# Bayesian Inference of Species Trees from Multilocus Data using *BEAST

Joseph Heled, Alexei J Drummond and Walter Xie

July 26, 2011

## Introduction

We describe a full Bayesian framework for species tree estimation. We have attempted to combine the best aspects of previous methods to provide joint inference of a species tree topology, divergence times, population sizes, and gene trees from multiple genes sampled from multiple individuals across a set of closely related species. We have achieved this by extending BEAST to *BEAST (pronounced "star beast"), which is published below:

Joseph Heled and Alexei J. Drummond Bayesian Inference of Species Trees from Multilocus Data Mol. Biol. Evol. 2010 27: 570-580.

You will need the following software at your disposal:

- **BEAST** - this package contains the BEAST program, BEAUti, TreeAnnotator and other utility programs. This tutorial is written for BEAST v1.6.x, which is available for download from `http://beast.bio.ed.ac.uk/`.
- **Tracer** - this program is used to explore the output of BEAST (and other Bayesian MCMC programs). It graphically and quantitively summarizes the distributions of continuous parameters and provides diagnostic information. At the time of writing, the current version is v1.5. It is available for download from `http://beast.bio.ed.ac.uk/`.
- **FigTree** - this is an application for displaying and printing molecular phylogenies, in particular those obtained using BEAST. At the time of writing, the current version is v1.3.1. It is available for download from `http://tree.bio.ed.ac.uk/`.

## *BEAST

This tutorial will guide you through the analysis of three loci sampled from 26 individuals representing nine species of pocket gophers. This is a subset of previous published

data [?]. The objective of this tutorial is to estimate the species tree that is most probable given the multi-individual multi-locus sequence data. The species tree has 9 taxa, whereas each gene tree has 26 taxa. *BEAST will co-estimate three gene trees embedded in a shared species tree (see Heled and Drummond, 2010 for details).

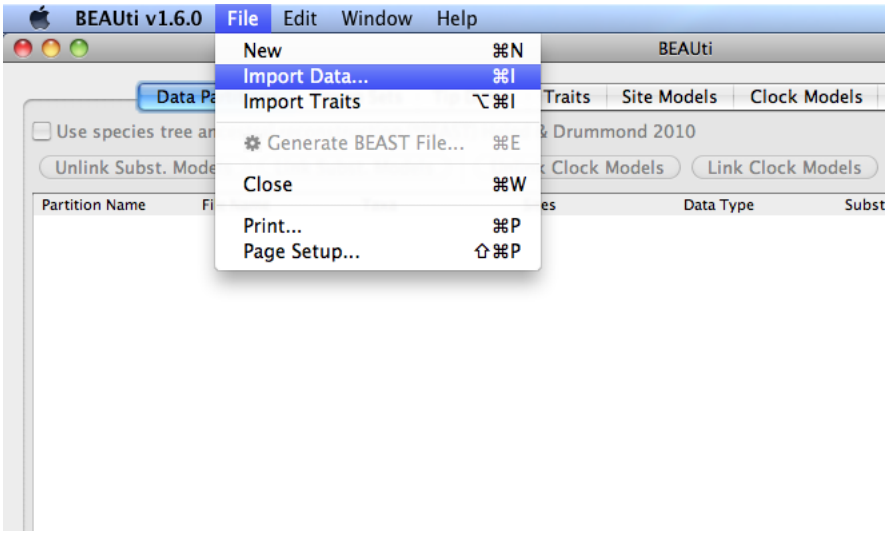
The first step will be to convert a NEXUS file with a DATA or CHARACTERS block into a BEAST XML input file. This is done using the program BEAUti (Bayesian Evolutionary Analysis Utility). This is a user-friendly program for setting the evolutionary model and options for the MCMC analysis. The second step is to actually run BEAST using the input file that contains the data, model and settings. The final step is to explore the output of BEAST in order to diagnose problems and to summarize the results.

## BEAUti

Run BEAUti by double clicking on its icon.

### Loading the NEXUS file

To load a NEXUS format alignment, simply select the `Import Data...` option from the File menu:

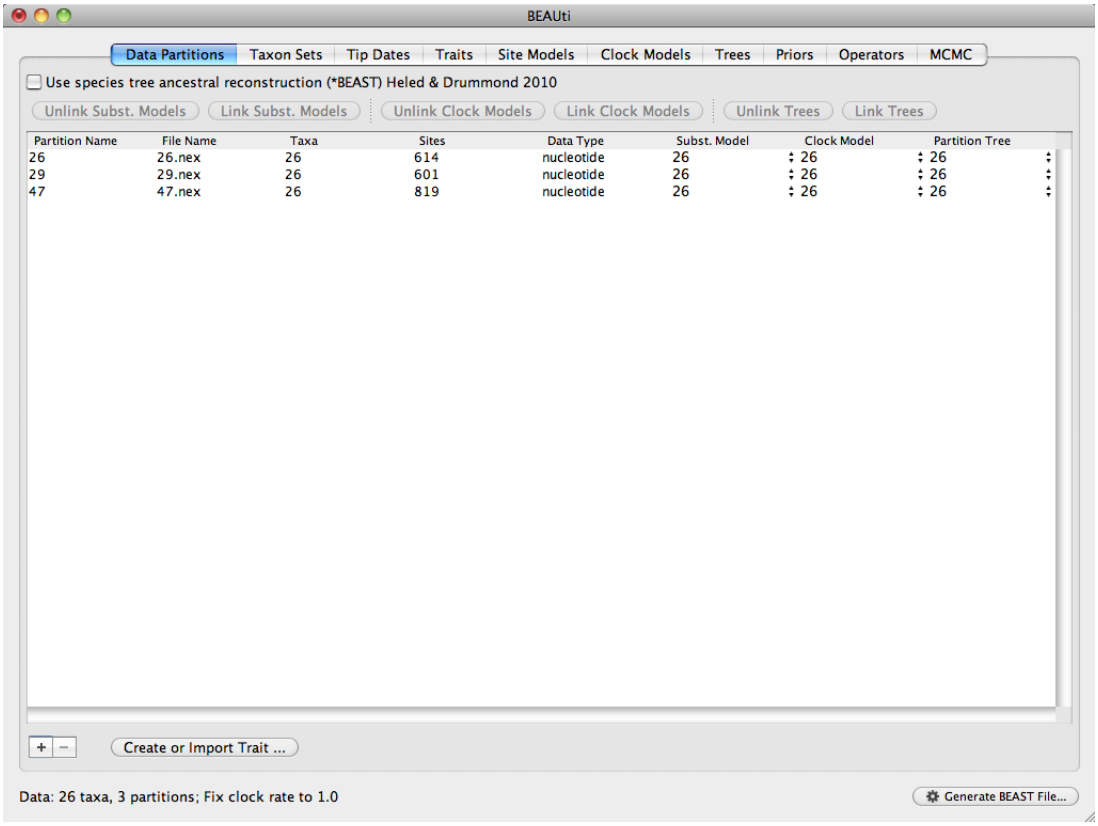



Select three files called `26.nex, 29.nex, 47.nex` by holding `shift` key. Each file contains an alignment of sequences of from an independent locus. The `26.nex` looks like this (content has been truncated):

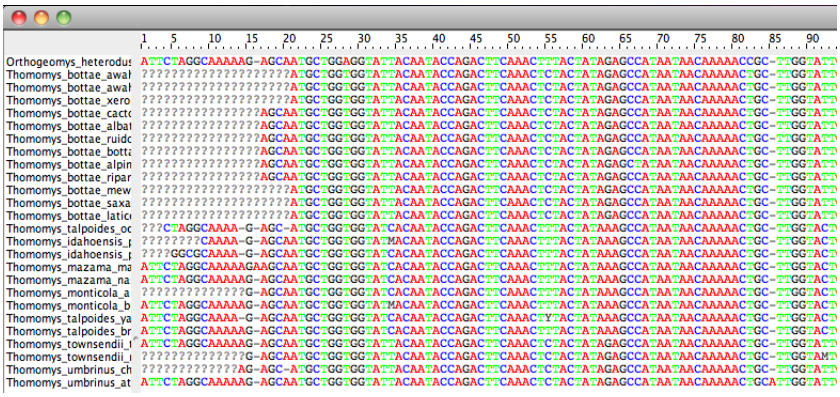
```
#NEXUS
[TBO26oLong]
BEGIN DATA;
DIMENSIONS  NTAX =26 NCHAR=614;
FORMAT DATATYPE = DNA GAP = - MISSING = ?;
```

```
MATRIX
Orthogeomys_heterodus          ATTCTAGGCAAAAAGAGCAATGC ...
Thomomys_bottae_awahnee_a      ??????????????????????ATGCTG ...
Thomomys_bottae_awahnee_b      ??????????????????????ATGCTG ...
Thomomys_bottae_xerophilus     ??????????????????????ATGCTG ...
Thomomys_bottae_cactophilus    ??????????????????AGCAATGCT ...

          ... ...

;
END;
```

