A	C	G	T
From	A	–	$\beta$	$\alpha$	$\beta$
	C	$\beta$	–	$\beta$	$\alpha$
	G	$\alpha$	$\beta$	–	$\beta$
	T	$\beta$	$\alpha$	$\beta$	–

**Question 1.** Provide a matrix for a model in which there is one rate for transversions, one rate for transitions among pyrimidines and another for transitions among purines.

We are going to use the *General Time Reversible (GTR)* model. Time reversible means that the probability of going from state  $x$  to state  $y$  when going from node  $A$  to node  $B$ , is the same as going from state  $y$  to state  $x$  when going from node  $B$  to node  $A$ . When you use a reversible model, the root of the tree does not have to be defined. All the models we have looked at so far are reversible. You might assume that the GTR model has six rates and looks like this:

		To			
		A	C	G	T
From	A	–	$\alpha$	$\beta$	$\gamma$
	C	$\alpha$	–	$\delta$	$\epsilon$
	G	$\beta$	$\delta$	–	$\sigma$
	T	$\gamma$	$\epsilon$	$\sigma$	–

Under this model the rate of change between any two states is the same in either direction. This model actually has five free parameters, not six, because the entire matrix must be constrained to evolve at rate 1. However, you can make this model be even more general and still be time reversible by including the equilibrium frequencies of each base ( $\pi$ ).

		To			
		A	C	G	T
From	A	–	$\pi_C \alpha$	$\pi_G \beta$	$\pi_T \gamma$
	C	$\pi_A \alpha$	–	$\pi_G \delta$	$\pi_T \epsilon$
	G	$\pi_A \beta$	$\pi_C \delta$	–	$\pi_T \sigma$
	T	$\pi_A \gamma$	$\pi_C \epsilon$	$\pi_G \sigma$	–

This adds three new parameters to the model for a total of eight. It is three and not four, because all the equilibrium frequencies have to add to one. In practice many programs do not actually fit the equilibrium frequencies, but instead use the empirical frequencies, that is to say the actual frequencies of the bases in the sequence. The empirical frequencies are usually very close to the ML frequencies. This is the case for *RAxML*, but not *MrBayes*. Nevertheless, we can still consider these free parameters.

### ***Gamma-distributed Rates***

Does every site in the sequence evolve at the same rate? Probably not. This feature of gene evolution is often approximated using the gamma distribution. The idea is that there is a probability that any site will evolve at a given rate and that probability is drawn from the gamma distribution. For each site, the likelihood is calculated for every possible rate; the likelihood under each rate is multiplied by the probability of that rate under the gamma distribution; and all those likelihoods are added together to calculate the total likelihood for the site.

This distribution has two parameters,  $\alpha$ , which controls the shape of the distribution, and  $\beta$ , which controls the spread of the distribution. The  $\beta$  parameter is held constant for all these models and only the shape of the distribution is varied. Thus this adds one parameter to our model.

### ***Invariant Sites***

Another possibility is that a certain portion of sites can not change, as a consequence of strong stabilizing selection. We can deal with this by assuming that there is a probability,  $I$ , that any site can not change. It then works just like the Gamma-distributed rates so that the likelihood for each site is calculated for the case where the site can change and for the case where the site can not change and those likelihoods are added together. Thus if a site has the same nucleotide in every sequence, then that site has probability 1 under the invariant model, and the likelihood of the entire model at that site is  $I + (1-I) * \text{Likelihood if it can change}$ . On the other hand, if there is some variation at that site then, then that site has probability 0 under the invariant model, because there is no way you could get differences in nucleotides between species, if that site does not change, so the total likelihood for that site is just  $(1-I) * \text{Likelihood if it can change}$ .

### ***The Tree***

Often when comparing models the tree is not a free parameter in the analysis, because the tree is already known. For example if we were looking at orthologous genes instead of paralogous genes, then we could bring many characters to bear on the phylogeny of the taxa, which would also imply the gene tree. However, in this case we are studying paralogous genes, and thus can only deduce their phylogeny from their gene sequence. Therefore, we should consider our tree as one of the nuisance parameters that we have to calculate in order to test our models. In fact both the programs that we are using today are usually used to deduce trees, and the other parameters of the model are usually considered nuisance parameters. Today we're taking a different approach.

