Phylogenetic Inference using RevBayes

Historical biogeography

Michael Landis

1 Introduction

This lab describes how to perform Bayesian inference of historical biogeography using RevBayes.

Outline

I. Introduction
- a) Workspace setup
- b) Range evolution models and data augmentation

II. DEC in RevBayes
- a) Tree, range, and geographic input
- b) DEC as a graphical model
- c) Monitors, moves, and MCMC
- d) Model selection with Bayes factors

III. Output and Analysis
- a) MCMC output in Tracer
- b) Ancestral range output
- c) Animate ancestral ranges

→ These little arrows indicate lines containing key information to progress through the lab. The rest of the text gives context for why we're taking these steps or what to make of results.

1.1 Handy links for this lab

RevBayes	https://github.com/revbayes/revbayes
Lab zip file	https://github.com/revbayes/revbayes/.../RB_biogeo_files.zip
Phylowood software	http://mlandis.github.io/phylowood
Phylowood manual	https://github.com/mlandis/phylowood/wiki
Tracer	http://tree.bio.ed.ac.uk/software/tracer

1.2 Setting up your workspace

The practical part of the lab will analyze a small dataset of 19 taxa distributed over 4 biogeographic areas. Parts of the lab will require entering terminal commands, which will assume you are using the Unix shell `bash`. Just ask for help if the commands don't seem to work.

→ Portions of this lab require Python is installed. No packages are needed.

→ This tutorial will assume you have successfully installed RevBayes and can be called from your current working directory.

→ Download and unzip the lab zip file, `RB_biogeo_files.zip`.

1.3 Motivation

XXX

1.4 Model and method

This section contains a brief description of the data, model, parameters, and method used in BayArea.

First, we define the range for taxon i as the bit vector X_i, where $X_{i,j} = 1$ if the taxon is present in area j and $X_{i,j} = 0$ if the taxon is absent. Each taxon range is a bit vector of length N areas. For example, if taxon B is present only in areas 2 and 3 out of $N = 3$ areas, its range is represented as $X_B = (0, 1, 1)$, which is translated to the bit string $X_B = 011$ for short. If we apply the labels A, B, and C to our three areas, this bit vector can also be represented as a set: $X_B = 011$ is the same as $X_B = \{B, C\}$, or $X_B = BC$ for short. The data matrix, $\mathbf{X}$, is analogous to a multiple sequence alignment where each element in the data matrix reports a discrete value for a homologous character shared by all taxa at column j.

2 Modeling anagenic range evolution

Next, we need a model of anagenic range evolution. Since we have discrete characters we'll use the continuous-time Markov chain, which allows us to compute transition probability of a character changing from i to j in time t through matrix exponentiation

$$\mathbf{P}_{i,j}(t) = \left[\exp\left\{\mathbf{Q}t\right\}\right]_{i,j},$$

where $\mathbf{Q}$ is the instantaneous rate matrix defining the rates of change between all pairs of characters, and $\mathbf{P}$ is the transition probability rate matrix. This technique of matrix exponentiation is powerful because it integrates over all possible scenarios of character transitions that could occur during t so long as the chain begins in state i and ends in state j. Remember, i and j represent different ranges, each of which is encoded as a set of occupied areas.

We can then encode range evolution events into the allowed character transitions of $\mathbf{Q}$ and parameterize the events so that we may infer their relative importance to generating our observed ranges. We'll take a simple model of range expansion (e.g. $BC \rightarrow ABC$) and range contraction (e.g. $BC \rightarrow C$). (Range expansion may also be referred to as dispersal or area gain and range contraction as extirpation or area loss.) The rates in the transition matrix for three areas might appear as

$$\mathbf{Q} = \begin{array}{c|cccccccc} & \emptyset & A & B & C & AB & AC & BC & ABC \\ \hline \emptyset & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A & e_A & - & 0 & 0 & d_{AB} & d_{AC} & 0 & 0 \\ B & e_B & 0 & - & 0 & d_{BA} & 0 & d_{BC} & 0 \\ C & e_C & 0 & 0 & - & 0 & d_{CA} & d_{CB} & 0 \\ AB & 0 & e_B & e_A & 0 & - & 0 & 0 & d_{AC} + d_{BC} \\ AC & 0 & e_C & 0 & e_A & 0 & - & 0 & d_{AB} + d_{CB} \\ BC & 0 & 0 & e_C & e_B & 0 & 0 & - & d_{BA} + d_{CA} \\ ABC & 0 & 0 & 0 & 0 & e_C & e_B & e_A & - \end{array},$$

where $e = (e_A, e_B, e_C)$ are the (local) extinction rates per area, and $d = (d_{AB}, d_{AC}, d_{BC}, d_{CB}, d_{CA}, d_{BA})$ are the dispersal rates between areas.

Q: For the three-area DEC rate matrix above, what is the rate of leaving state 101? That is, what is the absolute value of the diagonal term in the rate matrix for $Q_{AB,AB}$?

Note the rate of more than one event occurring simultaneously is zero, so a range must expand twice by one area in order to expand by two areas.

Q: What events might explain a transition from range ABC to range A? From range AB to range C?

Of course, this model can be specified for more than three areas.

Q: Imagine a DEC rate matrix with four areas, $ABCD$. What would be the dispersal rate for $Q_{BC,BCD}$? How many states does a DEC rate matrix with four areas have? What is the relationship between the number of areas and the number of states under the DEC model?

Let's consider what happens to the size of $\mathbf{Q}$ when the number of areas, N, becomes large. For three areas, $\mathbf{Q}$ is size 8×8. For ten areas, $\mathbf{Q}$ is size $2^{10} \times 2^{10} = 1024 \times 1024$, which approaches the largest size matrix that can be exponentiated in a practical amount of time. For biogeographic inference under large numbers of areas, see the Tutorial XXX (**?**).