Once loaded, the three partitions are displayed in the main panel:

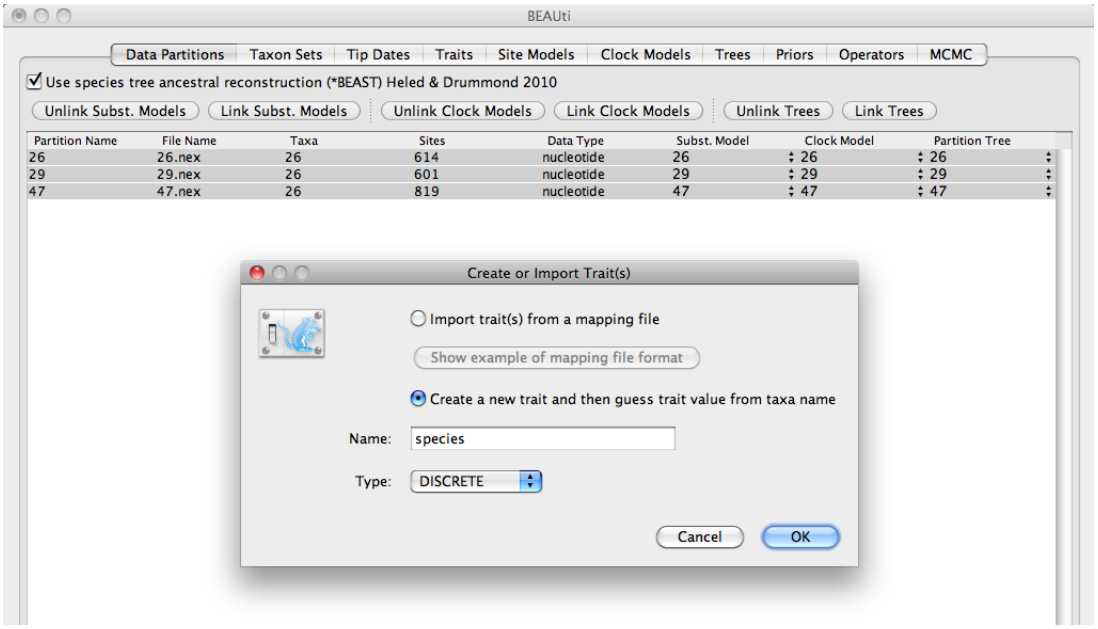



Double click any alignment (partition) to show its detail:



### Import trait(s) from a mapping file to fire *BEAST

To enable *BEAST in BEAST v1.6.x, simply click the check box labelled `Use species tree ancestral reconstruction (*BEAST) Heled & Drummond 2010` on the top of **Data Partitions** panel. Then, a **Create or Import Trait(s)** dialog will pop up.

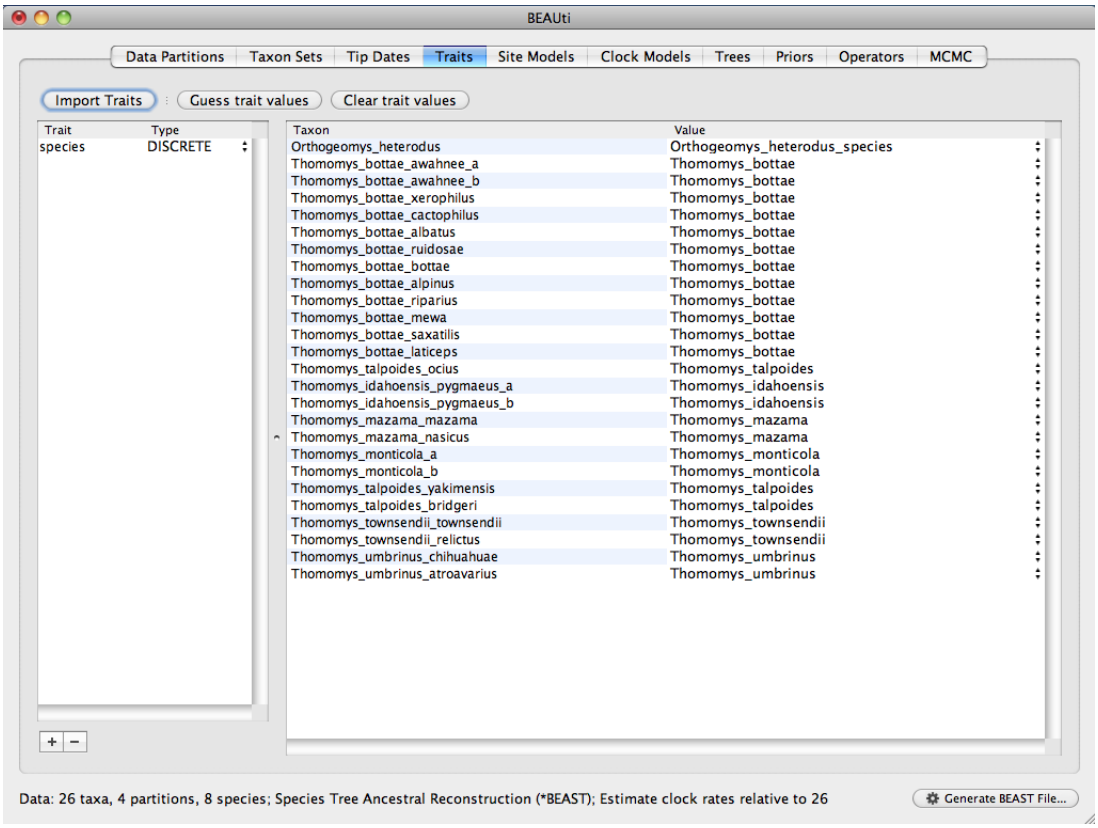



There are two options to be selected:

1. Import trait(s) from a mapping file;
2. Create a new trait and then guess trait value from taxa name `species`.

Choose the first option and click **OK** to load the mapping file, named `gopher_mapping.txt`.

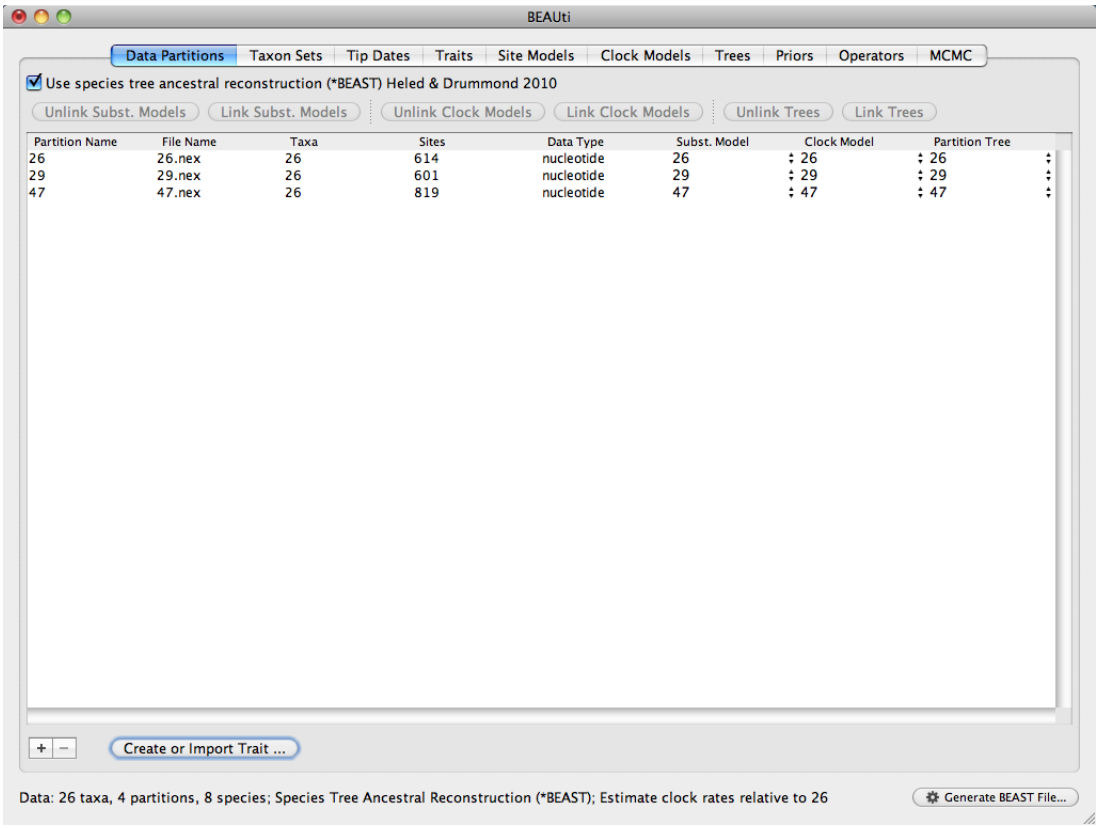
Once loaded, a message indicating the use of *BEAST will be displayed in the status at the bottom of the window, and a trait named `species` is created in the trait table in the **Traits** tab. Click it to show trait values.




