Worked-out examples

Richel Bilderbeek

2016-05-13

This document shows some work-out examples.

These worked-out examples show:

* the error if protractedness is absent
* the error if protractedness strong
* comparing these errors
* the error if protractedness is absent for multiple replicates
* the error if protractedness is strong for multiple replicates
* comparing these errors

## Workflow

Every experiment has the following steps:

* Starting up
* Creating parameter files
* Per parameter file:
* Simulate one incipient species tree per parameter file
* Once or more often, sample unique species from an incipient species tree
* Per sampled species tree, add an outgroup
* Simulate one or more alignments per species tree
* Once or more often, let BEAST2 infer a posterior per alignment
* Compare original, sampled species tree, to the posterior
* Per two parameter files:
* Compare original, sampled species tree, to the posterior

## Starting up

We will need to load some libraries:

library(ape)  
library(Cer2016)  
library(ggplot2)  
library(Hmisc)

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following object is masked from 'package:ape':  
##   
## zoom

## The following objects are masked from 'package:base':  
##   
## format.pval, round.POSIXt, trunc.POSIXt, units

library(nLTT)  
library(ribir)  
library(phangorn)

If we want to generate files fresh, set this to TRUE:

start\_fresh <- FALSE

We also set the resolution of our analysis:

dt <- 0.001

I use 0.1 while developing code, 0.001 for a decent analysis.

## Creating parameter files

This worked-out example will show:

* the error if protractedness is absent
* the error if protractedness strong
* comparing these errors
* the error if protractedness is absent for multiple replicates
* the error if protractedness is strong for multiple replicates
* comparing these errors

For that, I will create four parameter files:

* toy\_example\_1.RDa: protractedness is absent
* toy\_example\_2.RDa: protractedness strong
* toy\_example\_3.RDa: protractedness is absent for multiple replicates
* toy\_example\_4.RDa: protractedness is strong for multiple replicates

Here, we create the four parameter filenames:

filenames = c(  
 "toy\_example\_1.RDa",   
 "toy\_example\_2.RDa",   
 "toy\_example\_3.RDa",   
 "toy\_example\_4.RDa"  
)

To be sure to have a fresh analysis, we delete the files if they exist:

if (start\_fresh) {  
 for (filename in filenames) {  
 if (file.exists(filename)) {  
 file.remove(filename)  
 }  
 }  
}

Now the parameter files are created with the desired parameters:

rng\_seeds <- seq(1,4)  
sirgs <- rep(0.5, times = 4)  
siris <- rep(0.5, times = 4)  
scrs <- c(1.0e6, 1.0e-1, 1.0e6, 1.0e-1)  
ergs <- rep(0.1, times = 4)  
eris <- rep(0.1, times = 4)  
ages <- rep(5, times = 4)  
n\_species\_trees\_sampleses <- c(1, 1, 2, 2)  
mutation\_rates <- rep(0.01, times = 4)  
n\_alignmentses <- c(1, 1, 2, 2)  
sequence\_lengths <- rep(1000, times = 4)  
mcmc\_chainlengths <- rep(10000, times = 4)  
n\_beast\_runses <- c(1, 1, 2, 2)  
for (i in seq(1, 4)) {  
 save\_parameters\_to\_file(  
 rng\_seed = rng\_seeds[i],  
 sirg = sirgs[i],  
 siri = siris[i],  
 scr = scrs[i],  
 erg = ergs[i],  
 eri = eris[i],  
 age = ages[i],  
 n\_species\_trees\_samples = n\_species\_trees\_sampleses[i],  
 mutation\_rate = mutation\_rates[i],  
 n\_alignments = n\_alignmentses[i],  
 sequence\_length = sequence\_lengths[i],  
 mcmc\_chainlength = mcmc\_chainlengths[i],  
 n\_beast\_runs = n\_beast\_runses[i],  
 filename = filenames[i]  
 )   
}

* toy\_example\_1.RDa: protractedness is absent, thus speciation completion rate is high, in this case 1.0e6
* toy\_example\_2.RDa: protractedness strong, thus speciation completion rate is low, in this case 1.0e-1
* toy\_example\_3.RDa: protractedness is absent for multiple replicates, thus number of sampled species trees, alignments per species trees, and BEAST2 runs are all 2
* toy\_example\_4.RDa: protractedness is strong for multiple replicates, thus number of sampled species trees, alignments per species trees, and BEAST2 runs are all 2