2.1 Modeling cladogenic range evolution

In addition to dispersal and extinction, the DEC models cladogenic range evolution events. For each internal node in the reconstructed tree, one of two cladogenic events can occur: sympatry or allopatry. Say the range of a species approaching an internal node, i.e. that is about to speciate, is A. Since the species range is size one, this always results in sympatry, where both daughter lineages inherit the ancestral species range, so both lineages begin in state A. The notation $A \rightarrow A \mid A$ describes this event: the state is A before cladogenesis, the left daughter inherits range A after cladogenesis, as does the right daughter.

Now suppose the ancestral range is ABC. Under sympatric cladogenesis, one lineage identically inherits the ancestral species range, ABC, while the other lineage inherits only a single area, i.e. only A or B or C.

Under allopatric cladogenesis, the ancestral range is split evenly among daughter lineages, e.g. one lineage may inherit AB and the other inherits C. Both daughter lineages inherit the entire ancestral species range following widespread sympatric cladogenic events. For a general description of state transitions for cladogenic events, see **?**.

Q: What are the 6 possible states in the daughter lineages after cladogenesis given the state is AB before cladogenesis?

The probabilities of anagenic change along lineages must account for all combinations of starting states and ending states. For 3 areas, there are 8 states, and thus $8 \times 8 = 64$ probability terms for pairs of states. For cladogenic change, we need transition probabilities for all combinations of states before cladogenesis, after cladogenesis for the left lineage, and after cladogenesis for the right lineage. Like above, for three areas, there are 8 states, and $8 \times 8 \times 8 = 512$ cladogenic probability terms.

Q: For three areas, there are 3 + 12 + 6 = 21 possible sympatry events and 6 + 6 = 12 possible allopatry events. How many terms in the cladogenesis matrix are zero?

The DEC model ignores speciation events hidden by extinction or incomplete taxon sampling. The probability of cladogenesis and local extinction events would ideally be linked to a birth-death process, as it is in the GeoSSE model (**?**). Unfortunately, since this sort of model scales poorly, and DEC models remain the only option when the geography has more than two or three areas.

The rest of this tutorial will describe how to assemble the input, run the analysis, assess the output, and visualize the results.

2.2 Input

For this tutorial, we'll use a dataset for 19 species of *Psychotria* whose range spans the Hawaiian archipelago. The dataset was originally reported in **?** and analyzed using the maximum likelihood method, LAGRANGE, by **?**. We'll use this dataset for three reasons. First, it is relatively small, meaning we can produce results quickly. Second, the Hawaiian archipelago can be broken into naturally discrete areas and has a well-characterized geographical history that is uncomplicated to model. Third, it has previously been analyzed, which provides some basis for comparison to other methods. To simplify things, this model will assume we live in a gentler world where presence-absence characters are known without error.

For larger datasets to play with, see `input/sim_aus_50tip_33area.*` for a simulated model where all cladogenic events were sympatric (wide and narrow only).

2.3 RevBayes Analysis

There are six major parts to the analysis.

First, we need to read in the input files and assign analysis settings. Second, we need to construct our model. Third, we need to assign moves and monitors to our model parameters for use with Markov chain Monte Carlo (MCMC). Fourth, we will run an MCMC analysis assuming a complex model. Fifth, we will compare the complex model with a simple model using Bayes factors. Finally, we'll analyze our MCMC output.

3 Epoch models and ancestral state reconstruction

4 Large numbers of areas

→ Open the RevBayes console

```
$ revbayes
```

First, we'll assign all our input files to `String` variables.

```
RevBayes > in_fp <- "./input/"
RevBayes > data_fn <- "psychotria_range.nex"
RevBayes > area_fn <- "hawaii_dynamic.atlas.txt"
RevBayes > out_fp <- "./output/"
RevBayes > out_str <- "bg_2rate"
```

Then we'll create our range data, tree, and atlas objects

```
RevBayes > data <- readDiscreteCharacterData(in_fp + data_fn)
RevBayes > tree <- readTrees(in_fp + data_fn)[1]
RevBayes > atlas <- readAtlas(in_fp + area_fn)
```

4.1 Creating the model

Here, we will compose our rate matrix, $\mathbf{Q}$, parameterized by the transition rates, λ, and the distance dependent dispersal power parameter, β.

First, for λ, we will create a vector of two rates, where `glr[1]` corresponds to the rate of area loss (local extinction) and `glr[2]` corresponds to the rate of area gain (dispersal). Each rate will be drawn from an exponential distribution with rate 10.0 (mean 0.1). Because our tree is in units of millions of years, this means our prior expectation is that any given species undergoes one dispersal or extinction event per area per ten million years.

To introduce this to the model, type

```
RevBayes > for (i in 1:2) glr[i] ~ dnExponential(10.0)
```

Next, we will create `dp`, which determines the importance of geographical distance to dispersal. Remember that values of β far from zero means distance is important. So, if we we assign a prior that pulls β towards zero, then posterior values of β far from zero indicate the range data are informative of the importance of distance to dispersal. We'll use an exponential distribution with rate 10.0 (mean 0.1) as a prior for `dp`.

We will also create a deterministic node to modify the rate of dispersal between areas by evaluating `dp` and `atlas`. This node is determined by the function `fnBiogeoGRM`, where GRM stands for "geographical rate modifier", and plays the role of the $\eta(\cdot)$ rate-modifier function mentioned earlier. We will tell the `fnBiogeoGRM` function to modify dispersal rates based on distances and whether or not the area exists during an epoch.

```
RevBayes > dp ~ dnExponential(10.0)
RevBayes > grm := fnBiogeoGRM(atlas=atlas, distancePower=dp, useAvailable=true,
   useDistance=true)
```

Now we need a deterministic node to represent the rate matrix, $\mathbf{Q}$. To determine the value of this node, we'll use the function `fnBiogeoDE` to assign our model parameters to transition rates as described in the introduction. As input, we'll pass our gain and loss rates, `glr`, and our geographical rate modifier, `grm`. In addition, we'll inform the function of the number of areas in our analysis and whether we will allow species to be absent in all areas (i.e. have the null range).

```
RevBayes > Q := fnBiogeoDE(gainLossRates=glr, geoRateMod=grm, numAreas=4,
   forbidExtinction=true)
```