A proper trait file is tab delimited. The first row is always `traits` followed by the keyword `species` in the second column and separated by tab. The rest of the rows map each individual taxon name to a species name: the taxon name in the first column and species name in the second column separated by tab. For example:

```
traits species
taxon1 speciesA
taxon2 speciesA
taxon3 speciesB
... ...
```

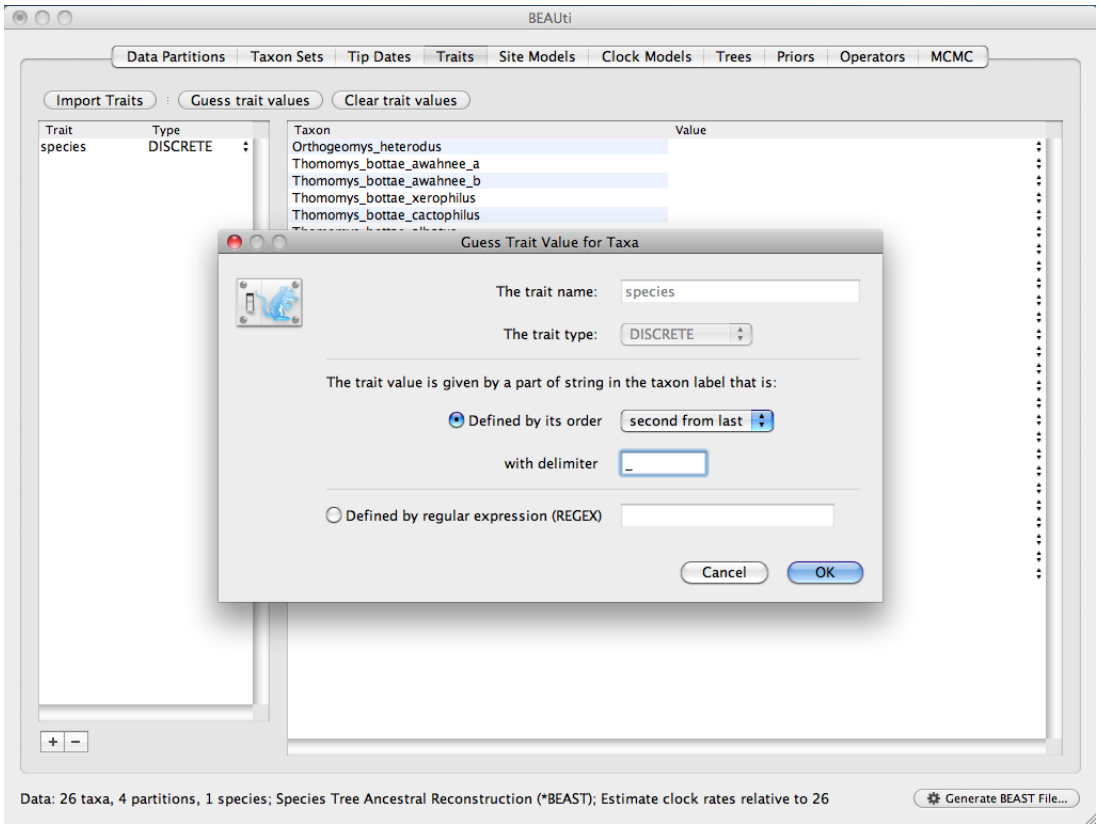
For multi-locus analyses, BEAST can link or unlink substitutions models across the loci by clicking buttons on the top of **Data Partitions** panel. The default of *BEAST is unlinking all models: substitution model, clock model, and tree models. Note that you should only unlink the tree model across data partitions that are actually genetically unlinked. For example, in most organisms all the mitochondrial genes are effectively linked due to a lack of recombination and they should be set up to use the same tree model in a *BEAST analysis.

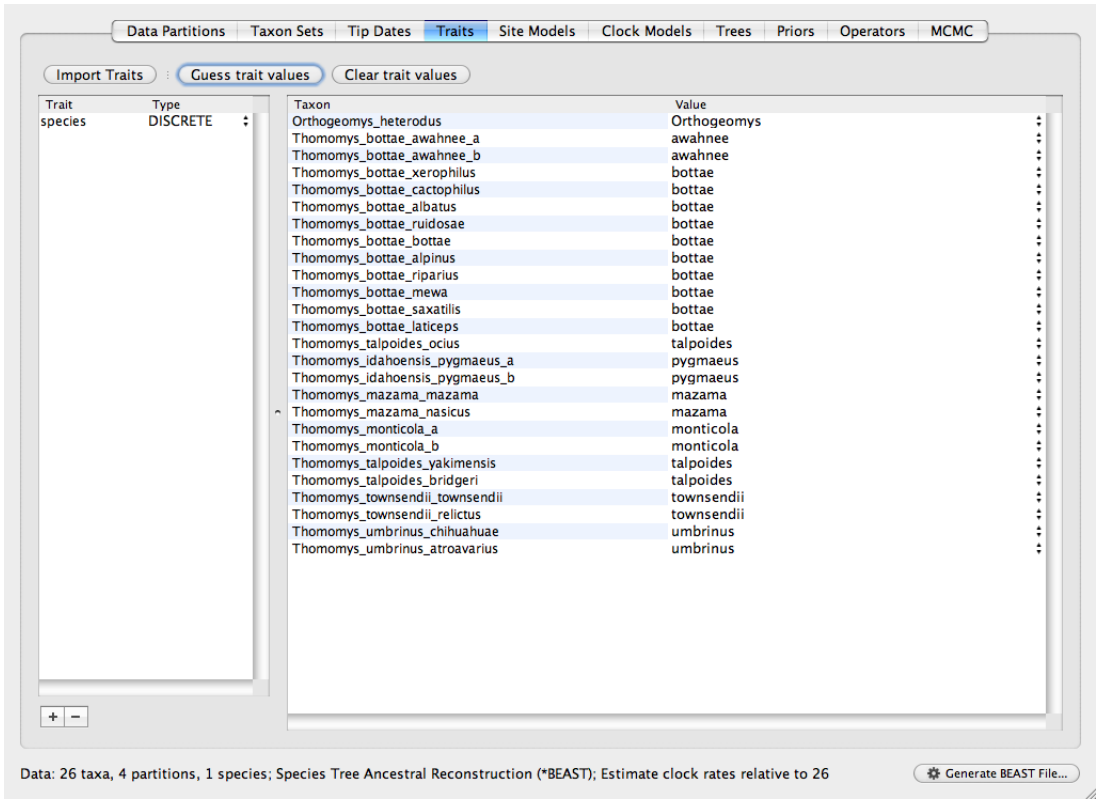



### Alternatively: Create a species trait from taxa names

The advantage of using the **Traits** panel is that we can extract species names from the taxa names if they already contain that information. Let's go to **Data Partitions** panel and unselect the check box labelled `Use species tree ancestral reconstruction (*BEAST) Heled & Drummond 2010`. As we can see in the status bar on the bottom, the analysis has been reverted to a standard BEAST analysis.

To enable *BEAST again, click the `Use species tree ancestral reconstruction (*BEAST) Heled & Drummond 2010` on the top of **Data Partitions** panel, and then choose the second option in **Create or Import Trait(s)** dialog this time. Click **OK** to continue, and then we will get to **Traits** panel and click on the **Guess trait values** at the top to pop out **Guess Trait Value for Taxa** dialog. Choose **second from last** in the drop list of **Defined by its order**, and input _ as separator. Click **OK**, and *BEAST is applied again.