Here the parameters are shown in a nicer format:

show\_parameter\_files(filenames)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | rng\_seed | sirg | siri | scr | erg | eri | age | n\_species\_trees\_samples | mutation\_rate | n\_alignments | sequence\_length | mcmc\_chainlength | n\_beast\_runs | version |
| toy\_example\_1.RDa | 1 | 0.5 | 0.5 | 1e+06 | 0.1 | 0.1 | 5 | 1 | 0.01 | 1 | 1000 | 10000 | 1 | 0.1 |
| toy\_example\_2.RDa | 2 | 0.5 | 0.5 | 1e-01 | 0.1 | 0.1 | 5 | 1 | 0.01 | 1 | 1000 | 10000 | 1 | 0.1 |
| toy\_example\_3.RDa | 3 | 0.5 | 0.5 | 1e+06 | 0.1 | 0.1 | 5 | 2 | 0.01 | 2 | 1000 | 10000 | 2 | 0.1 |
| toy\_example\_4.RDa | 4 | 0.5 | 0.5 | 1e-01 | 0.1 | 0.1 | 5 | 2 | 0.01 | 2 | 1000 | 10000 | 2 | 0.1 |

The parameter settings used in this example are identical, except for their speciation completion rate and number of replicates.

The experimental setup of this research has multiple steps, which we will follow closely here.

These worked-out examples show the data produced in its raw form and does not care too much about aesthetics.

## Example 1: Weak protractedness

This example answers the question: what is the base level error of the analyses in this research?

The base level error can be obtained by using parameters for a constant-rate birth-death model. All tools used assume this model, but there will be noise (thus error) added in the process.

The parameter settings of example 1 have a high speciation completion rate , which makes the constant-rate protracted speciation model fall back to a constant-rate birth-death model, as incipient species become good species (close to) instantaneously.

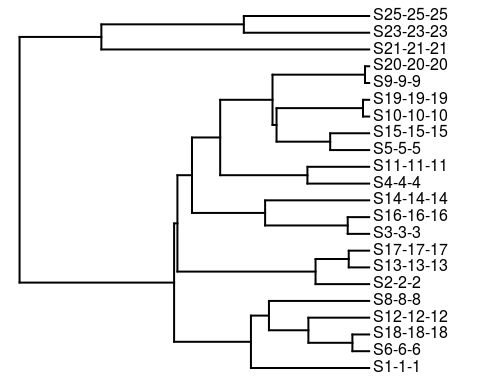
### Simulate one incipient species tree per parameter file

Here we simulate the 'true' incipient species tree:

filename <- filenames[1]  
testit::assert(is\_valid\_file(filename))  
add\_pbd\_output(filename)

This is how the incipient species tree looks like:

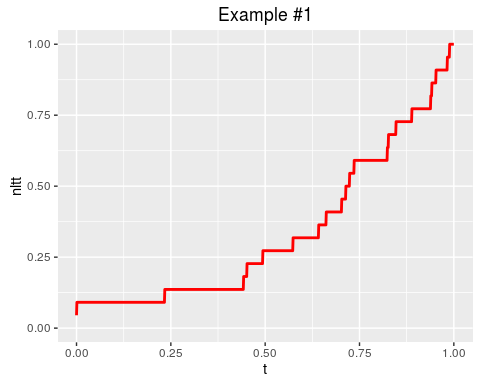
colors <- setNames(c("gray","black"), c("i","g"))  
testit::assert(length(read\_file(filename)$pbd\_output$igtree.extant$tip.label) > 0)  
phytools::plotSimmap(  
 read\_file(filename)$pbd\_output$igtree.extant,   
 colors = colors  
)



The taxon labels are S[genus]-[species]-[sub-species]'.

Now, we plot the nLTT plot of this phylogeny:

nltt\_values <- get\_nltt\_values(  
 phylogenies = list(read\_file(filename)$pbd\_output$tree),   
 dt = 0.001  
)  
qplot(  
 t, nltt, data = nltt\_values, geom = "blank", ylim = c(0,1),  
 main = "Example #1"  
) + stat\_summary(  
 fun.data = "mean\_cl\_boot", color = "red", geom = "smooth"  
)



Note that get\_nltt\_values is supplied a list of the one incipient species tree. This is because get\_nltt\_values works on a list of phylogenies.

