

BSC3402L (6): Experimental Biology — Comparative Genomics

Laboratory Section: Thursdays from 4:00 to 6:00 PM.

Phylogenetic Analysis 2

Week Seven, Thursday, February 22, 2006

Author and Instructor: Fredrik Ronquist

Lab covers: (1) Running ML analysis in PAUP; (2) Running Bayesian MCMC analysis in MrBayes; (3) Running PAUP or MrBayes in batch mode.

Fredrik Ronquist
School of Computational Science
Florida State University
Tallahassee, FL 32306-4120
ronquist@scs.fsu.edu
850-645-1325

1. Getting Ready

In this lab, we will be using PAUP for maximum likelihood analyses and then we will switch to MrBayes and use that program to run Bayesian MCMC analyses. Just like last week, we will first run through the analyses using the small `primates.nex` dataset and then we will be running similar analyses using the `SRY.DNA.nex` dataset. Finally, you will be asked to run ML and Bayesian MCMC analyses on your own data.

To keep us from overwriting our old results, we will first create a separate directory called `phylo2` for today's exercise, and then run all jobs from that directory. To do this, log onto your machine and make sure you are in your home directory. Then use `mkdir phylo2` to create the directory `phylo2`, and then copy the two data files from the directory `phylo1` into `phylo2` by using `cp phylo1/*.* phylo2/` (copy all files in the directory `phylo1` that end in `.*` to directory `phylo2`). Now change to directory `phylo2` using `cd phylo2` and check that the two data files are there using `ls`.

2. Running ML Analyses in PAUP

Start PAUP by typing `paup`. You should see the following text:

```
P A U P *
Portable version 4.0b10 for Unix
Mon Oct 11 19:37:51 2004
```

```
-----NOTICE-----
This is a beta-test version. Please report any crashes,
apparent calculation errors, or other anomalous results.
There are no restrictions on publication of results obtained
with this version, but you should check the WWW site
frequently for bug announcements and/or updated versions.
See the README file on the distribution media for details.
-----
```

`paup>`

If you type `help`, you will see a list of the available commands. It should look like this:

The following commands are always available:

!	Edit	Help	Quit
CD	Execute	Leave	Set
Defaults	Factory	Log	Time
DSet	FStatus	LSet	ToNEXUS

The following commands require data from a `DATA` (or `TAXA` and `CHARACTERS`) or `DISTANCES` block (* = requires only `TAXA` block):

*Agree	*DerootTrees	*LoadConstr	Reweight	SurfCheck
AllTrees	*DescribeTrees	LScores	*RootTrees	*TaxPartition
AncStates	DScores	*MatrixRep	SaveAssum	*TaxSet
Assume	Exclude	MPRsets	SaveDist	*TreeDist
BandB	Export	NJ	*SaveTrees	*TreeInfo
BaseFreqs	ExSet	*Outgroup	ShowAnc	*TreeWts
Bootstrap	*Filter	PairDiff	*ShowConstr	*TStatus
CharPartition	GammaPlot	Permute	ShowDist	TypeSet
CharSet	*GenerateTrees	PScores	ShowMatrix	*Undelete

*ClearTrees	*GetTrees	PSet	ShowCharParts	UPGMA
Condense	HomPart	Puzzle	ShowRateSets	UserType
*Constraints	HSearch	RandTrees	ShowTaxParts	Weights
*ConTree	Include	*RateSet	*ShowTrees	Wts
CStatus	*Ingroup	Reconstruct	ShowUserTypes	WtSet
CType	Jackknife	*Restore	*SortTrees	
*Delete	Lake	RevFilter	StarDecomp	

Type "HELP COMMANDS" or "HELP CMDS" for a one-line description of each command.

Type "<cmdname> ?" to see brief usage and current default settings.

PAUP is a complex program and provides many commands, options and settings. The online help system only provides basic help on commands; a more complete manual is available at <http://paup.csit.fsu.edu/downl.html>. Today, we will make extensive use of the 'LSet' command to change the parameters of the Maximum Likelihood analysis. Before we look into this, however, we first need to read in the data by using **execute primates.nex**. We also need to set the search criterion to likelihood by issuing **set criterion=likelihood**. PAUP uses the 'set criterion' mechanism to determine what kind of analysis it is going to perform.

Now type **lset ?** to see a list of the available lset options and the current settings. The list should look something like this:

Usage: LSet [options...] ;

Available options:

Keyword	----- Option type -----	Current default setting --
NST	1 2 6	2
TRatio	<real-value> Estimate Previous	2
RMatrix	(<rAC><rAG><rAT><rCG><rCT>) Estimate Previous	(1 1 1 1 1)
RClass	(<cAC><cAG><cAT><cCG><cCT><cGT>)	(a b c d e f)
Variant	HKY F84	HKY
BaseFreq	Empirical Equal Estimate Previous (<frqA><frqC><frqG>)	Empirical
Rates	Equal Gamma SiteSpec	Equal
Shape	<real-value> Estimate Previous	0.5
NCat	<integer-value>	4
RepRate	Mean Median	Mean
PInvar	<real-value> Estimate Previous	0
SiteRates	Partition[:<charpartition-name>] Rateset[:<rateset-name>] Previous	<none>
Wts	RepeatCnt Ignore	RepeatCnt
InitBrLen	Rogers LS <real-value>	Rogers
LCollapse	No Yes	Yes
MaxPass	<integer-value>	20
Delta	<real-value>	1e-06
UseApprox	No Yes	Yes
ApproxLim	<real-value>	5
AdjustAppLim	No Yes	Yes
LogIter	No Yes	No
ZeroLenTest	No Full Crude	No
Recon	Marginal Joint	Marginal
AllProbs	No Yes	No
Clock	No Yes	No
UserBrLens	No Yes	No

MinMemReq	No Yes	No
StartVals	ParsApprox Arbitrary	ParsApprox
ParamClock	Standard Rambaut BrLens SplitTimes MDRambaut Thorne	Standard
MLDistforLS	No Yes	No
ShowQMatrix	No Yes	No

