Lab 12:
Gene Tree - Species Tree Reconstruction Using RevBayes and the Multispecies Coalescent

By Will Freyman
updated by Ixchel González-Ramírez

1 Before you begin

Please download and install **RevBayes**. RevBayes is a command line program that runs from the terminal, please make sure you can get it running before lab. Downloads are here: `https://revbayes.github.io/download`

For this lab, please preserve the structure of the zipped filed that I have given you. Also, use the RevBayes version that is that folder.

The scripts used here have been modified from the excellent tutorial by Bastien Boussau and Sebastian Höhna posted at `http://revbayes.com`.

2 Introduction to the multispecies coalescent model

In lecture we have discussed many reasons that gene trees may differ from species trees. These processes include horizontal tree transfer, introgression, incomplete lineage sorting (ILS), and gene tree error. The multispecies coalescent models ILS and deep coalescence (see Figure 1). It does not model reticulate evolutionary processes such as introgression or hybrid speciation, though the multispecies coalescent has been extended (with varying degrees of success) to model those processes.

There is a long history of methods that reconcile gene trees with species trees. There are parsimony methods such as Minimising Deep Coalescences (MDC) that try to minimize gene duplications and losses, algorithmic approaches such as ASTRAL (Accurate Species TRee ALgorithm) that are consistent with the coalescent but are not explicitly using a probabilistic method. There are also methods that reconcile already estimated gene tree posterior distributions such as BUCKy (Bayesian Concordance Analysis) and BEST (Bayesian Estimation of Species Trees).

However, the full probabilistic multispecies coalescent model that estimates gene trees simultaneously with the species tree is only implemented in a couple pieces of software. The most popular is *BEAST (starBEAST), which is just BEAST with some extensions. It is very easy to use, but can be a bit of a black-box if you simply rely on default priors. Today we'll implement the multispecies coalescent in RevBayes, and step through setting up the entire model as well as the MCMC proposals needed to perform inference.

2.1 Calculating coalescent probabilities

We talked in lecture about how to calculate coalescent probabilities, but it is important to keep in mind that two parameters of the species tree have an impact on the expected gene tree/species tree incongruence:

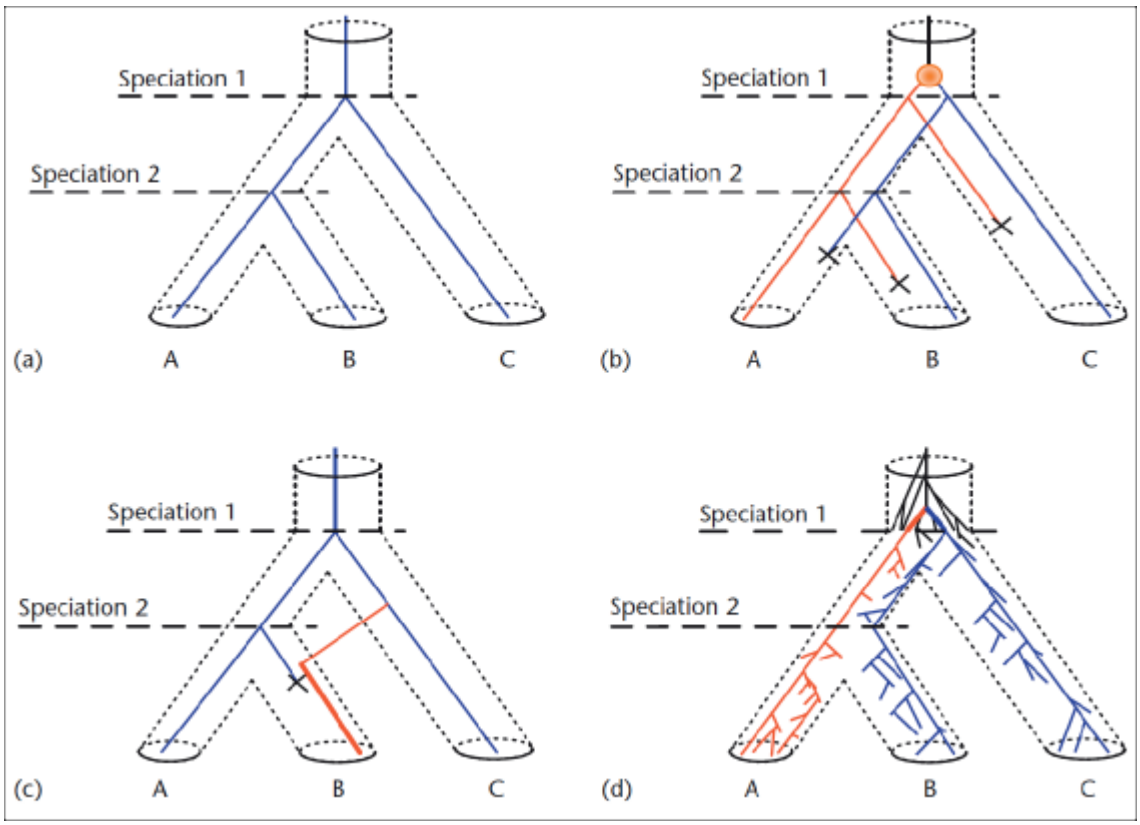

Figure 1: The processes of discord. The species tree is represented as a tubular structure. Gene trees are blue and red lines running along the species trees. a) A gene tree that perfectly matches the species tree. b) The gene tree and the species tree differ because of gene duplications and losses. c) The gene tree and the species tree differ because of gene transfer and gene loss. d) The gene tree and the species tree differ because of incomplete lineage sorting. [Replicated from Fig. 2 in Boussau et al. (2009).]

1. **Time between speciation events:** As branches get shorter, the pool of alleles within the branch has less time to undergo change. So we can expect deep coalescence and ILS more frequently when branch lengths are short.

2. **Effective population size:** In large populations alleles are more likely to be carried by large numbers of individuals, so chance events are less likely to wipe out alleles. In small populations chance events can wipe out entire alleles, resulting in less polymorphism. So we can expect deep coalescence and ILS more frequently when effective population sizes are large.

2.2 A note about "species" trees

In this lab and much of the literature, we talk about gene trees nested within "species" trees. Note that a species tree could more precisely be called a "population" tree, and that both populations and species have fuzzy boundaries that may change depending on the organisms and/or the scientists. Regardless of the term you use, the important point is that the multispecies coalescent explicitly models polymorphic lineages and ILS.

3 Implementing the multispecies coalescent in RevBayes

Today, we'll apply the multispecies coalescent model to 10 gene alignments from 23 primate species.

3.1 Loading our sequence data

Start up RevBayes and load our data.
Attention: If you get an error because RevBayes cannot find the files, please specify the full path to your folder.

```
locus_names = ["COIII", "FGA", "GHRmeredith", "lrpprc_169",
              "npas3", "sim1", "tex2", "ttr", "zfy", "zic3"]

num_loci = locus_names.size()

# read in each data matrix separately
for ( i in 1:num_loci ) {
    data[i] <- readDiscreteCharacterData("data/" + locus_names[i] + ".fasta")
}

# Get some useful variables from the data. We need these later on.
primate_tree = readTrees("data/primates.tree")[1]
n_species <- primate_tree.ntips()
taxa <- primate_tree.taxa()
n_branches <- 2 * n_species - 1 # number of branches in a rooted tree

# set my MCMC move index
mi = 0
```

3.2 Species tree model

We'll use a constant-rate sampled-birth-death process as the prior on the species tree. The birth-death process takes both speciation and extinction rates as its parameters. Note, we could easily use a Yule (birth-only) prior here instead if we wanted. We could also use rate-heterogenous birth-death processes as well. This is the *sampled* birth-death process because it requires an additional parameter ρ (or `rho`) which represents the proportion of extant lineages sampled.

First, set up the prior on speciation and extinction. There are multiple ways to parameterize these, we will use this:

```
speciation ~ dnGamma(2,2)
relativeExtinction ~ dnBeta(1,1)

#transform the diversification and turnover rates into speciation and extinction rates
extinction := speciation * relativeExtinction
```

We can set up ρ as such, because we know we sampled 23 out of the roughly 450 named primate species:

```
sampling_fraction <- 23 / 450
```

Now we will calibrate the tree by assuming that the crown age of primates is around 75 mya. Our informed guess is about 75-80 mya, thus, we specify a normal distribution with mean 75 and standard deviation 2.5 as the prior on the root age. Since the root age can only be a

positive real number we truncate the normal distribution at 0. We could have additionally used fossils, but for simplicity we'll stick to just the root calibration.

```
root ~ dnNormal(mean=75,sd=2.5,min=0.0, max=Inf)
```

Now we need to set up moves so that the MCMC can explore parameter space. First set up moves on the diversification rates and root age:

```
moves[++mi] = mvSlideBactrian(speciation,tune=true,weight=2)
moves[++mi] = mvSlideBactrian(relativeExtinction,tune=true,weight=2)
moves[++mi] = mvScaleBactrian(speciation,lambda=1,tune=true,weight=2)
moves[++mi] = mvScaleBactrian(relativeExtinction,lambda=1,tune=true,weight=2)
moves[++mi] = mvSlide(root,delta=1,tune=true,weight=0.2)
```