### Once or more often, sample unique species from an incipient species tree. Also add an outgroup

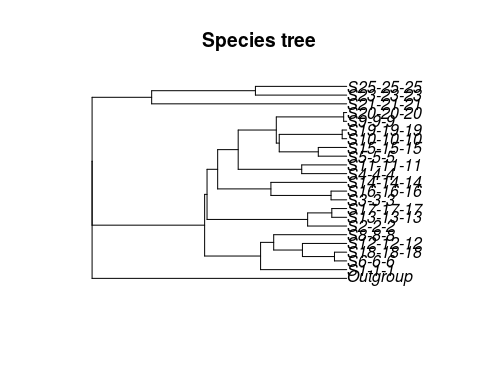
From that incipient species tree, we create a species tree by sampling one individual per incipient species. Because speciation is for a constant-rate birth death model, there exist no multiple individuals per species. To being able to root our phylogenies in later steps, an outgroup is added as well.

add\_species\_trees\_with\_outgroup(filename)

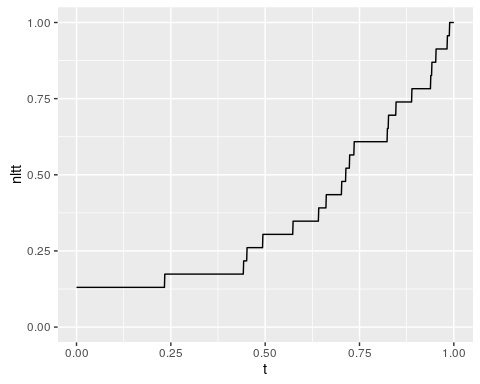
## [1] "Adding species\_trees\_with\_outgroup to file toy\_example\_1.RDa"  
## [1] " \* Setting seed to 2"  
## [1] "Added species\_trees\_with\_outgroup to file toy\_example\_1.RDa"

Here we observe the sample species tree:

plot\_species\_tree\_with\_outgroup(filename)



plot\_species\_tree\_with\_outgroup\_nltt(filename, dt = 0.001)



Note that in example #3, there will be multiple sampled species trees.

### Simulate one or more alignments per species tree

Knowing the evolutionary distances between species, DNA sequence alignments can be simulated fitting the species tree with outgroup. To do so, the parameters for sequence length and mutation rate are used.

Here we simulate one DNA sequence on our one sampled species tree with outgroup:

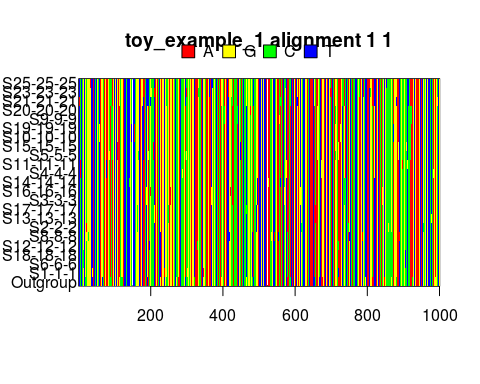
add\_alignments(filename)

## [1] " \* Created and saved alignments[1]"  
## [1] "file toy\_example\_1.RDa has gotten its 1 alignments (per species tree)"

Note that this research assumes a simple Jukes-Cantor model, and does so as well in later steps.

Let's see the alignment:

plot\_alignments(filename)



### Once or more often, let BEAST2 infer a posterior per alignment

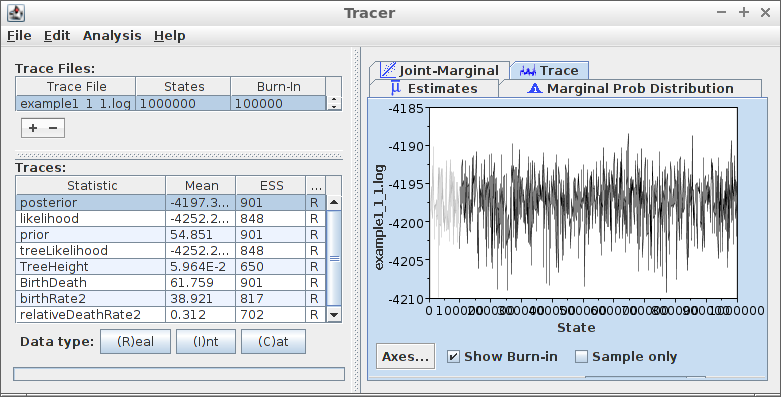
With BEAST2 we can now obtain a posterior. A posterior consists of a representative sample of all possible trees (and parameter estimates), yet with more probable trees being present more often.