There is a bewildering array of options for the `lset` command listed above but we are only going to use a few of them today. We'll start from the top of the list. The 'NST' (Number of Substitution Types) option allows us to easily choose one of three major types of substitution models. When NST=1 then all substitutions have the same rate. When NST=2 then transitions and transversions have different rates. Finally, when NST=6 then all reversible substitution types have different rates. When NST is set to 2 then the transition/transversion rate ratio is set to the numerical value specified by the parameter TRatio. If TRatio is set to 'Estimate', then the rate ratio is estimated from data. Similarly, when NST=6 then RMatrix specifies the six different substitution rates unless RMatrix is set to 'Estimate' and these values are estimated from data. The final option we will worry about now is 'BaseFreq'. This parameter defines the stationary frequencies of the nucleotides. The default behavior is to estimate these frequencies by simply counting the frequency of the different nucleotides in the available sequences. This is referred to as 'Empirical' base frequencies. The base frequencies can also be set to be equal by choosing 'Equal' instead of 'Empirical'. Finally, the base frequencies can be estimated from the evolutionary model and the data, in which case we have 'BaseFreq=Estimate'.

As we will soon discover, there is a significant time penalty in maximum likelihood for every parameter we estimate from data. If we're willing to lock the parameter to a particular value, there is no such time penalty. Let's first analyze our data using the simplest of all possible nucleotide substitution models, the Jukes-Cantor model. Recall that this model assumes equal base frequencies and equal rates for all substitution types. This corresponds to `lset nst=1 basefreq=equal`. Type `lset ?` to make sure that the options for nst and basefreq changed accordingly. Finally start a heuristic search by typing `hsearch`. You should get output similar to:

```

Heuristic search settings:
  Optimality criterion = likelihood
  Likelihood settings:
    Number of substitution types = 1
    Assumed nucleotide frequencies (set by user):
      A=0.25000 C=0.25000 G=0.25000 T=0.25000
    Among-site rate variation:
      Assumed proportion of invariable sites = none
      Distribution of rates at variable sites = equal
    These settings correspond to the JC69 model
    Number of distinct data patterns under this model = 308
    Molecular clock not enforced
    Starting branch lengths obtained using Rogers-      Swofford approximation
method
    Trees with approximate likelihoods 5% or further from the target score are
rejected without additional iteration
    Branch-length optimization = one-dimensional Newton-      Raphson with pass
limit=20, delta=1e-06
    -ln L (unconstrained) = unavailable due to missing-data and/or ambiguities
    Starting tree(s) obtained via stepwise addition
    Addition sequence: as-is
    Number of trees held at each step during stepwise addition = 1
    Branch-swapping algorithm: tree-bisection-reconnection (TBR)

```

```

Steepest descent option not in effect
Initial 'MaxTrees' setting = 100
Branches collapsed (creating polytomies) if branch length is less than or
equal to 1e-08
'MulTrees' option in effect
Topological constraints not enforced
Trees are unrooted

```

```

Heuristic search completed
Total number of rearrangements tried = 546
Score of best tree(s) found = 6424.20245
Number of trees retained = 1
Time used = 2 sec (CPU time = 1.81 sec)

```

Use **showtrees** to see the tree. You can save the tree to file by using the command **savetrees file=<name>**, where you replace “<name>” with whatever name you want to give the file. This search is surprisingly fast so there should be no problem obtaining a bootstrap estimate of clade support based on 100 replicates (since 1 search takes about 2 seconds (see above), expect this analysis to take about 200 seconds). Start the bootstrap analysis by typing **bootstrap**. Heuristic search is the default for the bootstrap command and the lset options are persistent and stay in effect until you change them or set the search criterion to something other than likelihood.

Compare the resulting tree to the parsimony and NJ trees you obtained last week. It should be similar.

Now, let's assume that transitions and transversions have different rates, and let's estimate the transition/transversion rate ratio in a phylogenetic analysis. First type **lset nst=2 tratio=estimate** to set up the likelihood model. Then do **hsearch** as before. As you will discover, just adding one more parameter to be estimated from the data increased the search time with an order of magnitude (to about 30 seconds on my machine). To see the likelihood of the tree and the estimated transition/transversion ratio, type lscores. You should see something like:

```

Likelihood scores of tree(s) in memory:
Likelihood settings:
Number of substitution types = 2
Transition/transversion ratio estimated via ML
Assumed nucleotide frequencies (set by user):
A=0.25000 C=0.25000 G=0.25000 T=0.25000
Among-site rate variation:
Assumed proportion of invariable sites = none
Distribution of rates at variable sites = equal
These settings correspond to the K80(K2P) model
Number of distinct data patterns under this model = 339
Molecular clock not enforced
Starting branch lengths obtained using Rogers-Swofford approximation method
Branch-length optimization = one-dimensional Newton-Raphson with pass
limit=20, delta=1e-06
-ln L (unconstrained) = unavailable due to missing-data and/or ambiguities

```

```

Tree 1
-----
-ln L 6142.42908
Ti/tv:
exp. ratio 2.286026
kappa 4.572053

```

Time used to compute likelihoods = <1 sec (CPU time = 0.05 sec)

The log likelihood of the tree is -6142.4 and the transition/transversion rate ratio (kappa) is 4.57. Thus, in our data, transitions are about 4.6 times as likely as transversions.

In the next step, let us also estimate the base frequencies from the data. Type **lset basefreq=estimate** and then **hsearch**. Just adding three free parameters (there are four base frequencies but knowing three is sufficient to determine the fourth, so there are only three free parameters) to estimate slows down the likelihood search so much that we will not obtain the result within a reasonable period of time. When you are tired of watching the screen, press **<ctrl>+C** to stop the search. We can still show the best tree found so far and its likelihood score by using **showtrees** and **lscores**. When you use the latter command, you can examine the estimated base frequencies. Note that the base composition in this sequence is strongly skewed in that G:s are underrepresented (the estimated stationary frequency of G is likely to be close to 0.1).

```
Tree 1
-----
-ln L 6024.94811
Base frequencies:
  A 0.291133
  C 0.307688
  G 0.101037
  T 0.300143
```

In practice, we often need to invoke rather complex models to adequately explain the evolution of molecular sequences. To obtain ML estimates within reasonable amounts of time for such complex models, people usually take an iterative approach. First, they find a good initial estimate of the tree using a fast method, such as parsimony or neighbor-joining. Then they estimate the parameters of the substitution model on that tree using the ML criterion without changing the tree. Once the ML estimates of the substitution model parameters have been obtained, ML is used in a heuristic search to find, if possible, a better tree. During this phase, the substitution model parameters are fixed to their starting values, they are not estimated. If the heuristic tree search succeeds in finding a better tree, the substitution model parameters are reoptimized on that tree and a new round of heuristic search is tried with the new substitution model parameter estimates. This cycle alternating parameter estimation with heuristic tree searches continues until no better trees can be found. We will not have time to explore the iterative approach now so we'll stop here and shift to Bayesian inference. Type **quit** to leave PAUP.

