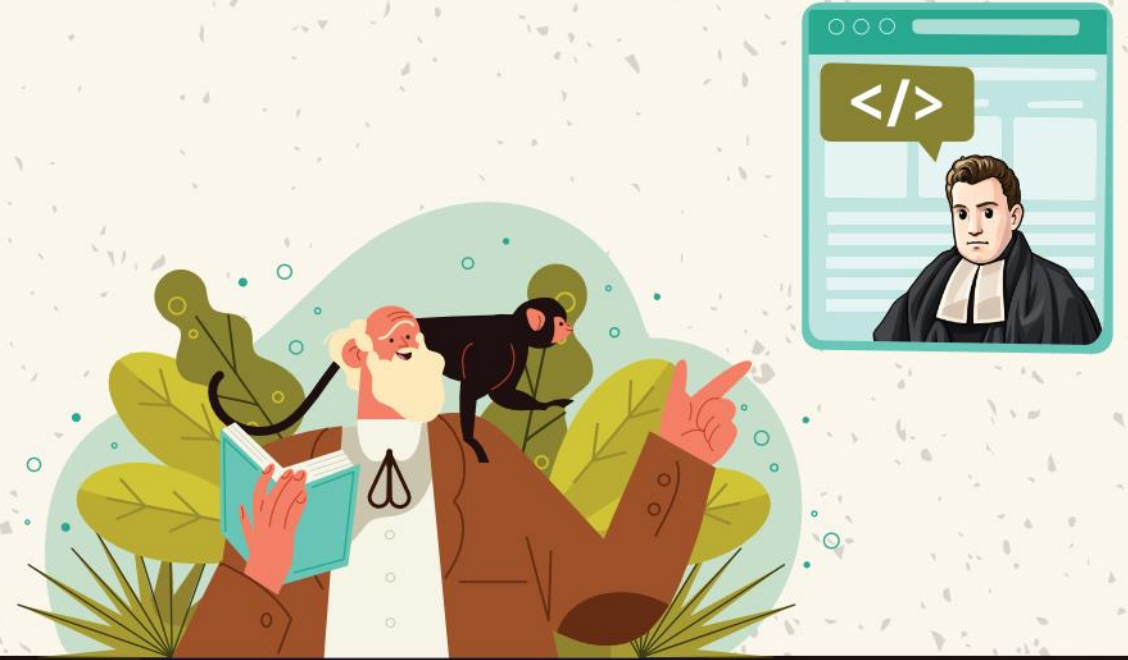


RevBayes para humanos:

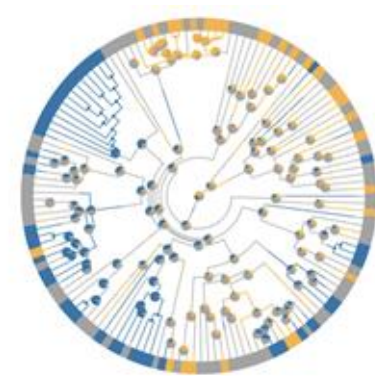
Entender los modelos
para entender el código



¡Bienvenidas y
bienvenidos!



Juan Diego Rodríguez-Rodríguez

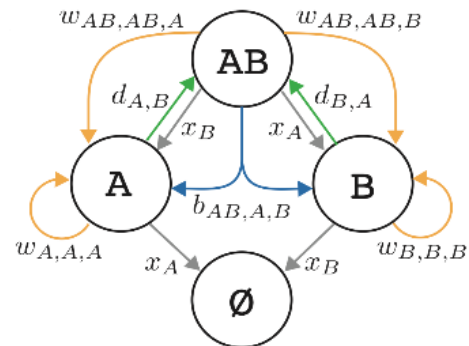




Ixchel González Ramírez

<https://ixchelgzlzl.github.io>

<https://ib-unisse.github.io>



Anagenetic events
(d)ispersal
e(x)tinction

Cladogenetic events
(w)ithin-region
speciation
(b)etween-region
speciation'



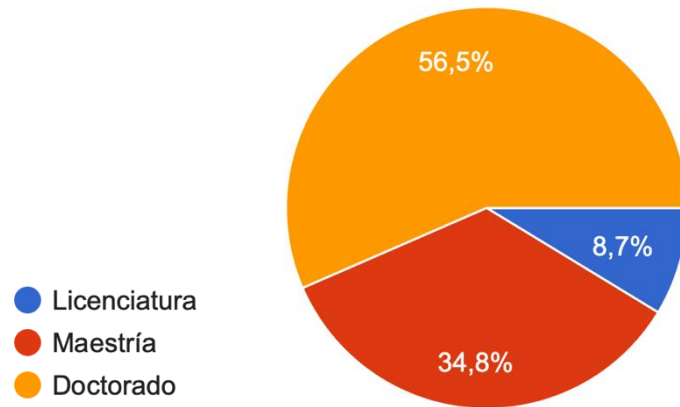
Objetivos del taller

1. Introducir **conceptos fundamentales** de estadística bayesiana.
2. Discutir cómo la **estadística bayesiana se aplica** al estudio de procesos evolutivos.
3. Introducir ideas de **modelado evolutivo**.
 - Ejemplificar el uso de **modelos gráficos**.
 - Familiarizarse con el **código de RevBayes**.
4. Practicar código y modelado evolutivo sin preocuparse por detalles técnicos de instalación de software.

El grupo:

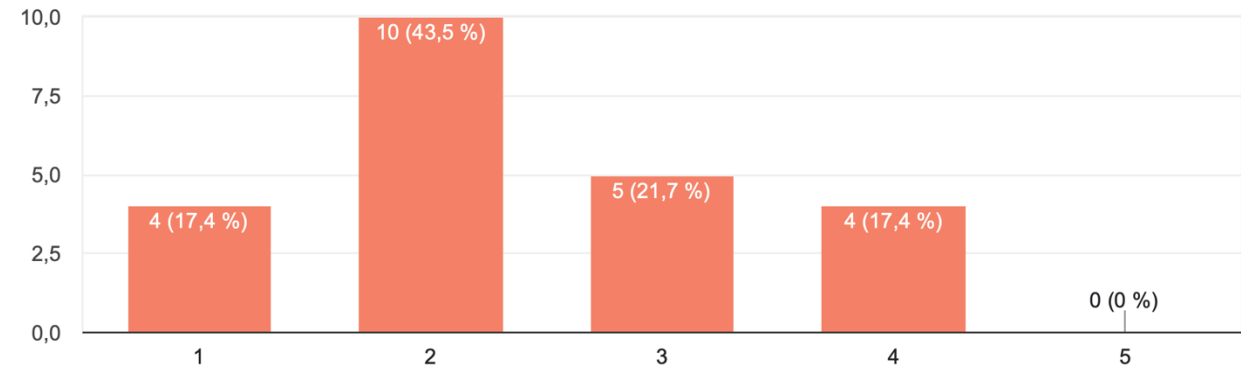
Nivel académico

23 respuestas



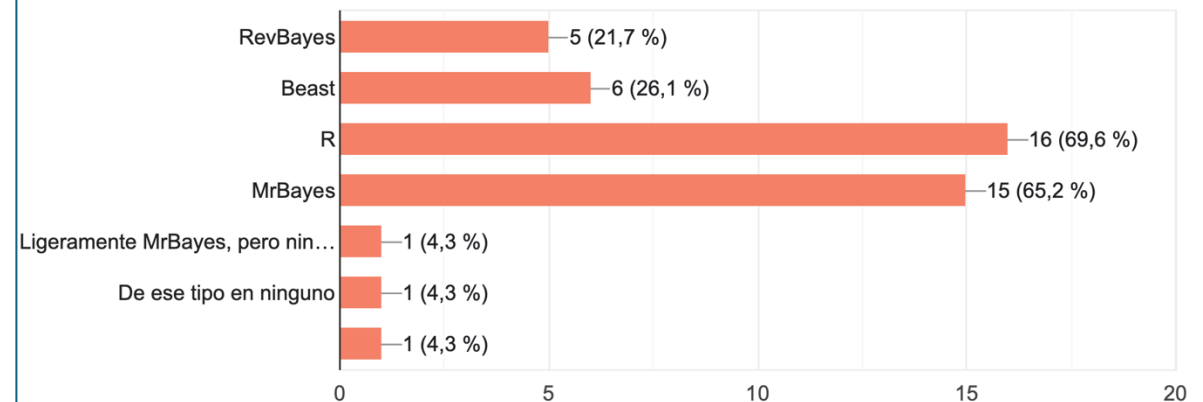
¿Qué tanta experiencia tienes con temas de estadística bayesiana?