To extract information for the frequencies of different cladogenic event types, we will create a Dirichlet-distributed stochastic node. The simplex is over three events, subset sympatry (index 0), allopatry (index 1), and widespread sympatry (index 2), but not over narrow sympatry whose range size is one. The prior parameter `[1,1,1]` is known as a flat prior, meaning all event types are expected to occur at equal

frequency. If there is information in the data of a dominant cladogenic mode of range evolution, the posterior simplex values in `csf` will reflect this.

```
RevBayes > csf ~ dnDirichlet([1,1,1])
```

For the model's final node, we create the stochastic node for the continuous-time Markov chain (CTMC). This node's distribution is `dnPhyloDACTMC` where `DA` indicates the CTMC uses data-augmentation to compute the likelihood rather than Felsenstein's pruning algorithm. To create the distribution, we must pass it our `tree` and `Q` objects, but additionally inform the distribution that it will be using a biogeographic model, that it will introduce the simple cladogenic range evolution events described in `?` (`useCladogenesis=true`), and that it will assign zero probability to a transition away from the null range state.

```
RevBayes > M ~ dnPhyloDACTMC(tree=tree, Q=Q, C=csf, type="biogeo", forbidExtinction=true
    , useCladogenesis=true)
```

So we may evaluate the graphical model's likelihood, we tell the CTMC to observe the `data` object, which will prime the model with data-augmented character histories. Now `M` has a defined likelihood value.

```
RevBayes > M.clamp(data)
RevBayes > M.lnProb
   -56.0288
```

Finally, we encapsulate our graphical model into a `Model` object, which can learn the model's structure and dependencies from any model parameter.

```
RevBayes > my_model <- model(glr)
```

4.2 Running an MCMC analysis

Now that we have our `Model` object, we can soon run an MCMC analysis. Remember that MCMC approximates the posterior distribution by repeatedly proposing new model parameter values, accepting or rejecting those new parameter values based on the model likelihood (and on biases in the proposal distribution), then reporting the sampled parameter values.

First, let's assign moves to our model parameters. These parameters are all supported for real positive values, which is appropriate for use with scale-multipler proposal, `mvScale()`. To inspect our model parameter types and the proposal argument types, enter

```
RevBayes > type(dp)
   RealPos
```

```
RevBayes > type(glr)
   RealPos[]
RevBayes > mvScale
   Move_Scale function (RealPos x, RealPos lambda, Bool tune, RealPos weight)
```

The arguments for `mvScale` are fairly typical as far as RevBayes `Move` objects go: `x` is the stochastic node the `Move` will update, `lambda` is proportional to how radically different proposed parameter values will tend to be, `tune` allows `lambda` to be adjusted automatically as the MCMC runs, and `weight` tells the MCMC how many times to perform the `Move` during a single MCMC generation (e.g., `weight=2.0`) means each generation will call that `Move` for the parameter `x` twice).

```
RevBayes > moves[1] <- mvScale(x=glr[1], lambda=0.5, tune=false, weight=5.0)
RevBayes > moves[2] <- mvScale(x=glr[2], lambda=0.5, tune=false, weight=5.0)
RevBayes > moves[3] <- mvScale(x=dp,  lambda=0.5, tune=false, weight=5.0)
RevBayes > moves[4] <- mvSimplexElementScale(csf, alpha=10.0, tune=false, weight=4.0)
```

In addition to proposing new model parameter values, we must also propose new data-augmented states and events to properly integrate over the space of possible range histories. The major challenge to sampling character histories is ensuring the character histories are consistent with the observations at the tip of the tree. The proposals in this tutorial use **?**'s rejection sampling algorithm, with some modifications to account for cladogenic events and epoch-based rate matrices.

The basic idea is simple. Each time a character history proposal is called, it selects a node at random from the tree. Path history proposals (`mvPathCHRS()`) propose a new character history for the lineage leading to that node. Node history proposals (`mvNodeCHRS()`) propose a new character history for the node and for the three lineages incident to that node. The character history proposal also samples some number of areas to update, ranging from one to all of the areas. Once the new character history is proposed, the likelihood of the model is evaluated and the MCMC accepts or rejects the new state according to e.g. the Metropolis-Hastings algorithm.

Because these `Move` objects update the character histories stored in the data-augmented CTMC node, e.g. `M`, they require access to a `TimeTree` object to know which lineages are sisters and whether the lineages span various epochs, and a `RateMap_Biogeography` object to propose new character histories. The `lambda` argument gives what proportion of areas' character histories to update. Here, if `lambda=0.2`, then the proposal will redraw character histories for each area with probability 0.2 (in addition to one random area with probability 1). Below, we use two moves of each type with `lambda=0.2` and `lambda=1.0` for partial and full character history updates, respectively. Indicating `type="biogeo"` informs the `Move` object to be aware of special character history constraints, such as cladogenic events and forbidden null ranges. The `weight` parameter should be assigned a value proportional to the number of nodes in the analysis to ensure proper mixing.

Let's create the character history moves as follows.

```
RevBayes > n_nodes <- tree.nnodes()
RevBayes > moves[5] <- mvNodeCHRS(ctmc=M, qmap=Q, tree=tree, lambda=0.2, type="biogeo",
    weight=2.0*n_nodes)
RevBayes > moves[6] <- mvPathCHRS(ctmc=M, qmap=Q, tree=tree, lambda=0.2, type="biogeo",
    weight=2.0*n_nodes)
RevBayes > moves[7] <- mvNodeCHRS(ctmc=M, qmap=Q, tree=tree, lambda=1.0, type="biogeo",
    weight=n_nodes)
RevBayes > moves[8] <- mvPathCHRS(ctmc=M, qmap=Q, tree=tree, lambda=1.0, type="biogeo",
    weight=n_nodes)
```

