

# Phylogenetic Inference using RevBayes

## *Model Selection & Data Partitioning*

### Overview

RevBayes is a software program for inferring phylogenetic parameters in a Bayesian statistical framework.

[TAH comment: write an overview of RevBayes]

This tutorial demonstrates how to set up and perform an analysis that calculates Bayes factors to select among partitioning schemes... [TAH comment: add brief description of exercise]

### Probabilistic Graphical Models

[TAH comment: write description of graphical models; use figure for GM for GTR+G model]

### The Rev Language

[TAH comment: write about Rev language basics]

### Getting Started

This tutorial assumes that you have already downloaded, compiled, and installed RevBayes. We also recommend that—if you are working on a Unix machine—you put the `rb` binary in your path.

For the exercises outlined in this tutorial, we will use RevBayes interactively by typing commands in the command-line console. The format of this exercise uses **lavender blush shaded boxes** to delineate important steps. The various RevBayes commands and syntax are specified using **typewriter text**. And the specific commands that you should type (or copy/paste) into RevBayes are indicated by shaded box and prompt. For example, after opening the RevBayes program, you can load your data file:

```
RevBayes > D <- readCharacterData("data/conifer_atpB.nex")
```

For this command, type in the command and its options:

`D <- readCharacterData("data/conifer_atpB.nex")`. **DO NOT** type in “RevBayes >”, the prompt is simply included to replicate what you see on your screen.

This tutorial also includes hyperlinks: bibliographic citations are **burnt orange** and link to the full citation in the references, external URLs are **cerulean**, and internal references to figures and equations are **purple**.

The various exercises in this tutorial take you through the steps required to perform phylogenetic analyses of the example datasets. In addition, we have provided the output files for every exercise so you can verify your results. (Note that since the MCMC runs you perform will start from different random seeds, the output files resulting from your analyses *will not* be identical to the ones we provide you.)

- Download data and output files from: [\[TAH comment: add link to data files\]](#)

## 1 Model Selection & Partitioning using Bayes Factors

Variation in the evolutionary process across the sites of nucleotide sequence alignments is well established, and is an increasingly pervasive feature of datasets composed of gene regions sampled from multiple loci and/or different genomes. Inference of phylogeny from these data demands that we adequately model the underlying process heterogeneity; failure to do so can lead to biased estimates of phylogeny and other parameters (?). To accommodate process heterogeneity within and/or between various gene(omic) regions, we will evaluate the support for various partition schemes using Bayes factors to compare the marginal likelihoods of the candidate partition schemes.

Accounting for process heterogeneity involves adopting a ‘mixed-model’ approach, (?) in which the sequence alignment is first parsed into a number of partitions that are intended to capture plausible process heterogeneity within the data. The determination of the partitioning scheme is guided by biological considerations regarding the dataset at hand. For example, we might wish to evaluate possible variation in the evolutionary process within a single gene region (*e.g.*, between stem and loop regions of ribosomal sequences), or among gene regions in a concatenated alignment (*e.g.*, comprising multiple nuclear loci and/or gene regions sampled from different genomes). The choice of partitioning scheme is up to the investigator and many possible partitions might be considered for a typical dataset.

Next, a substitution model is specified for each predefined process partition (using a given model-selection criterion, such as Bayes factors, the hierarchical likelihood ratio test, or the Akaike information criterion). This results in a composite model, in which all sites are assumed to share a common tree topology, denoted  $\tau$ , and proportional branch lengths,  $\nu$ , but subsets of sites (‘data partitions’) are assumed to have independent substitution model parameters (*e.g.*, for the relative substitution rates,  $\theta_{ij}$ , stationary frequencies,  $\pi_i$ , degree of gamma-distributed among-site rate variation,  $\alpha$ , etc.). This composite model is referred to as a *mixed model*.

Finally, we perform a separate MCMC simulation to approximate the joint posterior probability density of the phylogeny and other parameters. Note that, in this approach, the mixed model is a fixed assumption of the inference (*i.e.*, the parameter estimates are conditioned on the specified mixed model), and the parameters for each process partition are independently estimated.

For most sequence alignments, several (possibly many) partition schemes of varying complexity are plausible *a priori*, which therefore requires a way to objectively identify the partition scheme that balances estimation bias and error variance associated with under- and over-parameterized mixed models, respectively. Increasingly, mixed-model selection is based on *Bayes factors* (*e.g.*, ?), which involves first calculating the marginal likelihood under each candidate partition scheme and then comparing the ratio of the marginal likelihoods for the set of candidate partition schemes (???). The analysis pipeline that we will use in this tutorial is depicted in Figure 1.

Given two models,  $M_0$  and  $M_1$ , the Bayes factor comparison assessing the relative plausibility of each model as an explanation of the data,  $BF(M_0, M_1)$ , is:

$$BF(M_0, M_1) = \frac{\text{posterior odds}}{\text{prior odds}}.$$

The posterior odds is the posterior probability of  $M_0$  given the data,  $\mathbf{X}$ , divided by the posterior odds of