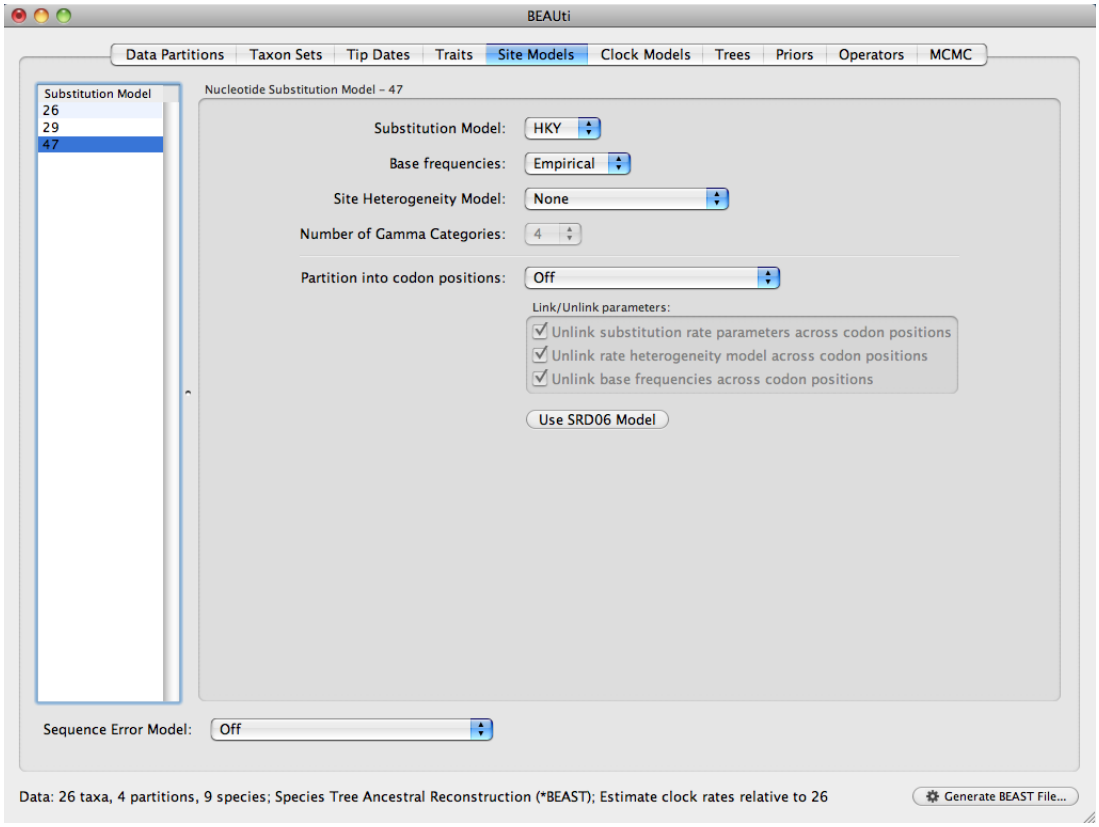

BEAUti
Data Partitions
Taxon Sets
Tip Dates
Traits
Site Models
Clock Models
Trees
Priors
Operators
MCMC
Import Traits
Guess trait values
Clear trait values
Trait
Type
species
DISCRETE
Taxon
Value
Orthogeomys_heterodus
Thomomys_bottae_awahnee_a
Thomomys_bottae_awahnee_b
Thomomys_bottae_xerophilus
Thomomys_bottae_cactophilus
Guess Trait Value for Taxa
The trait name:
species
The trait type:
DISCRETE
The trait value is given by a part of string in the taxon label that is:
Defined by its order
second from last
with delimiter
_
Defined by regular expression (REGEX)
Cancel
OK
Data: 26 taxa, 4 partitions, 1 species; Species Tree Ancestral Reconstruction (*BEAST); Estimate clock rates relative to 26
Generate BEAST File...


Data Partitions
Taxon Sets
Tip Dates
Traits
Site Models
Clock Models
Trees
Priors
Operators
MCMC
Import Traits
Guess trait values
Clear trait values
Trait
Type
species
DISCRETE
Taxon
Value
Orthogeomys_heterodus
Orthogeomys
Thomomys_bottae_awahnee_a
awahnee
Thomomys_bottae_awahnee_b
awahnee
Thomomys_bottae_xerophilus
bottae
Thomomys_bottae_cactophilus
bottae
Thomomys_bottae_albatus
bottae
Thomomys_bottae_ruidosae
bottae
Thomomys_bottae_bottae
bottae
Thomomys_bottae_alpinus
bottae
Thomomys_bottae_riparius
bottae
Thomomys_bottae_mewa
bottae
Thomomys_bottae_saxatilis
bottae
Thomomys_bottae_laticeps
bottae
Thomomys_talpoides_ocius
talpoides
Thomomys_idahoensis_pygmaeus_a
pygmaeus
Thomomys_idahoensis_pygmaeus_b
pygmaeus
Thomomys_mazama_mazama
mazama
Thomomys_mazama_nasicus
mazama
Thomomys_monticola_a
monticola
Thomomys_monticola_b
monticola
Thomomys_talpoides_yakimensis
talpoides
Thomomys_talpoides_bridgeri
talpoides
Thomomys_townsendii_townsendii
townsendii
Thomomys_townsendii_relictus
townsendii
Thomomys_umbrinus_chihuahuae
umbrinus
Thomomys_umbrinus_atroavarius
umbrinus
Data: 26 taxa, 4 partitions, 1 species; Species Tree Ancestral Reconstruction (*BEAST); Estimate clock rates relative to 26
Generate BEAST File...

### Setting the substitution model

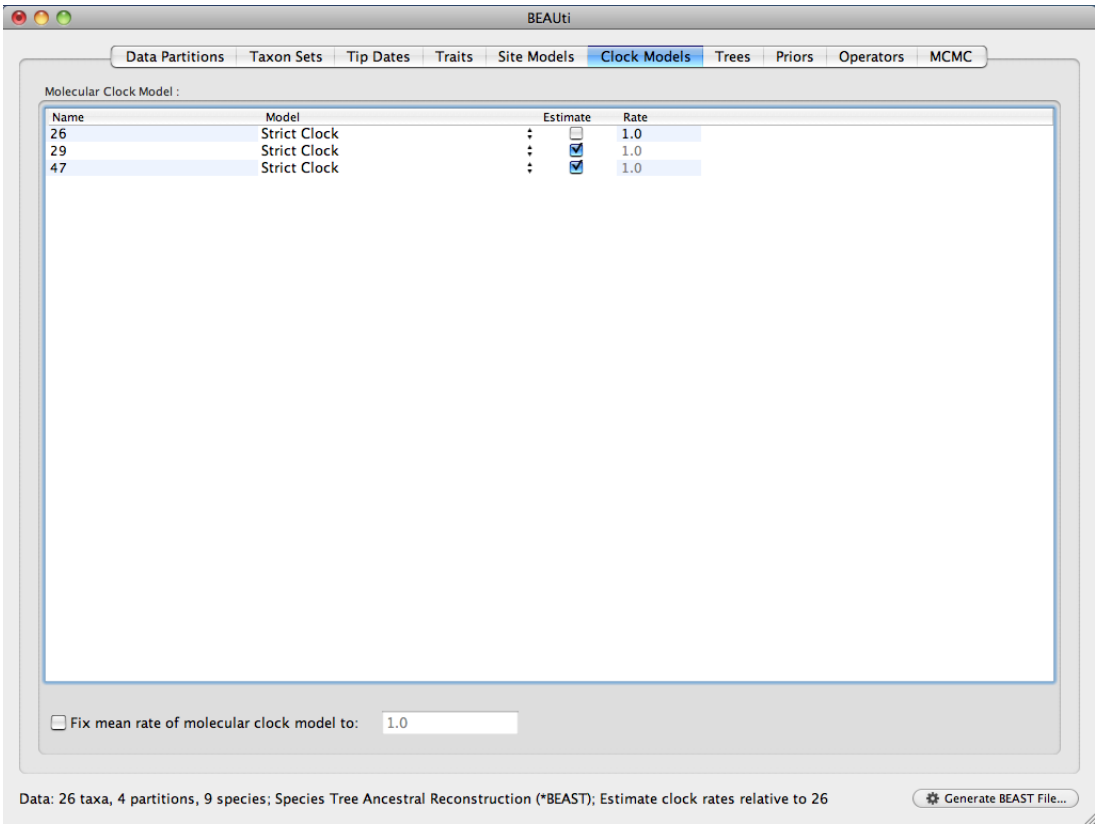
The next thing to do is to click on the **Site Models** tab at the top of the main window. This will reveal the evolutionary model settings for BEAST. Exactly which options appear depend on whether the data are nucleotides, or amino acids, or binary data, or general data. The settings that will appear after loading the data set will be the default values so we need to make some changes.

Most of the models should be familiar to you. For this analysis, we will select each substitution model listed on the left side in turn to make the following change: select **Empirical** for the **Base frequencies**.




