Lab 11:
Diversification models: Pure-Birth and Birth-death Process; Joint character evolution and diversification analyses using BiSSE;

By Ixchel González-Ramírez

1 Before you begin

Make sure you have downladed and decompressed the files needed for this lab.
We will also require RevBayes. If you haven't downloaded it, do it here: `https://revbayes.github.io/download`. I recommend to have the execute file of RevBayes in the same folder where you have the data for this lab.
Tip: Every time you open a new RevBayes window, start by setting your working directory in the same way you would do in R.

```
setwd("This/is/the/full/path/to/the/folder/where/yo/want/to/work/")
```

2 Introduction

The study of macroevolutionary processes such as adaptive radiation, diversity-dependent and character-dependent diversification, key innovations, and mass extinction requires the estimation of diversification (speciation − extinction) rates. In this lab, we will infer diversification rates under two models (pure birth and birth-dead). Then, we will go through the BiSSE analysis, that allows us to infer correlations on the evolution of traits and diversification rates.
The material for this lab was modified by me from the online tutorials of RevBayes url-https://revbayes.github.io/tutorials/, and was written by:

- Sebastian Höhna - Introduction to Diversification Rate Estimation
- Sebastian Höhna and Tracy Heath - Simple Diversification Rate Estimation
- Sebastian Höhna, Will Freyman, and Emma Goldberg - State-dependent diversification with BiSSE and MuSSE
- Carrie Tribble - R code

3 Bayesian inference of diversification rates

Models of speciation and extinction are fundamental to any phylogenetic analysis of macroevolutionary processes (e.g., divergence time estimation, diversification rate estimation, continuous and discrete trait evolution, and historical biogeography). First, a prior model describing the distribution of speciation events over time is critical to estimating phylogenies with branch lengths proportional to time. Second, stochastic branching models allow for inference of speciation and extinction rates. These inferences allow us to investigate key questions in evolutionary biology.

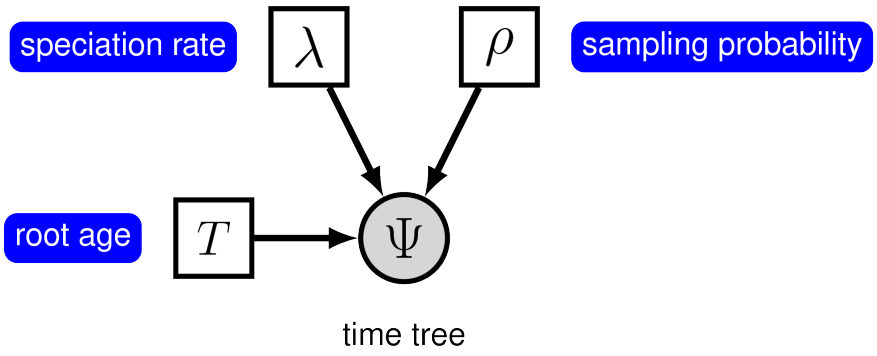

Figure 1: The graphical model representation of the pure-birth (Yule) process.
Image from https://revbayes.github.io/tutorials/divrate/simple.html

3.1 The pure birth model

The simplest branching model is the pure-birth process described by Yule (1925). Under this model, we assume that at any instant in time, every lineage has the same speciation rate, λ. Thus, the speciation rate remains constant over time. The pure-birth branching model does not allow for lineage extinction (i.e., the extinction rate μ=0). However, the model depends on a second parameter, ρ, which is the probability of sampling a species in the present time. It also depends on the time of the start of the process, whether that is the origin time or root age. Therefore, the probabilistic graphical model (Figure 1) of the pure-birth process is quite simple, where the observed time tree topology and node ages are conditional on the speciation rate, sampling probability, and root age.

3.1.1 Reading the data

We will use a tree of 233 primates from the APE package in R. Keep in mind that we are assuming that we know the number of extant primate species: 367. Begin by reading the observed tree and get the list of taxa from it.

```
T <- readTrees("data/primates_tree.nex")[1]
taxa <- T.taxa()
```

Aditionally, we can initialize a variable for out vector moves and monitors:

```
moves    = VectorMoves()
monitors = VectorMonitors()
```

3.1.2 Specifying the model

Now we will specify the priors accordingly with the biological information we have (prior-beliefs). We will start by specifying the prior of the **birth_rate**:

```
birth_rate_mean <- ln( ln(367/2) / T.rootAge() )
birth_rate_sd <- 0.587405
birth_rate ~ dnLognormal(mean=birth_rate_mean,sd=birth_rate_sd)
```