The tree consists of two different aspects, the topology and the branch lengths. Since we have 5 terminals, there are 15 possible trees. There's a complicated formula that describes how to calculate this number, but it is really not that tough. If you have  $n$  terminals, then start with the number  $2 * n - 5$ , and multiply that number by every odd number below it down to one. Thus if you have 5 terminals, you start with  $2 * 5 - 5 = 5$  and then multiply by all the odd numbers below it:  $5 * 3 * 1 = 15$ . The tree also has 7 branches, and each of those branches has an independent branch length. You can always calculate the number of branches for a fully bifurcating tree as  $2 * n - 3$ .

### ***The Model Test***

We are going to compare a model in which all these parameters are held equal throughout the gene to one in which one set of parameters is fit for the exons and another is fit for the introns. The only parameter that we will hold constant in both the introns and the exons is the tree topology, as there is good reason to expect that every part of the gene has the same evolutionary history.

Note that our whole gene model is nested within our intron/exon model. Thus we can do a likelihood ratio test; but first we must calculate how many extra parameters our more general model has. There are 8 parameters from our substitution mode, 1 gamma shape parameter, 1 parameter for the fraction of invariable sites and 7 branch lengths for a total of 17 extra parameters.

## **Maximum Likelihood Model Test**

## ***RAxML***

For the ML model test we are going to use *RaxML* by Alexandros Stamatakis. This program is actually designed for the fast inference of phylogenies with many taxa using ML. We are going to use it today, because it is free and has the models that we want. Our tree is so small that its fast search strategies are really not very helpful.

*RaxML* uses the *Phylip* format for data input files, rather than the *Nexus* format we are already familiar with. *Mesquite* is capable of converting files between these formats, should you find the need to do so. Just open the file, and save it normally to convert to *Nexus* or use **File>Export** to convert to *Phylip*. I have already done that conversion for you. Open the *PtK\_nem\_phyl* file in a text editor to see what it looks like. As you can see, this format is very simple, pretty much just the sequences and the names, with a few numbers to indicate the number of taxa and the number of characters.

*Nexus* files can contain a great deal of information other than the sequence, but *Phylip* files can only contain the sequences; other information must be passed with other files. For example the information breaking this sequence into introns and exons can be found in the *Parts* file. Open it in a text editor to see what it looks like. Pretty simple format, huh?

We are going to have to open another utility to run *RAxML*.

**For Macs:** Open the terminal by going to **Go>Utilities>terminal**

**For PCs:** Open the command window. **Start>Run**, then type “**cmd**” and hit **OK**.

In both cases a screen should open telling you what directory you’re in. (If you don’t understand what I’m talking about at all, that’s OK for this lab.) Type “**cd** ”; the space after the **cd** is very important. Then drag the ***RAxML*** folder that you created for the lab prep into the window. A path location will be added to your command. Hit **Enter**. Now whenever this terminal/command window looks for a file it will start by looking in that folder.

## ***Constrained Model***

We are going to use *RAxML* to calculate a maximum likelihood for our constrained model, that is to say the model in which both introns and exons evolve at the same rate. From now on we will call this model *modell*, just for simplicity’s sake.

RAxML is a command line program, which means it runs by typing commands at the command line, or from a control file containing those same commands.

I’m now going to relay to you a series of things that you should type in your Terminal window (Macs) or Command window (PCs).

You should type all these things on one line without hitting enter between commands (put spaces in-between the commands, though):

1. The name of the *RAxML* program on your computer. I can’t tell you what that is, because it is platform dependent. It could be something like

*“raxmlHPC.PowerMac5”* or *“RAxML-7.0.3-WIN”*. You don’t actually have to type it, just drag it onto your window.