Now that we have moves for all our parameters and the character histories, we'll proceed with assigning `Monitor` objects to record their values. The first two `Monitor` objects are fairly standard and found in most RevBayes MCMC analyses. `mnScreen` reports the values for any nodes assigned to `RevObject ...` every `printgen` generations to the terminal screen. `mnModel` reports the values for all nodes in the `Model` object every `printgen` generations to the file assigned to `filename`, which is delimited by the `separator` character.

```
RevBayes > mnScreen
   Mntr_Screen function (RevObject ..., Natural printgen, Bool posterior, Bool
   likelihood, Bool prior)
RevBayes > monitors[1] <- mnScreen(printgen=10, glr, dp, csf)
RevBayes > mnModel
   Mntr_Model function (String filename, Natural printgen, String separator,
   Bool posterior, Bool likelihood, Bool prior, Bool append, Bool
   stochasticOnly)
RevBayes > monitors[2] <- mnModel(filename=out_fp+params_fn, printgen=10)
```

Like any parameter, we can sample the augmented range histories from the MCMC to approximate the posterior distribution of range histories. This is statistically equivalent to generating ancestral state reconstructions from a posterior distribution via stochastic mapping. We will extract these reconstructions using special monitors designed for the `dnPhyloDACTMC` distribution.

Next, we will create `Mntr_CharacterHistoryNewickFile` objects to record the sampled character history states for each node in the tree. This `Monitor` has two `style` options: `counts` reports the number of gains and losses per branch in a tab-delimited Tracer-readable format; `events` reports richer information of what happens along a branch, anagenically and cladogenically, using an extended Newick format. How to read these file formats will be discussed in more detail in Section **??**.

```
RevBayes > mnCharHistoryNewick
   Mntr_CharacterHistoryNewickFile function (String filename,
   AbstractCharacterData ctmc, TimeTree tree, Natural printgen, String
   separator, Bool posterior, Bool likelihood, Bool prior, Bool append, String
   style = events|counts
   , String type = biogeo
```

```
   )
RevBayes > monitors[3] <- mnCharHistoryNewick(filename=fp+out_str+".events.txt", ctmc=M,
     tree=tree, printgen=100, style="events")
RevBayes > monitors[4] <- mnCharHistoryNewick(filename=fp+out_str+".counts.txt", ctmc=M,
     tree=tree, printgen=100, style="counts")
```

As our last monitor, the `Mntr_CharacterHistoryNhxFile` records character history values throughout the MCMC analysis, then stores some simple posterior summary statistics as a Nexus file. These summary statistics could be computed from the previously mentioned `Monitor` output files, but `mnCharHistoryNhx` provides a simple way to produce Phylowood-compatible files. We will also discuss this file's format in more detail later in the tutorial.

```
RevBayes > mnCharHistoryNhx
   Mntr_CharacterHistoryNhxFile function (String filename, AbstractCharacterData
   ctmc, TimeTree tree, RlAtlas atlas, Natural samplegen, Natural maxgen,
   Probability burnin, String separator, Bool posterior, Bool likelihood, Bool
   prior, String type = biogeo
   )
RevBayes > monitors[5] <- mnCharHistoryNhx(filename=fp+out_str+".nhx.txt", ctmc=M, tree=
    tree, atlas=atlas, samplegen=100, maxgen=25000, burnin=0.2)
```

Before we run the MCMC, we'd like to get the node index of the ancestor When analysing the output, we'll take some special interest in the branch for the most recent common ancestor of *P. kaduana* and *P. hatheway*. We will identify this lineage by its index, 23, which is meaningful only for a fixed tree topology,

```
RevBayes > names = tree.names()
RevBayes > names[15]
   P_kaduana_PuuKukuiAS
RevBayes > names[16]
   P_hathewayi_1
RevBayes > mrcaIndex(tree=tree,clade=clade(names[15],names[16]))
   23
```

4.3 Running an MCMC analysis

Now all that's left is to configure and run our MCMC analysis. For this, we create an `Mcmc` object, which we give our `Move` vector, our `Monitor` vector, and our `Model object`

```
RevBayes > mcmc
   MCMC function (Model model, Monitor[] monitors, Move[] moves, String
   moveschedule = sequential|random|single
   )
RevBayes > my_mcmc <- mcmc(my_model, monitors, moves)
```

MCMC typically requires some period of burn-in before it reaches stationarity, i.e. from a random starting point, it takes some time for the chain to produce valid samples from the posterior distribution. By running `burnin()`, we tell the `Mcmc` object to propose and reject new states but *not* to record anything to file. After burn-in is complete, we call `run()`, where we begin recording valid posterior samples under our model.

```
RevBayes > my_mcmc.burnin(generations=1000, tuningInterval=50)
RevBayes > my_mcmc.run(generations=25000)
```

Everything we've done is contained in the file `biogeography_DEC_2rate.Rev`. You can modify this file as you like then re-run the analysis by typing

```
RevBayes > source("./scripts/biogeography_DEC_2rate.Rev")
```

4.4 Model selection using Bayes factors

Bayes factors (BFs) are used to select which of two models better describes the observed data, $\mathbf{X}_{obs}$, and are computed as the ratio of marginal likelihoods for those two models. One might prefer to analytically compute the marginal likelihood, but it's the same intractable quantity we intentionally avoid computing when using MCMC in a Bayesian context. Instead, we must estimate the marginal likelihood from our posterior distribution samples. Here, we will use thermodynamic integration (**?**) and stepping-stone approximation (**?**). The exact details of these techniques will not be covered here, but there is an important practical point to mention: both methods rely on computing a number of "power posterior" distributions. Computing more power posteriors increases the marginal likelihood estimator's accuracy at the cost of computational time.

Moving on, we'll compute the Bayes factor to compare a simple one-rate model, which asserts the rate of area gain and loss are always equal, to a two-rate model which allows these rates to vary independently. Rather than specifying the model manually, we will load (source) the model definition from a file then enter the commands to compute its marginal likelihood. For faster results, we will use two separate RevBayes sessions, one for each model. For each session, the power posterior analysis run for 1000 generations during burn-in then 1000 generations per each of 30 power posterior categories.