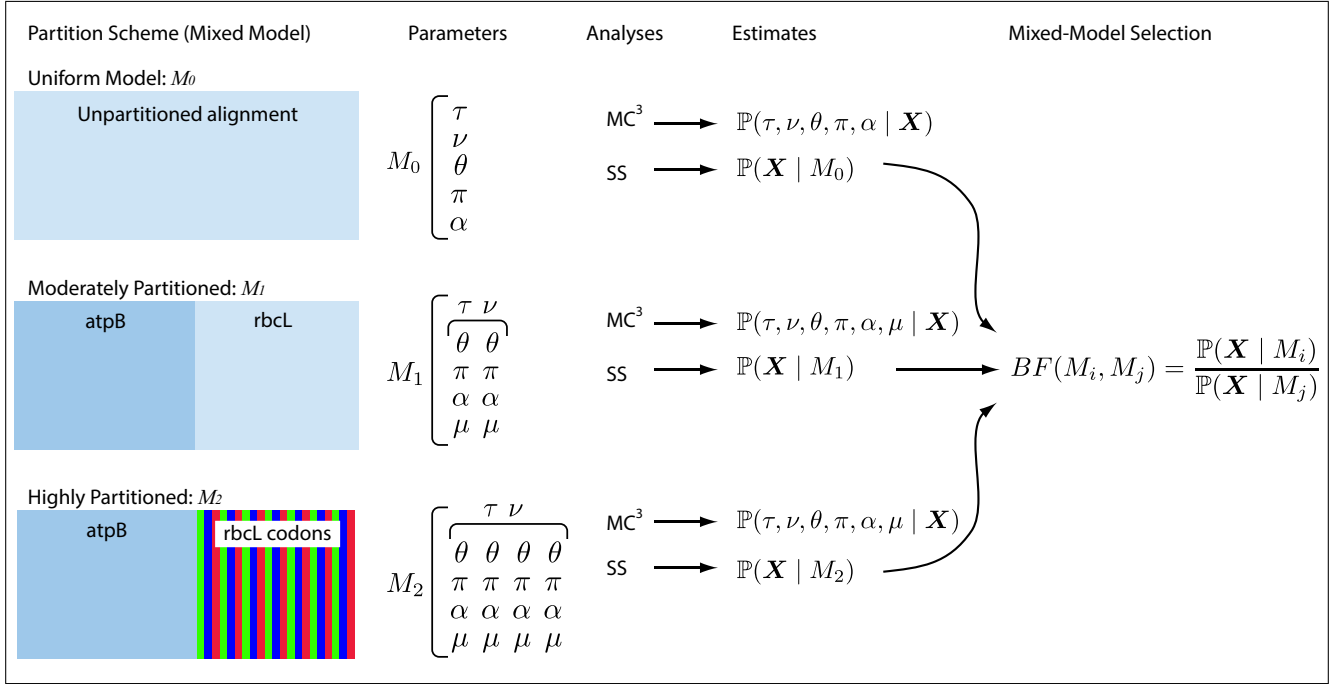


Figure 1: The analysis pipeline for Exercise 1. We will explore three partition schemes for the conifer dataset. The first model (the ‘uniform model’,  $M_0$ ) assumes that all sites evolved under a common GTR+ $\Gamma$  substitution model. The second model (the ‘moderately partitioned’ model,  $M_1$ ) invokes two data partitions corresponding to the two gene regions (atpB and rbcL), and assumes each subset of sites evolved under an independent GTR+ $\Gamma$  model, each with its own rate multiplier,  $\mu$ . The final mixed model (the ‘highly partitioned’ model,  $M_2$ ) invokes four data partitions—the first partition corresponds to the atpB gene region, and the remaining partitions correspond to the three codon positions of the rbcL gene region—and each data partition is assumed evolved under an independent GTR+ $\Gamma$  substitution model. Note that we assume that all sites share a common tree topology,  $\tau$ , and branch-length proportions,  $\nu$ , for each of the candidate partition schemes. We perform two separate sets of analyses for each mixed model—a Metropolis-coupled MCMC simulation to approximate the joint posterior probability density of the mixed-model parameters, and a ‘stepping-stone’ MCMC simulation to approximate the marginal likelihood for each mixed model. The resulting marginal-likelihood estimates are then evaluated using Bayes factors to assess the fit of the data to the three candidate mixed models.

$M_1$  given the data:

$$\text{posterior odds} = \frac{\mathbb{P}(M_0 \mid \mathbf{X})}{\mathbb{P}(M_1 \mid \mathbf{X})},$$

and the prior odds is the prior probability of  $M_0$  divided by the prior probability of  $M_1$ :

$$\text{prior odds} = \frac{\mathbb{P}(M_0)}{\mathbb{P}(M_1)}.$$

Thus, the Bayes factor measures the degree to which the data alter our belief regarding the support for  $M_0$  relative to  $M_1$  (?):

$$BF(M_0, M_1) = \frac{\mathbb{P}(M_0 \mid \mathbf{X}, \theta_0)}{\mathbb{P}(M_1 \mid \mathbf{X}, \theta_1)} \div \frac{\mathbb{P}(M_0)}{\mathbb{P}(M_1)}. \quad (1)$$

This, somewhat vague, definition does not lead to clear-cut identification of the “best” model. Instead, you must decide the degree of your belief in  $M_0$  relative to  $M_1$ . Despite the absence of any strict “rule-of-thumb”, you can refer to the scale (outlined by ?) for interpreting these measures (Table 1).

Table 1: The scale for interpreting Bayes factors by Harold ?.

$BF(M_0, M_1)$	Strength of evidence
$< 1 : 1$	Negative (supports $M_1$ )
$1 : 1$ to $3 : 1$	Barely worth mentioning
$3 : 1$ to $10 : 1$	Substantial
$10 : 1$ to $30 : 1$	Strong
$30 : 1$ to $100 : 1$	Very strong
$> 100 : 1$	Decisive

For a detailed description of Bayes factors see ?

Unfortunately, direct calculation of the posterior odds to prior odds ratio is unfeasible for most phylogenetic models. However, we can further define the posterior odds ratio as:

$$\frac{\mathbb{P}(M_0 | \mathbf{X})}{\mathbb{P}(M_1 | \mathbf{X})} = \frac{\mathbb{P}(M_0) \mathbb{P}(\mathbf{X} | M_0)}{\mathbb{P}(M_1) \mathbb{P}(\mathbf{X} | M_1)},$$

where  $\mathbb{P}(\mathbf{X} | M_i)$  is the *marginal likelihood* of the data marginalized over all parameters for  $M_i$ ; it is also referred to as the *model evidence* or *integrated likelihood*. More explicitly, the marginal likelihood is the probability of the set of observed data ( $\mathbf{X}$ ) under a given model ( $M_i$ ), while averaging over all possible values of the parameters of the model ( $\theta_i$ ) with respect to the prior density on  $\theta_i$

$$\mathbb{P}(\mathbf{X} | M_i) = \int \mathbb{P}(\mathbf{X} | \theta_i) \mathbb{P}(\theta_i) dt. \quad (2)$$

If you refer back to equation 1, you can see that, with very little algebra, the ratio of marginal likelihoods is equal to the Bayes factor:

$$BF(M_0, M_1) = \frac{\mathbb{P}(\mathbf{X} | M_0)}{\mathbb{P}(\mathbf{X} | M_1)} = \frac{\mathbb{P}(M_0 | \mathbf{X}, \theta_0)}{\mathbb{P}(M_1 | \mathbf{X}, \theta_1)} \div \frac{\mathbb{P}(M_0)}{\mathbb{P}(M_1)}. \quad (3)$$

Therefore, we can perform a Bayes factor comparison of two models by calculating the marginal likelihood for each one. Alas, exact solutions for calculating marginal likelihoods are not known for phylogenetic models (see equation 2), thus we must resort to numerical integration methods to estimate or approximate these values. In this exercise, we will estimate the marginal likelihood for each partition scheme using both the harmonic-mean (unreliable) and stepping-stone (preferable) estimators.