Here, the stochastic node called birth rate represents the speciation rate, λ. The prior for this node is taken from a lognormal distribution, so we need to specify the hyperpriors `birth_rate_mean` and the `birth_rate_sd` of this distribution. We chose the mean so that it

is centered around observed number of species (i.e., the expected number of species under a Yule process will thus be equal to the observed number of species) and a prior standard deviation of 0.587405 which creates a lognormal distribution with 95% prior probability spanning exactly one order of magnitude. If you want to represent more prior uncertainty by, e.g., allowing for two orders of magnitude in the 95% prior probability then you can simply multiply birth_rate_sd by a factor of 2.
Next, to estimate the value of λ, we assign a proposal mechanism to operate on this node. Remember that in RevBayes these MCMC sampling algorithms are called **moves**.

```
moves.append( mvScale(birth_rate,lambda=1,tune=true,weight=3) )
```

Another prior belief is that we sampled 233 out of 367 living primate species. To account for this we can set the sampling parameter as a constant node with a value of 233/367

```
rho <- T.ntips()/367
```

Any stochastic branching process must be conditioned on a time that represents the start of the process. In his case, we will condition on the age of the root of the tree.

```
root_time <- T.rootAge()
```

Now we have all of the parameters we need to specify the full pure-birth model. We can initialize the stochastic node representing the time tree. Note that we set the μ parameter to the constant value 0.0, cause we are not taking into account extinction.

```
timetree ~ dnBDP(lambda=birth_rate, mu=0.0, rho=rho, rootAge=root_time,
samplingStrategy="uniform", condition="survival", taxa=taxa)
timetree.clamp(T)
```

The second line of code is "clamping" the model with the "observed" tree, i.e. the data. In other words, we are fixing the value of the time tree to our observed tree.
Finally, we can create a workspace object of our whole model using the `model()` function. Workspace objects are initialized using the = operator, this helps to distinguish the grafical model objects (using the `"<-"` operator) from the objects used by the MCMC anlysis. The model() function traverses all of the connections and finds all of the nodes we specified.

```
mymodel = model(birth_rate)
```

3.1.3 Running the MCMC analysis

For our MCMC analysis, we need to set up a vector of monitors to record the states of our Markov chain. The first line of code create the output file were we will track the parameters of the model. The second line creates a screen monitor that constantly report the `birth_rate`:

```
monitors.append( mnModel(filename="output/primates_Yule.log",printgen=10,
separator = TAB) )
monitors.append( mnScreen(printgen=1000, birth_rate) )
```

With a fully specified model, a set of monitors, and a set of moves, we can now set up the MCMC algorithm that will sample parameter values in proportion to their posterior probability. The mcmc() function will create our MCMC object:

```
mymcmc = mcmc(mymodel, monitors, moves, nruns=2, combine="mixed")
```

Now, run the MCMC:

```
mymcmc.run(generations=50000)
```

Question 1
Locate the .log output files in your computer and open them in `Tracer`.
Did the chains reach convergence? What is the likelihood value for this analysis?
What's the mean posterior estimate of the birth_rate and what is the estimated HPD (high posterior density)?
Explain with your words what does HPD tell us?

3.2 The birth-death process model

As we saw, the pure-birth model does not account for extinction. This assumption is fairly unrealistic for most phylogenetic datasets on a macroevolutionary time scale since the fossil record provides evidence of extinct lineages. Kendall (1948) described a more general branching process model to account for lineage extinction called the birth-death process. Under this model, at any instant in time, every lineage has the same rate of speciation λ and the same rate of extinction μ. This is the **constant-rate birth-death process**, which considers the rates constant over time and over the tree (Nee et al. 1994; Höhna 2015). In principle, we can specify a model with prior distributions on speciation and extinction rates directly, like we did for λ in the previous section. However, it is more common to specify prior distributions on a transformation of the speciation and extinction rate (diversification rate and turnover) because, for example, we want to enforce that the speciation rate is always larger than the extinction rate (Figure 2).