23 respuestas



¿En qué softwares tienes experiencia?

23 respuestas



Actividades



1. Presentación del taller.
2. **Parte 1**- Introducción al pensamiento bayesiano, y su uso en filogenética.

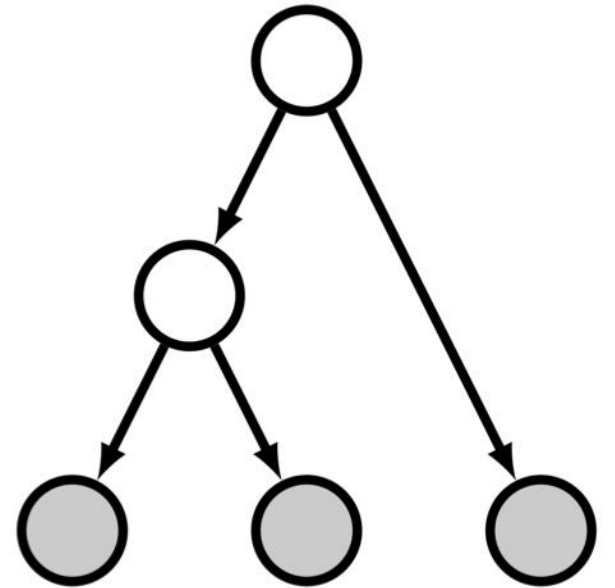
----Cafecito---

3. **Parte 2** - Introducción al modelado evolutivo con RevBayes.
4. **Actividad I**: construye tu modelo.

----Comida---

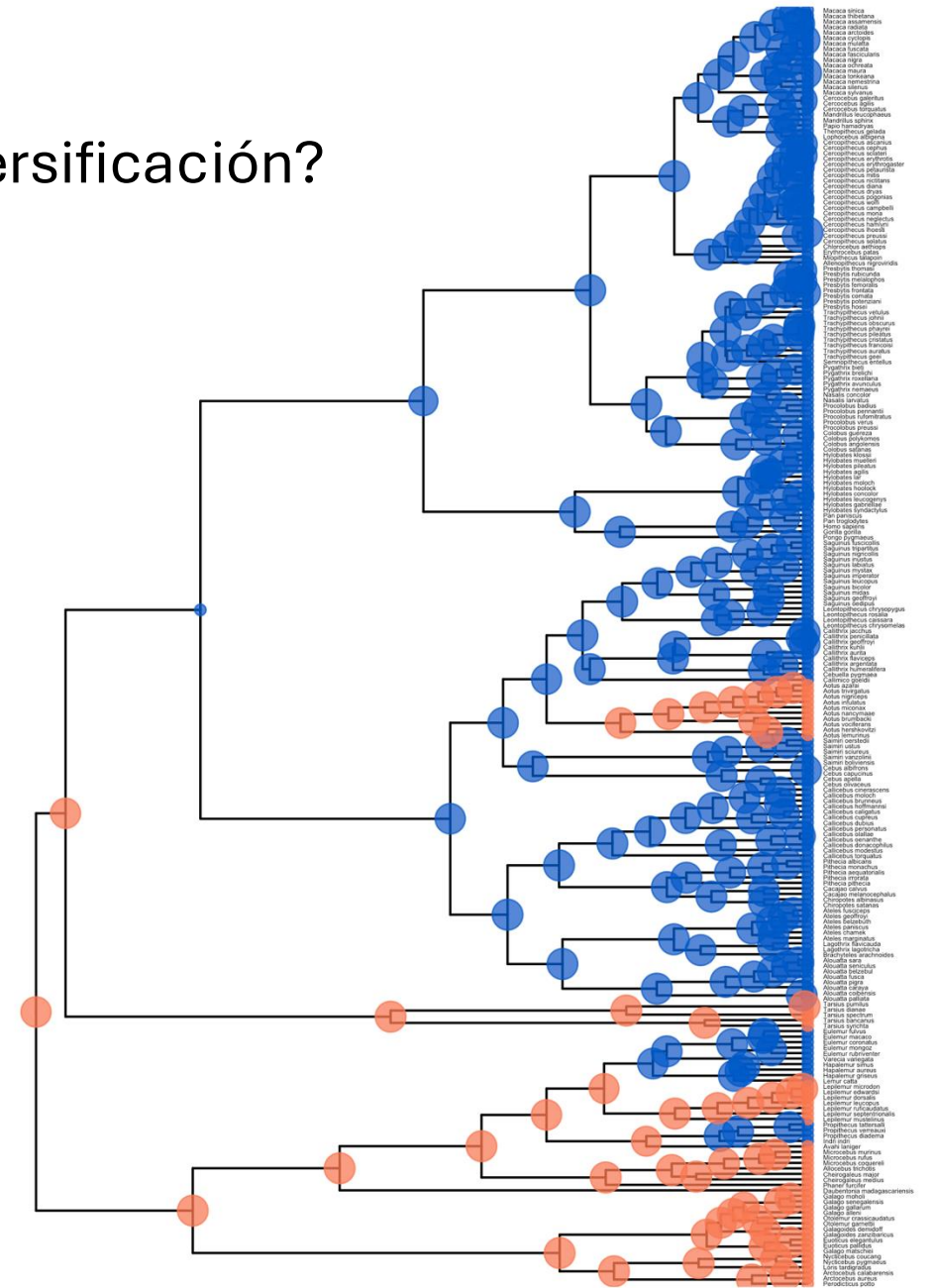
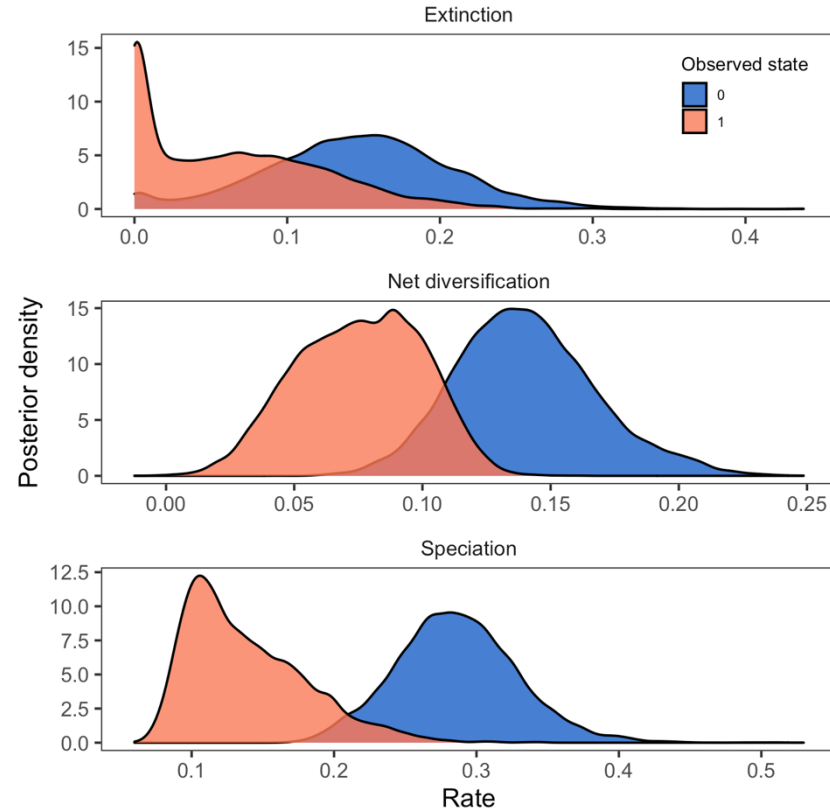
5. **Actividad II**: descifra el código.
6. Live-coding y preguntas.