- Open the file **conifer\_atpB.nex** in your text editor. This file contains the sequences for the atpB gene sampled from 9 species (Box 1). The elements of the **DATA** block indicate the type of data, number of taxa, and length of the sequences.

Box 1: A fragment of the NEXUS file containing the atpB sequences for this exercise.

```
#NEXUS

Begin data;
  Dimensions ntax=9 nchar=1394;
  Format datatype=dna gap=-;
  Matrix
Ginkgo_biloba      TTATTGGTCCAGTACTGGATGTAGCTTTTCCCCGGGCAATATGCCTAATATTTACAATTCTTTG...
Araucaria_araucana -----GGTCCGGTACTGGATGTATCTTTTCTCCAGATGAAATGCCCTATATTTACAATTCTTTG...
```

```

Cedrus_deodara      TCATTGGCCCAGTACTGGA?GTCTCTTTTCCTCCAGGTAATATGCCTAATATTTACAATTCATTG...
Cupressus_arizonica -----GATGTATCTTTCCCTCCAGGTAGTATGCCTAGAATTTACAATTCCTTG...
Juniperus_communis -----
Pinus_densiflora    TCATTGGCCCAGTACTGGATGTCTCTTTTCCTCCAGGTAATATGCCTAATATTTACAATTCATTG...
Podocarpus_chinensis TCATCGGCCCTGTACTGGATGTATCTTTTCCTCCAGATGGTATGCCTTTTATTACAATTCCTTA...
Sciadopitys_verticillata TCATTGGTCCAGTACTAGATGTATCTTTCCCTCCAGGCAATATGCCTAGAATTTACAATTCCTTG...
Taxus_baccata       TTATCGGCCCCAGTACTAGATGTCTCTTTTCCTCCAGGTAATATGCCTAAAATTTACAATTCCTTA...
;
End;
```

- Open the Rev file, [TAH comment: Rev file name], in a text editor. This file contains all of the commands required to perform the necessary analyses to explore various partition schemes (unpartitioned, partitioned by gene region, and partitioned by gene region+codon position). The details of each command are described in adjacent comments, after a #; *e.g.*, # **this is a Rev comment**.

Typically, we would perform these analyses by simply sourcing this file in RevBayes. For the purposes of this exercise, however, we will walk through the different steps interactively in the command line.

## Launch RevBayes

Execute the RevBayes binary. If this program is in your path, then you can simply type in your Unix terminal:

- `> rb`

When you execute the program, you will see the program information, including the current version number and functions that will provide information about the program — `contributors()` and `license()`. Execute the `help()` function by typing:

```
RevBayes > help()
```

[TAH comment: will there be help information by this time?]

This displays a list of the different elements and commands available in RevBayes. The `help` command also provides more detailed information about each of these items.

For example, we can view the `help()` information about the `log()` function:

```
RevBayes > ?mcmc
```

[TAH comment: this is the only really good help file that exists]

The `mcmc()` function...

Additionally, RevBayes will print the correct usage of a function if it is executed without any arguments:

```
RevBayes > mcmc()
Error: Argument mismatch for call to function 'mcmc'(). Correct usage is:
MCMC function (Model model, VectorRbPointer<Monitor> monitors,
VectorRbPointer<Move> moves, String moveschedule = sequential|random|single)
```

## 1.1 An Unpartitioned Analysis

### *Load Data*

First load in the sequences using the **readCharacterData** function:

[TAH comment: It'd be nice if we could get the function indexing again...]

```
RevBayes > D <- readCharacterData("data/conifer_atpB.nex")
RevBayes > data_atpB <- D[1]
RevBayes > D <- readCharacterData("data/conifer_rbcL.nex")
RevBayes > data_rbcL <- D[1]
```

Concatenate the two data matrices using the **+** operator. This returns a single data matrix with both genes.

```
RevBayes > data <- data_atpB + data_rbcL
```

To report the current value of any variable, simply type the variable name and press enter. For the **data** matrix, this provides information about the alignment:

```
RevBayes > data
  Origination:                conifer_atpB.nex
  Number of taxa:              9
  Number of characters:        2659
  Number of included characters: 2659
  Datatype:                    DNA
```

[TAH comment: can we specify an outgroup?]

Next we will specify some useful variables based on our dataset. The variable **data** has *member functions* that we can use to retrieve information.

```
RevBayes > n_species <- data.ntaxa()
RevBayes > ns <- data.nchar()
```

```
RevBayes > n_sites <- ns[1]
RevBayes > names <- data.names()
RevBayes > n_branches <- 2 * n_species - 3
```

### The GTR Parameters

The first analysis in this exercise involves performing an analysis on our unpartitioned alignment. This corresponds to the assumption that the process that gave rise to our data was homogeneous across all sites of the alignment. Specifically, we will assume that both genes evolved under the same GTR+ $\Gamma$  model (Fig. 1).

[TAH comment: describe setting up model and some background on GTR...]