3.2.1 Specifying the model

For this section, make sure you start in a new RevBayes session. We will start by reading the data and initializing a vector for moves and monitors. We will also set some variables that are the sames as in the pure-birth model like ρ and root_time:

```
T <- readTrees("data/primates_tree.nex")[1]
taxa <- T.taxa()
moves    = VectorMoves()
monitors = VectorMonitors()
rho <- T.ntips()/367
root_time <- T.rootAge()
```

Now we are taking into account the extinction, the extant number of species is a good prior information about the magnitude of the diversification. So, We will use the same prior for the diversification rate as we used before for the birth rate.

```
diversification_mean <- ln( ln(367.0/2.0) / T.rootAge() )
diversification_sd <- 0.587405
diversification ~ dnLognormal(mean=diversification_mean,
sd=diversification_sd)
moves.append( mvScale(diversification,lambda=1.0,tune=true,weight=3.0) )
```

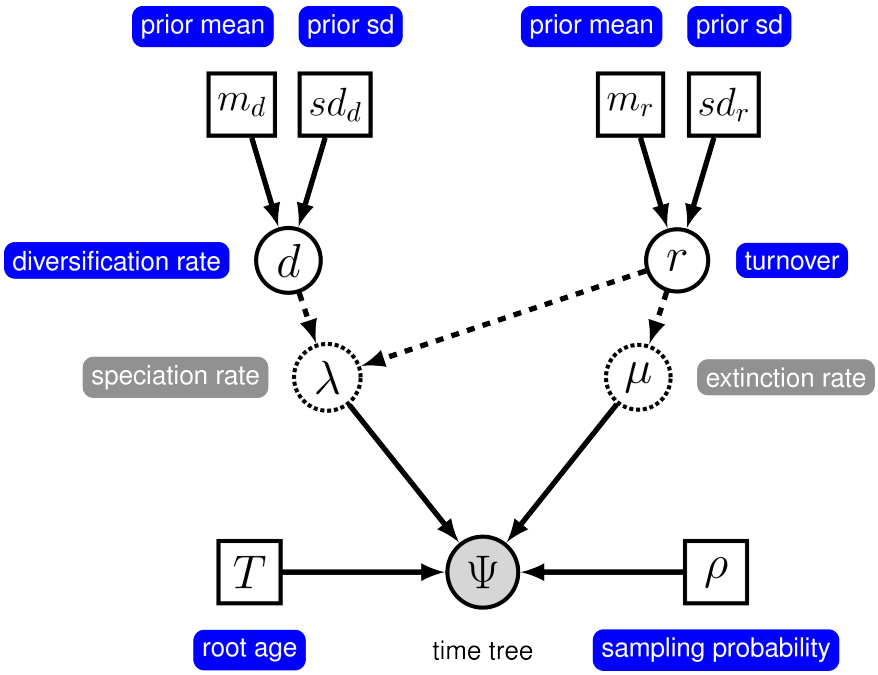

Figure 2: The graphical model representation of the birth-death process with uniform sampling parameterized using the diversification and turnover.
Image from https://revbayes.github.io/tutorials/divrate/simple.html

Unfortunately, we have less prior information about the turnover rate. The turnover rate is the rate at which one species is replaced by another species due to a birth plus death event. Hence, the turnover rate represent the longevity of a species. For simplicity we use the same prior on the turnover rate but with two orders of magnitude prior uncertainty.

```
turnover_mean <- ln( ln(367.0/2.0) / T.rootAge() )
turnover_sd <- 0.587405*2
turnover ~ dnLognormal(mean=turnover_mean,sd=turnover_sd)
moves.append( mvScale(turnover,lambda=1.0,tune=true,weight=3.0) )
```

The birth and death rates are both deterministic nodes. We compute them by simple parameter transformation. Note that the death rate is in fact equal to the turnover rate.

```
birth_rate := diversification + turnover
death_rate := turnover
```

Initialize the stochastic node representing the time tree. The main difference now is that we provide a stochastic parameter for the extinction rate μ

```
timetree ~ dnBDP(lambda=birth_rate, mu=death_rate, rho=rho,
rootAge=root_time, samplingStrategy="uniform", condition="survival",
taxa=taxa)
timetree.clamp(T)
```

Finally we have a full specified model, and we can set the monitors and the output file.