Introducción al modelado evolutivo con RevBayes.



Motivación: Entender patrones evolutivos

Por ejemplo, ¿el carácter de mi interés afecta la diversificación?



Recordatorio (de hace 10 min)

- La inferencia bayesiana trata a los parámetros de interés como **variables aleatorias**.
 - Cada variable aleatoria (parámetro) puede ser estimada dada un modelo y datos.
- La inferencia bayesiana se basa en el teorema de Bayes: Prior, verosimilitud, verosimilitud marginal.

$$P(\theta | X) = \frac{P(X | \theta)P(\theta)}{P(X)}$$

- La **MCMC** es un algoritmo que permite **aproximar la probabilidad posterior** sin necesidad de calcular verosimilitudes marginales.

Ventajas prácticas de la inferencia bayesiana:

- Incorpora **medidas de incertidumbre** directamente
 - Probabilidades posteriores, intervalos de credibilidad, etc.
- Los modelos modulares son fáciles de implementar mediante el teorema de Bayes.
 - Se pueden incorporar **distintas fuentes de información**.
 - Es relativamente sencillo **implementar modelos de inferencia conjunta**.

Sin embargo, hay un **trade-off**:

- Al igual que en ML, los resultados son **sensibles** al modelo.
- Especificar **priors**, puede ser complicado y tener efectos no anticipados.
- Se usan MCMCs, y estas pueden **consumir mucho tiempo y en ocasiones fallar**.

Podemos realizar nuestra inferencia bayesiana con varios programas:



RevBayes



BEAST

Bayesian Evolutionary Analysis Sampling Trees



Mr. Bayes

¿Por qué RevBayes?



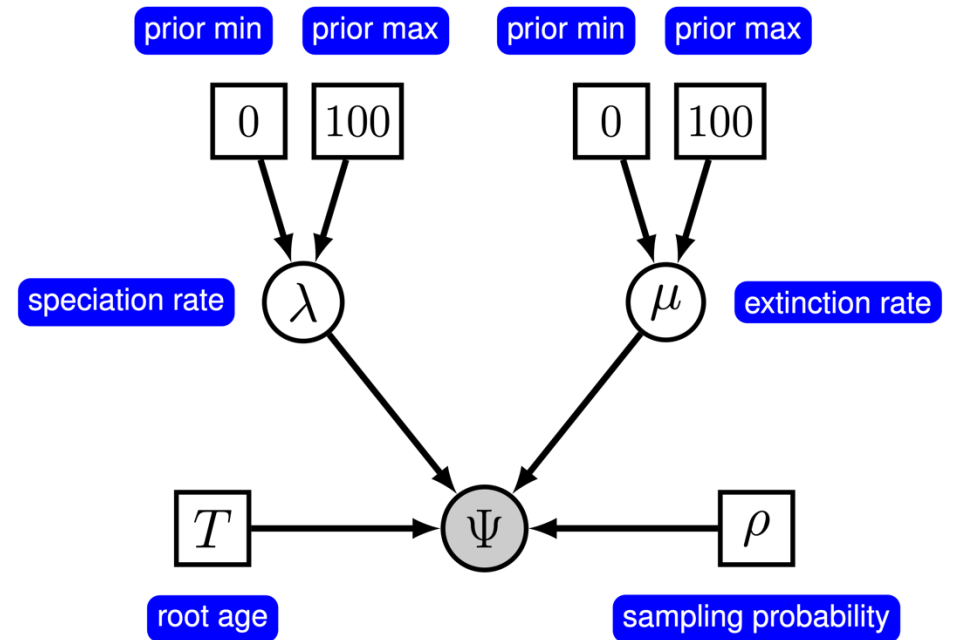
```
19
20
21 # CONSTANT RATE DIVERSIFICATION MODEL
22
23 # empirical prior on the diversification rate
24 diversification_prior_mean <- ln(total_taxa) / origin_time
25 diversification_prior_sd <- H
26
27 # the diversification rate
28 diversification ~ dnLognormal( ln(diversification_prior_mean) - diversification_prior_sd * diversification,
29 moves.append( mvScale(diversification, weight = 1.0) )
30
31 # the relative extinction rate
32 relext ~ dnUniform(0, 1)
33 moves.append( mvSlide(relext, weight = 1.0) )
34
35 # transform to real parameters
36 lambda := abs(diversification / (1 - relext))
37 mu := abs(lambda * relext)
38
39
40 # CONSTANT-RATE FOSSILIZATION MODEL
41
42 # empirical prior on the fossilization rate
43 fossilization_prior_mean <- num_fossils * diversification_prior_mean / (exp(diversification_prior_mean * o
44 fossilization_prior_sd <- 2 * H
45
46 # the fossilization rate
47 psi ~ dnLognormal( ln(fossilization_prior_mean) - fossilization_prior_sd * fossilization_prior_sd * 0.5, f
48 # add moves
49 moves.append( mvScale(psi, weight = 1.0) )
50
51
52 # make the FBD tree
53
54 timetree ~ dnFBDP(
55 | | originAge = origin_time,
```

- Lenguaje de programación enfocado en filogenética.
- Basado en R
- Dinámico; permite construir modelos complejos y especificar tus propios priors
- Te obliga a pensar en tu modelo y en tus priors.

¿Qué es un modelo evolutivo?

¿Qué es un modelo evolutivo?

Una representación de cómo entendemos un **proceso**. Los **parámetros** del modelo son características del proceso que se pueden cuantificar. En el modelo establecemos explícitamente **relaciones de dependencia** entre los parámetros.



En RevBayes

- El modelo gráfico nos permite plasmar nuestro entendimiento de como modelamos la evolución.
- Entender el modelo gráfico nos permite visualizar la estructura del código.