First, we will define and specify a prior on the exchangeability rates of the GTR model. We will use a flat Dirichlet prior on these six rates. To do this, we must begin by defining a constant node that specifies the vector of concentration values of the Dirichlet prior using the `v()` function:

```
RevBayes > er_prior <- v(1,1,1,1,1,1)
```

The constant variable `er_prior` defines the parameters of the Dirichlet prior distribution on the exchangeability rates. Thus, we can create a stochastic node for the exchangeability rates using the `dnDirichlet()` function, which takes a vector of values as an argument and the `~` operator. Together, these create a stochastic node named “`er`”:

```
RevBayes > er ~ dnDirichlet(er_prior)
```

The Dirichlet distribution assigns probability densities to grouped parameters: *e.g.*, those that measure proportions and must sum to 1. Above, we specified a 6-parameter Dirichlet prior on the relative rates of the GTR model, where the placement of each value specified represents one of the 6 relative rates: (1)  $A \rightleftharpoons C$ , (2)  $A \rightleftharpoons G$ , (3)  $A \rightleftharpoons T$ , (4)  $C \rightleftharpoons G$ , (5)  $C \rightleftharpoons T$ , (6)  $G \rightleftharpoons T$ . The input parameters of a Dirichlet distribution are called shape parameters or concentration parameters and a value is specified for each of the 6 GTR rates. The expectation and variance for each variable are related to the sum of the shape parameters. The prior above is a ‘flat’ or symmetric Dirichlet since all of the shape parameters are equal (1,1,1,1,1,1), thus we are specifying a model that allows for equal rates of change between nucleotides, such that the expected rate for each is equal to  $\frac{1}{6}$  (?). Figure 2a shows the probability density of each rate under this model. If we parameterized the Dirichlet distribution such that all of the parameters were equal to 100, this would also specify a prior with an expectation of equal exchangeability rates (Figure 2b). However, by increasing the shape parameters of the Dirichlet distribution, `er_prior <- v(100,100,100,100,100,100)`, would heavily restrict the MCMC from sampling sets of GTR rates in which the values were not equal or very nearly equal (*i.e.*, this is a very *informative* prior). We can consider a different Dirichlet parameterization if we had strong prior belief that transitions and transversions occurred at different rates. In this case, we could specify a more informative prior density: `er_prior <- v(4,8,4,4,8,4)`. Under this model, the expected rate for transversions would be  $\frac{4}{32}$  and the expected rate for transitions would equal  $\frac{8}{32}$ , and there

would be greater prior probability on sets of GTR rates that matched this configuration (Figure 2c). An alternative informative prior would be one where we assumed that each of the 6 GTR rates had a different value conforming to a  $\text{Dirichlet}(2,4,6,8,10,12)$ . This would lead to a different prior probability density for each rate parameter (Figure 2d). Without strong prior knowledge about the pattern of relative rates, however, we can better capture our statistical uncertainty with a vague prior on the GTR rates. Notably, all patterns of relative rates have the same probability under `er_prior <- v(1,1,1,1,1,1)`.

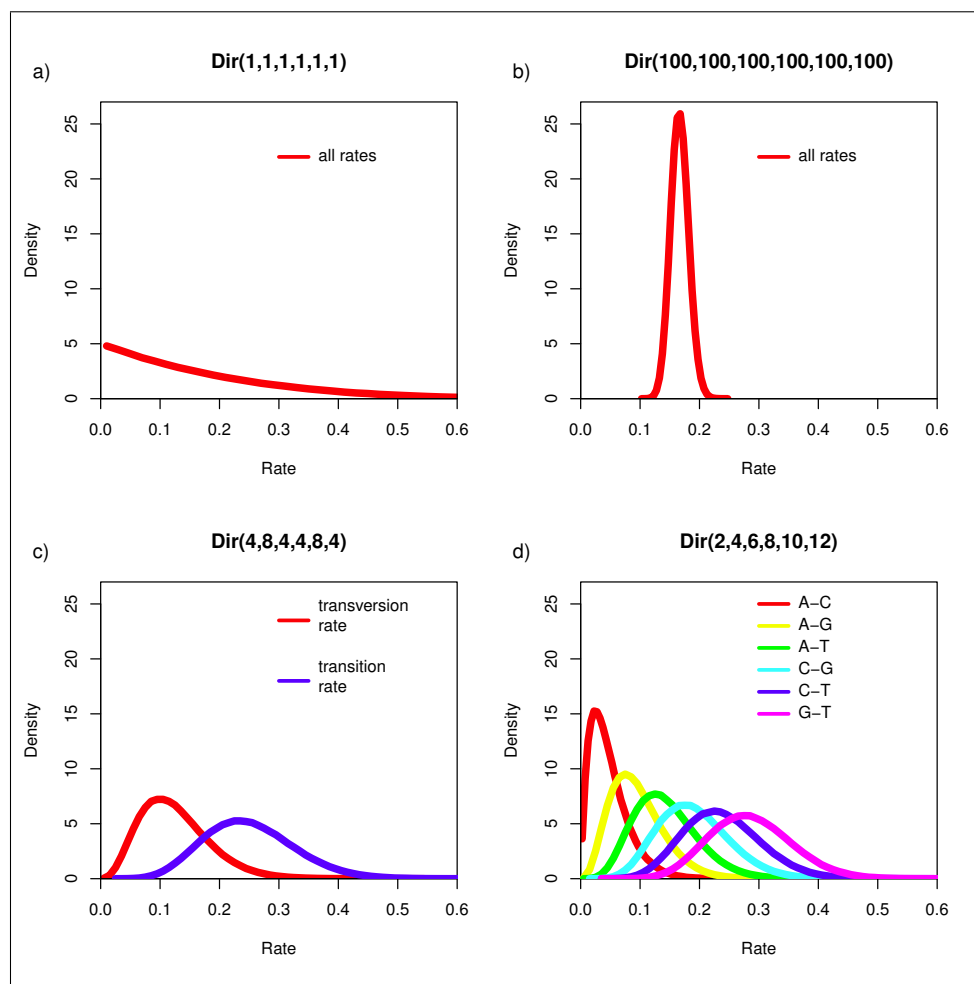


Figure 2: Four different examples of Dirichlet priors on exchangeability rates.

[TAH comment: implement the move for each stochastic parameter at the same time...]

For each stochastic node in our model, we must also specify a proposal mechanism if we wish to sample that value. The Dirichlet prior on our parameter `er` creates a *simplex* of values that sum to 1. In RevBayes, there are many different proposal mechanisms — called “moves” — and each move operates on a specific data type. [TAH comment: more here...]