First session:

```
RevBayes > source("./scripts/biogeography_DEC_1rate.Rev")
RevBayes > pp_fn <- out_fp + out_str + ".pp.txt"
RevBayes > pow_p <- powerPosterior(my_model, moves, pp_fn, cats=30)
RevBayes > pow_p.burnin(generations=1000,tuningInterval=100)
RevBayes > pow_p.run(generations=5000)
```

Second session:

```
RevBayes > source("./scripts/biogeography_DEC_2rate.Rev")
RevBayes > pp_fn <- out_fp + out_str + ".pp.txt"
RevBayes > pow_p <- powerPosterior(my_model, moves, pp_fn, cats=30)
RevBayes > pow_p.burnin(generations=1000,tuningInterval=100)
RevBayes > pow_p.run(generations=5000)
```

Each power posterior analysis will write their contents to the file given in `pp_fn`. These files are `bg_1rate.pp.txt` and `bg_2rate.pp.txt` for the simple and complex models, respectively. This may take a few minutes. When complete, the power posterior files may then be used to compute marginal likelihoods. For example, from the RevBayes session analyzing the simple one-rate model

```
RevBayes > ss <- steppingStoneSampler(file=pp_fn)
RevBayes > ss.marginal()
RevBayes > ps <- pathSampler(file=pp_fn)
RevBayes > ps.marginal()
```

For a given model, the path sampling and stepping stone sampling methods should produce similar marginal likelihood estimates. Values should be within one log likelihood unit of one another. If the values are extremely different, this may indicate `powerPosterior` should be re-run with a larger number of `cats`. We chose `cats=30` which should suffice, and we see no problem. Then from the complex two-rate model RevBayes session using the same commands as above. Finally, we can compute the Bayes factor, which is simply the ratio of marginal likelihoods.

```
RevBayes > exp(-51.7202)/exp(-52.3158)
   1.81412
```

A value of one would mean both models had equal marginal likelihoods. A value less than one would indicate the first model, the simple model, had a larger marginal likelihood, and was therefore favored by model testing. But that's not the case, the value is greater than one, and the complex two-rate model is favored. Similar to frequentist interpretations of significance for p-values, there is no universal and objective criterion of significance with Bayes factors, but most would agree a factor of 1.8 (or 1.0/1.8) indicates weak support for one model over the other.

5 Output

5.1 Sampled parameters from `ScreenMonitor`

The `mnScreen` monitor reports model parameter values to the screen, where each row corresponds to the current accepted MCMC state, and each column reports some model feature, such as the model likelihood or a parameter value. Every 20 iterations, this monitor re-prints the column headers.

```
Running MCMC simulation for 25000 iterations
The simulator uses 8 different moves in a random
move schedule with 241 moves per iteration

Iteration  |   Posterior |        dp  |     glr[1] |     glr[2] |     csf[1] |     csf
    [2] |     csf[3]
-----------------------------------------------------------------------------------------------------------

0          |    -51.3307 |   0.0570518 |    0.175137 |   0.0580957 |    0.330891 |
    0.255308 |    0.413801
10         |    -54.4257 |   0.0416423 |    0.166936 |    0.178402 |   0.0549136 |
    0.148854 |    0.796233
20         |    -58.0696 |   0.0991853 |    0.136495 |    0.122135 |    0.308418 |
    0.448892 |    0.242689
30         |    -46.5049 |     0.10676 |   0.0958918 |   0.0959592 |    0.543837 |
    0.363871 |   0.0922922
40         |    -42.8697 |    0.173549 |    0.158565 |   0.0662419 |    0.569416 |
    0.126439 |    0.304145
50         |    -43.5319 |    0.117868 |    0.196497 |   0.0523307 |    0.440269 |
    0.257171 |     0.30256

...
```

For the complex 2-rate model, our model parameters are `dp`, the distance power parameter, and the rates of area loss and gain, `glr[1]` and `glr[2]`, respectively, and the frequencies for subset sympatry, allopatry, and widespread sympatry `csf[1]`, `csf[2]`, and `csf[3]`, respectively. If you notice the value of some parameter is rarely updated from iteration to iteration, the MCMC is probably mixing poorly therefore it's not generating samples from the posterior distribution (the MCMC's stationary distribution). In this case, you may want to re-run the analysis with different arguments for the `Move` object assigned to that parameter.

5.2 Sampled parameters from `ModelMonitor`

This tab-delimited file contains parameter samples from the posterior distribution. As with the `ScreenMonitor`, columns are model or parameter values and rows are MCMC cycles.

```
0          -51.3307   -56.0288 4.69806 0.175137 0.0580957 0.057051 0.330891 0.255308
    0.413801
10         -54.4257   -58.1568 3.73110 0.166936 0.1784020 0.041642 0.054913 0.148854
    0.796233
20         -58.0696   -62.0923 4.02274 0.136495 0.1221350 0.099185 0.308418 0.448892
    0.242689
30         -46.5049   -51.1197 4.61480 0.095891 0.0959592 0.106760 0.543837 0.363871
    0.092292
```

```
40          -42.8697   -46.4870 3.61735 0.158565 0.0662419 0.173549 0.569416 0.126439
    0.304145
50          -43.5319   -47.4659 3.93394 0.196497 0.0523307 0.117868 0.440269 0.257171
    0.302560

...
```

→ Open Tracer, select the fields for the posterior probability and area gain rate, `glr[2]`, then click the Joint-Marginal tab.

Here, we see a strong negative correlation between the posterior probability and the area gain rate, which is expected. Next, click the Estimates tab then select the three `csf` parameters.

5.3 Biogeographic event counts from `mnCharHistoryNewick`

Recording stochastic mappings in a Tracer-compatible format requires some summarization. This monitor generates a tab-delimited file where the number of events of each type for each branch is recorded.