2. **-s PTK\_Nem\_phyl.txt** will tell *RAxML*, where to find the sequence data.
3. **-m GTRGAMMAI** sets the substitution model to GTR with gamma distributed rates and a portion of invariant sites
4. **-n modell.txt** will add that phrase to the end of all the output files. I put .txt to make reading them easier.
5. Now hit **Enter**

A bunch of information will be printed to the screen. We can find all this information in one of the output files, so let’s not worry about what the screen says. You will find five new files in your **RAxML** folder.

Open the *RaxML\_info.modell* file. First thing you will see is a ton of warnings. Don’t worry about those, they come from the fact that I cut a bunch of sequences out of this data set, so several sites have no sequence data. The sequences with those sites removed can be found in the *reduced* file; we won’t worry about that. Then you will see a bunch of information about the way we set up the run: the models we used, etc. At the bottom of the group of data called “Partition: 0” you will find the empirical base frequencies. Take note of those. A few lines after that we will find the Maximum Likelihood values of the parameters from our analysis. Take a look over these. Do you understand what all these parameters are? Finally, near the bottom you will find the actual likelihood score for this run. Copy all these values, (ie. Parameters and likelihood) and paste them into a spreadsheet program, like Excel, or whatever.

The *RAxML\_log* file just shows some of the likelihoods calculated as the algorithm searched for the ML. The *RAxML\_parsimonyTree* file shows the starting tree for the search. We won’t bother with either of these.

We do care about the *RAxML\_result* file, as it contains our ML tree. You could just open it up and look at it, but it’s probably better if we can get a graphical interpretation. Open this file in *Mesquite*. Select **Phylip(trees)**. Name the nexus file whatever you want and save it. Click on the tree icon to open the tree window. Select **Drawing>Tree form>Square Tree**. Then select **Drawing> Branches Proportional to Lengths**. This tree is actually unrooted; we have no basis for determining where the root is. However, I find it is much easier for me to tell if unrooted trees have the same topology by picking an arbitrary root. Pick the longest branch and use the root tool to root the tree there. When comparing to other trees we will always root with this same tip of our gene tree, but keep in mind that this root is arbitrary.

## More General Model

Now we're going to do the same test, but we're going to distinguish between the introns and exons and allow all their parameters (except the tree topology) to vary independently. We will use the *Parts* file to distinguish between the different parts of our genes. The command you should use for this run is:

```
RAxML_program_name -s PTK_Nem_phyl.txt -m GTRGAMMAI -q Parts.text -M -n model2.txt
```

It is important that the commands are put in that order. **-q** identifies the partition and automatically separates all the parameters- except the tree parameters- between the partitions. The **-M** command also separates the tree branch lengths between partitions. We will save these files as *model2* instead of *model1*, just to distinguish them.

This will produce the exact same files as the last run, except there will be 3 *result* files. Open the *info* file. Page down to the bottom of the file where you will find all the parameter values. As you can see, there are two sets of parameter values: one set for partition 0 and one set for partition 1. Which partition is the introns and which the exons? Copy the likelihoods and the parameter values into your spread sheet, so that the partitions and the models can easily be compared.

There are three *result* files, because you have two different sets of branch lengths, one for each partition. The first file shows the average branch lengths. *PARTITION.0* and *PARTITION.1* show the branch lengths for each partition. Open all three of these trees in *Mesquite*, draw them with branch lengths and root them with the same taxon you used before.

## Model Comparison

First let's look at the trees. Do the topologies differ? What about the branch lengths? Which trees have different branch lengths under the different models?

Now let's look at the model parameters. Obviously there are two sets of model parameters for each partition of *model2*; look over them one by one. Do you see big differences between the partitions or between either partition and the constrained model? For example the proportion of invariable sites is very low for the combined model and for the introns, but pretty high for the exons. This makes sense, because stabilizing selection should have a much stronger affect on exons than it does on introns, so there should be more invariant sites in exons. The parameter in the combined analysis is obviously dominated by the affects of the introns.

**Question 2.** Other than the proportion of invariable sites, which parameter has the largest relative difference? For this parameter which two values are closest to each other? (For example you would say proportion of invariant sites are closest between the combined analysis and the introns partition.)