Check the value type of the variable `er` using the `structure()` function:

```
RevBayes > structure(er)
```



```
_valueType      = Vector<RealPos>
_objectType     = Container
_value          = [ 1, 1, 1, 1, 1, 1 ]
```

All moves go in a vector...instantiate the first element of our vector of moves by setting the proposal on the exchangeability rates:

```
RevBayes > moves[1] <- mvSimplexElementScale(er, alpha=10, tune=true, weight=3)
```

We can use the same type of distribution as a prior on the 4 base frequencies ( $\pi_A, \pi_C, \pi_G, \pi_T$ ) since these parameters also represent proportions. Specify a flat Dirichlet prior density on the base frequencies:

```
RevBayes > sf_prior <- v(1,1,1,1)
RevBayes > sf ~ dnDirichlet(sf_prior)
```

And specify the simplex scale move:

```
RevBayes > moves[2] <- mvSimplexElementScale(sf, alpha=10, tune=true, weight=2)
```

[TAH comment: Now say something about the instantaneous rate matrix here and deterministic nodes]

```
RevBayes > Q := gtr(er,sf)
```

### *Gamma-Distributed Site Rates*

We will also assume that the substitution rates vary among sites according to a gamma distribution, which has two parameters: the shape parameter,  $\alpha$ , and the scale parameter,  $\beta$ . In order that we can interpret the branch lengths as the expected number of substitutions per site, this model assumes that the mean site rate is equal to 1. The mean of the gamma is equal to  $\alpha/\beta$ , so a mean-one gamma is specified by setting the two parameters to be equal,  $\alpha = \beta$ . Therefore, we need only consider the single shape parameter,  $\alpha$  (?). The degree of among-site substitution rate variation (ASRV) is inversely proportional to the value of the shape parameter—as the value of  $\alpha$ -shape parameter increases, the gamma distribution increasingly resembles a normal distribution with decreasing variance, which corresponds to decreasing levels of ASRV (Figure 3). If  $\alpha = 1$ , then the gamma distribution collapses to an exponential distribution with a rate parameter equal to  $\beta$ . By contrast, when the value of the  $\alpha$ -shape parameter is  $< 1$ , the gamma distribution assumes a concave distribution that places most of the prior density on low rates but allows some prior mass on sites with very high rates, which corresponds to high levels of ASRV (Figure 3).

Alternatively, we might not have good prior knowledge about the variance in site rates, thus we can place an uninformative, or diffuse prior on the shape parameter. For this analysis, we will use an exponential

distribution with a rate parameter, **shape\_prior**, equal to **0.05**. Under an exponential prior, we are placing non-zero probability on values of  $\alpha$  ranging from 0 to  $\infty$ . The rate parameter, often denoted  $\lambda$ , of an exponential distribution controls both the mean and variance of this prior such that the expected (or mean) value of  $\alpha$  is:

$$\mathbb{E}[\alpha] = \frac{1}{\lambda}.$$

Thus, if we set  $\lambda = 0.05$ , then  $\mathbb{E}[\alpha] = 20$ .

```
RevBayes > shape_prior <- 0.05
RevBayes > shape ~ dnExponential(shape_prior)
```

[TAH comment: write something about setting up and normalizing gamma rates...]

```
RevBayes > norm_gamma_rates := discretizeGamma( shape, 4 )
```

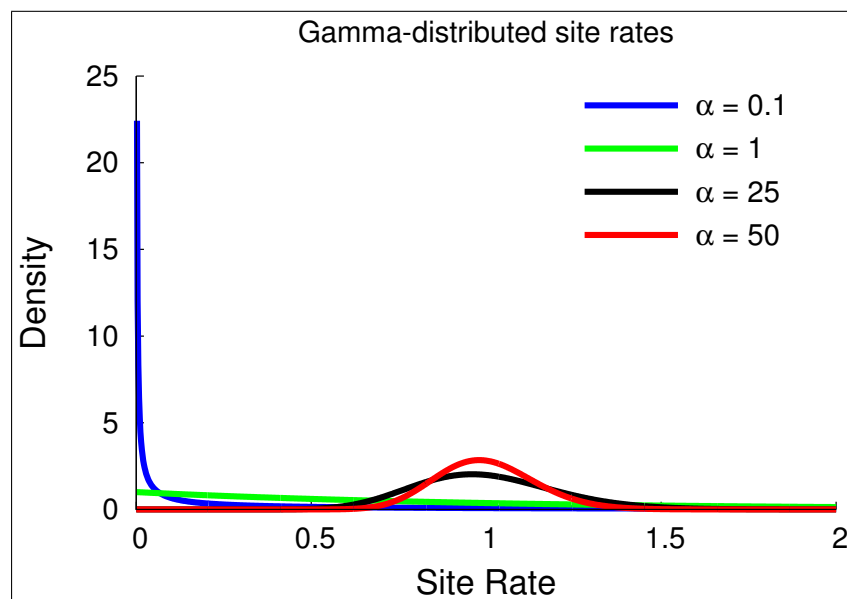


Figure 3: The probability density of mean-one gamma-distributed rates under different shape parameters.

The random variable that controls the rate variation is the stochastic node **shape**. This variable is a single, real positive value (**RealPos** datatype in RevBayes). Thus, we will apply a simple scale move to this parameter:

```
moves[3] <- mvScale(shape, lambda=1.0, tune=true, weight=1.0)
```

### *Tree Topology and Branch Lengths*

```
RevBayes > topology ~ uniformTopology(n_species, names)
```

```
RevBayes > moves[4] <- mvNNI(topology, weight=5.0)
```

```
RevBayes > for (i in 1:n_branches) {  
RevBayes +   br_lens[i] ~ dnExponential(1.0)  
RevBayes +   moves[i+4] <- mvScale(br_lens[i], lambda=1, tune=true, weight=1)  
RevBayes + }
```

```
RevBayes > phylogeny := treeAssembly(topology, br_lens)
```

### *Putting it All Together*

```
RevBayes > phyloSeq ~ phyloCTMC(tree=phylogeny, Q=Q, siteRates=  
  norm_gamma_rates, nSites=n_sites, type="DNA")
```

```
RevBayes > phyloSeq.clamp(data)
```

```
RevBayes > mymodel <- model(sf)
```

Now we have specified a simple, single-partition analysis—each parameter of the model will be estimated from every site in our alignment.