→ Open `./output/bg_2rate.counts.txt` in a text editor.

```
0           -51.3307   -56.0288 4.69806      9       9       18      0       0       0
          1       1       0       ...
10          -54.4257   -58.1568 3.73110      9       10      17      0       0       1
          1       1       0       ...
20          -58.0696   -62.0923 4.02274      11      9       15      2       1       0
          2       1       1       ...
30          -46.5049   -51.1197 4.61480      8       8       18      0       0       0
          1       1       0       ...
40          -42.8697   -46.4870 3.61735      7       7       18      0       0       0
          1       1       0       ...
50          -43.5319   -47.4659 3.93394      7       7       18      0       0       0
          1       1       0       ...
...
```

For example, `b2_s1` gives the number of areas that are gained for the branch leading to the node indexed 2. `b2_c` gives the cladogenic event type that gives rise to the node indexed 2, where narrow sympatry, subset sympatry, allopatry, and widespread sympatry are recorded as `0`, `1`, `2`, and `3`, respectively. The columns `t_s0` and `t_s1` give the sum of events over all branches. `t_c0`, `t_c1`, and `t_c2` give the total number of narrow sympatric, subset sympatric, allopatric, and widespread sympatric cladogenic events over the entire tree.

Because the expected number of gain events should be proportional to the area gain rate, we expect to

see the same negative correlation between posterior probability and number of events as we did with the posterior and rate in the `parameters.txt` file.

Open Tracer, select the fields for the posterior probability and the number of gained areas over the tree, `t1`, then click the Joint-Marginal tab.

One interesting facet of this output is there are never fewer than six events. In fact, since we assume a stratified geography and that only one event may occur per instant, it is impossible to describe the data we see at the tips with fewer than six gain events. That is, six gain events is part of the maximum parsimony solution.

5.4 Biogeographic event histories from `mnCharHistoryNewick`

For more detailed data exploration, this analysis also provides annotated Newick strings with the complete character mappings for the tree.

→ Open `./output/bg_2rate.events.txt` in a text editor.

```
Iteration      Posterior      Likelihood     Prior  Tree
0      -51.3307       -56.0288       4.69806 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...
10     -54.4257       -58.1568       3.7311 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...
20     -58.0696       -62.0923       4.02274 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...
30     -46.5049       -51.1197       4.6148 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...
40     -42.8697       -46.4870       3.61735 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...
50     -43.5319       -47.4659       3.93394 (((((((P_hawaiiensis_WaikamoiL1[&index
    =18;nd=0010;pa=0010;ev={}]:0.96 ...

...
```

Each iteration records the data-augmented character history (stochastic mapping) using metadata labels, which, for an internal node, looks like

```
[&index=23;nd=0110;pa=0010;ch0=0010;ch1=0110;cs=s;bn=16;ev={{t:0.2513,a:1.1195,s:1,i:1}}
```

`index=23` indicates this branch leads to the node indexed 23. The branch began in the ancestral state `pa=0100` and terminated in the state `nd=0110`. Since this node is not a tip node, it represents a speciation event, so the daughter ranges are also given, `ch0=0010` and `ch1=0110`. The cladogenic state for this speciation event was subset sympatric, `cs=s`, rather than sympatric (wide or narrow; `w` or `n`) or allopatric

(a). Anagenic dispersal and extinction events occurring along the lineage leading to node 19 are recorded in `events`, where each event has a time (relative to the absolute branch length), absolute age, state (into), and character index (`t`, `a`, `s`, `i`, resp.). For this posterior sample of the character history for the branch leading to node 22, the species range expanded into Oahu at age 1.1195.

To manipulate this data format, we'll use Python scripts. Below are a few examples of interesting posterior features.

→ Open a Python console and read in the events.

```
> cd scripts
> python27

...

>>> from bg_parse import *
>>> dd=get_events(fn="../output/bg_2rate.events")
```

By default, `get_events()` extracts a dictionary where each node index maps to a branch's character history as reported in `./input/bg_2rate.events.txt`. Each branch is a dictionary whose keys are various parts of the MCMC state and whose values the MCMC samples.

```
>>> dd[23].keys()
['ch1', 'iteration', 'bn', 'nd', 'ch0', 'prior', 'posterior', 'cs', 'ev', 'likelihood']
>>> dd[23]['posterior'][0:5]
[-48.6952, -60.1832, -53.2286, -57.5778, -53.4633]
```

To get the $n = 1$ highest-valued sample for a branch by its posterior value

```
>>> get_best(dd[23],n=1,p='posterior')
{'prior': [4.48225], 'iteration': [14890], 'bn': [22], 'nd': [[0, 1, 1, 0]], 'ch0': [[0,
    1, 1, 0]], 'ch1': [[0, 0, 1, 0]], 'posterior': [-34.7139], 'pa': [[0, 1, 0, 0]], '
   cs': ['subset_sympatry'], 'ev': [[{'age': 1.5637, 'state': 1, 'idx': 2, 'time':
   0.8611}]], 'likelihood': [-39.1962]}
```

To get the probability that area i and area j are both part of the species range as the branch for node 23 terminates, just before the speciation event

```
>>> get_area_pair(dd[23])
[[0.0816, 0.0188, 0.0628, 0.0000],
 [0.0188, 0.7081, 0.4390, 0.0000],
```

```
[0.0628, 0.4390, 0.7141, 0.0000],
[0.0000, 0.0000, 0.0000, 0.0000]]
```

showing area 3 was occupied nearly with probability 0.71 and both areas 2 and 3 were occupied with probability 0.44. Note, Hawaii was submerged until approximately 0.5 million years ago, and thus the probability of being in that area is 0.0.

If the range is size one during a speciation event, the cladogenic event state is always narrow sympatric, `'narrow_sympatry'`. But given the opportunity for non-sympatric events, i.e. that the range is larger than size one, we can get the probability of cladogenic state using For the probability for cladogenic event state given the range was larger than size one

```
>>> get_clado_state(dd[23])
{'allopatry': 0.0224, 'subset_sympatry': 0.1463, 'widespread_sympatry': 0.3183, '
   narrow_sympatry': 0.5130}
>>> get_clado_state(dd[23],minSize=2)
{'allopatry': 0.0460, 'subset_sympatry': 0.3005, 'widespread_sympatry': 0.6535, '
   narrow_sympatry': 0.0000}
>>> get_clado_state(get_best(dd[23],n=100),minSize=2)
{'allopatry': 0.1290, 'subset_sympatry': 0.6774, 'widespread_sympatry': 0.1936, '
   narrow_sympatry': 0.0000}
```

Depending on your question, different aspects of the posterior cladogenic state will interest you. Narrow sympatry is the favored ancestral state, but wide sympatry is favored for ranges of size $n > 1$. However, when we look at the 100 most probable samples, subset sympatry becomes most favored.