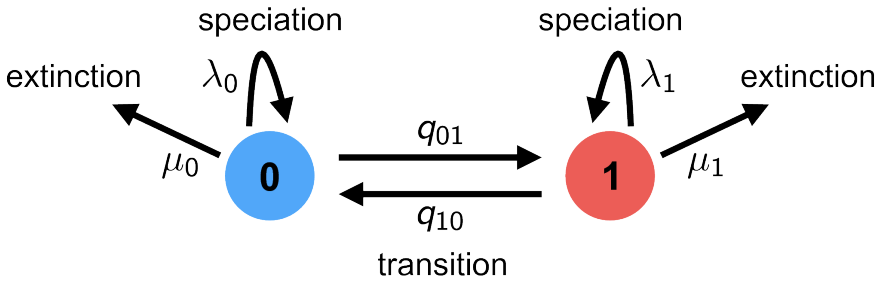

Figure 3: A schematic overview of the BiSSE model. Each lineage has a binary trait associated with it, so it is either in state 0 (blue) or state 1 (red). When a lineage is in state 0, it can either (a) speciate with rate $\lambda 0$ which results into two descendant lineage both being in state 0; (b) go extinct with rate $\mu 0$; or (c) transition to state 1 with rate q01. The same types of events are possible when a lineage is in state 1 but with rates $\lambda 1$, $\mu 1$, and q10, respectively. Image from urlhttps://revbayes.github.io/tutorials/sse/bisse-intro.html

```
mymodel = model(birth_rate)
monitors.append( mnModel(filename="output/primates_BD.log",printgen=10,
separator = TAB))
monitors.append(mnScreen(printgen=1000, birth_rate, death_rate))
```

3.2.2 Running the BD model

Let's run the model:

```
mymcmc = mcmc(mymodel, monitors, moves, nruns=2, combine="mixed")
mymcmc.run(generations=50000)
```

Question 2
Take a look at your log file in tracer.
A) What are the estimates for speciation rate and extinction rate? Send a screenshot of the Marginal Probability distributions for both parameters (you can select multiple statistics at once in Tracer). B) In this exercise we modeled one constant speciation rate and one constant extinction rate. Do you think this a valid model for diversification? How would you like to alter this model to make it more biologically reasonable? Check out tutorials available for RevBayes under the Diversification Rate Estimation subheading: https://revbayes.github.io/tutorials/ Is your altered model described in one of these tutorials?

4 State-dependent diversification with BiSSE

State-dependent speciation and extinction (SSE) models are a birth-death process where the diversification rates are dependent on the state of an evolving character. The original model of this type considered a binary character (a trait with two discrete state values) and it is called BiSSE, (Maddison et al. 2007). Several variants have also been developed for other types of traits (FitzJohn 2010; Goldberg et al. 2011; Goldberg and Igić 2012; Magnuson-Ford and Otto 2012; FitzJohn 2012; Beaulieu and O'Meara 2016; Freyman and Höhna 2018). For practical reasons, we will focus on the BiSSE model in this lab (Figure 3).
The BiSSE model assumes two discrete states (i.e., a binary character), and that the state of each extant species is known (i.e., the discrete-valued character is observed). The general

approach adopted by BiSSE and related models is to derive a set of ordinary differential equations (ODEs) that describe how the probability of observing a descendant clade changes along a branch in the observed phylogeny. Each equation in this set describes how the probability of observing a clade changes through time if it is in a particular state over that time period. Computing the likelihood proceeds by establishing an initial value problem. We initialize the procedure by observing the character states of some lineages, generally the tip states. Then starting from those probabilities (e.g., species X has state 0 with probability 1 at the present), we describe how those probabilities change over time (described by the ODEs), working our way back until we have computed the probabilities of observing that collection of lineages at some earlier time (e.g., the root).

4.1 The data

For this exercise we will use the same tree that we already know well: 233 species of primates, plus an additional file on a binary state mapped for all the terminals: primatesactivity.nex. Explore the file in a text editor. The state) indicates that this species is diurnal, while the state 1 codes for a nocturnal activity period. We may have an a priori hypothesis that diurnal species have higher rates of speciation. Estimating the diversification rates of lineages associated with that trait will allow us to explore this hypothesis.
Let's start by reading the data and extracting the number of taxa:

```
observed_phylogeny <- readTrees("data/primates_tree.nex")[1]
data <- readCharacterData("data/primates_activity_period.nex")
num_taxa <- observed_phylogeny.ntips()
```

As always, we need to initialize a variable for our vector of moves and monitors.

```
moves    = VectorMoves()
monitors = VectorMonitors()
```

Finally, create a helper variable that specifies the number of states that the observed character has. Using this variable allows us to easily change our script and use a different character with a different number of states, essentially changing our model from BiSSE (Maddison et al. 2007) to one that allows for more than 2 states–i.e., the MuSSE model (FitzJohn 2012).

```
NUM_STATES = 2
```

4.2 Specifying the model

The basic idea behind the model in this example is that speciation and extinction rates are dependent on the state of a binary character, and the character transitions between the two possible states (Maddison et al. 2007). We start by specifying our character state-specific speciation and extinction rate parameters. Because we will use the same prior for each rate, it's easy to specify all of them in a for-loop. We will use a log-uniform distribution as a hyper prior on the speciation and extinction rates. The loop also allows us to apply moves to each of the rates we are estimating and create a vector of deterministic nodes representing the rate of diversification ($\lambda - \mu$) associated with each character state.