Simple Diversification Rate Estimation

Comparing different constant-rate models of lineage diversification

Sebastian Höhna and Tracy Heath

Last modified on March 9, 2022

Overview

Prerequisites

- [Getting Started with RevBayes and Rev Language Syntax](#)
- [Introduction to Markov chain Monte Carlo \(MCMC\) Sampling](#)
- [Introduction to Diversification Rate Estimation](#)

Table of Contents

- [Estimating Constant Speciation & Extinction Rates](#)
- [Pure-Birth \(Yule\) Model](#)
 - [Read the tree](#)
 - [Specifying the model](#)
 - [Running an MCMC analysis](#)
 - [Exercise 1](#)
- [Estimating the marginal likelihood of the model](#)
 - [Exercise 2](#)
- [Birth-Death Process](#)
 - [Birth rate and death rate](#)
 - [Diversification and turnover](#)
 - [The time tree](#)
 - [Exercise 3](#)

Data files and scripts

Data Files

- [primates.tre](#)
- [primates_tree.nex](#)
- [primates_tree.nex](#)

Scripts

- [mcmc_Yule.Rev](#)
- [ml_Yule.Rev](#)
- [plot_BD_rates.R](#)
- [plot_Yule_rates.R](#)

Other

- [primates.tre](#)
- [primates_tree.nex](#)
- [primates_tree.nex](#)
- [primates_tree.nex](#)

Estimating Constant Speciation & Extinction Rates

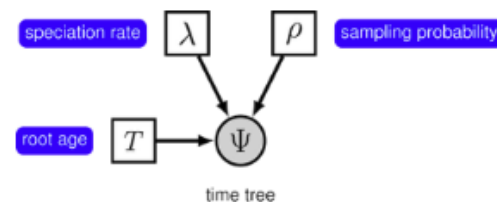
This tutorial describes how to specify basic birth-death models in RevBayes. Specifically, we will use the pure-birth (Yule; [Yule \(1925\)](#)) process and the constant-rate birth-death process ([Kendall 1948](#); [Thompson 1975](#); [Nee et al. 1994](#)). The probabilistic graphical model is given for each component of this tutorial. After each model is specified, you will estimate speciation and extinction rates using Markov chain Monte Carlo (MCMC). Finally, you will estimate the marginal likelihood of the model and evaluate the relative support using Bayes factors.

You should read first the [Introduction to Diversification Rate Estimation](#) tutorial, which explains the theory and gives some general overview of diversification rate estimation.

Pure-Birth (Yule) Model

In this section, we will walk through specifying a pure-birth process model and estimating the speciation rate. Then, we will continue by estimating the marginal likelihood of the model so that we can perform model comparison. The section about the birth-death process will be less detailed because it will build up on this section.

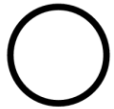
The simplest branching model is the *pure-birth process* described by [Yule \(1925\)](#). Under this model, we assume at any instant in time, every lineage has the same speciation rate, λ . In its simplest form, the speciation rate remains constant over time. As a result, the waiting time between speciation events is exponential, where the rate of the exponential distribution is the product of the number of extant lineages (n) at that time and the speciation rate: $n\lambda$ ([Yule 1925](#); [Aldous 2001](#); [Höhna 2015](#)). The pure-birth branching model does not allow for lineage extinction (*i.e.*, the extinction rate $\mu = 0$). 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 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 ([Figure 1](#)).



Tipos de nodos en RevBayes:



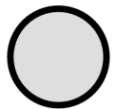
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo constante

```
# Definir el número de taxa  
n_taxa <- 88
```

- Se define con una flecha.
- Es una variable con un valor fijo.
- La MCMC no puede cambiar nodos fijos.
- Comunmente se usa para definir características de los datos o hiperpriors.

Tipos de nodos en RevBayes:



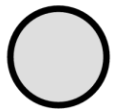
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo estocástico:

```
### diversification = birth_rate - death_rate
### assume an exponential prior distribution
### and apply a scale proposal
diversification ~ dnExponential(10.0)
```

- Se define con “~”.
- Es una variable que se define aleatoriamente de una distribución.

Tipos de nodos en RevBayes:



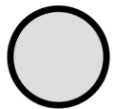
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo estocástico:

```
### diversification = birth_rate - death_rate  
### assume an exponential prior distribution  
### and apply a scale proposal  
diversification ~ dnExponential(10.0)
```

- Se define con “~”.
- Es una variable que se define aleatoriamente de una distribución.

Tipos de nodos en RevBayes:



a) Constant node



b) Stochastic node

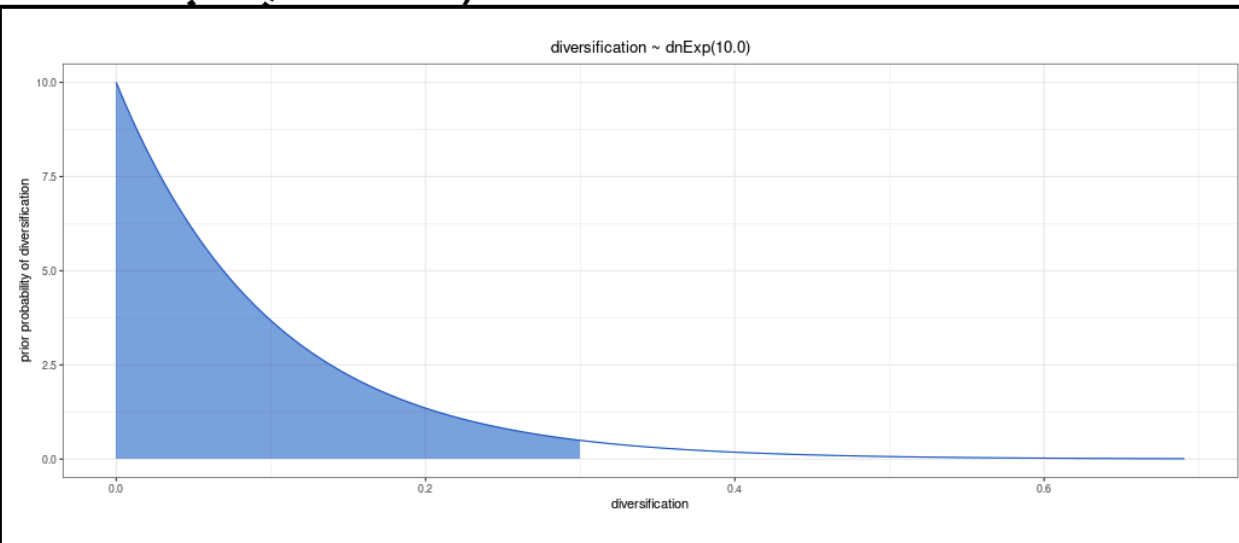


c) Deterministic node

Nodo estocástico:

```
### diversification = birth_rate - death_rate  
### assume an exponential prior distribution  
### and apply a scale proposal  
diversification ~ dnExponential(10.0)
```

- Se define con “~”.
- Es una variable que se define aleatoriamente de una distribución.



Tipos de nodos en RevBayes:



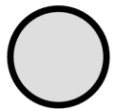
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo estocástico:

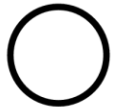
```
### diversification = birth_rate - death_rate  
### assume an exponential prior distribution  
### and apply a scale proposal  
diversification ~ dnExponential(10.0)
```

- Se define con “~”.
- Es una variable que se toma aleatoriamente de una distribución.
- La MCMC puede actuar sobre nodos estocásticos.
- Comunmente son parámetros de interés que queremos estimar.

Tipos de nodos en RevBayes:



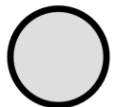
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo determinista:

```
# draw the log of the mean from a uniform distribution
mole_clock_rate_mean ~ dnLoguniform(1e-10, 1)

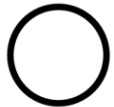
# get the rate on a log scale
mole_clock_rate_mean_log := ln(mole_clock_rate_mean)
```

- Se define con “:=”.
- Es un nodo cuyo valor depende matemáticamente de otro nodo.
- El valor de este nodo se actualiza automáticamente cuando el valor de otra variable cambia.

Tipos de nodos en RevBayes:



a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Nodo “pinzado/clampeado”:

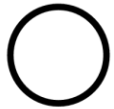
```
# clamp the fixed topology  
timetree.clamp(fixed_tree)
```

- Es un nodo estocástico (que viene de una distribución), pero con una **realización observada**.
- Los nodos pinzados es la manera en la que incluimos datos en el modelo.