Now, we can draw our species tree (`psi` or ψ) from the sampled-birth-death process:

```
psi ~ dnBDP(lambda=speciation, mu=extinction, rootAge=root, rho=sampling_fraction,
taxa=taxa)
```

And let's add some moves for the species tree:

```
moves[++mi] = mvNarrow(psi, weight=5.0)
moves[++mi] = mvNNI(psi, weight=1.0)
moves[++mi] = mvFNPR(psi, weight=3.0)
moves[++mi] = mvGPR(psi, weight=3.0)
moves[++mi] = mvSubtreeScale(psi, weight=3.0)
moves[++mi] = mvNodeTimeSlideUniform(psi, weight=5.0)
```

Mixing in the multispecies coalescent model is particularly challenging. Although, in principle and assuming our MCMC is working as intended, after an infinite amount of time we would converge to a correct estimate of the posterior distribution over the species tree, gene trees and other parameters, it is advisable to start from reasonable starting values for the species tree and the gene trees. In our case, we will use previously reconstructed Maximum A Posteriori (MAP)candidate gene trees:

```
##Read each MAP gene trees:
for ( i in 1:num_loci ) {
gene_trees[i] <- readTrees("output_GeneTrees/" + locus_names[i] + "_MAP.tree")[1]
print("Gene tree "+i+ " has "+ gene_trees[i].ntips() + " tips.")
}
```

These MAP trees will provide good starting gene trees, and will also be used to generate a starting species tree using a function that computes a maximum tree. The maximum tree *is the tree with the largest possible speciation times in the space of species trees restricted by available gene trees* (Liu et al. 2010).

```
recTree <- maximumTree(gene_trees)
psi.setValue(recTree)
root.setValue(recTree.rootAge())
write("\t\tProposed starting species tree: ")
write( psi)
write("\t\tWith root age: " + root)
```

3.3 Gene tree model

Now that we have a species tree, we can specify the prior on the gene trees by using a multispecies coalescent process.
First, we will assume that each branch of the species tree, which represents a population, has its own population size. Note that we use fixed population sizes for the terminal branches because we have only a single individual per species and thus have no information about its population size.

```
for (i in 1:n_species) {
Ne[i] <- 10.0
}
for (i in (n_species+1):n_branches) {
Ne[i] ~ dnExponential(0.01)
moves[++mi] = mvScale(Ne[i],lambda=.1,tune=true,3.0)
moves[++mi] = mvSlide(Ne[i],tune=true,2.0)
}
```

Now we specify the gene trees, using the multispecies coalescent model. Note that we use the good starting point for gene trees that we have previously specified from the MAP trees.

```
for (i in 1:num_loci) {
taxon_map = readTaxonData("data/species_maps/primates_" + locus_names[i] +
"_species_map.txt")
geneTree[i] ~ dnMultiSpeciesCoalescent(speciesTree=psi, Ne=Ne, taxa=taxon_map)
geneTree[i].setValue(gene_trees[i])
}
```

Now that we have established the relationship between gene trees and species tree, we can set moves to sample this relationship:

```
## Joint species tree/gene tree moves
move_species_narrow_exchange = mvSpeciesNarrow( speciesTree=psi, weight=20 )
move_species_subtree_scale_beta = mvSpeciesSubtreeScaleBeta(psi, weight=5)
move_species_subtree_scale = mvSpeciesSubtreeScale(psi, weight=5)

## Moves that alter each gene tree:
for (i in 1:num_loci) {
moves[++mi] = mvNNI(geneTree[i], 5.0)
moves[++mi] = mvNarrow(geneTree[i], 5.0)
moves[++mi] = mvFNPR(geneTree[i], 3.0)
moves[++mi] = mvGPR(geneTree[i], 2.0)
moves[++mi] = mvSubtreeScale(geneTree[i], 5.0)
moves[++mi] = mvTreeScale(geneTree[i], 1.0, true, 3.0)
moves[++mi] = mvNodeTimeSlideUniform(geneTree[i], 20.0)
move_species_narrow_exchange.addGeneTreeVariable( geneTree[i] )
move_species_subtree_scale_beta.addGeneTreeVariable( geneTree[i] )
move_species_subtree_scale.addGeneTreeVariable( geneTree[i] )
}
## We must not forget to include the joint moves into the vector of moves!
moves[++mi] = move_species_narrow_exchange
moves[++mi] = move_species_subtree_scale_beta
moves[++mi] = move_species_subtree_scale
```