3. Starting MrBayes

MrBayes is a program for the Bayesian inference of phylogeny written by John Huelsenbeck at the University of California, San Diego and me. If your computers are set up correctly, you should be able to start MrBayes by simply typing **mb**. You should be greeted by the following information:

MrBayes v3.1.2

(Bayesian Analysis of Phylogeny)

by

John P. Huelsenbeck and Fredrik Ronquist

Section of Ecology, Behavior and Evolution
Division of Biological Sciences
University of California, San Diego
johnh@biomail.ucsd.edu

School of Computational Science
Florida State University
ronquist@csit.fsu.edu

Distributed under the GNU General Public License

Type "help" or "help <command>" for information
on the commands that are available.

MrBayes >

If the system complains that the command is not recognized or if you get some other strange information, we need to install a symbolic link from your directory to the copy of MrBayes on your machine. **Only follow the instructions in the remainder of this paragraph if typing 'mb' did not start the program!** Install the link by typing `ln -s /usr/common/i686-linux/mb mb` (given that the path to the program is `"/usr/common/i686-linux/mb"`). You will now have a shortcut to MrBayes in the directory from which you ran the `ln` command; the shortcut is called `mb`. Your current working directory is probably not included in your path, which means that the system will not look for the command `mb` in your current working directory if you simply type `mb`. To make sure you invoke the copy of the `mb` command (or program) in your working directory, precede `mb` with `./` (meaning 'in the directory right here'). That is, type `./mb`. This should produce the welcome screen reproduced above.

4. Running Bayesian MCMC Analyses Using MrBayes

At the prompt, type **help** to see a list of the commands available in MrBayes. Most commands allow you to set values (options) for a range of parameters. If you type **help <command>**, where `<command>` is any of the listed commands, you will see the help information for that command as well as a description of the available options. The help facility also provides a way to see the current settings. For instance, typing **help lset** results in a list of the options and, at the end, a table giving the current settings.

A complete list of commands and options is given in the command reference, which can be downloaded from the program web site (<http:// mrbayes.net>) or, alternatively, you can browse the help information on the program site. Note that MrBayes, like PAUP, supports abbreviation of commands and options, so in many cases it is sufficient to type the first few letters of a command or option instead of the full name. If the abbreviation is ambiguous, MrBayes will not execute the command, so there is no risk involved in trying to abbreviate commands and options when working interactively with the program.

To read data into MrBayes from the file `primates.nex`, simply type **execute primates.nex** or simply **exe primates.nex**. At a minimum, two commands are required to specify the evolutionary model that will be used in the analysis, namely 'lset' and 'prset'. The 'showmodel' command is also useful for showing the current model settings. In general, **lset** is used to define the structure of the model and **prset** is used to define the prior probability distributions on the parameters of the model. In the following, we will specify a GTR + Γ model for the evolution of the mitochondrial sequences. In this model, all base frequencies and all reversible substitution rates are estimated from the data; in addition, rates are allowed to vary among sites according to a gamma distribution. In short, the model is incredibly complex compared to the models we analyzed previously using the ML criterion. It is the computational efficiency of the Bayesian approach that allows us to do this.

In general, a good start is to type **help lset**. Ignore the help information for now and concentrate on the table at the bottom of the output, which specifies the current settings. It should look like this:

Model settings for partition 1:

Parameter	Options	Current Setting
Nucmodel	4by4/Doublet/Codon	4by4
Nst	1/2/6	1
Code	Universal/Vertmt/Mycoplasma/ Yeast/Ciliates/Metmt	Universal
Ploidy	Haploid/Diploid	Diploid
Rates	Equal/Gamma/Propinv/Invgamma/Adgamma	Equal
Ngammacat	<number>	4
Nbetacat	<number>	5
Omegavar	Equal/Ny98/M3	Equal
Covarion	No/Yes	No
Coding	All/Variable/Noabsencesites/ Nopresencesites	All
Parsmodel	No/Yes	No

First, note that the table is headed by `Model settings for partition 1`. By default, MrBayes divides the data into one partition for each type of data you have in your `DATA` block. If you have only one type of data, all data will be in a single partition.

The `Nucmodel` setting allows you to specify the general type of DNA model. The `Doublet` option is for the analysis of paired stem regions of ribosomal DNA and the `Codon` option is for analyzing the DNA sequence in terms of its codons. We will analyze the data using a standard nucleotide substitution model (the only model we cover in this course), in which case the default `4by4` option is appropriate, so we will leave `Nucmodel` at its default setting.

The general structure of the substitution model is determined by the `Nst` setting, which is essentially the same as in PAUP. By default, all substitutions have the same rate (`Nst=1`), corresponding to the F81 model (which is equivalent to the JC model except that the stationary state frequencies are allowed to be different).

We want the GTR model ($N_{st}=6$) instead of the F81 model so we type `lset nst=6`. MrBayes should acknowledge that it has changed the model settings.

The `Code` setting is only relevant if the `Nucmodel` is set to `Codon`. The `Floidy` setting is also irrelevant for us. However, we need to change the `Rates` setting from the default `Equal` (no rate variation across sites) to `Gamma`. Do this by typing `lset rates=gamma`. Again, MrBayes will acknowledge that it has changed the settings. We could have changed both `lset` settings at once if we had typed `lset nst=6 rates=gamma` in a single line.