Tipos de nodos en RevBayes:



a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

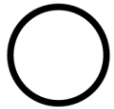
Nodo “pinzado/clampeado”:

```
# definimos la variable estocástica del árbol
timetree ~ dnGLHBDSP(  
  rootAge      = root_age,  
  lambda       = clado_rates,  
  lambdaTimes  = time_bounds,  
  mu           = mu,  
  eta          = Q_DEC,  
  etaTimes     = time_bounds,  
  omega        = clado_probs,  
  omegaTimes   = time_bounds,  
  pi           = rf,  
  rho          = 0.44,  
  condition    = "survival",  
  taxa         = taxa,  
  nStates      = n_states - 1,  
  nProc        = 1,  
  zeroIndex    = TRUE)  
  
# clamp the fixed topology  
timetree.clamp(fixed_tree)
```

Tipos de nodos en RevBayes:



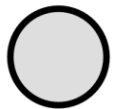
a) Constant node



b) Stochastic node



c) Deterministic node



d) Clamped node
(observed)



e) Plate

Bandeja

```
# tasas para cada rama
for(i in 1:n_branches) {

  # draw the log of the rate
  mole_branch_rates_log[i] ~ dnLogExponential(1 / mole_clock_rate_mean)
  moves.append(_mvSlide(mole_branch_rates_log[i]))
}
```

- Representa un for-loop.
- Lo utilizamos cuando tenemos que repetir una misma asignación de variable.

Ejemplo de un modelo evolutivo

Birth-Death Process

Birth-death process / Proceso de Nacimiento-Muerte

Nee et al. (1994)

- Modelo que describe el Nacimiento (especiación) y muerte (extinción) de los linajes a través del tiempo.
- Este modelo constituye la base de muchos modelos de diversificación.

Birth-death process / Proceso de Nacimiento-Muerte

Birth-death model.

λ - especiación

μ - extinción

ρ - sampling fraction



Birth-death process / Proceso de Nacimiento-Muerte

Birth-death model.

λ - especiación

μ - extinción

ρ - sampling fraction



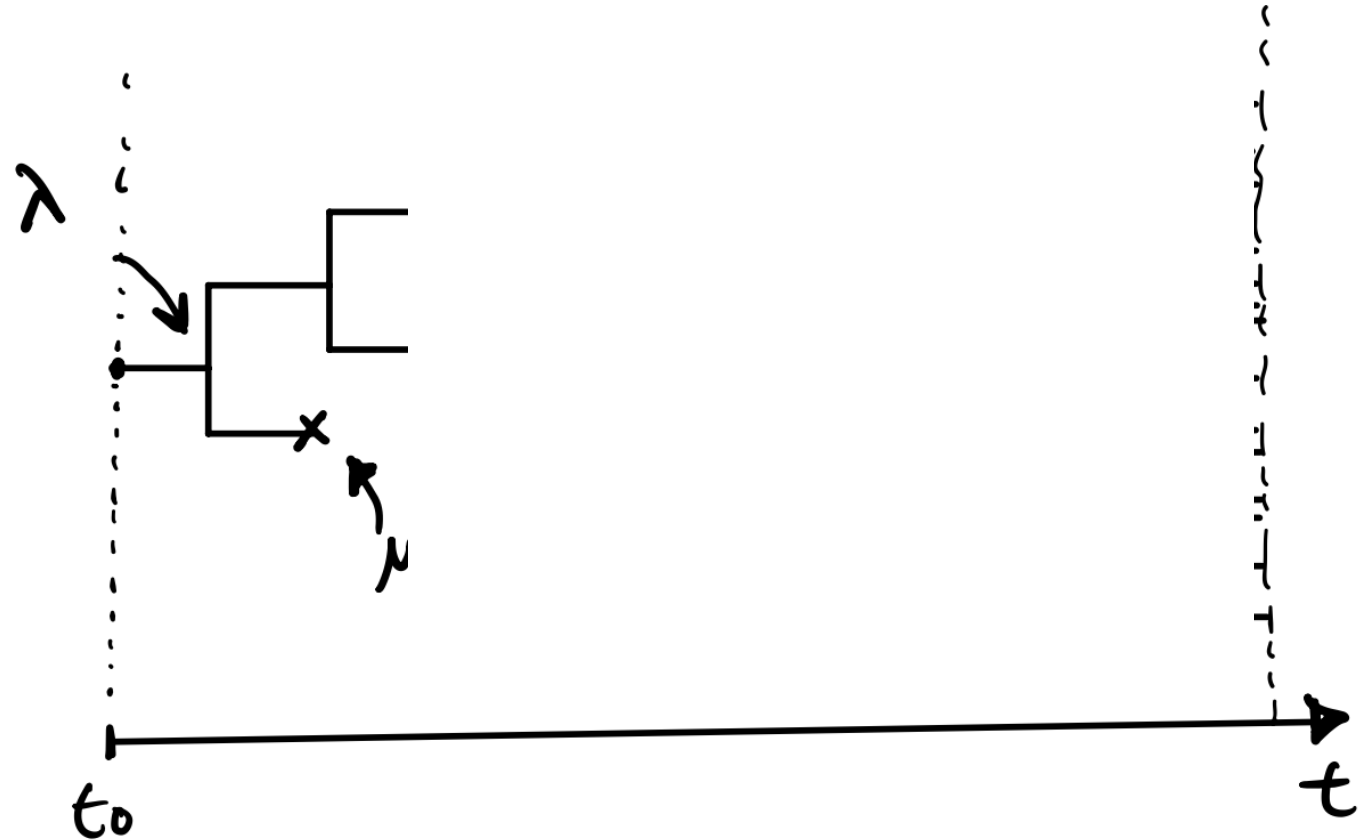
Birth-death process / Proceso de Nacimiento-Muerte

Birth-death model.

λ - especiación

μ - extinción

ρ - sampling fraction



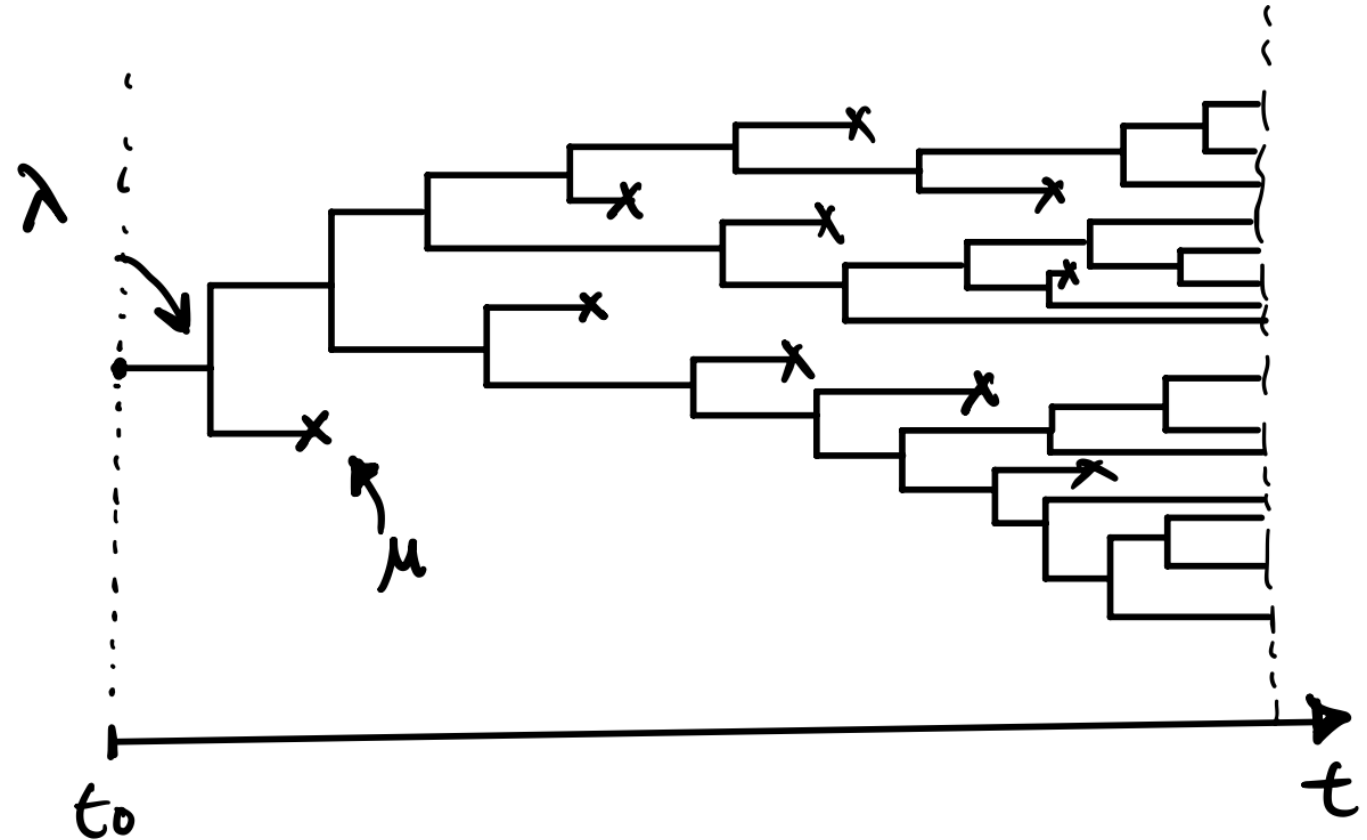
Birth-death process / Proceso de Nacimiento-Muerte