Finally we need to do the model test. For maximum likelihood values from nested models you can use the Likelihood Ratio Test. The  $LRT = 2 \cdot \ln(\text{Likelihood of the general model} / \text{likelihood of the constrained model})$ . The LRT can then be compared to a chi-squared

distribution with degrees of freedom equal to the difference in the number of parameters between the two models.

First calculate the LRT. We already have the log likelihoods, not the actual likelihoods, so the LRT, is just twice the difference between them. We already calculated that there are 17 extra parameters in the more complex model. Now compare that to a chi-squared distribution:

**Open Office:** =CHIDIST(*chi-squared value*;17)

**Excel:** =CHIDIST(*chi-squared value*,17)

**R:** pchisq(*chi-squared value*,17,lower.tail=FALSE)

That p-value represents our rejection of the null hypothesis that the rates are the same in both the introns and the exons. I wonder what would happen if we constrained the branch lengths, reducing the number of parameters by seven. Would that model be supported over the completely constrained model? Would the completely general model be selected over it?

## **Bayesian Model Test**

### ***MrBayes***

*MrBayes* is a tree searching program by Fredrik Ronquist, John Huelsenbeck, Bret Larget, and Paul van der Mark, that uses Bayesian estimation of the tree topology. We are going to use it to estimate the marginal likelihood for these two models. We will compare those likelihoods to each other in order to see if the one model is significantly better than the other. In the process we will also estimate the other parameters of the model, but our estimation of the likelihood will not be predicated on our estimate of the model parameters.

*MrBayes* is easier to use than *RaxML*. It uses the regular *Nexus* file format that we have already become familiar with with a few modifications. *Mesquite* can also export the appropriate type of *Nexus* file. Furthermore you don't have to open a command window to run it.

Execute your *MrBayes* program by double-clicking on it. To get our data into the program type:

**execute PTK\_Nem\_small.nex**

It will tell you that it read each part of the file.

### ***Constrained Model***

First thing that we want to do is set up our model of nucleotide substitution. Type:

**help lset**

You will get a list describing all the options. Below that is a table telling you how the options are set. I will not discuss all these options now, but if you are going to use this program



for your own analysis, you should definitely make yourself more familiar with them. We are going to reset two of these options to match the model we used in our ML test. We will reset *nst* to 6, this will establish our GTR model, which had six different rates independent of the equilibrium frequencies. We will also set *rates* to **Invgamma** to set our model to gamma-distributed rates with a proportion of invariable sites. To do this type:

**lset nst=6 rates=Invgamma**

Type **help lset** again to make sure that your parameters were set correctly.

Now we will turn to our priors. We did not have to worry about the priors for the maximum likelihood model, but they are of great importance for a Bayesian analysis. Type:

**help prset**

You will see a description of the priors followed by a table of their current settings. We will only concern ourselves with the priors that are relevant to our search. Revmatpr, the prior for the substitution rates, and Statefreqpr, the prior for the equilibrium frequencies, have a flat Dirichlet distribution for their default prior. This assumes basically no prior knowledge about these values. Shapepr, the prior for the gamma shape parameter, is set to uniform. This actually excludes some values for that parameter, but it assumes that all reasonable values are equally likely. Pinvarpr, the prior for the proportion of invariable sites, is a uniform prior between 0 and 1, which implies no prior knowledge. Topologypr, the prior for the topology, can be used to constrain our search to certain trees, but we have no reason to do that. Brlenspr, the prior for the branch lengths, could be made ultrametric, but this is an unrooted tree. So, we'll leave all these as defaults.

Finally we will set the parameters for the Metropolis-Hastings algorithm. Type:

**help mcmc**

You should see the same basic layout. These parameters are set for a complicated search of tree space, but we have a simple little tree, so we can set this run to take much less time.

**mcmcp Ngen=65000 Nruns=1 Nchains=2 Samplefreq=50 Printfreq=50  
Filename=model1**

*MrBayes* does a number of things to make sure it is fully exploring tree space. For one thing it does multiple searches for each run, some that are more liberal in their exploration of tree space than others. We will do only 2. It also does multiple independent runs at the same time to make sure that they converge on the same tree. We won't bother with that. We will also only run the algorithm for 65,000 generations, and sample every 50. Finally, we set the name of the output files to *model1*, just to keep everything straight.