Here we let BEAST2 do so:

add\_posteriors(filename, skip\_if\_output\_present = TRUE)

## [1] " \* Setting seed to 2"  
## [1] " \* Storing posterior #1 for alignment #1 for species tree #1 at posterior\_index #1"  
## [1] "file toy\_example\_1.RDa has gotten its posteriors"

After running BEAST2 on our DNA sequence, the full posterior must be verified to be eligible for further analysis. Using Tracer, we can open the .log file generated by BEAST2, which is then displayed here:



Example 1 trace

(note: this picture is not freshly generated and may thus be incorrect)

It can be seen that the values for ESS (Effective Sample Size) are above 200 and that the trace log shows a well-mixed chain. An ESS of 200 is used as a minimum in this research.

A posterior contains many phylogenies. To visualize these, they are extracted from a .trees file here:

trees\_filename <- "toy\_example\_1\_1\_1\_1.trees"  
testit::assert(file.exists(trees\_filename))  
phylogenies <- rBEAST::beast2out.read.trees(trees\_filename)

## Warning: replacing previous import 'data.table::melt' by 'reshape2::melt'  
## when loading 'rBEAST'

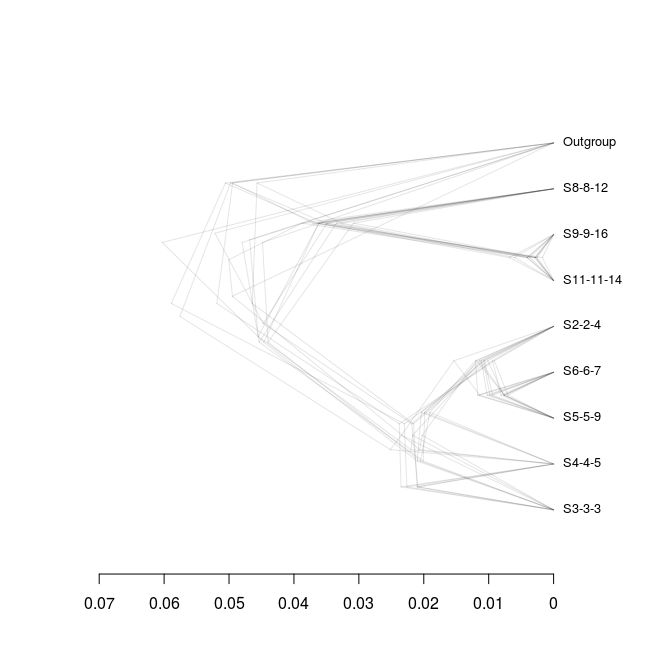
## Warning: replacing previous import 'data.table::dcast' by 'reshape2::dcast'  
## when loading 'rBEAST'

## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_1\_1\_1\_1.trees'

# To get the densiTree function working, phylogenies must be of class multiphylo  
class(phylogenies) <- "multiPhylo"

Then, these posterior trees are plotted:

densiTree(  
 phylogenies,   
 type = "cladogram",   
 alpha = 1/length(phylogenies)  
)

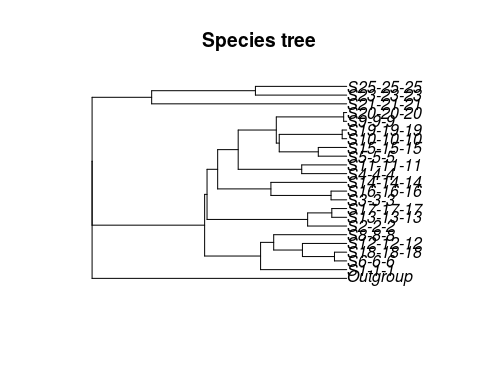


From this plot, it can be seen which phylogeny configurations pop up most in the posterior.