[TAH comment: list all in model]

```
RevBayes > mymodel
```

### Calculating the Marginal Likelihood

```
RevBayes > pow_p <- powerPosterior(mymodel, moves, "pow_p_uniform.out",  
  cats=50)
```

```
RevBayes > pow_p.burnin(generations=1000,tuningInterval=100)
```

```
RevBayes > pow_p.run(generations=1000)
```

```
RevBayes > ss <- steppingStoneSampler(file="pow_posterior_uniform.out",
  powerColumnName="power", likelihoodColumnName="likelihood")
```

Compute the marginal likelihood under stepping-stone sampling using the member function `marginal()` of the `ss` variable and record the value in table 2.

```
RevBayes > ss.marginal()
```

```
RevBayes > ps <- pathSampler(file="pow_posterior_uniform.out", powerColumnName
  ="power", likelihoodColumnName="likelihood")
```

```
RevBayes > ps.marginal()
```

## 1.2 Partitioning by Gene Region

The uniform model used in the previous section assumes that all sites in the alignment evolved under the same process described by a shared tree, branch length proportions, and parameters of the GTR+ $\Gamma$  substitution model. However, our alignment contains two distinct gene regions—atpB and rbcL—so we may wish to explore the possibility that the substitution process differs between these two gene regions. This requires that we first specify the data partitions corresponding to these two genes, then define an independent substitution model for each data partition.

### *Clear Workspace and Reload Data*

[TAH comment: this time reading from 2 different files]

```
RevBayes > clear()
```

```
RevBayes > filenames <- v("data/conifer_atpb.nex", "data/conifer_rbcL.
  nex")
```

```
RevBayes > n_parts <- filenames.size()
```

```
RevBayes > for (i in 1:n_parts){
RevBayes +   data[i] <- readCharacterData(filenames[i])[1]
RevBayes + }
```

```
RevBayes > n_species <- data[1].ntaxa()
RevBayes > names <- data[1].names()
RevBayes > n_branches <- 2 * n_species - 3
```

### *Specify the Parameters by Looping Over Partitions*

```
mv_it <- 0
for (i in 1:n_parts){
  ## index i=1 : atpB gene
  ## index i=2 : rbcL gene

  # Exchangeability rates #
  er_prior[i] <- v(1,1,1,1,1,1)
  er[i] ~ dnDirichlet(er_prior[i])
  moves[mv_it++] <- mvSimplexElementScale(er[i], alpha=10, tune=true, weight=3)

  # Stationary frequencies #
  sf_prior[i] <- v(1,1,1,1)
  sf[i] ~ dnDirichlet(sf_prior[i])
  moves[mv_it++] <- mvSimplexElementScale(sf[i], alpha=10, tune=true, weight=2)

  # GTR Rate Matrix (deterministic node) #
  Q[i] := gtr(er[i],sf[i])

  # Gamma-dist site rates #
  shape_prior[i] <- 0.05
  shape[i] ~ dnExponential( shape_prior[i] )
  norm_gamma_rates[i] := discretizeGamma( shape[i], 4 )
  moves[mv_it++] <- mvScale(shape[i], lambda=1.0, tune=true, weight=1.0)
}
```

### *Uniform Topology and Branch Lengths*

```
# Uniform Topology #
topology ~ uniformTopology(n_species, names)
moves[mv_it++] <- mvNNI(topology, weight=1.0)
```

```
# Branch lengths #
for (i in 1:n_branches) {
  br_lens[i] ~ exponential(1.0)
  moves[mv_it++] <- mvScale(br_lens[i], lambda=1, tune=true, weight=1)
}

# Create the phylogeny by combining topology & branch lengths #
phylogeny := treeAssembly(topology, br_lens)
```

### *Putting it All Together*

```
for (i in 1:n_parts){
  phyloSeq[i] ~ phyloCTMC(tree=phylogeny, Q=Q[i], siteRates=
    norm_gamma_rates[i], nSites=data[i].nchar()[1], type="DNA")
  phyloSeq[i].clamp(data[i])
}
```

```
RevBayes > mymodel <- model(topology)
```

### Calculating the Marginal Likelihood

```
RevBayes > pow_p <- powerPosterior(mymodel, moves, file="
  pow_posterior_twogene.out", cats=50)
RevBayes > pow_p.burnin(generations=1000, tuningInterval=100)
RevBayes > pow_p.run(generations=1000)
```

```
RevBayes > ss <- steppingStoneSampler(file="pow_posterior_twogene.out",
  powerColumnName="power", likelihoodColumnName="likelihood")
RevBayes > ss.marginal()
```

```
RevBayes > ps <- pathSampler(file="pow_posterior_twogene.out",
  powerColumnName="power", likelihoodColumnName="likelihood")
RevBayes > ps.marginal()
```

## 1.3 Partitioning by Codon Position and by Gene

Because of the genetic code, we often find that different positions within a codon (first, second, and third) evolve at different rates. Thus, using our knowledge of biological data, we can devise a third approach that

further partitions our alignment. For this exercise, we will partition sites within the rbcL gene by codon position.

### *Clear Workspace and Reload Data*

```
RevBayes > clear()
RevBayes > data[1] <- readCharacterData("data/conifer_atpB.nex")[1]
RevBayes > data_rbcL <- readCharacterData("data/conifer_rbcL.nex")[1]
```

### *Specify Data Matrices for Each Codon Position*

```
RevBayes > n_sites_rbcL <- data_rbcL.nchar()[1]
RevBayes > vec_1pos <- seq(1, n_sites_rbcL, 3)
RevBayes > vec_2pos <- seq(2, n_sites_rbcL, 3)
RevBayes > vec_3pos <- seq(3, n_sites_rbcL, 3)
```

