Phylogenetic Inference using `RevBayes`

Total-evidence Dating under the FBD Model

Tracy A. Heath, April Wright, and Walker Pett

1 Introduction

?

1.1 Models

1.1.1 Sequence Evolution

Point to other tutorials (e.g., GTR stuff)

1.1.2 Morphological Character Change

Mk models and ascertainment bias

1.1.3 Lineage-Specific Substitution Rates

Clocks (**?**) and relaxing them

1.1.4 Lineage Diversification and Sampling

Birth-death processes and FBD

2 Prerequisites

What do you need to know before doing this?

2.1 Requirements

We assume that you have read and hopefully completed the following tutorials:

- RB_Getting_Started
- RB_Basics_Tutorial

Note that the RB_Basics_Tutorial introduces the basic syntax of `Rev` but does not cover any phylogenetic models. You may skip the RB_Basics_Tutorial if you have some familiarity with `R`. We tried to keep this tutorial very basic and introduce all the language concepts on the way. You may only need the RB_Basics_Tutorial for a more in-depth discussion of concepts in `Rev`.

3 Data and files

We provide the data file(s) which we will use in this tutorial. You may want to use your own data instead. In the **data** folder, you will find the following files

- **bears_extant_cytb.nex**: description of file/data (we also need more descriptive file names).

4 Exercise: Title

4.1 Getting Started

We will complete this analysis in RevBayes by entering the Rev code interactively.

Note that some PDF viewers render some characters differently and if you copy/paste directly from this document, you may introduce erroneous characters. This can cause commands to fail. For learning, it's often better to 'live code' and type the commands in manually, rather than copying and pasting.

Hints for navigating in the RevBayes console:

- **Ctrl+A** – jump to the beginning of the line
- **Ctrl+E** – jump to the end of the line
- Pressing the up key will pull up previous commands, and allow you to edit them
- If you enter the first few letters of a RevBayes keyword and then the Tab key, this will 'autocomplete' the remaining letters.
- Help – if you type **?** followed immediately by a RevBayes keyword, this will print the help pages for that keyword to the screen. (Example **?dnBDP**)

4.2 Loading the Data

→ Download data and output files (if you don't have them already) from:
http://revbayes.github.io/tutorials.html

Start up RevBayes at the command line. You should do this from within the **RB_TotalEvidenceDating_FBD_Tutorial** directory.

```
./rb
```

If you've done this correctly, you should see the prompt shown below

```
>
```

First load in the sequences using the **readDiscreteCharacterData()** function and assign it to a variable called **cytb**.

```
cytb <- readDiscreteCharacterData("data/bears_cytb.nex")
```

Executing these lines initializes the data matrix as the respective `Rev` variables. To report the current value of any variable, simply type the variable name and press enter. For the **cytb** data matrix, this provides information about the alignment:

```
cytb
     DNA character matrix with 10 taxa and 1000 characters
   ======================================================
   Origination:                  bears_cytb.nex
   Number of taxa:               10
   Number of included taxa:      10
   Number of characters:         1000
   Number of included characters: 1000
   Datatype:                     DNA
```

Next, import the morphological character matrix and assign it to the variable **morpho**.

```
morpho <- readDiscreteCharacterData("data/bears_morphology.nex")
```

The information about the morphological character matrix should look like this:

```
morpho
    Standard character matrix with 18 taxa and 81 characters
    ========================================================
    Origination:                bears_morphology.nex
    Number of taxa:             18
    Number of included taxa:    18
    Number of characters:       81
    Number of included characters: 81
    Datatype:                   Standard
```

Now we read in the full list of taxa and create a workspace object with the total number of taxa.

```
taxa <- readTaxonData("data/bears_taxa.tsv", delimiter=TAB)
n_taxa <- taxa.size() # the number of taxa
```

If you open the bears taxa file (**bears_taxa.tsv**), you'll notice that this is a tab-separated file of all of the taxon names, with the age in millions of years ago (mya) in the second column. An age of 0.0 indicates an extant bear species. We will use this information to allow fossils to be incorporated as tips in the analysis.

Notice that the two data matrices have different numbers of taxa. The last bit of data preparation we will do is to add any taxa that are not found in the molecular partition (i.e. are only found in the fossil data) to the molecular partition as missing data, and vice versa. In order for all the taxa to appear on the same tree, they all need to be part of the same dataset, as opposed to present in separate datasets. This ensures that there is a unified taxon set that contains all of our tips.

```
cytb.addMissingTaxa( taxa )
morpho.addMissingTaxa( taxa )
```

4.3 Specify Clades

RevBayes allows researchers to use clade constraints to put a zero probability on trees not containing certain clades. In our case, Ursinae (brown bears, black bears and sloth bears) and Ailuropodinae (pandas and their close relatives) are both very well supported subfamilies. We will add constraints for these two groups.

```
clade_ursinae = clade("Melursus_ursinus","Ursus_arctos","Ursus_maritimus",
                      "Helarctos_malayanus","Ursus_americanus","Ursus_thibetanus",
                      "Ursus_abstrusus","Ursus_spelaeus")
clade_pandas = clade("Ailuropoda_melanoleuca","Indarctos_vireti",
                     "Indarctos_arctoides","Indarctos_punjabiensis",
                     "Ailurarctos_lufengensis","Agriarctos_spp",
                     "Kretzoiarctos_beatrix")
constraints = v(clade_pandas, clade_ursinae)
```