We will leave the `Ngammacat` setting (the number of discrete categories used to approximate the gamma distribution) at the default of 4. This is a setting that only affects the way in which the program approximates the gamma distribution. In most cases, four rate categories are sufficient. It is possible to increase the accuracy of the likelihood calculations by increasing the number of rate categories. However, the time it will take to complete the analysis will increase in direct proportion to the number of rate categories you use, and the effects on the results will be difficult or impossible to detect in most cases.

Of the remaining settings, it is only `Covarion` and `Parsmodel` that are relevant for single nucleotide models. We will use neither the parsimony model nor the covarion model for our data, so we will leave these settings at their default values. If you type `help lset` now to verify that the model is correctly set, the table should look like this:

Model settings for partition 1:

Parameter	Options	Current Setting
Nucmodel	4by4/Doublet/Codon	4by4
Nst	1/2/6	6
Code	Universal/Vertmt/Mycoplasma/ Yeast/Ciliates/Metmt	Universal
Floidy	Haploid/Diploid	Diploid
Rates	Equal/Gamma/Propinv/Invgamma/Adgamma	Gamma
Ngammacat	<number>	4
Nbetacat	<number>	5
Omegavar	Equal/Ny98/M3	Equal
Covarion	No/Yes	No
Coding	All/Variable/Noabsencesites/ Nopresencesites	All
Parsmodel	No/Yes	No

We now need to set the priors for our model. A simple solution is to use the default priors, which is what we are going to do. If you are interested in more on the priors read on, otherwise jump to **showmodel** below.

There are five types of parameters in the model: the topology, the branch lengths, the four stationary frequencies of the nucleotides, the six different nucleotide substitution rates, and the shape parameter of the gamma distribution of rate variation. It is a good idea to type `help prset` to obtain a list of the default settings for these parameter types. The table at the end of the help information reads:

Parameter	Options	Current Setting
Tratioopr	Beta/Fixed	Beta(1.0,1.0)
Revmatpr	Dirichlet/Fixed	Dirichlet(1.0,1.0,1.0,1.0,1.0,1.0)
Aamodelpr	Fixed/Mixed	Fixed(Poisson)
Aarevmatpr	Dirichlet/Fixed	Dirichlet(1.0,1.0,...)
Omegapr	Dirichlet/Fixed	Dirichlet(1.0,1.0)
Ny98omegalpr	Beta/Fixed	Beta(1.0,1.0)
Ny98omega3pr	Uniform/Exponential/Fixed	Exponential(1.0)
M3omegapr	Exponential/Fixed	Exponential
Codoncatfreqs	Dirichlet/Fixed	Dirichlet(1.0,1.0,1.0)
Statefreqpr	Dirichlet/Fixed	Dirichlet(1.0,1.0,1.0,1.0)
Treeheightpr	Exponential/Gamma	Exponential(1.0)
Ratepr	Fixed/Variable=Dirichlet	Fixed
Shapepr	Uniform/Exponential/Fixed	Uniform(0.0,200.0)
Ratecorrpr	Uniform/Fixed	Uniform(-1.0,1.0)
Pinvarpr	Uniform/Fixed	Uniform(0.0,1.0)
Covswitchpr	Uniform/Exponential/Fixed	Uniform(0.0,100.0)
Symdirihyperpr	Uniform/Exponential/Fixed	Fixed(Infinity)
Topologypr	Uniform/Constraints	Uniform
Brlenspr	Unconstrained/Clock	Unconstrained:Exp(10.0)
Speciationpr	Uniform/Exponential/Fixed	Uniform(0.0,10.0)
Extinctionpr	Uniform/Exponential/Fixed	Uniform(0.0,10.0)
Sampleprob	<number>	1.00
Thetapr	Uniform/Exponential/Fixed	Uniform(0.0,10.0)

We need to focus on `Revmatpr` (for the six substitution rates of the GTR rate matrix), `Statefreqpr` (for the stationary nucleotide frequencies of the GTR rate matrix), `Shapepr` (for the shape parameter of the gamma distribution of rate variation), `Topologypr` (for the topology), and `Brlenspr` (for the branch lengths).

The default prior probability density is a flat Dirichlet (all values are 1.0) for both `Revmatpr` and `Statefreqpr`. This is appropriate if we want estimate these parameters from the data assuming no prior knowledge about their values. It is possible to fix the rates and nucleotide frequencies but this is generally not recommended. However, it is occasionally necessary to fix the nucleotide frequencies to be equal, for instance in specifying the JC and SYM models. This would be achieved by typing `prset statefreqpr=fixed(equal)`.

If we wanted to specify a prior that put more emphasis on equal nucleotide frequencies than the default flat Dirichlet prior, we could for instance use `prset statefreqpr = Dirichlet(10,10,10,10)` or, for even more emphasis on equal frequencies, `prset statefreqpr=Dirichlet(100,100,100,100)`. Note that the sum of the numbers in the Dirichlet distribution determines how focused the distribution is, and the balance between the numbers determines the expected proportion of each nucleotide (in the order A, C, G, and T).

In our analysis, we will leave the prior on state frequencies at its default setting. If you have changed the setting according to the suggestions above, you need to change it back by typing `prset statefreqpr=Dirichlet(1,1,1,1)` or `prset = Dir(1,1,1,1)` if you want to save some typing.

The `Shapepr` parameter determines the prior for the alpha (shape) parameter of the gamma distribution of rate variation. We will leave it at its default setting, a uniform distribution spanning a wide range of `_` values.

For topology, the default `Uniform` setting for the `Topologypr` parameter puts equal probability on all distinct, fully resolved topologies. The alternative is to constrain some nodes in the tree to always be present but we will not attempt that in this analysis.

The `Brlenspr` parameter can either be set to unconstrained or clock-constrained. For trees without a molecular clock (unconstrained) the branch length prior can be set either to exponential or uniform. The default exponential prior with parameter 10.0 should work well for most analyses. It has an expectation of $1/10 = 0.1$ but the distribution is still sufficiently vague that it will have little influence on the posterior unless the data are very weak.