### Setting the clock model

Second, click on the **Clock Models** tab at the top of the main window. In this analysis, we use the **Strict Clock** molecular clock model as default. Your model options should now look like this:

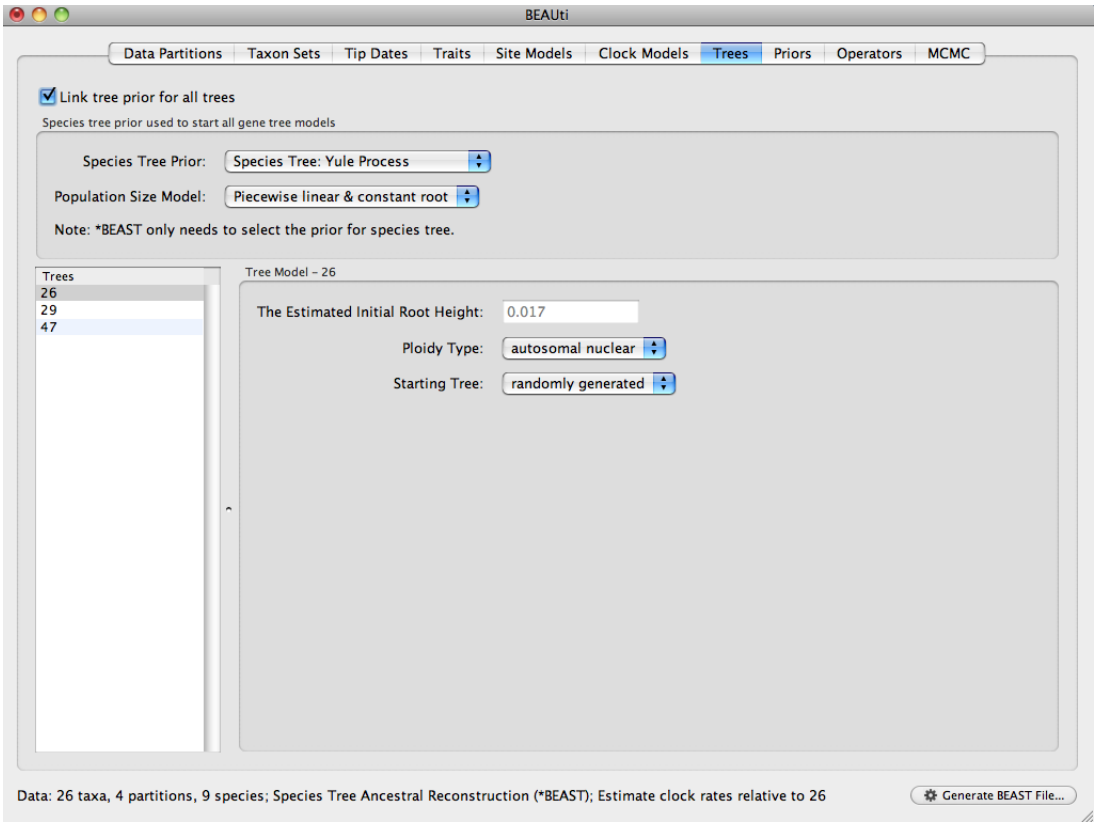



The **Estimate** check box is unchecked for the first clock model and checked for the rest clock models, because we wish to estimate the mutation rate of each subsequent locus relative to the first locus whose rate is fixed to 1.0.

### Trees

The **Trees** panel allows priors to be specified for each parameter in the model, which can be defined on the top of the panel. *BEAST has a different tree prior panel where users can only configure the species tree prior not gene tree priors (which are automatically specified by the multispecies coalescent). Currently, we have two species tree priors: **Yule Process** and **Birth-Death Process**; and three population size models: **Piecewise linear and constant root**, **Piecewise linear**, and **Piecewise constant**. In this analysis, we use the default options.

The bottom right panel is used to configure the corresponding starting trees. The **Ploidy Type** menu determines the type of sequence (mitochondrial, nuclear, X, Y). This matters since different modes of inheritance give rise to different effective population sizes. The **Starting Tree** menu provides three options, where the **user-specified** starting tree has to be loaded from the data file (e.g. NEXUS file) in advance. In this analysis, we simply use a random starting tree.




### Priors and Operators

The **Priors** panel allows priors to be specified for each parameter in the model. The **Operators** panel is used to configure technical settings that affect the efficiency of the MCMC program (see Notes for details). We leave these two panels unchanged in this analysis.

### Setting the MCMC options

The next tab, **MCMC**, provides more general settings to control the length of the MCMC and the file names.

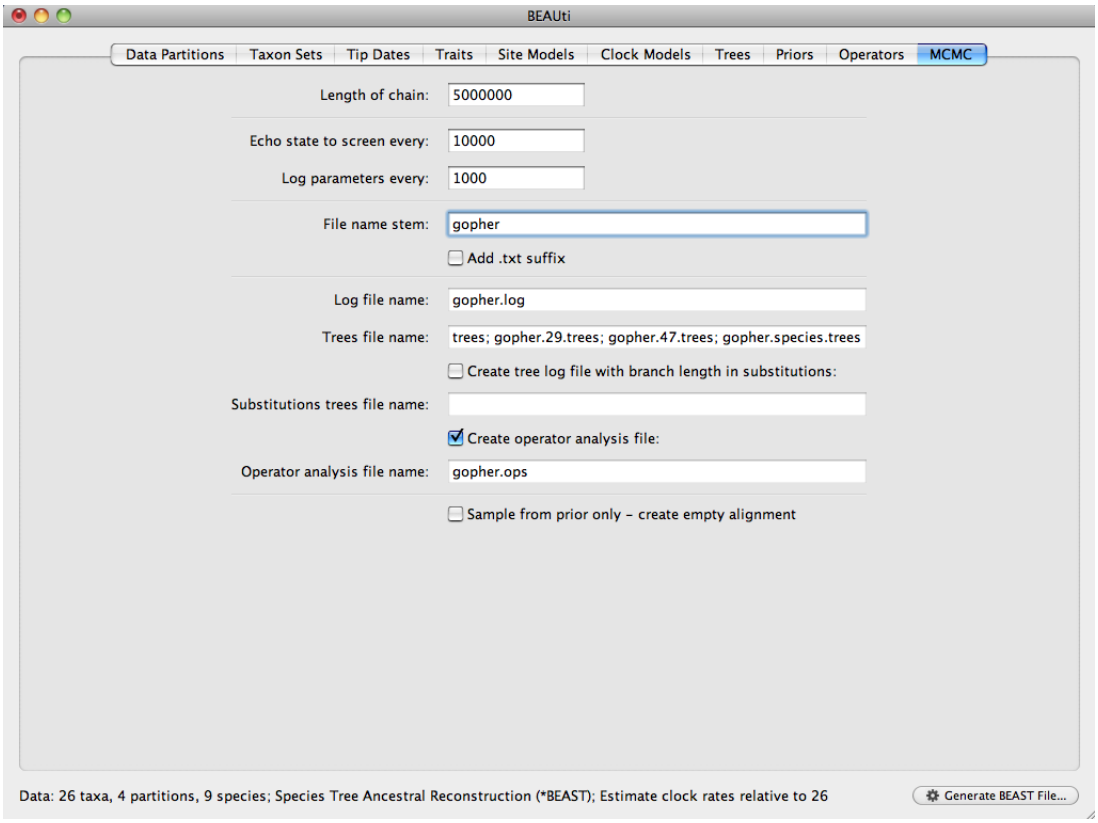
Firstly we have the **Length of chain**. This is the number of steps the MCMC will make in the chain before finishing. The appropriate length of the chain depends on the size of the data set, the complexity of the model and the accuracy of the answer required. The default value of 10,000,000 is entirely arbitrary and should be adjusted according to the size of your data set. For this data set let's initially set the chain length to 5,000,000 as this will run reasonably quickly on most modern computers (less than 20 minutes).

The next options specify how often the parameter values in the Markov chain should be displayed on the screen and recorded in the log file. The screen output is simply for monitoring the programs progress so can be set to any value (although if set too small,

the sheer quantity of information being displayed on the screen will actually slow the program down). For the log file, the value should be set relative to the total length of the chain. Sampling too often will result in very large files with little extra benefit in terms of the precision of the analysis. Sample too infrequently and the log file will not contain much information about the distributions of the parameters. You probably want to aim to store no more than 10,000 samples so this should be set to no less than chain length / 10,000.

For this exercise we will set the screen log to 10000 and the file log to 1000. The final two options give the file names of the log files for the sampled parameters and the trees. These will be set to a default based on the name of the imported NEXUS file.

If you would like to save the operator analysis into a file, you need to check **Create operator analysis file** which will generate a file with the suffix `.ops`.




- If you are using windows then we suggest you add the suffix `.txt` to both of these (so, `gopher.log.txt` and `gopher.trees.txt`) so that Windows recognizes these as text files.