3.4 Sequence evolution models and clock rates

Next we need clock rates for each gene. These scale the branch lengths in expected number of substitutions into time. Here we will assume a strict clock for each gene, however we could easily use relaxed clocks by drawing a separate rate for each branch.

```
for ( i in 1:num_loci ) {
clock_rate[i] ~ dnExponential(1.0)
moves[++mi] = mvScale(clock_rate[i], weight=2.0)
moves[++mi] = mvSlide(clock_rate[i], weight=3.0)
}
```

We will assign each gene the HKY+G model of nucleotide evolution. The HKY model takes the parameter `kappa` which is the ratio of the transition to transversion rates, and the parameter `pi` which are the equilibrium frequencies for each of the four bases. Let's set up the rate matrix `Q` for each gene:

```
for ( i in 1:num_loci ) {
kappa[i] ~ dnLognormal(0,1)
moves[++mi] = mvScale(kappa[i],weight=1.0)
moves[++mi] = mvSlide(kappa[i], weight=1.0)
pi_prior[i] <- v(1,1,1,1)
pi[i] ~ dnDirichlet(pi_prior[i])
moves[++mi] = mvSimplexElementScale(pi[i],weight=2.0)
Q[i] := fnHKY(kappa[i],pi[i])
}
```

And now we'll model the gamma distributed rate variation among sites (the +G part of the HKY+G model). As is standard, we use a discretized gamma distribution with four rate categories. We'll utilize an exponential prior on the `alpha` shape parameter.

```
for ( i in 1:num_loci ) {

    alpha_prior[i] <- 0.05
    alpha[i] ~ dnExponential( alpha_prior[i] )
    gamma_rates[i] := fnDiscretizeGamma( alpha[i], alpha[i], 4, false )
    moves[++mi] = mvScale(alpha[i],weight=2)

}
```

Finally, just like in last week's lab, we set up continuous-time Markov chain (CTMC) distributions for sequence evolution, one for each gene. We then clamp our sequence data to the CTMCs. We don't set up moves for the CTMC because they are the observed values and are therefore fixed.

```
for ( i in 1:num_loci ) {
    seq[i] ~ dnPhyloCTMC(tree=geneTree[i], Q=Q[i], branchRates=clock_rate[i],
    siteRates=gamma_rates[i], type="DNA")

    seq[i].clamp(data[i])
}
```

And now we have completed our model. Let's make a workspace variable to represent the model. We can use any node of our model, here we choose to use the species tree `psi`.

```
mymodel = model(psi)
```

3.5 Running the MCMC analysis

We're now ready to perform inference on our model. Just like before, let's set up some MCMC monitors.

```
monitors[1] = mnScreen(printgen=100)
monitors[2] = mnModel(filename="output/primates_root_calibration.log", printgen=10,
    separator = TAB)
monitors[3] = mnFile(filename="output/primates_root_calibration.trees", printgen=10,
    separator = TAB, psi)
```

Let's additionally include monitors for each of the gene trees:

```
for ( i in 1:num_loci ) {
    monitors[i+3] = mnFile(filename="output/primates_root_calibration_" + locus_names[i] +
".trees", printgen=10, separator = TAB, geneTree[i])
}
```

Here we'll perform a simple MCMC analysis. You could also set `nruns=2` for a replicated analysis or use `mcmcmc` to perform Metropolis coupled MCMC with heated chains.

```
mymcmc = mcmc(mymodel, monitors, moves, nruns=1)
```

And finally let's start the analysis.

```
mymcmc.run(generations=1000)
mymcmc.operatorSummary()
```

This will take about 10 minutes or so.

3.6 Summarizing the species tree from the posterior

Now let's summarize the posterior distribution of species trees. We first read in the tree trace, and then produce a *maximum a posteriori* (MAP) tree.

```
treetrace = readTreeTrace("output/primates_root_calibration.trees", treetype="clock")
mapTree(treetrace,"output/primates_root_calibration.tree")
```

Check out `primates_root_calibration.tree` in FigTree.

Questions:

1. Send me a PDF or screenshot of your MAP tree (you can open it in FigTree).
2. Take a look at your log file in tracer. Does it look like your analysis has converged?
3. Name a couple assumptions we made that seem unrealistic to you? According to your experience with RevBayes, do they seem to be easy to solve?
4. Concatenation is still used much more frequently than the full multispecies coalescent. Summarize in a sentence or two the pros and cons of using concatenation versus the multispecies coalescent.

5. Contrast your output tree with the original tree we used as prior "data/primates.tree". Open them in Figtree. Do you observe differences in topology and time estimation? What big conclusion can you make from your observations.

References