In summary, we will not change the default priors. To check the model before we start the analysis, type `showmodel`. This will produce the following output:

Model settings:

```
Datatype = DNA
Nucmodel = 4by4
Nst = 6
      Substitution rates, expressed as proportions
      of the rate sum, follow a Dirichlet
      (1.00,1.00,1.00,1.00,1.00,1.00)
Covarion = No
# States = 4
      State frequencies have a Dirichlet prior
Rates = Gamma
      Gamma shape parameter is uniformly dist-
      ributed on the interval (0.05,50.00).
      Gamma distribution is approximated using 4 categories.
```

Active parameters:

```
Parameters
-----
Revmat      1
Statefreq   2
Shape       3
Topology    4
Brlens      5
-----

1 -- Parameter = Revmat
   Prior      = Dirichlet(1.00,1.00,1.00,1.00,1.00,1.00)
2 -- Parameter = Statefreq
   Prior      = Dirichlet
3 -- Parameter = Shape
   Prior      = Uniform(0.05,50.00)
4 -- Parameter = Topology
   Prior      = All topologies equally probable a priori
5 -- Parameter = Brlens
   Prior      = Branch lengths are Unconstrained:Exponential(10.0)
```

This gives a nice overview of the model settings. The analysis is started by issuing the `mcmc` command. However, before doing this, we recommend that you review the run settings by typing `help mcmc`. One of the reasons for this is that you cannot stop a MrBayes run once it is started without abandoning the session completely (in the worst case, use `ctrl-c` on Windows or Unix machines to accomplish this). The `help mcmc` command will produce the following table at the bottom of the output:

Parameter	Options	Current Setting
Seed	<number>	1160489349
Swapseed	<number>	1160489349
Ngen	<number>	1000000
Nruns	<number>	2
Nchains	<number>	4
Temp	<number>	0.200000
Reweight	<number>, <number>	0.00 v 0.00 ^
Swapfreq	<number>	1
Nswaps	<number>	1
Samplefreq	<number>	100
Printfreq	<number>	100
Printall	Yes/No	Yes
Printmax	<number>	8
Mcmcdiag	Yes/No	Yes
Diagnfreq	<number>	1000
Minpartfreq	<number>	0.10
Allchains	Yes/No	No
Allcomps	Yes/No	No
Relburnin	Yes/No	Yes
Burnin	<number>	0
Burninfrac	<number>	0.25
Stoprule	Yes/No	No
Stopval	<number>	0.01
Filename	<name>	primates.nex.<p/t>
Startingtree	Random/User	Random
Nperts	<number>	0
Savebrlens	Yes/No	Yes
Ordertaxa	Yes/No	No

The `Seed` is simply the seed for the random number generator. The `Ngen` setting is the number of generations for which the analysis will be run. It is useful to run a small number of generations first to make sure that the analysis is correctly set up and to get an idea of how long it will take to complete a longer analysis. We will start with only 10,000 generations. To change the `Ngen` setting, we *cannot* use the `mcmc` command because this will start the analysis. Instead we use the `mcmcp` command, which is equivalent to `mcmc` except that it does not start the analysis. Type `mcmcp ngen=10000` to set the number of generations to 10,000.

The `Samplefreq` setting determines how often the chain is sampled. By default, the chain is sampled every 100th generation, and this works well for most analyses. When the chain is sampled, the current values of the model parameters are printed to file. The substitution model parameters are printed to a `.p` file (in our case, the file will be called `primates.nex.p`), which is a tab delimited text file that can be imported into most statistics and graphing programs. The topology and branch lengths are printed to a `.t` file (in our case, the

file will be called `primates.nex.t`), which is a Nexus tree file that can be imported into programs like PAUP*. The root of the `.p` and `.t` file names can be altered using the `Filename` setting.

By default, MrBayes uses Metropolis coupling to improve the MCMC sampling of the target distribution. The `Swapfreq`, `Nchains`, and `Temp` settings together control the Metropolis coupling behavior. When `Nchains` is set to 1, no heating is used. When `Nchains` is set to a value n larger than 1, then $n - 1$ heated chains are used. By default, `Nchains` is set to 4, meaning that MrBayes will use 3 heated chains and one "cold" chain. In our experience, heating is essential for problems with more than about 50 taxa, whereas smaller problems often can be analyzed successfully without heating. It is still largely an open question whether adding more than three heated chains is helpful in analyzing large and difficult data sets. The time complexity of the analysis is directly proportional to the number of chains used.

MrBayes uses an incremental heating scheme, in which chain i is heated by raising its posterior probability by the power $1 / (1 + i_{_})$, where $_$ is the temperature controlled by the `Temp` parameter. The effect of the heating is to flatten out the posterior probability, such that the heated chains more easily find isolated peaks in the posterior distribution and can help the cold chain move more rapidly between these peaks. Every `Swapfreq` generation, two chains are picked at random and an attempt is made to swap their states. For many analyses, the default settings should work nicely.

The `Printfreq` parameter controls the frequency with which the state of the chains is printed to screen. Leave `Printfreq` at the default value (print to screen every 100th generation).

The `Startingtree` parameter can be used to feed the chain(s) with a user-specified starting tree. The default behavior is to start each chain with a different random tree.

To help us determine when to stop the analysis, MrBayes by default runs two independent analyses. The tree samples from these analyses are then compared with each other at some predetermined frequency interval (the 'diagnfreq' setting, by default every 1000th generation). The metric that is used for these comparisons is the average standard deviation of the split frequencies (these are the support values or posterior probabilities for each group in the tree, ranging numerically from 0.0 to 1.0). If the MCMC procedure has reached convergence and we have two or more independent samples of the posterior probability distribution of high quality, this value should be close to 0.0. If you want to be on the safe side, you might want to run three or four independent analyses but we will go with the default of two for now. The more parallel analyses you start, the longer your analysis will take, of course, for the same total number of generations.

Now we are ready to start the analysis. Type `mcmc`. MrBayes will first print information about the model and then list the proposal mechanisms that are used in sampling from the posterior distribution. In our case, the proposals are the following:

```
The MCMC sampler will use the following moves:  
  With prob. Chain will change
```

```

4.35 % param. 1 (revmat) with Dirichlet proposal
4.35 % param. 2 (state frequencies) with Dirichlet proposal
4.35 % param. 3 (gamma shape) with multiplier
65.22 % param. 4 (topology and branch lengths) with extending TBR
21.74 % param. 4 (topology and branch lengths) with LOCAL

```

Note that MrBayes will spend most of its effort changing topology and branch lengths. In our experience, topology and branch lengths are the most difficult parameters to integrate over and we therefore let MrBayes spend a large proportion of its time proposing new values for these parameters. The proposal probabilities can be changed with the `props` command but be warned that inappropriate changes of proposal probabilities may destroy any hopes of achieving convergence.