Check your model again:

**showmodel**

If everything looks good, run it:

```
mcmc
```

A bunch of numbers will appear on the screen telling you how much longer the model will run for. When it's done hit “n” and Enter. You do not need to worry about what is printed to the screen, because our analysis is so simple, but if you were doing a real tree search, it would be very important.

### ***Analyzing the Output from Metropolis-Hastings***

The Metropolis-Hastings algorithm works by sampling values from our distribution in proportion to their probability under the distribution. In this case that distribution is the posterior probability of our model. To understand how this works imagine that we have a die, there is a 1 in 6 probability that any number between 1 in 6 will come up each time you role it. If you role that die 1200 times, then you would expect to get each number about 200 times. Under the Metropolis-Hastings algorithm each sample is like a role of the die. The fraction of times that any particular combination of parameter values comes up is the posterior probability of that combination. Furthermore, the fraction of times that any value comes up for any one parameter is the marginal posterior probability of that value, the posterior probability of that value integrated over all the values of the other parameters.

There should be three new files in the MrBayes folder. Open up *modell.p*. This shows the parameter values from our run. Each row is a separate sample from the MCMC. Open up *modell.t*. This shows the tree topologies and branch lengths from each sample corresponding to the parameter values in the equivalent row of the .p file.

Erase the first line with some numbers in brackets from the *modell.p* file, save it, and open it in R:

```
data1<-read.table(filename,header=TRUE)
```

Check the dimensions to make sure that it's 1301 by 15. Then check the column names.

```
dim(data1)  
colnames(data1)
```

Now plot the log likelihoods against the sample number:

```
plot(data1[,2])
```

As you can see, the likelihoods starts out low and then stabilize. This early period of low values is called burn-in and it is the time that it took the program to find the peak in likelihood space. It spent the rest of the time exploring that peak. We should throw out the burn-in values. To be conservative let's throw out the first 301 samples:

```
data1<-data1[302:1301,]
```

Check the dimensions and plot it again.

Now let's investigate some of our parameters. To see the probability distribution of a parameter we can just make a histogram:

**hist(data1[,4])**

That is the distribution of the A to C rate. For the proportion of invariable sites type:

**hist(data1[,15])**

Finally let's calculate the expectation of our values under the posterior probability distribution:

**mean(data1[,3:15])**

It is important to realize what this value is. Remember that the probability of sampling any particular value for parameter  $\alpha$  is:

$$\frac{\int P(Data|\alpha, \Theta) P(\alpha) P(\Theta) d\Theta}{\iint P(Data|\alpha, \Theta) P(\alpha) P(\Theta) d\Theta d\alpha}$$

Where  $\Theta$  is all the other parameters. Therefore the mean of a sample from this distribution is:

$$\frac{\iint \alpha P(Data|\alpha, \Theta) P(\alpha) P(\Theta) d\Theta d\alpha}{\iint P(Data|\alpha, \Theta) P(\alpha) P(\Theta) d\Theta d\alpha}$$

This is a little different than what you normally think of a mean as being. How do these values compare to the values calculated under the maximum likelihood *model*? Look back at your original spread sheet. Are any of them way off? If so try making a histogram of this parameter, and look at where the ML value falls. Is it near the peak?

The whole purpose of this exercise is to calculate the marginal likelihood of this model, that is to say, the denominator of our posterior probability equation, or the total likelihood summed over all parameter values:

$$\int P(Data|\Theta) P(\Theta) d\Theta$$

Where  $\Theta$  is all the parameters. This is a little tricky. We can calculate the marginal likelihood of our model as the harmonic mean of our likelihoods, where the harmonic mean is 1

over the mean of  $(1 / \text{likelihood})$ . This may seem strange, but if you consider the above equations it makes sense. You can ask me if you want a detailed description of why (or read the Wikipedia page on harmonic means to see what they are good for). To calculate the harmonic mean use the following formula:

```
temp.mean<-mean(data1[,2])
like.adj<-exp(data1[,2]-temp.mean)
inv.mean<-mean(1/like.adj)
like.model1<-temp.mean-log(inv.mean)
```

*temp.mean* had no bearing on the formula. I just put it in there to deal with the fact that *R* can't actually calculate  $\exp(-13000)$ , cause it's too small. Make sure to keep track of *like.model1* we will use it soon.