Birth-death model.

λ - especiación

μ - extinción

ρ - sampling fraction



Birth-death process / Proceso de Nacimiento- Muerte

Paso 1

- ¿Qué parámetros definen el proceso?

Paso 2

- ¿Qué tipo de nodo es cada parámetro?

Paso 3

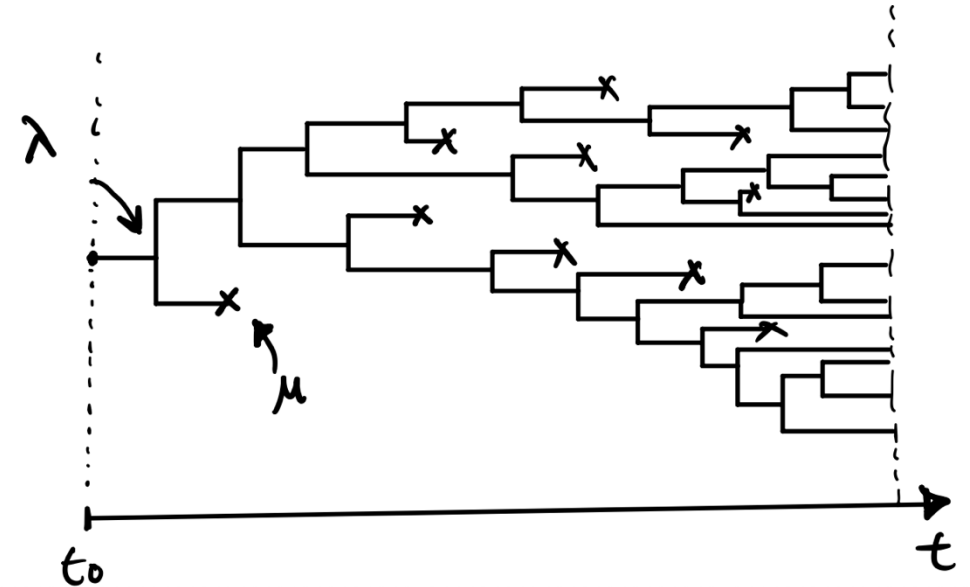
- Definir mis priors.

Birth-death model.