After the initial log likelihoods, MrBayes will print the state of the chains every 100th generation, like this:

```

Chain results:

  1 -- [-7425.887] (-7679.393) (-7788.508) (-7865.109) * [-7507.457] (-7711.838) (-7812.353) (-7849.566)
100 -- (-6837.197) (-6957.222) [-6638.953] (-6877.411) * [-6600.482] (-6757.712) (-6779.327) (-6777.602) -- 0:01:39
200 -- (-6545.817) (-6354.106) [-6242.265] (-6311.658) * [-6242.782] (-6359.851) (-6402.040) (-6328.580) -- 0:00:49
300 -- (-6296.616) (-6080.818) [-6072.028] (-6050.853) * (-6123.103) (-6264.465) (-6148.979) [-6094.193] -- 0:01:04
400 -- (-6121.174) [-5948.266] (-5952.140) (-6005.619) * [-5987.812] (-6165.219) (-6021.067) (-6038.789) -- 0:01:12
500 -- (-6011.477) [-5907.528] (-5917.751) (-5929.179) * [-5930.754] (-6120.004) (-5972.281) (-6009.348) -- 0:00:57
600 -- (-5951.584) (-5870.154) [-5834.123] (-5872.068) * [-5908.744] (-6036.368) (-5919.192) (-5974.685) -- 0:01:02
700 -- (-5937.809) (-5844.145) [-5792.316] (-5864.596) * [-5836.377] (-5948.057) (-5891.798) (-5907.714) -- 0:01:06
800 -- (-5844.860) (-5831.864) [-5796.581] (-5804.880) * [-5767.053] (-5943.500) (-5880.383) (-5852.889) -- 0:00:57
900 -- (-5812.108) (-5808.852) [-5769.247] (-5802.321) * [-5763.083] (-5928.195) (-5841.726) (-5842.519) -- 0:01:00
1000 -- (-5785.351) (-5800.538) [-5758.632] (-5807.451) * [-5740.435] (-5883.480) (-5834.830) (-5829.896) -- 0:01:03

Average standard deviation of split frequencies: 0.117851

...

9100 -- (-5735.992) [-5721.282] (-5729.222) (-5735.859) * [-5729.749] (-5728.892) (-5735.633) (-5734.412) -- 0:00:06
9200 -- (-5740.708) [-5720.207] (-5735.005) (-5728.959) * [-5722.369] (-5725.822) (-5733.410) (-5726.371) -- 0:00:05
9300 -- (-5748.757) [-5725.245] (-5734.551) (-5733.714) * (-5725.401) [-5720.981] (-5728.135) (-5732.691) -- 0:00:04
9400 -- (-5734.807) [-5724.413] (-5736.371) (-5733.678) * [-5727.694] (-5723.414) (-5733.792) (-5728.815) -- 0:00:04
9500 -- (-5733.603) [-5725.293] (-5732.989) (-5736.105) * (-5729.254) [-5726.885] (-5725.746) (-5735.932) -- 0:00:03
9600 -- [-5724.564] (-5724.201) (-5729.236) (-5730.766) * [-5726.381] (-5733.251) (-5730.594) (-5728.112) -- 0:00:02
9700 -- (-5730.260) [-5723.910] (-5729.155) (-5726.642) * [-5726.313] (-5729.738) (-5726.408) (-5724.949) -- 0:00:02
9800 -- (-5732.039) (-5732.065) [-5730.287] (-5728.816) * [-5723.578] (-5728.654) (-5723.603) (-5727.953) -- 0:00:01
9900 -- [-5728.920] (-5732.020) (-5733.553) (-5730.998) * [-5719.771] (-5727.824) (-5724.544) (-5723.332) -- 0:00:00
10000 -- (-5723.307) (-5733.976) (-5729.283) [-5720.703] * [-5721.329] (-5729.010) (-5736.245) (-5722.009) -- 0:00:00

Average standard deviation of split frequencies: 0.005169

Continue with analysis? (yes/no):

```

The first column lists the generation number. The two sets of four columns with negative numbers, separated by a star, each correspond to one chain (or rather, one physical location in computer memory). The star separates the two independent runs. The numbers are the log likelihood values of the chains. The chain that is currently the cold chain has its value surrounded by square brackets. When two chains successfully change states, they trade places in computer memory because this is by far the most computationally efficient implementation. This results in their trading column positions. In a healthy run, the cold chain should move around among the columns because this means that the cold chain successfully swaps states with the heated chains. If the cold chain gets stuck in one of the columns, then the heated chains are not successfully contributing states to the cold chain. The analysis may then have to be run longer or the temperature difference between chains may have to be lowered.

The last column gives the time left to completion of the specified number of generations. It will take a while before these values stabilize because it is difficult to estimate the time to completion in the beginning of the analysis. After a while, though, these numbers stabilize and you should have a fairly predictable count-down. This

analysis approximately takes 1 second per 100 generations. At the end of the run, MrBayes asks whether you want to continue with the chain.

Since the standard deviation of split frequencies is very low, we need not continue this analysis. If the value had been higher, though, we could have chosen to type in a number of extra generations to run the analysis at this point. A good stopping criterion for moderate to large analyses is a standard deviation of split frequencies below 0.05.

Another symptom of convergence may be obtained by observing the log likelihood (or, more exactly, the log probability of the data given the parameter values) of the cold chain. In the beginning of the run, the log likelihood of the cold chain typically increases rapidly. This phase of the run is referred to as the burn-in and the samples from this phase are typically discarded. Once the likelihood of the cold chain stops to increase and starts to randomly fluctuate within a more or less stable range, the run may have reached stationarity. At stationarity, we typically also observe that the cold and heated chains are exchanging states frequently, resulting in the cold chain moving around freely among the columns. By default, MrBayes uses a burn-in of 25 % of the samples when it calculates its convergence diagnostic, the average standard deviation of split frequencies.