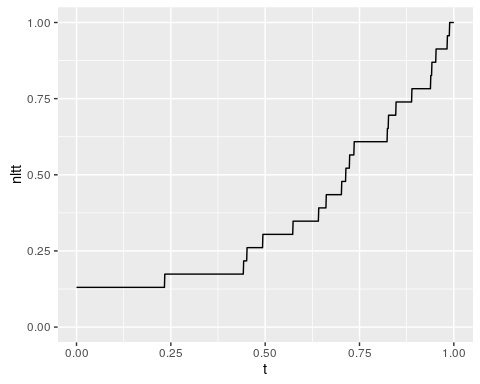
### Compare original, sampled species tree, to the posterior

To refresh our mind, the true sampled species tree is:

plot\_species\_tree\_with\_outgroup(filename)



plot\_species\_tree\_with\_outgroup\_nltt(filename, dt = dt)

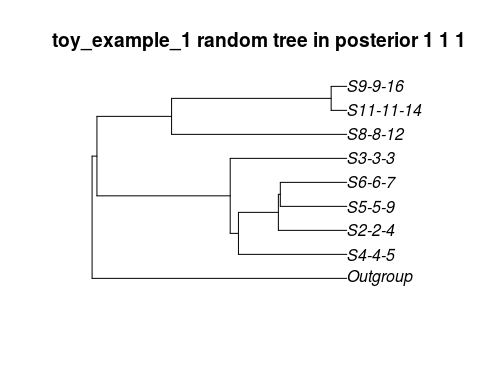


Now that the full posterior is assumed to be correct, first, I will now highlight one of its posterior states, before going back to the full picture.

In this case, I choose a random state to zoom in on. From this last state, I show the tree only:

plot\_posterior\_samples(filename)

## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_1\_1\_1\_1.trees'



This random tree may be very different by chance, as unlikely trees are present, yet in low abundances.

The original sampled tree is very probably different from this random posterior tree.

How different then?

There are multiple ways to quantify this:

* The gamma statistic
* The nLTT statistic

#### The gamma statistic

The gamma value is a summary statistic described in [2]

First, I collect the gamma statistics from the species tree(s) with outgroup(s):

true\_gamma\_statistics <- collect\_species\_tree\_gammas(filename = filename)  
knitr::kable(head(true\_gamma\_statistics))

|  |  |
| --- | --- |
| gamma\_stat | species\_tree |
| 0.035416 | 1 |

In this case, there will be one statistics, as we only have one species tree.

Then, I collect the gamma statistics from the posterior(s):

posterior\_gamma\_statistics <- collect\_posterior\_gammas(  
 filename = filename  
)

## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_1\_1\_1\_1.trees'

knitr::kable(head(posterior\_gamma\_statistics))

|  |  |  |  |
| --- | --- | --- | --- |
| species\_tree | alignment | beast\_run | gamma |
| 1 | 1 | 1 | -0.2329420 |
| 1 | 1 | 1 | -0.4906083 |
| 1 | 1 | 1 | -0.6150782 |
| 1 | 1 | 1 | -0.6800620 |
| 1 | 1 | 1 | -0.2933330 |
| 1 | 1 | 1 | -0.4585116 |

#### The nLTT statistic

The nLTT value is a summary statistic described in [1]

First, I collect the nLTT values from the species tree(s) with outgroup(s):

true\_nltt\_values <- collect\_species\_tree\_nltts(filename = filename, dt = dt)  
knitr::kable(head(true\_nltt\_values))

|  |  |  |
| --- | --- | --- |
| species\_tree | t | nltt |
| 1 | 0.000 | 0.1304348 |
| 1 | 0.001 | 0.1304348 |
| 1 | 0.002 | 0.1304348 |
| 1 | 0.003 | 0.1304348 |
| 1 | 0.004 | 0.1304348 |
| 1 | 0.005 | 0.1304348 |

Then, I collect the nLTT values from the posterior(s):

posterior\_nltt\_values <- collect\_posterior\_nltt\_values(  
 filename = filename, dt = dt  
)

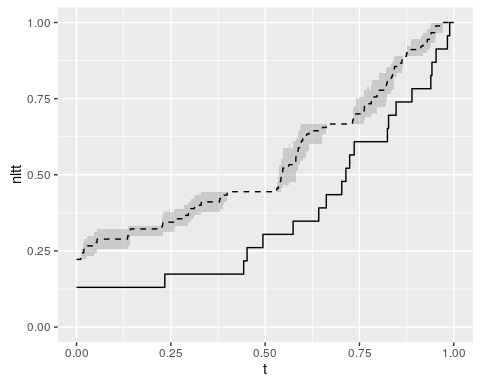
## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_1\_1\_1\_1.trees'

knitr::kable(head(posterior\_nltt\_values))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| species\_tree | alignment | beast\_run | t | nltt |
| 1 | 1 | 1 | 0.000 | 0.2222222 |
| 1 | 1 | 1 | 0.001 | 0.2222222 |
| 1 | 1 | 1 | 0.002 | 0.2222222 |
| 1 | 1 | 1 | 0.003 | 0.2222222 |
| 1 | 1 | 1 | 0.004 | 0.2222222 |
| 1 | 1 | 1 | 0.005 | 0.2222222 |