We will also define a clade for the extant bears that will not be constrained. This will allow us to monitor the age of this node in our log file (in Section **??**).

```
clade_extant = clade("Ailuropoda_melanoleuca","Tremarctos_ornatus","Melursus_ursinus",
                     "Ursus_arctos","Ursus_maritimus","Helarctos_malayanus",
                     "Ursus_americanus","Ursus_thibetanus")
```

4.4 Create a Helper Variable

Now that our data are loaded and prepared, we can specify the fossilized birth-death model, and the MCMC moves associated with it. As we work, we will be creating moves to govern how values are sampled for our various priors and parameters. We'll first create an iterator to hold all of our moves.

```
mvi = 1
```

4.5 Specify the fossilized birth-death process

Two key parameters of the FBD process are the birth rate (the rate at which lineages are added to the tree) and the death rate (the rate at which lineages are removed from the tree). We'll place exponential priors on both of these values. An exponential prior with a $\lambda = 10$ places a higher probability on values less than 0.4 for these parameters.

```
birth_rate ~ dnExponential(10)
death_rate ~ dnExponential(10)
```

Now that the priors have been specified, we give RevBayes some information on how to sample values for our parameters. We'll use a scaling move, which changes the value sampled multiplicatively with the tuning parameter. We will use three different tuning parameters, which govern the size of the move. Including multiple tuning parameters improves mixing.

```
moves[mvi++] = mvScale(birth_rate, lambda=0.01, weight=3.0)
moves[mvi++] = mvScale(birth_rate, lambda=0.1, weight=3.0)
moves[mvi++] = mvScale(birth_rate, lambda=1.0, weight=3.0)
moves[mvi++] = mvScale(death_rate, delta=0.01, weight=3.0)
moves[mvi++] = mvScale(death_rate, delta=0.1, weight=3.0)
moves[mvi++] = mvScale(death_rate, delta=1, weight=3.0)
```

In order to print nodes in our graph to output (also called *monitoring*), we need to create deterministic nodes for the diversification and turnover. Deterministic nodes are value transformations between existing stochastic nodes. So we will define diversification and turnover as deterministic nodes.

```
diversification := birth_rate - death_rate
turnover := death_rate/birth_rate
```

All extant bears are represented in this dataset. Therefore, we can fix the sampling probability of extant lineages to 1.

```
rho <- 1.0
```

The rate of sampling fossils (ψ), on the other hand is not known. We will use an exponential prior on this parameter as well, and use a slide move to sample values from our distribution.

```
psi ~ dnExponential(10)
moves[mvi++] = mvScale(birth_rate, lambda=0.01, weight=3.0)
moves[mvi++] = mvScale(birth_rate, lambda=0.1, weight=3.0)
moves[mvi++] = mvScale(birth_rate, lambda=1.0, weight=3.0)
```

Under the FBD model, the process is conditioned on the age of the origin, or the start of the process. We will specify a uniform distribution on the age of the origin. If you looked in the bears taxa file, you might notice that the age of the oldest fossil is slightly younger than the upper bound of the uniform distribution on the origin age. For this parameter, we will use a sliding window move. A sliding window move samples within an interval (defined by delta) . Sliding window moves can be tricky for small values, as the window may overlap zero. However, for parameters such as the origin, there is little risk of this being an issue.

```
origin_time ~ dnUnif(37.0, 55.0)
moves[mvi++] = mvSlide(origin_time, delta=0.01, weight=10.0)
moves[mvi++] = mvSlide(origin_time, delta=0.1, weight=10.0)
moves[mvi++] = mvSlide(origin_time, delta=1, weight=10.0)
```

All the parameters of the FBD process are now defined. The next step is to combine these parameters to define the tree prior as the FBD.

```
tree_prior = dnFBDP(origin=origin_time, lambda=birth_rate, mu=death_rate, psi=psi, rho
    =rho, taxa=taxa)
```

Next, we will define the **fbd_tree** variable as a random variable. It will be used to generate trees under the FBD process that conform to our clade constraints.

```
fbd_tree ~ dnConstrainedTopology(tree_prior, constraints)
```

Finally, we can also create deterministic nodes for other quantities we might be interested in monitoring. Below, we will define a monitor that prints the number of fossils that are inferred to be 'sampled ancestors' - lineages that are present in the phylogeny, and have descendants present on the tree. We will also define a deterministic node for the age of the crown group of bears, using the previously-defined extant bear constraint (Section **??**).

```
sa := fbd_tree.numSampledAncestors();
crown := tmrca(fbd_tree, clade_extant)
```

Version dated: December 19, 2016