Since it looks like we have convergence, answer `no` to the question of whether the chain should be extended. MrBayes will respond by printing various run statistics. Pay particular attention to the table of acceptance rates:

```
Acceptance rates for the moves in the "cold" chain of run 1:
  With prob. Chain accepted changes to
    28.22 % param. 1 (revmat) with Dirichlet proposal
     9.98 % param. 2 (state frequencies) with Dirichlet proposal
    27.38 % param. 3 (gamma shape) with multiplier
    11.98 % param. 4 (topology and branch lengths) with extending TBR
    14.72 % param. 4 (topology and branch lengths) with LOCAL
Acceptance rates for the moves in the "cold" chain of run 2:
  With prob. Chain accepted changes to
    28.86 % param. 1 (revmat) with Dirichlet proposal
    11.27 % param. 2 (state frequencies) with Dirichlet proposal
    28.93 % param. 3 (gamma shape) with multiplier
    11.70 % param. 4 (topology and branch lengths) with extending TBR
    15.93 % param. 4 (topology and branch lengths) with LOCAL
```

As a rough rule of thumb, an efficient Metropolis-Hastings MCMC sampler will have acceptance rates somewhere in the range 10 % to 70 %. If the acceptance rate is too high, then the proposal mechanism is making too modest suggestions of new states. If the rate is too low, on the other hand, then the proposals are too bold. Both situations are likely to lead to slow mixing, which means that you will have to run the chain longer to obtain convergence. The acceptance rates can be changed by modifying the tuning parameters using the `props` command. The topology proposals are difficult, however, and one often has to be satisfied with relatively low acceptance rates for these. In our analysis, the acceptance rates are OK for all proposals.

Finally, let's examine the chain swap information:

Chain swap information for run 1:

	1	2	3	4
1		0.54	0.30	0.12
2	1659		0.62	0.32
3	1709	1695		0.61
4	1615	1654	1668	

Chain swap information for run 2:

	1	2	3	4
1		0.43	0.18	0.06
2	1711		0.56	0.25
3	1655	1702		0.55
4	1651	1567	1714	

The bottom row of the upper diagonal contains the acceptance rates for the swaps between chains separated by only one heating step. Again, a rough rule of thumb is that these acceptance rates should lie in the range 10 % to 70 %. If the acceptance rates are too low, you can try to lower the temperature difference; if the rates are too high, the temperature should be increased instead. In our case, the acceptance rates are all within the suggested range.

During the run, samples of the substitution model parameters have been written to two separate files, one called `primates.nex.run1.p` and the other `primates.nex.run2.p`. These files look something like this (you can check it by opening a new terminal window and displaying the file using, for instance, **more**):

```
[ID: 5848203808]
  Gen  LnL      TL      r(G<->T)  ...  pi (G)    pi (T)    alpha
    1 -7433.991  2.098  0.166667  ...  0.250000  0.250000  0.500000
  100 -6601.275  1.939  0.143314  ...  0.226052  0.282844  0.732008
  ...
 9900 -5727.425  2.709  0.026515  ...  0.079022  0.249481  0.382468
10000 -5728.934  3.177  0.015705  ...  0.084921  0.243379  0.387159
```

From left to right, the columns contain: (1) the generation number; (2) the log likelihood of the cold chain; (3) the total tree length (the sum of all branch lengths; (4) the six GTR rate parameters; (5) the four stationary nucleotide frequencies; and (6) the shape parameter of the gamma distribution of rate variation.

To summarize the information in the `.p` files, type `sump burnin=25`. By default, `sump` will summarize the information in the `.p` file generated most recently, but the filename can be changed if necessary. Beware that the burn-in value is given as the number of samples to be discarded as the burn-in, not as the number of generations to be discarded. Since we have been sampling every 100th generation and we ran the chain for 100,000 generations, a burn-in of 25 % corresponds to 2,500 generations or 25 samples (to be exact, it is equivalent to 26 samples since the first generation is always sampled). The `sump` command will generate a plot of the generation versus the log probability of the data. If we are at stationarity, this plot should look like 'white noise', that is, there should be no tendency of steady increase or decrease. At the bottom of the output, there is a table summarizing the samples of the parameter values:

Parameter	Mean	Variance	95% Cred. Interval		Median	PSRF *
			Lower	Upper		
TL	2.872467	0.049941	2.400000	3.310000	2.865000	1.070
r(A<->C)	0.045923	0.000049	0.031628	0.059208	0.046094	1.009
r(A<->G)	0.484462	0.001604	0.404829	0.582137	0.482340	0.995
r(A<->T)	0.038671	0.000067	0.024350	0.056935	0.038432	1.083
r(C<->G)	0.030986	0.000176	0.007817	0.053856	0.031923	0.994
r(C<->T)	0.384893	0.001157	0.313217	0.459387	0.385211	1.008
r(G<->T)	0.015066	0.000123	0.000102	0.045086	0.012413	0.994
pi(A)	0.350097	0.000147	0.330104	0.376843	0.349321	0.999
pi(C)	0.321788	0.000097	0.302723	0.339289	0.320625	1.016
pi(G)	0.081197	0.000030	0.069921	0.091668	0.080984	1.013
pi(T)	0.246919	0.000093	0.228528	0.265909	0.247857	1.012
alpha	0.404507	0.001502	0.344946	0.496584	0.399295	0.994

* Convergence diagnostic (PSRF = Potential scale reduction factor [Gelman and Rubin, 1992], uncorrected) should approach 1 as runs converge. The values may be unreliable if you have a small number of samples. PSRF should only be used as a rough guide to convergence since all the assumptions that allow one to interpret it as a scale reduction factor are not met in the phylogenetic context.

The table should be self-explanatory. Note that the six rate parameters of the GTR model are given as proportions of the rate sum (the so-called Dirichlet parameterization).