Last, I plot both of these:

ggplot2::ggplot(  
 data = posterior\_nltt\_values,  
 ggplot2::aes(t, nltt),  
 main = "nLTT plots"  
) + ggplot2::geom\_point(  
 color = "transparent"  
) + ggplot2::scale\_x\_continuous(  
 limits = c(0, 1)  
) + ggplot2::scale\_y\_continuous(  
 limits = c(0, 1)  
) + ggplot2::stat\_summary(  
 fun.data = "mean\_cl\_boot", size = 0.5, lty = "dashed",  
 color = I("black"), geom = "smooth"  
) + ggplot2::geom\_step(  
 data = true\_nltt\_values,  
 ggplot2::aes(t, nltt)  
)



How do we interpret these results? Well, even *if* we follow all assumptions, this will be standard errors we will make; this is the minimal level of imperfection.

Note that the error is not quantified only visualized. TODO: Issue #21

## Example 2: Strong protractedness

This example answers the question: how do the analyses of this research look like for strong protractedness?

The pipeline is identical, except one parameter is changed: the speciation completion rate is set to a low value.

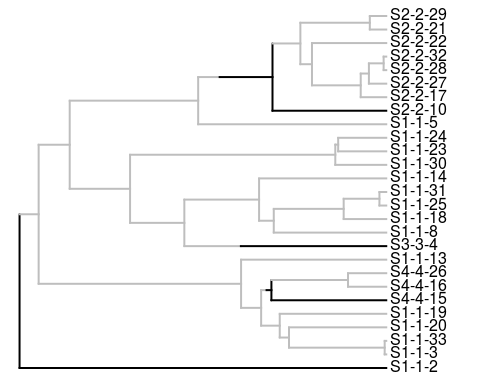
### Simulate one incipient species tree per parameter file

Here we simulate the 'true' incipient species tree:

filename <- filenames[2]  
testit::assert(is\_valid\_file(filename))  
add\_pbd\_output(filename)

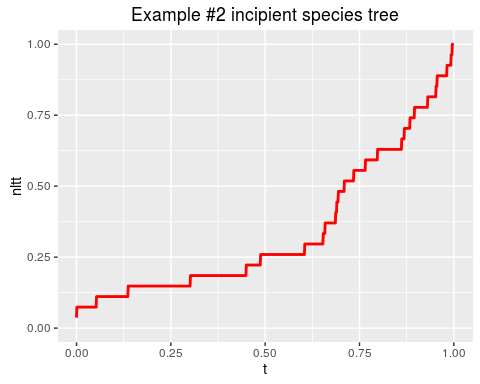
This is how the incipient species tree looks like:

colors <- setNames(c("gray","black"), c("i","g"))  
testit::assert(length(read\_file(filename)$pbd\_output$igtree.extant$tip.label) > 0)  
phytools::plotSimmap(  
 read\_file(filename)$pbd\_output$igtree.extant,   
 colors = colors  
)



Now, we plot the nLTT plot of this phylogeny:

nltt\_values <- get\_nltt\_values(  
 phylogenies = list(read\_file(filename)$pbd\_output$tree),   
 dt = 0.001  
)  
qplot(  
 t, nltt, data = nltt\_values, geom = "blank", ylim = c(0,1),  
 main = "Example #2 incipient species tree"  
) + stat\_summary(  
 fun.data = "mean\_cl\_boot", color = "red", geom = "smooth"  
)



### Once or more often, sample unique species from an incipient species tree. Also add an outgroup

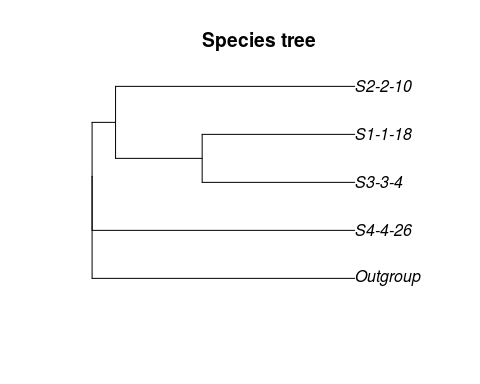
From that incipient species tree, we create a species tree by sampling one individual per incipient species. Because speciation is for a constant-rate birth death model, there exist no multiple individuals per species. To being able to root our phylogenies in later steps, an outgroup is added as well.