More script functions are found in `./scripts/bg_parse.py`.

5.5 New Hampshire extended format file (`./output/bg_2rate.nhx`)

Because this data is very high-dimensional, we'll use an external data exploration tool to look at range evolution.

This file summarizes the input and output from a BayArea analysis using NEXUS format containing a New Hampshire eXtended (NHX) tree string. NHX allows you to annotate nodes in a Newick string with meta-information, which BayArea uses to report the probabilities in the `my_run.area_probs.txt` file. The `geo` block gives the geographical latitudes and longitudes for the areas in the order they are reported as probabilities. Like the `my_run.area_probs.txt` file, this file is not written until the analysis is complete. This annotation is used for the two visualization programs covered in the next section, Phylowood and BayArea-Fig. The anatomy of the Phylowood and BayArea-Fig settings blocks will also be explained there.

6 Visualization

Here we'll explore two options for visualizing ancestral range reconstructions. I'll walk you through some of the basic functionality, but feel free to play around as you like.

6.1 Phylowood

Phylowood generates interactive animations to explore biogeographic reconstructions.

→ Open `http://mlandis.github.io/phylowood`.

→ Drag and drop `./output/bg_2rate.nhx.txt` into the text field.

→ Click the Play button to view the animation.

There are three control panels to help you filter data: the media panel, the map panel, and the phylogeny panel. The media buttons correspond to Beginning, Slow/Rewind, Play, Stop, Fast Forward, Ending (from left to right). The animation will play the timeframe corresponding to the slider.

→ Drag the slider to the right (the present).

→ Pan and zoom around the map.

Marker colors correspond to the phylogenetic lineages in the phylogeny panel. Markers are split into slices and (loosely) sorted phylogenetically, so nearby slices are generally closely related. At divergence events, a marker's radius is proportional to the marginal posterior probability the node was present in the area at that time. Between divergence events, marker's radius is simply an interpolation of the values at the two endpoints. Some information about geological constraints and cladogenic events is lost.

→ Mouseover an area to learn which lineage it belongs to and its presence probability.

Since it's difficult to see how specific clades evolve with so many taxa, Phylowood offers two ways to filter taxa from the animation. We call the set of a lineage, all its ancestral lineages towards the root, and all descendant lineages a phylogenetic heritage. The root's heritage is the entire clade. A leaf node's heritage is a path from the tip to the root.

→ Mouseover a lineage to temporarily highlight the lineage's heritage. Remove the mouseover to remove the highlight effect.

The highlight effect is temporary and quickly allows you to single out lineages of interest during animation. Phylowood also offers a masking effect that persists until an unmask command is issued.

→ Double-click the white root branch to mask the root node's heritage (all lineages). Single click a lineage

to unmask that lineage's heritage.

Now that the masking effects are in place, you're free to interact with other map components. In addition, the area of marker sizes is only distributed among unmasked lineages.

→ Visit `https://github.com/mlandis/phylowood/wiki` to learn more about Phylowood.

Version dated: January 30, 2015

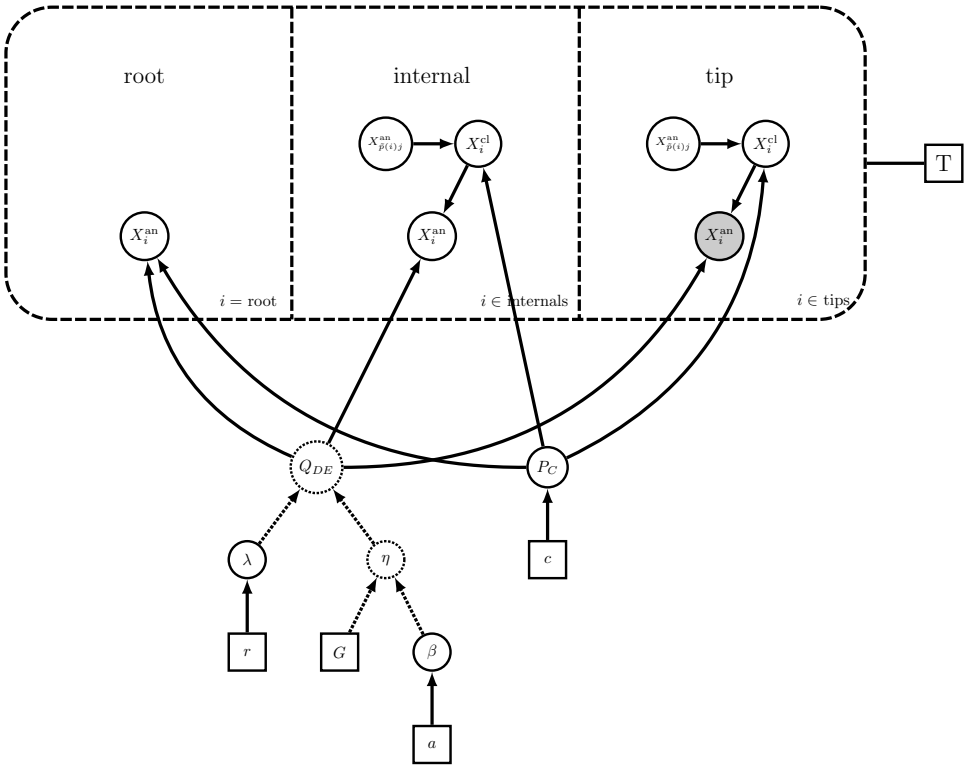

Figure 1: Graphical model of DEC. The tree plate's topology is fixed by T, where each internal node has both an anagenic and cladogenic random variable (X_i^{an} and X_i^{cl}, resp.) that represents an ancestral species before and after it speciated. Anagenic change is modeled by a continuous time Markov process, where Q_{DE} is the instantaneous rate matrix of area gain and loss, as parameterized by λ. The geographic distance rate modifier function, η, takes in the geographical distances and strata as G, and the distance power parameter, β. Cladogenic change is modeled by P_C, a Dirichlet-distributed simplex with a flat prior.