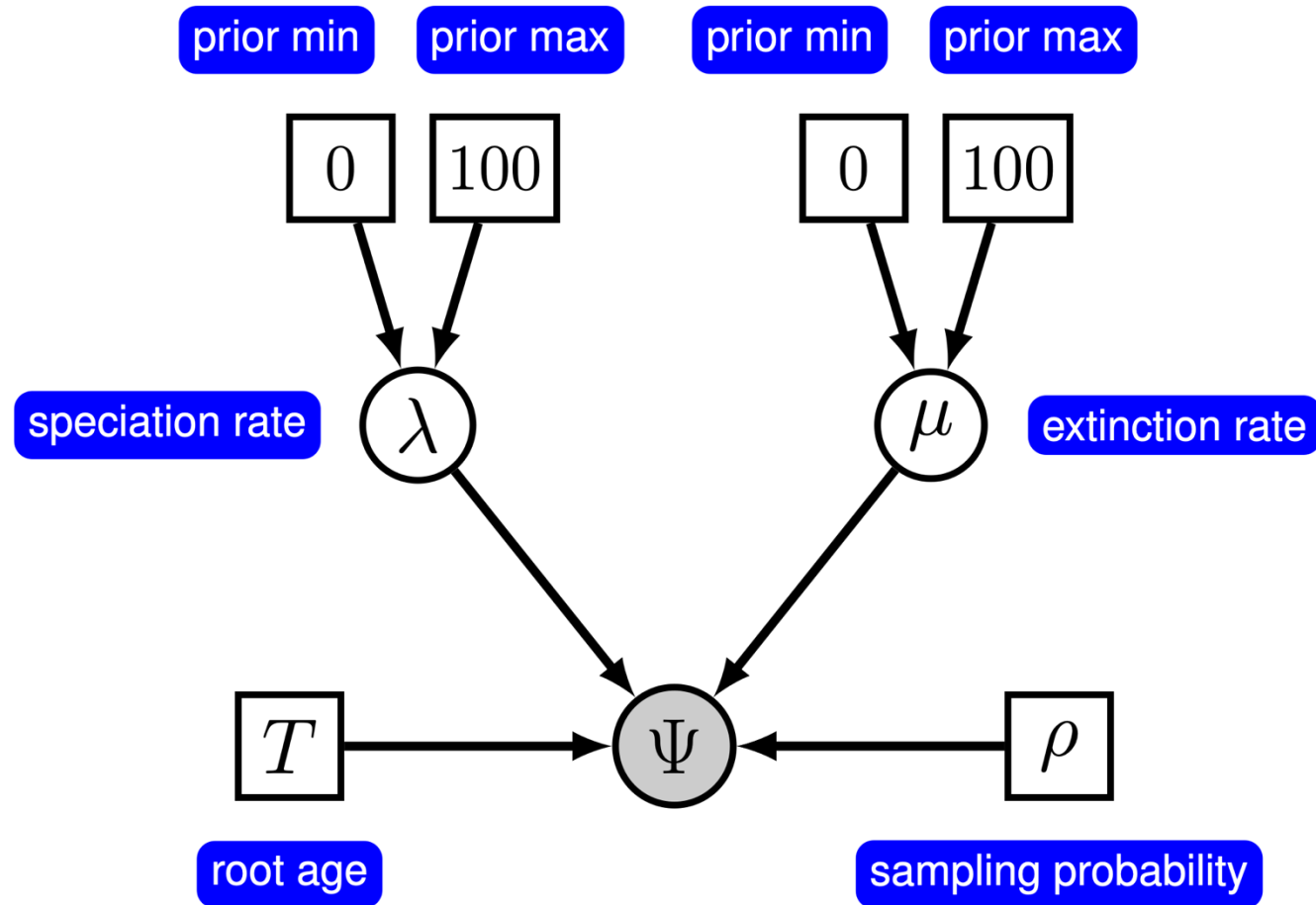
λ - especiación

μ - extinción

ρ - sampling fraction



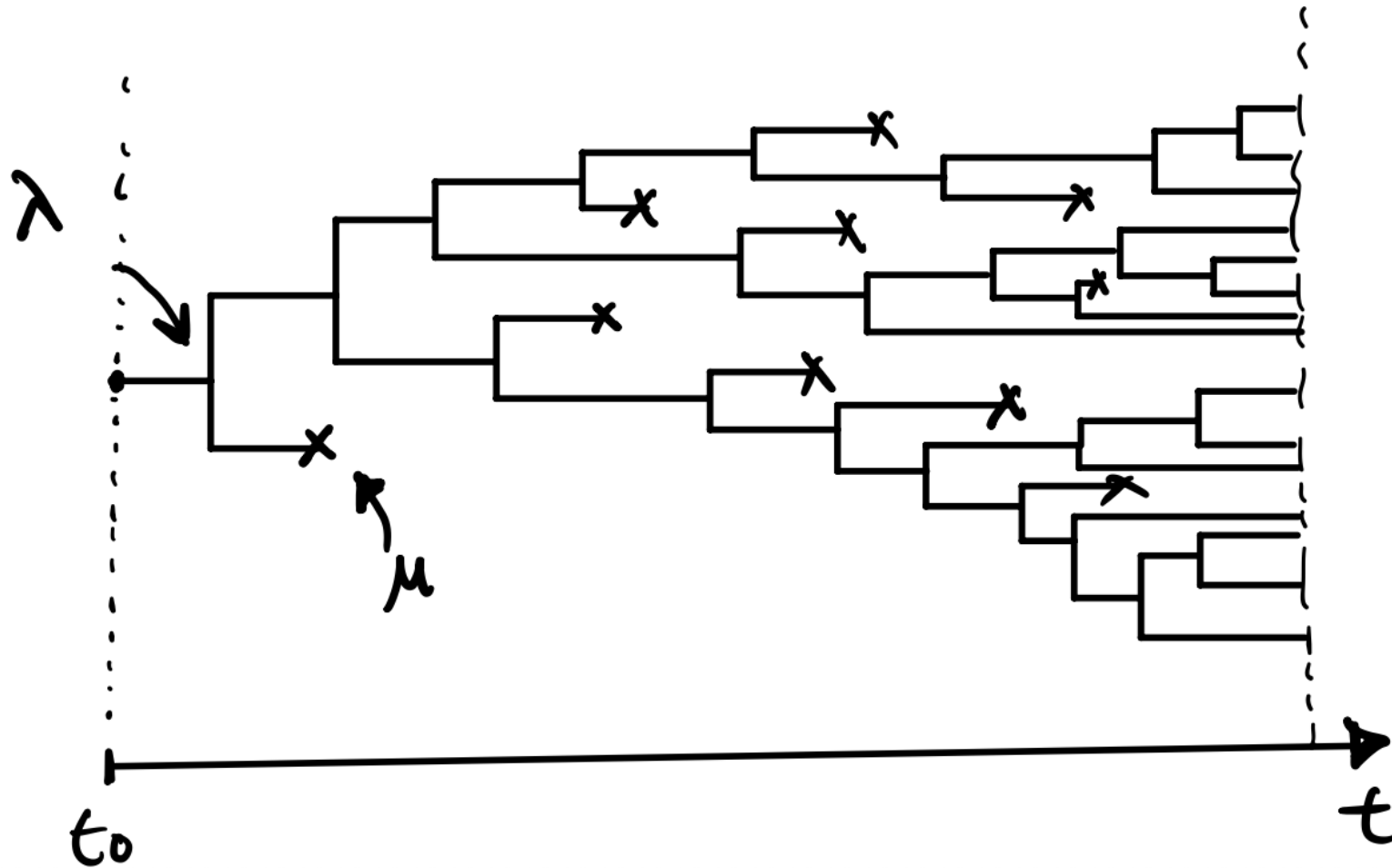
Birth-death process / Proceso de Nacimiento-Muerte



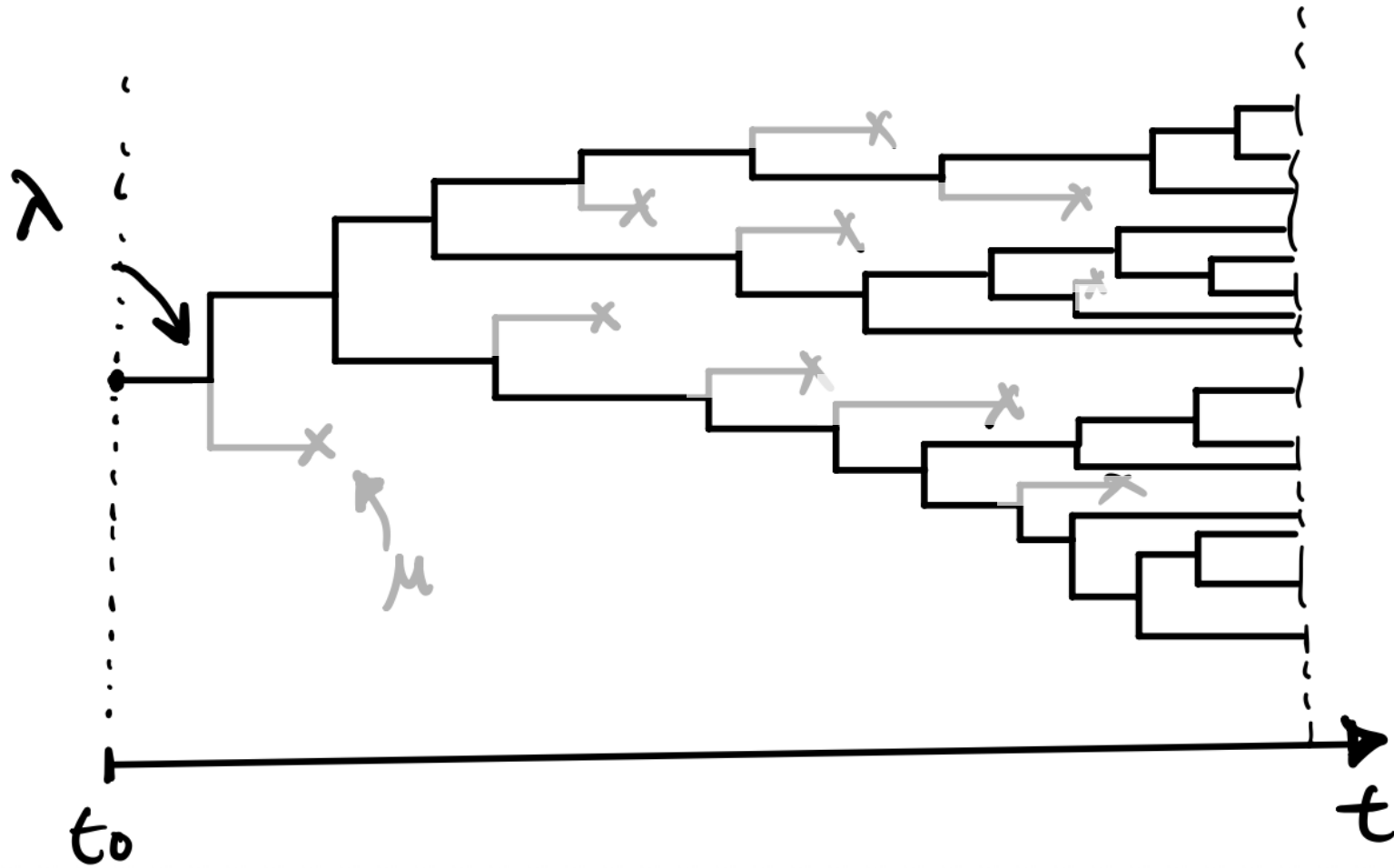
Birth-death process / Proceso de Nacimiento-Muerte