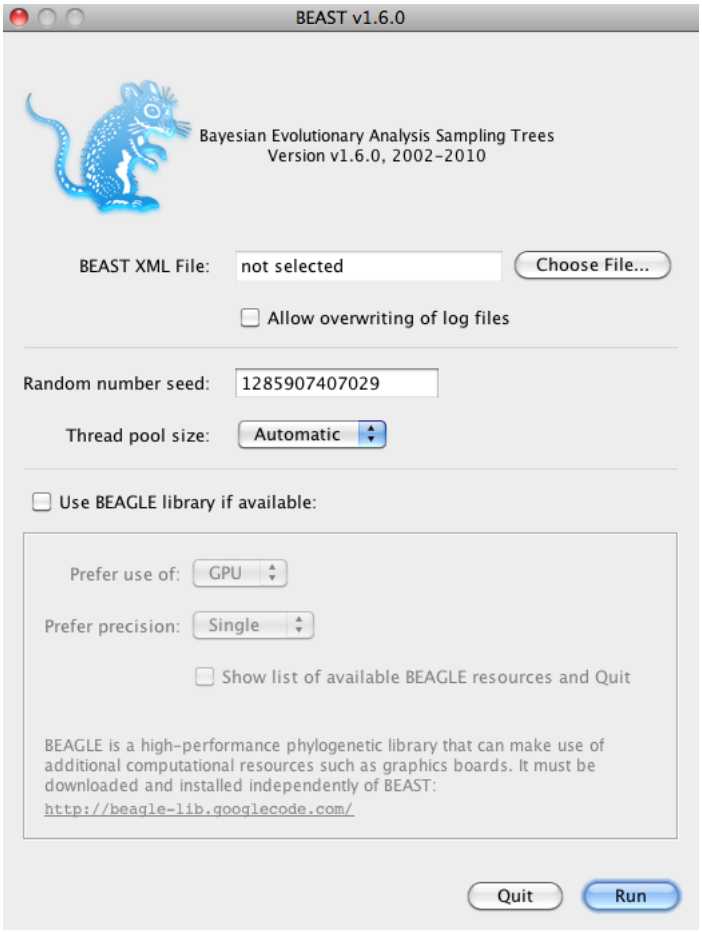
### Generating the BEAST XML file

We are now ready to create the BEAST XML file. To do this, either select the **Generate BEAST File...** option from the **File** menu or click the similarly labelled button

at the bottom of the window. Check the default priors setting and click **Continue**. Save the file with an appropriate name (we usually end the filename with `.xml`, i.e., `gopher.xml`). We are now ready to run the file through BEAST.

## Running BEAST

Now run BEAST and when it asks for an input file, provide your newly created XML file as input by click **Choose File ...**, and then click **Run**.




BEAST will then run until it has finished reporting information to the screen. The actual results files are saved to the disk in the same location as your input file. The output to the screen will look something like this:

```
                BEAST v1.6.0, 2002-2010
     Bayesian Evolutionary Analysis Sampling Trees
                 Designed and developed by
 Alexei J. Drummond, Andrew Rambaut and Marc A. Suchard

              Department of Computer Science
                  University of Auckland
```