[TAH comment: assign the rbcL data to the vector of data matrices]

```
RevBayes > data[2] <- data_rbcL # codon position 1
RevBayes > data[3] <- data_rbcL # codon position 2
RevBayes > data[4] <- data_rbcL # codon position 3
```

```
RevBayes > for(i in vec_1pos){
RevBayes +   data[3].excludeCharacter(i)
RevBayes +   data[4].excludeCharacter(i)
RevBayes + }
```

```
RevBayes > for(i in vec_2pos){
RevBayes +   data[2].excludeCharacter(i)
RevBayes +   data[4].excludeCharacter(i)
RevBayes + }
```

```
RevBayes > for(i in vec_3pos){
RevBayes +   data[2].excludeCharacter(i)
RevBayes +   data[3].excludeCharacter(i)
RevBayes + }
```

[TAH comment: create a vector of dataset sizes]

```
RevBayes > n_sites[1] <- data[1].nchar()[1]
RevBayes > n_sites[2] <- vec_1pos.size()
RevBayes > n_sites[3] <- vec_2pos.size()
RevBayes > n_sites[4] <- vec_3pos.size()
```

```
RevBayes > n_parts <- data.size()
```

```
RevBayes > n_species <- data[1].ntaxa()
RevBayes > names <- data[1].names()
RevBayes > n_branches <- 2 * n_species - 3
```

### *Specify the Parameters by Looping Over Partitions*

```
mv_it <- 0
for (i in 1:n_parts){
  ## index i=1 : atpB gene
  ## index i=2 : rbcL gene position 1
  ## index i=3 : rbcL gene position 2
  ## index i=4 : rbcL gene position 3

  # Exchangeability rates #
  er_prior[i] <- v(1,1,1,1,1,1)
  er[i] ~ dnDirichlet(er_prior[i])
  moves[mv_it++] <- mvSimplexElementScale(er[i], alpha=10, tune=true, weight=3)

  # Stationary frequencies #
  sf_prior[i] <- v(1,1,1,1)
  sf[i] ~ dnDirichlet(sf_prior[i])
  moves[mv_it++] <- mvSimplexElementScale(sf[i], alpha=10, tune=true, weight=2)

  # GTR Rate Matrix (deterministic node) #
  Q[i] := gtr(er[i],sf[i])

  # Gamma-dist site rates #
  shape_prior[i] <- 0.05
  shape[i] ~ dnExponential( shape_prior[i] )
  norm_gamma_rates[i] := discretizeGamma( shape[i], 4 )
  moves[mv_it++] <- mvScale(shape[i], lambda=1.0, tune=true, weight=1.0)
}
```

### *Uniform Topology and Branch Lengths*



```
# Uniform Topology #
topology ~ uniformTopology(n_species, names)
moves[mv_it++] <- mvNNI(topology, weight=1.0)

# Branch lengths #
for (i in 1:n_branches) {
  br_lens[i] ~ exponential(1.0)
  moves[mv_it++] <- mvScale(br_lens[i], lambda=1, tune=true, weight=1)
}

# Create the phylogeny by combining topology & branch lengths #
phylogeny := treeAssembly(topology, br_lens)
```

### *Putting it All Together*

```
for (i in 1:n_parts){
  phyloSeq[i] ~ phyloCTMC(tree=phylogeny, Q=Q[i], siteRates=
    norm_gamma_rates[i], nSites=data[i].nchar()[1], type="DNA")
  phyloSeq[i].clamp(data[i])
}
```

```
RevBayes > mymodel <- model(topology)
```

### Calculating the Marginal Likelihood

```
RevBayes > pow_p <- powerPosterior(mymodel, moves, file="
  pow_posterior_genecodon.out", cats=50)
RevBayes > pow_p.burnin(generations=1000, tuningInterval=100)
RevBayes > pow_p.run(generations=1000)
```

```
RevBayes > ss <- steppingStoneSampler(file="pow_posterior_genecodon.out",
  powerColumnName="power", likelihoodColumnName="likelihood")
RevBayes > ss.marginal()
```

```
RevBayes > ps <- pathSampler(file="pow_posterior_genecodon.out",
  powerColumnName="power", likelihoodColumnName="likelihood")
RevBayes > ps.marginal()
```

## 1.4 Compute Bayes Factors and Select Model

Now that we have estimates of the marginal likelihood under each of our different models, we can evaluate their relative plausibility using Bayes factors. Use Table 2 to summarize the marginal log-likelihoods estimated using the stepping-stone and path-sampling methods.

Table 2: Estimated marginal likelihoods for different partition configurations\*.

Partition	Marginal lnL estimates	
	Stepping-stone	Path sampling
1.1 uniform ( $M_1$ )		
1.2 moderate ( $M_2$ )		
1.3 extreme ( $M_3$ )		

\*you can edit this table

Phylogenetics software programs log-transform the likelihood to avoid [underflow](#), because multiplying likelihoods results in numbers that are too small to be held in computer memory. Thus, we must use a different form of equation 3 to calculate the ln-Bayes factor (we will denote this value  $\mathcal{K}$ ):

$$\mathcal{K} = \ln[BF(M_0, M_1)] = \ln[\mathbb{P}(\mathbf{X} | M_0)] - \ln[\mathbb{P}(\mathbf{X} | M_1)], \quad (4)$$

where  $\ln[\mathbb{P}(\mathbf{X} | M_0)]$  is the *marginal lnL* estimate for model  $M_0$ . The value resulting from equation 4 can be converted to a raw Bayes factor by simply taking the exponent of  $\mathcal{K}$

$$BF(M_0, M_1) = e^{\mathcal{K}}. \quad (5)$$

Alternatively, you can interpret the strength of evidence in favor of  $M_0$  using the  $\mathcal{K}$  and skip equation 5. In this case, we evaluate the  $\mathcal{K}$  in favor of model  $M_0$  against model  $M_1$  so that:

if  $\mathcal{K} > 1$ , then model  $M_0$  wins  
 if  $\mathcal{K} < -1$ , then model  $M_1$  wins.

Thus, values of  $\mathcal{K}$  around 0 indicate ambiguous support.

Using the values you entered in Table 2 and equation 4, calculate the ln-Bayes factors (using  $\mathcal{K}$ ) for the different model comparisons. Enter your answers in Table 3 using the stepping-stone and the path-sampling estimates of the marginal log likelihoods.