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

### Create a loguniform distributed variable for the diversification rate
```

```
    speciation[i] ~ dnLoguniform( 1E-6, 1E2)
    moves.append( mvScale(speciation[i],lambda=0.20,tune=true,weight=3.0) )

    ### Create a loguniform distributed variable for the turnover rate
    extinction[i] ~ dnLoguniform( 1E-6, 1E2)
    moves.append( mvScale(extinction[i],lambda=0.20,tune=true,weight=3.0) )

    ###Calculate the diversification rate
    diversification[i] := speciation[i] - extinction[i]

}
```

Now, if you type `speciation` or `extinction` you should get a vector of values indicating the prior rates for each character state.
Next we specify the transition rates between the states 0 and 1: q01 and q10. As a prior, we choose that each transition rate is drawn from an exponential distribution with a mean of 10 character state transitions over the entire tree. This is reasonable because we use this kind of model for traits that transition not-infrequently, and it leaves a fair bit of uncertainty. Note that we will actually use a for-loop to instantiate the transition rates so that our script will also work for non-binary characters.

```
rate_pr := observed_phylogeny.treeLength() / 10

for ( i in 1:(NUM_STATES*(NUM_STATES-1)) ) { #for each pair of states
    transition_rates[i] ~ dnExp(rate_pr) #calculate the transition rates
    moves.append( mvScale(transition_rates[i],lambda=0.20,tune=true,weight=3.0) )
}

transition rates #print the obtained transition rates
```

Here, rate[1] is the rate of transition from state 0 (diurnal) to state 1 (nocturnal), and rate[2] is the rate of going from nocturnal to diurnal.

Finally, we put the rates into a matrix, because this is what's needed by the function for the state-dependent birth-death process.

```
    rate_matrix := fnFreeK( transition_rates, rescaled=false)
```

Create a variable with the prior probabilities of each rate category at the root. We are using a flat Dirichlet distribution as the prior on each state. In this case we are actually estimating the prior frequencies of the root states. There has been some discussion about this in FitzJohn et al. (2009). You could also fix the prior probabilities for the root states to be equal (generally not recommended), or use empirical state frequencies.

```
    root_state_freq ~ dnDirichlet( rep(1, NUM_STATES) )
    moves.append( mvDirichletSimplex(root_state_freq,tune=true,weight=2) )
```

As in the previous examples we are taking into account the probability of sampling in the present, assuming that the total number of extant taxa are 367.

```
sampling <- num_taxa / 367
```

The birth-death process also depends on time to the most-recent-common ancestor–i.e., the root. In this exercise we use a fixed tree and thus we know the age of the tree.

```
root_age <- observed_phylogeny.rootAge()
```

Now we have all of the parameters we need to specify the full character state-dependent birth-death model. We initialize the stochastic node representing the time tree and we create this node using the dnCDBDP() function. And clamp it to our observed data.

```
timetree ~ dnCDBDP( rootAge           = root_age,
                    speciationRates   = speciation,
                    extinctionRates   = extinction,
                    Q                 = rate_matrix,
                    pi                = root_state_freq,
                    rho               = sampling )

timetree.clamp( observed_phylogeny ) #clamp tree data
timetree.clampCharData( data ) #clamp trait data
```

Finally, we create a workspace for our model:

```
mymodel = model(timetree)
```

4.3 Running the MCMC analysis

We set the outputfile and the screen monitors:

```
monitors.append( mnModel(filename="output/primates_activTime_BiSSE_mcmc.log",
printgen=1) )
monitors.append( mnFile(filename="output/primates_BiSSE_dataset.trees",
printgen=1, timetree) )
monitors.append( mnScreen(printgen=10, speciation, extinction) )
```

Finally, we can set up the MCMC function and run the anaysis:

```
mymcmc = mcmc(mymodel, monitors, moves, nruns=2, combine="mixed")
mymcmc.run(generations=5000, tuningInterval=200)
```

4.4 Making beautiful graphics in R

The RevBayes team is currently developing an R package that allows to process output data from RevBayes. In this section we will run a code for visualizing our results of the BiSSE analysis. In the lab_files folder, open the R script named **R_plots.R** and follow the steps. The code on this R script will export the results as PDF.

Question 3
Look at the PDF file. Are the rates state-dependent? What does this tell us about the trait we used in our analysis? Send me the figure of these plots.

Send me your answers to Questions 1 - 3 and the requested screenshots/ figures.