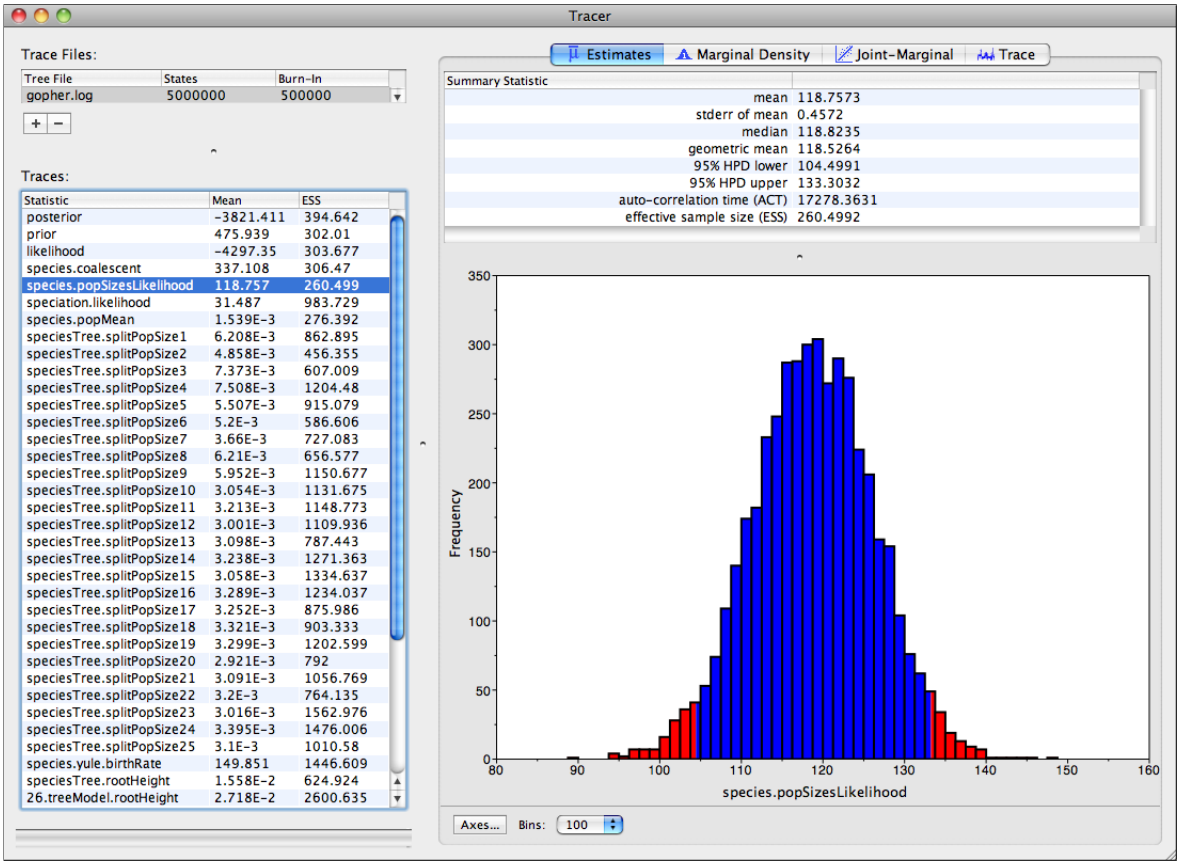
```
                  alexei@cs.auckland.ac.nz

              Institute of Evolutionary Biology
                   University of Edinburgh
                      a.rambaut@ed.ac.uk

               David Geffen School of Medicine
            University of California, Los Angeles
                      msuchard@ucla.edu

                 Downloads, Help & Resources:
                   http://beast.bio.ed.ac.uk

Source code distributed under the GNU Lesser General Public License:
              http://code.google.com/p/beast-mcmc

                       BEAST developers:
Alex Alekseyenko, Erik Bloomquist, Joseph Heled, Sebastian Hoehna,
Philippe Lemey, Wai Lok Sibon Li, Gerton Lunter, Sidney Markowitz,
Vladimir Minin, Michael Defoin Platel, Oliver Pybus, Chieh-Hsi Wu, Walter Xie

                           Thanks to:
     Roald Forsberg, Beth Shapiro and Korbinian Strimmer


Random number seed: 1285907407029

Parsing XML file: gopher.xml
  File encoding: MacRoman
Looking for plugins in /Users/local/EC/dxie004/Desktop/BEAST v1.6.0/*BEAST/plugins
Read alignment: alignment1
  Sequences = 26
      Sites = 614
   Datatype = nucleotide
Read alignment: alignment2
  Sequences = 26
      Sites = 601
   Datatype = nucleotide
Read alignment: alignment3
  Sequences = 26
      Sites = 819
   Datatype = nucleotide
Site patterns '26.patterns' created from positions 1-614 of alignment 'alignment1'
  pattern count = 144
Site patterns '29.patterns' created from positions 1-601 of alignment 'alignment2'
  pattern count = 71
Site patterns '47.patterns' created from positions 1-819 of alignment 'alignment3'
  pattern count = 153
Creating the tree model, '26.treeModel'
  initial tree topology = (((((((Thomomys_bottae_awahnee_b,Thomomys_bottae_laticeps),Thomomys_idahoensis_pygmaeus_b),(Thomomy
  tree height = 0.017
Creating the tree model, '29.treeModel'
  initial tree topology = ((((((((((((Thomomys_idahoensis_pygmaeus_b,Thomomys_monticola_a),Thomomys_idahoensis_pygmaeus_a),Th
  tree height = 0.016
Creating the tree model, '47.treeModel'
  initial tree topology = (((((((((Thomomys_bottae_awahnee_a,Thomomys_bottae_saxatilis),Thomomys_talpoides_yakimensis),((Thom
  tree height = 0.017
Using strict molecular clock model.
Using strict molecular clock model.
Using strict molecular clock model.
Creating state frequencies model: Using empirical frequencies from data = {0.40772, 0.20916, 0.19046, 0.19266}
Creating HKY substitution model. Initial kappa = 2.0
Creating site model.
Creating state frequencies model: Using empirical frequencies from data = {0.24545, 0.21384, 0.23701, 0.3037}
Creating HKY substitution model. Initial kappa = 2.0
```

```
Creating site model.
Creating state frequencies model: Using empirical frequencies from data = {0.2017, 0.21368, 0.21208, 0.37254}
Creating HKY substitution model. Initial kappa = 2.0
Creating site model.
TreeLikelihood(26.treeModel) using native nucleotide likelihood core
  Ignoring ambiguities in tree likelihood.
  With 144 unique site patterns.
Branch rate model used: strictClockBranchRates
TreeLikelihood(29.treeModel) using native nucleotide likelihood core
  Ignoring ambiguities in tree likelihood.
  With 71 unique site patterns.
Branch rate model used: strictClockBranchRates
TreeLikelihood(47.treeModel) using native nucleotide likelihood core
  Ignoring ambiguities in tree likelihood.
  With 153 unique site patterns.
Branch rate model used: strictClockBranchRates
Using Yule prior on tree
Likelihood is using -1 threads.
Creating the MCMC chain:
  chainLength=5000000
  autoOptimize=true
  autoOptimize delayed for 50000 steps
# BEAST v1.6.0, Build r3554
# Generated Fri Oct 01 17:30:48 NZDT 2010 [seed=1285907407029]
state Posterior    Prior        Likelihood   PopMean      26.rootHeight 29.rootHeight 47.rootHeight 26.clock.rate
29.clock.rate 47.clock.rate
0 -7327.1914   284.5565     -7611.7479   0.0170       1.7E-2       1.6E-2       1.7E-2       1.00000      1.00000      1.0000
10000 -4039.0459   398.7446     -4437.7905   0.0050       2.13417E-2   2.08796E-2   1.44625E-2   1.00000      0.68149      1.
20000 -3883.9551   445.5451     -4329.5002   0.0024       2.31188E-2   2.00516E-2   1.66494E-2   1.00000      0.88617      1.
30000 -3862.4974   449.2440     -4311.7414   0.0025       3.19515E-2   2.62574E-2   2.31006E-2   1.00000      0.72893      1.
40000 -3852.6398   449.6922     -4302.3320   0.0019       3.15963E-2   2.50161E-2   2.30687E-2   1.00000      0.83797      1.
50000 -3824.9321   471.8587     -4296.7907   0.0023       2.19453E-2   2.24944E-2   1.71097E-2   1.00000      0.89729      1.
60000 -3844.8398   454.9870     -4299.8269   0.0012       2.94943E-2   3.32895E-2   2.48704E-2   1.00000      0.74949      1.
70000 -3832.1382   461.6018     -4293.7400   0.0024       2.59893E-2   3.06152E-2   1.79898E-2   1.00000      0.87217      1.
80000 -3845.4199   471.8890     -4317.3089   0.0020       3.21837E-2   2.01324E-2   2.04069E-2   1.00000      0.85336      1.
90000 -3817.7446   489.1778     -4306.9224   0.0009       2.74968E-2   2.60995E-2   1.78024E-2   1.00000      0.86640      1.
100000 -3820.8950   476.0313     -4296.9263   0.0014       3.53535E-2   2.65514E-2   2.05705E-2   1.00000      0.73415      1

... ...

4990000 -3835.4728   484.6381     -4320.1110   0.0014       2.70561E-2   2.86596E-2   1.44919E-2   1.00000      0.75162
5000000 -3820.9055   489.0983     -4310.0037   0.0014       3.83597E-2   3.22498E-2   1.56354E-2   1.00000      0.83375

Operator analysis
Operator                                    Tuning   Count      Time     Time/Op  Pr(accept)  Performance suggestion
scale(26.kappa)                             0.344   1160       200      0.17     0.3207      good
scale(29.kappa)                             0.346   1133       161      0.14     0.3786      good
scale(47.kappa)                             0.437   1138       232      0.2      0.3805      good
scale(29.clock.rate)                        0.492   33065      4492     0.14     0.278       good
scale(47.clock.rate)                        0.57    33096      6467     0.2      0.2804      good
up:29.clock.rate 47.clock.rate species.yule.birthRate down:speciesTree species.popMean speciesTree.splitPopSize nodeHeights(2
Try setting scaleFactor to about 0.4813
subtreeSlide(26.treeModel)                  0.003   165032     16538    0.1      0.2607      good
Narrow Exchange(26.treeModel)                       165294     10898    0.07     0.2855      good
Wide Exchange(26.treeModel)                         32975      806      0.02     0.0213      good
wilsonBalding(26.treeModel)                         33122      1692     0.05     0.033       slightly low
scale(26.treeModel.rootHeight)              0.504   33333      3108     0.09     0.2723      good
uniform(nodeHeights(26.treeModel))                  331392     39147    0.12     0.5474      high
subtreeSlide(29.treeModel)                  0.003   166117     15469    0.09     0.2592      good
Narrow Exchange(29.treeModel)                       166240     10407    0.06     0.3106      good
Wide Exchange(29.treeModel)                         33214      643      0.02     0.0428      good
wilsonBalding(29.treeModel)                         33091      1541     0.05     0.0471      slightly low
scale(29.treeModel.rootHeight)              0.463   32870      2809     0.09     0.2929      good
uniform(nodeHeights(29.treeModel))                  331873     35779    0.11     0.5748      high
subtreeSlide(47.treeModel)                  0.003   165783     17558    0.11     0.225       good
```

```
Narrow Exchange(47.treeModel)                          165382    11719    0.07    0.2303    good
Wide Exchange(47.treeModel)                            33191     709      0.02    0.0128    good
wilsonBalding(47.treeModel)                            32745     1692     0.05    0.0177    slightly low
scale(47.treeModel.rootHeight)          0.591          33096     2660     0.08    0.2303    good
uniform(nodeHeights(47.treeModel))                     331242    42940    0.13    0.5344    high
up:down:nodeHeights(26.treeModel)       0.834          33125     4217     0.13    0.2202    slightly high
Try setting scaleFactor to about 0.8427
up:29.clock.rate down:nodeHeights(29.treeModel)  0.807  33350     3364     0.1     0.2195    slightly high
Try setting scaleFactor to about 0.8177
up:47.clock.rate down:nodeHeights(47.treeModel)  0.781  33332     4407     0.13    0.2304    slightly high
Try setting scaleFactor to about 0.7838
scale(species.popMean)                  0.525          55356     2978     0.05    0.2895    good
scale(species.yule.birthRate)           0.237          32988     1904     0.06    0.2911    good
scale(speciesTree.splitPopSize)         0.168          1036951   72638    0.07    0.2547    good
nodeReHeight(sptree,species)                           1038320   73042    0.07    0.3012    good none

12.4212 minutes
```

## Analyzing the results

Run the program called **Tracer** to analyze the output of BEAST. When the main window has opened, choose **Import Trace File...** from the **File** menu and select the file that BEAST has created called `gopher.log`. You should now see a window like the following:



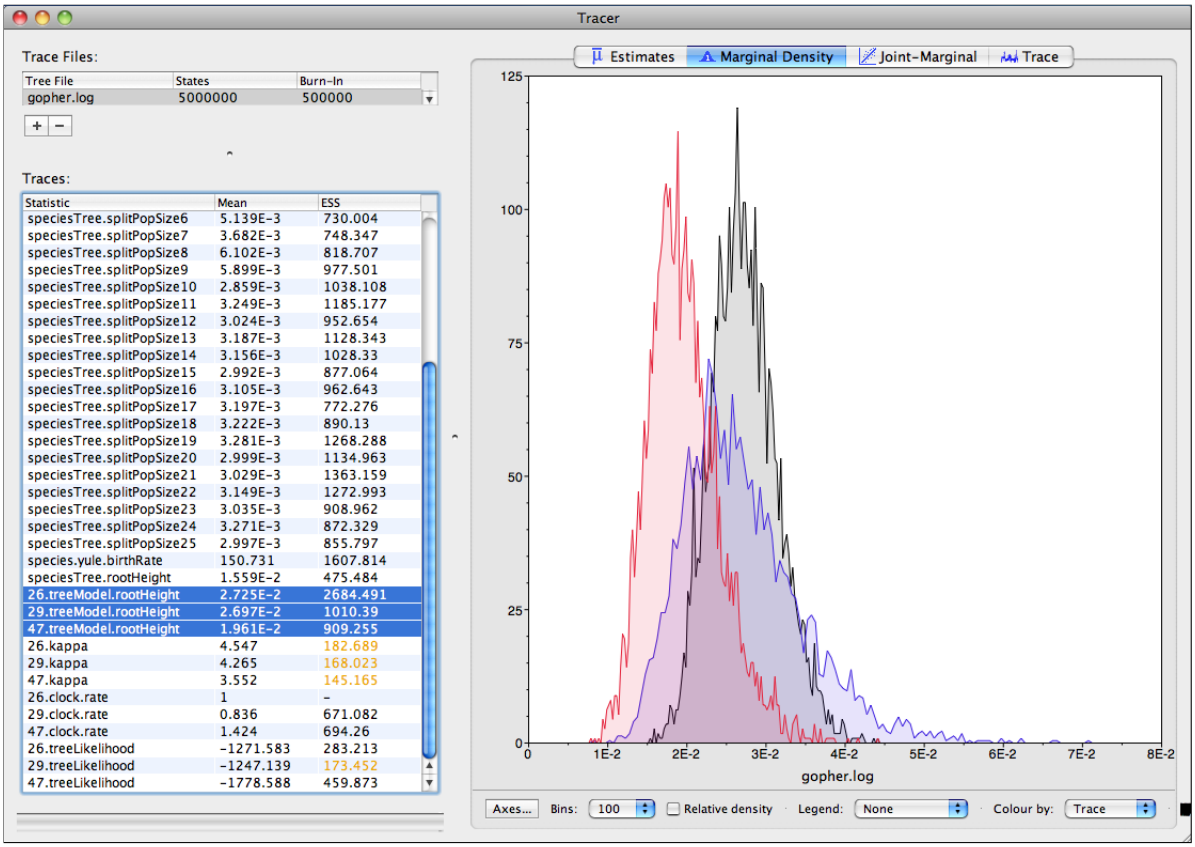

Remember that MCMC is a stochastic algorithm so the actual numbers will not be

exactly the same.

On the left hand side is a list of the different quantities that BEAST has logged. There are traces for the posterior (this is the log of the product of the tree likelihood and the prior probabilities), and the continuous parameters. Selecting a trace on the left brings up analyses for this trace on the right hand side depending on tab that is selected. When first opened, the 'posterior' trace is selected and various statistics of this trace are shown under the Estimates tab. In the top right of the window is a table of calculated statistics for the selected trace.

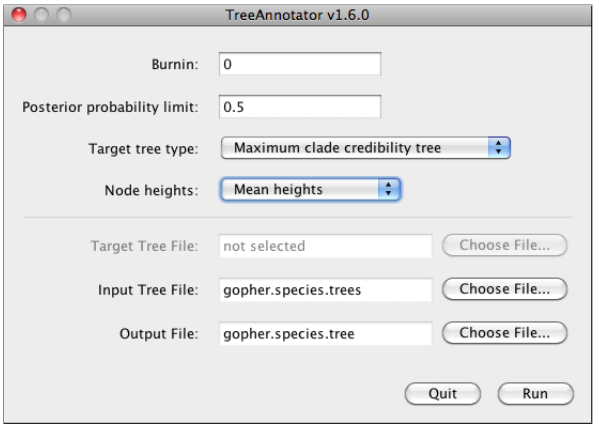
Tracer will plot a (marginal posterior) distribution for the selected parameter and also give you statistics such as the mean and median. The `95% HPD lower` or `upper` stands for *highest posterior density interval* and represents the most compact interval on the selected parameter that contains 95% of the posterior probability. It can be thought of as a Bayesian analog to a confidence interval.

Select the `treeModel.rootHeight` parameter and the next three (hold shift whilst selecting). This will show a display of the age of the root and the three gene trees. If you switch the tab at the top of the window to **Marginal Density** then you will get a plot of the marginal posterior densities of each of these date estimates overlayed:



## Obtaining an estimate of the phylogenetic tree

BEAST also produces a sample of plausible trees. These need to be summarized using the program **TreeAnnotator** (see Notes for details). This will take the set of trees and identify a single tree that best represents the posterior distribution. It will then annotate this selected tree topology with the mean ages of all the nodes as well as the 95% HPD interval of divergence times for each clade in the selected tree. It will also calculate the posterior clade probability for each node. Run the **TreeAnnotator** program and set it up to look like this:




The burnin is the number of trees to remove from the start of the sample. Unlike **Tracer** which specifies the number of steps as a burnin, in **TreeAnnotator** you need to specify the actual number of trees. For this run, we use the default setting.

The **Posterior probability limit** option specifies a limit such that if a node is found at less than this frequency in the sample of trees (i.e., has a posterior probability less than this limit), it will not be annotated. The default of 0.5 means that only nodes seen in the majority of trees will be annotated. Set this to zero to annotate all nodes.

For **Target tree type** you can either choose a specific tree from a file or ask TreeAnnotator to find a tree in your sample. The default option, **Maximum clade credibility tree**, finds the tree with the highest product of the posterior probability of all its nodes.

Choose **Mean heights** for node heights. This sets the heights (ages) of each node in the tree to the mean height across the entire sample of trees for that clade.

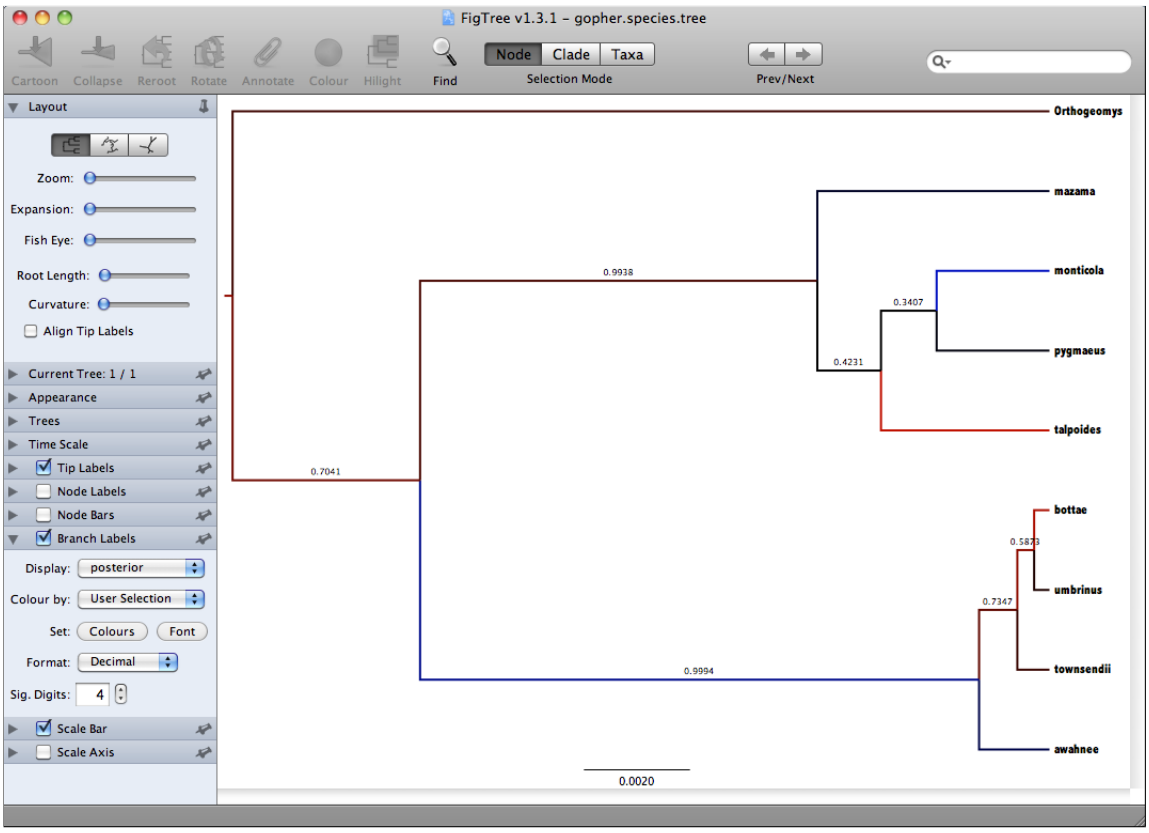
For the input file, select the trees file that BEAST created (by default this will be called `gopher.species.trees`) and select a file for the output (here we called it `gopher.species.tree`).

Now press `Run` and wait for the program to finish.

## Viewing the Tree

Finally, we can look at the tree in another program called **FigTree**. Run this program, and open the `gopher.species.tree` file by using the Open command in the File menu. The tree should appear. You can now try selecting some of the options in the control

panel on the left. Try selecting **Node Bars** to get node age error bars. Also turn on **Branch Labels** and select **posterior** to get it to display the posterior probability for each node. Under **Appearance** you can also tell FigTree to colour the branches by the rate. You should end up with something like this:




## Comparing your results to the prior

Using BEAUti, set up the same analysis but under the MCMC options, select the **Sample from prior only** option. This will allow you to visualize the full prior distribution in the absence of your sequence data. Summarize the trees from the full prior distribution and compare the summary to the posterior summary tree.