Once you complete Table 3, you will notice that the Bayes factor comparison indicates strong evidence in support of the highly partitioned model using both the stepping-stone and path sampling estimates of the marginal likelihoods. However, this does not mean that model  $M_3$  is the *true* partition model. We only considered three out of the many, many possible partitions for 2,659 sites (the number of possible partitions can be viewed if you compute the [2659<sup>th</sup> Bell number](#)). Given the strength of support for the highly partitioned model, it is possible that further partitioning is warranted for these data. In particular, partitioning the dataset by codon position for both atpB *and* rbcL is an important next step for this exercise (consider taking some time on your own to test this model).

Table 3: Bayes factor calculation\*.

Model comparison	ln-Bayes Factor ( $\mathcal{K}$ )	
	<i>Stepping-stone</i>	<i>Path sampling</i>
$M_1, M_2$		
$M_2, M_3$		
$M_1, M_3$		
Supported model?		

\*you can edit this table

Because of the computational costs of computing marginal likelihoods and the vast number of possible partitioning strategies, it is not feasible to evaluate all of them. New methods based on nonparametric Bayesian models have recently been applied to address this problem (???). These approaches use an infinite mixture model (the Dirichlet process; ??) that places non-zero probability on *all* of the countably-infinite possible partitions for a set of sequences. Bayesian phylogenetic inference under these models is implemented in the program [PhyloBayes](#) (?) and the [subst-bma](#) plug-in for [BEAST2](#) (?).

Note that Bayes factors based on comparison of HM-based marginal likelihoods often *strongly* favor the most extremely partitioned mixed model. In fact, the harmonic mean estimator has been shown to provide unreliable estimates of marginal likelihoods, compared to more robust approaches (???). Based on these studies, it is recommended that you avoid using HM-derived marginal likelihoods for Bayes factor comparisons. (The Canadian Bayesian Radford Neal says the harmonic mean is the “[worst Monte Carlo method ever](#)”.)

## 1.5 Perform MCMC Analysis Under Preferred Model

### *Clear Workspace and Load the Data and Model*

```
RevBayes > clear()
RevBayes > source("RevBayes_scripts/genecodonLoop_partition_model.Rev")
```

### *Specify Monitors*

```
RevBayes > monitors[1] <- modelmonitor(filename="conifer_genecodon_mcmc.
    log", printgen=100, separator = "      ")
```

```
RevBayes > monitors[2] <- filemonitor(filename="conifer_genecodon_mcmc.
    trees", printgen=100, separator = "      ", phylogeny)
```

```
RevBayes > monitors[3] <- screenmonitor(printgen=10, separator = "
", sf, shape)
```

### *Initialize and Run MCMC*

```
RevBayes > mymcmc <- mcmc(myModel, monitors, moves)
```

```
RevBayes > mymcmc.burnin(generations=10000, tuningInterval=1000)
```

```
RevBayes > mymcmc.run(generations=100000)
```

## 1.6 Summarize and Analyze MCMC Output

```
RevBayes > treeTrace <- readTreeTrace("conifer_genecodon_mcmc.trees")
RevBayes > treeTrace.summarize()
```

```
RevBayes > mapTree(treeTrace, "conifer_genecodon_MAP.tre")
```

The trees in these files are also annotated with various branch- or node-specific parameters or statistics in an extended Newick format called NHX. We can use FigTree to visualize these summary trees.

- Open the summary tree in FigTree: **conifer\_genecodon\_MAP.tre**.
- Use the tools on the side panel to display the posterior probabilities as node labels. An example is shown in Figure 4.

## Exercise 1 – Batch Mode

If you wish to run this exercise in batch mode, the files are provided for you.

You can carry out these batch commands by providing the file name when you execute the **rb** binary in your unix terminal (this will overwrite all of your existing run files).

- **> rb full\_analysis.Rev**

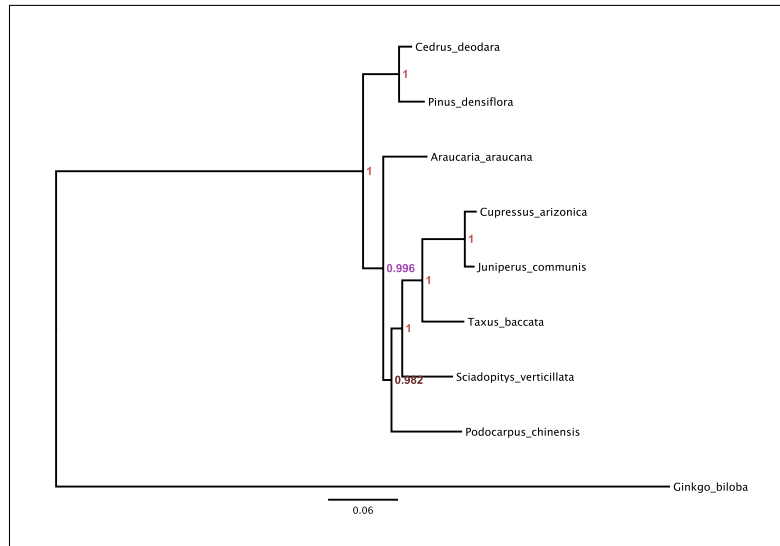



Figure 4: The summary tree from the uniform analysis, with posterior probabilities labeled at nodes. [TAH comment: need to make a new figure]

## Useful Links

- RevBayes: <https://github.com/revbayes/code>
- MrBayes: <http://mrbayes.sourceforge.net>
- PhyloBayes: [www.phylobayes.org](http://www.phylobayes.org)
- Tree Thinkers: <http://treethinkers.org>

Questions about this tutorial can be directed to:

- Tracy Heath (email: [tracyh@berkeley.edu](mailto:tracyh@berkeley.edu))
- Michael Landis (email: [mlandis@berkeley.edu](mailto:mlandis@berkeley.edu))
- Sebastian Höhna (email: [sebastian.hoehna@gmail.com](mailto:sebastian.hoehna@gmail.com))

 This tutorial was written by [Tracy Heath](#), [Michael Landis](#), and [Sebastian Höhna](#); licensed under a [Creative Commons Attribution 4.0 International License](#). (Some content in this tutorial is based on the [Phylogenetic Inference using MrBayes v3.2](#) tutorial written by Tracy Heath, Conor Meehan, and Brian Moore.)

Version dated: July 26, 2014