Trees and branch lengths are printed to two files that are called `primates.nex.run1.t` and `primates.nex.run2.t`. These files look something like this:

```
#NEXUS
[ID: 5848203808]
begin trees;
  translate
    1 Lemur_catta,
    2 Homo_sapiens,
    3 Pan,
    4 Gorilla,
    5 Pongo,
    6 Hylobates,
    7 Macaca_fuscata,
    8 M_mulatta,
    9 M_fascicularis,
    10 M_sylvanus,
    11 Saimiri_sciureus,
    12 Tarsius_syrichta;
  tree rep.1 =
  (((((2:0.100000,(8:0.100000,7:0.000748):0.197415):0.100000,(5:0.100000,6:0.100000):0.100000):0.100000,(9:0.100000,(10:0.100000,3:0.100000):0.100000):0.100000):0.100000,12:0.100000):0.100000,11:0.100000):0.100000,4:0.100000,1:0.100000);
  ...
  tree rep.10000 =
  (((6:0.126475,((4:0.079300,(3:0.049876,2:0.066929):0.040517):0.103191,5:0.138220):0.105224):0.147212,(10:0.059671,((8:0.035939,7:0.012332):0.037992,9:0.069477):0.060859):0.221336):0.149243,11:0.627561):0.180115,12:0.525606,1:0.339476);
end;
```

Note that branch lengths are printed to file only if this is requested when starting the analysis (in the `mcmc` or `mcmcpr` command).

To summarize the tree and branch length information, type `sumt burnin=25`. The `sumt` and `sump` commands each have separate burn-in settings so it is necessary to give the burn-in here again. Otherwise, many of the settings in MrBayes are persistent and need not be repeated every time a command is executed. To make sure the settings for a particular command are correct, you can always use `help <command>`.

The `sumt` command will output, among other things, a tree with clade credibility (posterior probability) values and a phylogram (if branch lengths have been saved). In the background, the command will create three additional files. The first is a `.parts` file, which contains the list of taxon bipartitions, their posterior probability (the proportion of sampled trees containing them), and the branch lengths associated with them (if branch lengths have been saved). The branch length values are based only on those trees containing the relevant bipartition. The second generated file has the suffix `.con` and includes two consensus trees. The first one has both the posterior probability of clades (as interior node labels) and the branch lengths (if they have been saved) in its description. A graphical representation of this tree can be generated in Rod Page's program TreeView. The second tree only contains the branch lengths and it can be imported into a wide range of phylogenetics programs. The third file generated by the `sumt` command is the `.trprobs` file, which contains the trees that were found during the MCMC search, sorted by posterior probability.

To quit MrBayes, simply type `quit`.

6. Analyze the SRY data

Now it is your turn. Switch your focus to the SRY data. If you wish, you can try to calculate an ML tree but the run will take too long (3-4 hours is a good guess) to complete so you may prefer to skip this step and jump directly to the Bayesian analysis. For the Bayesian analysis, you should be able to duplicate your analysis of the `primates.nex` analysis, except that you probably need to run the chain longer (a good guess is that you may need around 100,000 generations, which should take about 10 minutes) and use a more substantial burnin. Before you run the `SRY.DNA.nex` dataset in MrBayes, you need to make two changes to the file output produced by GCG, since MrBayes does not support the full NEXUS standard. Type `pico SRY.DNA.nex` to launch the pico text editor with the `SRY.DNA.nex` file (you can also use a GUI text editor for this task). Then find the format line close to the top of the sequence matrix and delete `equate="U=T"` and finally find the last line and change `endblock;` to `end;`. Finally save the file and exit pico. Now you should be able to execute the SRY data in MrBayes using **execute SRY.DNA.nex**.

Compare your results with those obtained last week. Do you get the same tree? How well supported are the groups? By looking at the `sump` table, you should get a good feeling for factors such as base composition bias and substitution rate profile. If you suspect that these parameters may vary over the sequence, you can run separate analyses on subsections of the alignment using the **exclude** command to temporarily exclude sites from the analysis, and then compare the parameter estimates.

7. Running PAUP and MrBayes in batch mode

When you run your own data, it is quite possible that your analyses are so demanding that you have to log out before they complete and log in later and check the results. What is the best way of doing this? First add a PAUP block at the end of your NEXUS data file where you give all the commands necessary to run the analysis. You may have to do this using a Unix text editor, such as pico. Type **pico primates.nex** to edit the primates.nex file, for instance (or use TextEdit). Finally, add the following line first in your 'paup' block, after the 'begin' statement: `set autoclose=yes warnreset=no warntsave=no increase=auto;` and this line: `quit;` last in the block. These statements will ensure that PAUP does not stop during an analysis to wait for confirmation from the user and that it quits automatically when the analysis is done. Then execute this data file once to check that the analysis is starting correctly. Interrupt this analysis using **<ctrl>C**. Finally start the real PAUP analysis by typing **paup batch.nex > log.txt &**, where 'batch.nex' is the name of your data (batch) file and 'log.txt' is the name of the file you want to use to store the screen output from PAUP (NB! This file will be overwritten if it already exists!). Do not forget the ampersand at the end; it ensures that Unix returns the control to you. If you forget it, type **<ctrl> C** and start the analysis again. The > sign is necessary and must have spaces before and after and precede 'log.txt' to ensure output redirection to 'log.txt' (output redirection is the printing of screen output to a file instead of to the terminal window). Once you have started your job you will get a job number. You can see how long your job has run and how much of the processor resources it consumes by giving the command **top**. You can see all of your current jobs by typing **ps**. To kill a job you need to find its PID number, for instance by using 'ps'. Then stop it by using **kill -9 <PID>**, where PID is the Process ID Number of the job. Never use the kill command unless you are absolutely certain that you are stopping one of the jobs you started; killing a system process may be fatal!

Running MrBayes in batch mode is essentially identical, except that the block at the end of the file should contain:

```
begin mrbayes;
  set autoclose=yes nowarn=yes;
  <your commands go here>
  quit;
end;
```

8. Homework Assignment

For your assignment, use a dataset of your own choice with at least 5 sequences in it and run either a ML analysis or a Bayesian MCMC analysis. Try to avoid too large data sets, about 20 sequences would be appropriate for a ML analysis and about 50 to 100 for a Bayesian MCMC analysis. Larger data sets are OK for your project but we do not want to make this homework assignment overly complicated. Describe your analysis (ML/Bayesian, chosen model, analysis settings, etc) in one paragraph. Present part of the result (a tree with support values, a sample of substitution model parameters, or something else that you find interesting) as a graph or table and discuss that result in one paragraph.