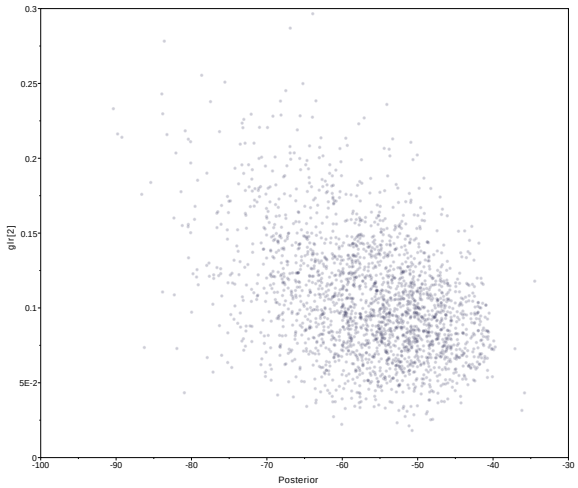

Figure 2: Joint-marginal distribution of posterior and area gain rate, λ_1.

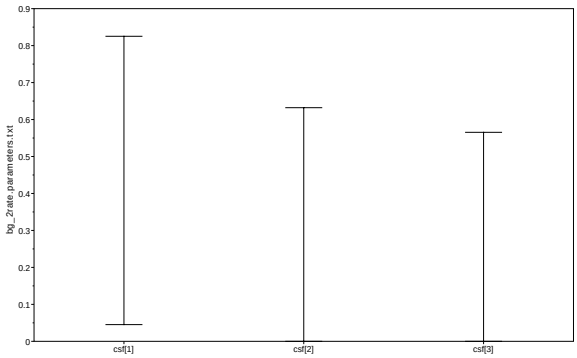

Figure 3: Mean values for the cladogenic state frequency simplex, where `csf[1]`, `csf[2]`, and `csf[3]` correspond to subset sympatry, allopatry, and wide sympatry whose mean posterior values are 0.45, 0.30, and 0.25, respectively.

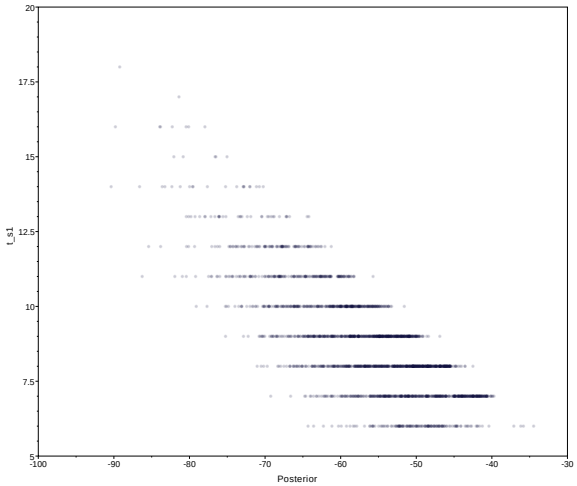

Figure 4: Joint-marginal distribution of posterior and number dispersal events summed over the tree.

Figure 5: Phylowood frame showing posterior ancestral range of root node.

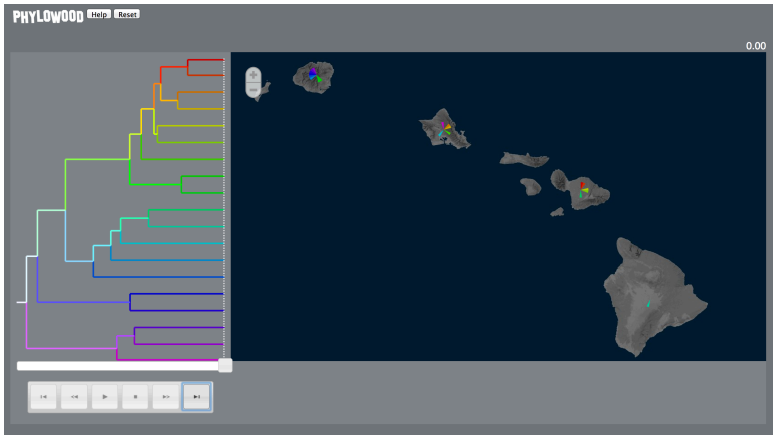

Figure 6: Phylowood frame showing distribution of extant taxon ranges.

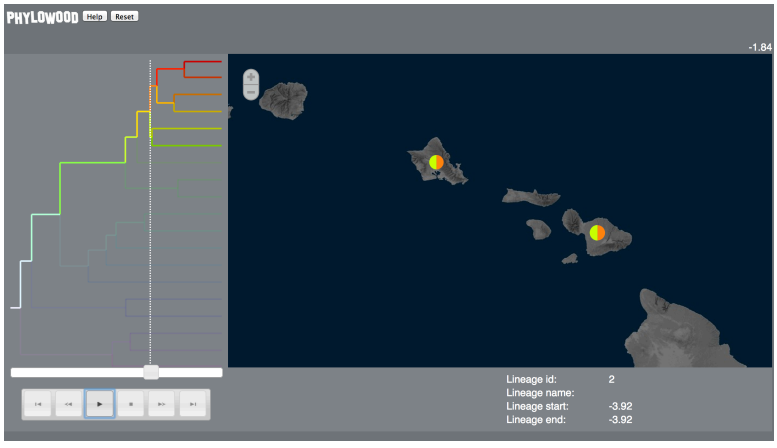

Figure 7: Phylowood frame highlighting the posterior range for the most recent common ancestor of *P. mauiensis* and *P. hawaiiensis*.