The last thing that we will look at is the trees. Remember that for the ML search we used only the one tree that the ML search concluded was best, just like the rest of the parameters. For the Bayesian model we summed over all the different possible tree topologies and branch lengths.

Open the *modell.t* file in *Mesquite*, and check out the trees. Make the branches proportional to the branch lengths and page through these trees. These are the trees from each sample of the MCMC. We must eliminate all the trees from burn-in. Select **Taxa&Trees>List of Trees**. Cut out the first 301 trees, by selecting them and then going to **List>Delete Selected Trees**. Now go back to your tree window.

It is OK to look at all these trees one by one, but it would be nice if we could summarize all this information in one figure. We'll make a consensus tree, a tree with all the clades that appear in more than 50% of our sampled trees. Go to **Tree>Tree Source>Consensus Tree**, then select **Stored Trees** and hit OK. Select **Majority Rule Consensus**, then hit **OK** twice. This may take a second.

There's your majority rule consensus tree. How does it compare to your Maximum Likelihood trees? Click on the **Branch Info Tool** (a question mark); use that to click on the two internal branches. The *consensus frequency* is the posterior probability of that division of taxa. Does this tree differ from your ML tree? Do you think that the other topologies had a large affect on your calculation of the marginal likelihood?

### ***More General Model***

Now we need to set *MrBayes* up to do the same analysis, but with our data partitioned. We already defined our partitions in the nexus file, and named our whole partition scheme *exin*. If you want to see how to do that, open the nexus file in a text editor. Now we need to establish that partition scheme as reality. Type:

```
set partition=exin
```

Now it's set, but we still have to establish which parameters are different for each parameter. First let's look at the matrix describing the partition:

```
showmodel
```

You will see a table labeled *active parameters*. The columns of this table are the partitions and the rows are the parameters. If both columns have the same number in a given row, then that parameters will be equal for both partitions. If you notice, all the model parameters are now back to the original settings so we'll have to reset our model:

```
lset nst=6 rates=Invgamma  
showmodel
```

Now we need to divide those parameters up, so that they vary between our partitions.

```
unlink statefreq=(all) revmat=(all) shape=(all) pinvar=(all) brlens=(all)  
showmodel
```

As you can see all of our parameters have now been unlinked, except the topology. We just need to reset our output file names, so that we don't write over our old files:

```
mcmc Filename=model2
```

Then we can run it:

```
mcmc
```

When it's done hit “n”, then you can shut down the program.

### ***Model Comparison***

So, we could compare all the parameters again from our different models, and compare those to our ML analysis, but I think that we've all had enough of that. You'll notice that there are two *model2.t* files. Those files have the trees with the branch lengths for the different partitions. The topologies should be the same.

Open the *model2.p* file in a text editor. As you can see each parameter appears twice, once for each partition.

We do still need to calculate the Bayes Factor for the comparison between these two models. The Bayes Factor is the ratio between the marginal likelihoods. It will not give us a p-value like the LRT did, but instead it implies the degree to which our data supports one model over another. In general Bayes Factors above 10 are considered good support, anything over 100 is great. You can actually calculate the Bayes Factors without doing separate runs for each model, by using reversible-jump MCMC, but that's a lesson for another time.

I'll describe the steps to calculate the Bayes Factor from our data, but won't give the commands:

- 1) Load the *model2.p* file into *R*. Don't forget to cut out the first line.
- 2) Examine and then remove the burn-in.
- 3) Calculate the marginal likelihood of the model.
- 4) Calculate the Bayes Factor. Remember that you actually have the logs of the marginal likelihoods, so what you want to calculate is  $\exp(\log \text{model2 like} - \log \text{model 1 like})$

**Question 3.** What is the Bayes Factor? Which model does it support? Is that strong support?