Vamos a transformar el modelo gráfico a código de RevBayes

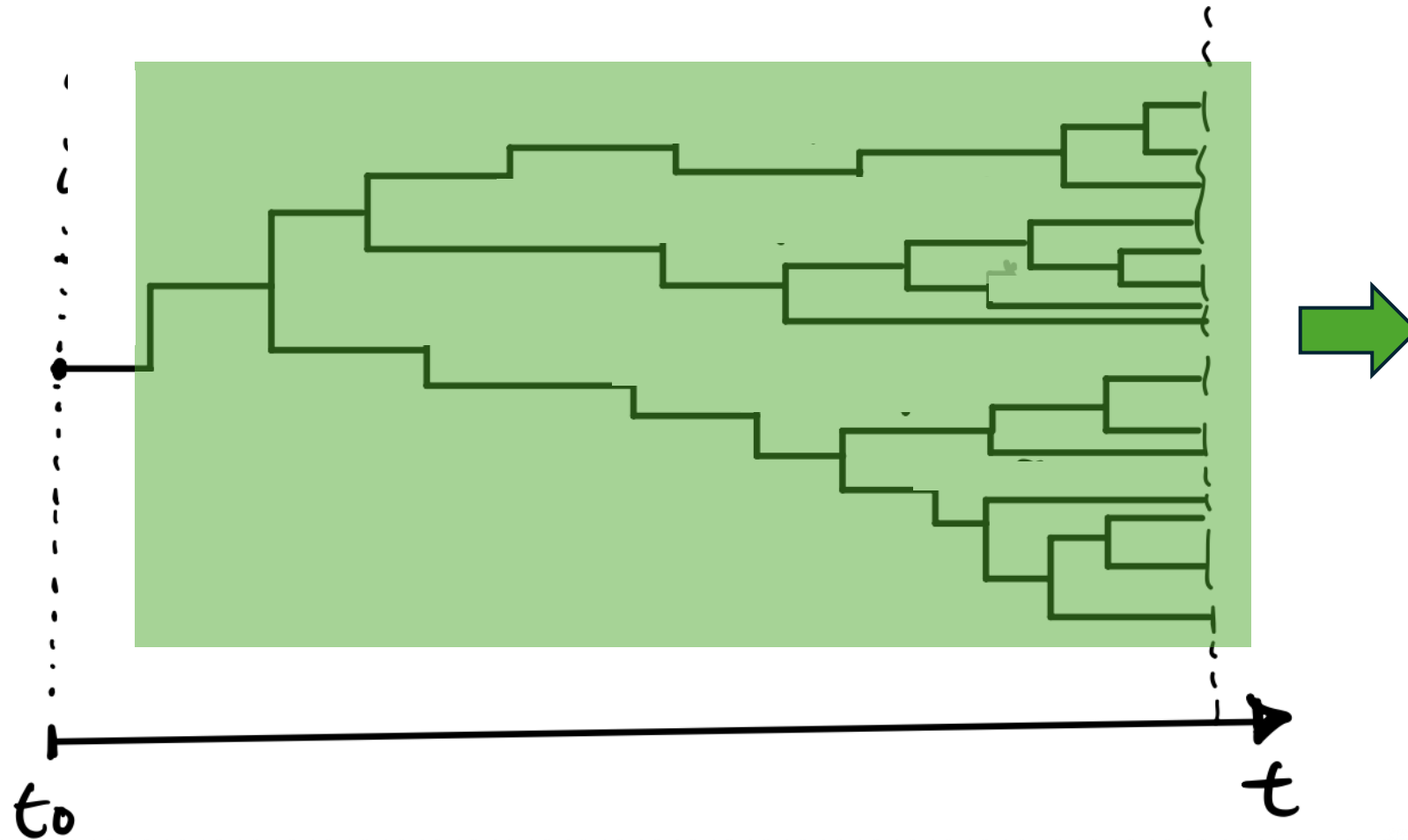
Birth-death process / Proceso de Nacimiento- Muerte



Birth-death process / Proceso de Nacimiento- Muerte



Birth-death process / Proceso de Nacimiento- Muerte



Simulación de distribución de árboles

Actividad I. Modelo gráfico de BiSSE

Especifica el modelo gráfico para BiSSE. Pasos sugeridos.

1. Identifica tus parámetros
2. Qué tipo de nodos son
3. Cómo se relacionan

Pista: El modelo tiene dos piezas de LEGO: la evolución del carácter y el modelo BD



Actividad I. Modelo gráfico de BiSSE

Especifica el modelo gráfico para BiSSE. Pasos sugeridos.

1. Identifica tus parámetros
2. Qué tipo de nodos son
3. Cómo se relacionan

```
timetree ~ dnCDBDP( rootAge      = root_age,  
                    speciationRates = speciation,  
                    extinctionRates = extinction,  
                    Q              = rate_matrix,  
                    pi             = root_state_freq,  
                    delta          = 1.0,  
                    rho            = sampling,  
                    condition      = "time")
```



Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

- 1) Definimos el modelo.
- 2) Leemos datos y características de nuestro sistema de estudio.
- 3) Especificamos características de la MCMC.
- 4) Le damos instrucciones a rb de cómo operar.

Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

1) Definimos el modelo.

```
# tasa de especiación
birth_rate ~ dnUniform(0, 100.0)

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

Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

2) Leemos datos y características de nuestro sistema de estudio.

```
# Cargar la filogenia desde el archivo NEXUS  
T <- readTrees("../docs/u1_PatDiv/subarbol_ingroup.nex") [1]  
  
# Obtener la lista de taxones en la filogenia  
taxa <- T.taxa()
```

Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

3) Especificamos características de la MCMC.

MOVIMIENTOS

```
# Inicializar un vector vacío para los movimientos (moves)  
moves = VectorMoves()  
  
# Agregar un movimiento MCMC para la tasa de especiación  
moves.append( mvScale(birth_rate, lambda=1.0, tune=true, weight=3.0) )
```

Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

3) Especificamos características de la MCMC.

MONITORES

```
# Crear un vector vacío de monitores
moves = VectorMonitors()

# Monitor para registrar los estados del modelo en un archivo de salida
monitors.append( mnModel(filename="output/diversification_Yule.log", printgen=10, separator=TAB) )

# Monitor para imprimir la tasa de especiación en la pantalla cada 1000 generaciones
monitors.append( mnScreen(printgen=1000, birth_rate) )
```

Más detalles del código de RevBayes

Cuando escribimos código de RevBayes:

3) Especificamos características de la MCMC.

CARACTERÍSTICAS DE LA MCMC

```
# Crear el objeto de modelo  
mymodel = model(birth_rate)  
  
# Inicializar el MCMC con dos cadenas combinadas en un solo análisis  
mymcmc = mcmc(mymodel, monitors, moves, nruns=1)  
  
# Correr una seccion de burnin en donde los movimientos se optimizan  
mymcmc.burnin(generations=2000, tuningInterval=10)  
  
# Ejecutar el MCMC por 50,000 generaciones  
mymcmc.run(generations=50000)
```

Actividad II. Descifra el código

1. Utiliza tres colores diferentes para colorear:

- Parametrización del modelo.
- Definición de datos.
- Instrucciones a RevBayes acerca de la MCMC.

2) Encuentra:

- un nodo estocástico
- un nodo determinista
- un nodo pinzado
- un “plate”
- una linea donde se define un prior
- una linea donde se agrega un movimiento