add\_species\_trees\_with\_outgroup(filename)

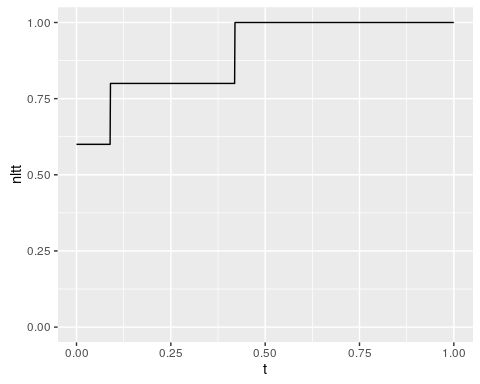
## [1] "Adding species\_trees\_with\_outgroup to file toy\_example\_2.RDa"  
## [1] " \* Setting seed to 3"  
## [1] "Added species\_trees\_with\_outgroup to file toy\_example\_2.RDa"

Here we observe the sample species tree:

plot\_species\_tree\_with\_outgroup(filename)



plot\_species\_tree\_with\_outgroup\_nltt(filename, dt = 0.001)



Note that in example #4, there will be multiple sampled species trees.

### Simulate one or more alignments per species tree

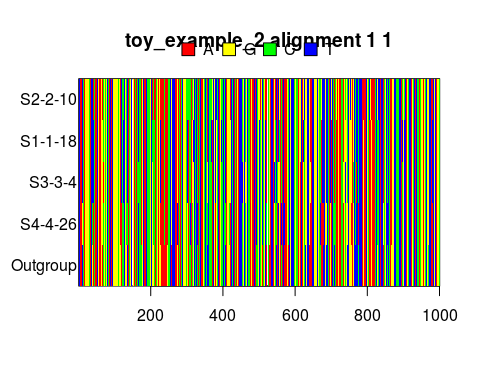
Here we simulate one DNA sequence on our one sampled species tree with outgroup:

add\_alignments(filename)

## [1] " \* Created and saved alignments[1]"  
## [1] "file toy\_example\_2.RDa has gotten its 1 alignments (per species tree)"

Let's see the alignment:

plot\_alignments(filename)



### Once or more often, let BEAST2 infer a posterior per alignment

Here we let BEAST2 infer a posterior:

add\_posteriors(filename, skip\_if\_output\_present = TRUE)

## [1] " \* Setting seed to 3"  
## [1] " \* Storing posterior #1 for alignment #1 for species tree #1 at posterior\_index #1"  
## [1] "file toy\_example\_2.RDa has gotten its posteriors"

A posterior contains many phylogenies. To visualize these, they are extracted from a .trees file here:

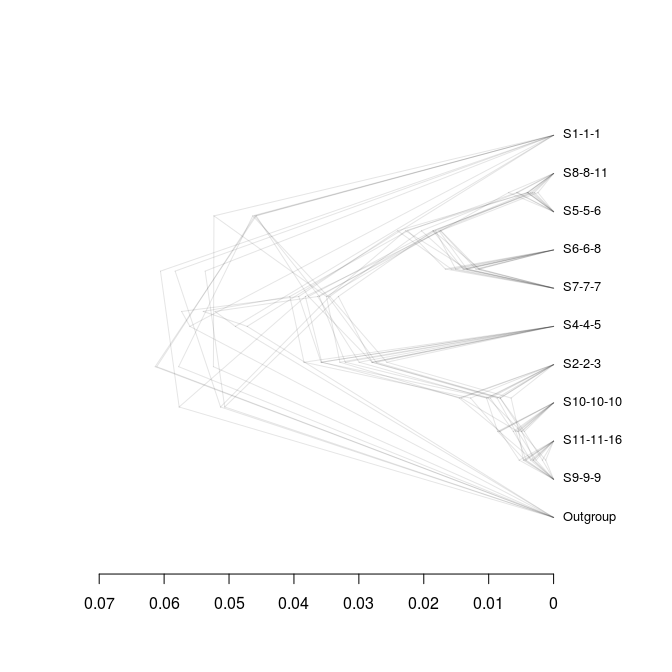
trees\_filename <- "toy\_example\_2\_1\_1\_1.trees"  
testit::assert(file.exists(trees\_filename))  
phylogenies <- rBEAST::beast2out.read.trees(trees\_filename)

## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_2\_1\_1\_1.trees'

# To get the densiTree function working, phylogenies must be of class multiphylo  
class(phylogenies) <- "multiPhylo"

Then, these posterior trees are plotted:

densiTree(  
 phylogenies,   
 type = "cladogram",   
 alpha = 1/length(phylogenies)  
)

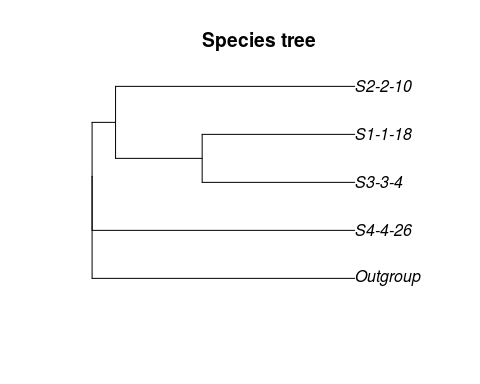


From this plot, it can be seen which phylogeny configurations pop up most in the posterior.

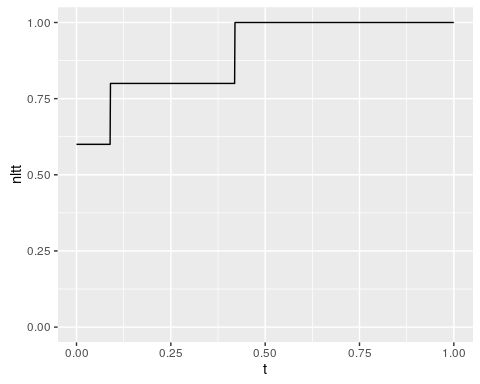
### Compare original, sampled species tree, to the posterior

To refresh our mind, the true sampled species tree is:

plot\_species\_tree\_with\_outgroup(filename)



plot\_species\_tree\_with\_outgroup\_nltt(filename, dt = dt)



To quantify the difference between original incipient species tree and posterior trees, the gamma statistic is used again.

#### The nLTT statistic

The nLTT value is a summary statistic described in [1]

First, I collect the nLTT values from the species tree(s) with outgroup(s):

true\_nltt\_values <- collect\_species\_tree\_nltts(filename = filename, dt = dt)  
knitr::kable(head(true\_nltt\_values))

|  |  |  |
| --- | --- | --- |
| species\_tree | t | nltt |
| 1 | 0.000 | 0.6 |
| 1 | 0.001 | 0.6 |
| 1 | 0.002 | 0.6 |
| 1 | 0.003 | 0.6 |
| 1 | 0.004 | 0.6 |
| 1 | 0.005 | 0.6 |

Then, I collect the nLTT values from the posterior(s):

posterior\_nltt\_values <- collect\_posterior\_nltt\_values(  
 filename = filename, dt = dt  
)

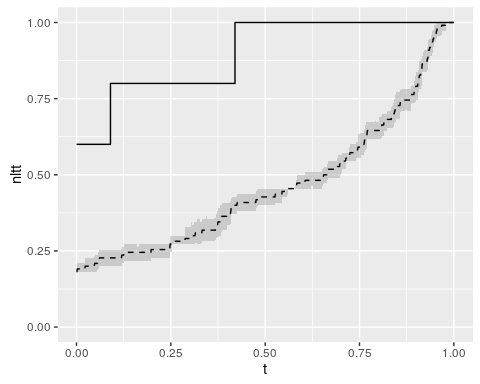
## Warning in readLines(file, n = 2000): incomplete final line found on  
## 'toy\_example\_2\_1\_1\_1.trees'

knitr::kable(head(posterior\_nltt\_values))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| species\_tree | alignment | beast\_run | t | nltt |
| 1 | 1 | 1 | 0.000 | 0.1818182 |
| 1 | 1 | 1 | 0.001 | 0.1818182 |
| 1 | 1 | 1 | 0.002 | 0.2727273 |
| 1 | 1 | 1 | 0.003 | 0.2727273 |
| 1 | 1 | 1 | 0.004 | 0.2727273 |
| 1 | 1 | 1 | 0.005 | 0.2727273 |

Last, I plot both of these:

ggplot2::ggplot(  
 data = posterior\_nltt\_values,  
 ggplot2::aes(t, nltt),  
 main = "nLTT plots"  
) + ggplot2::geom\_point(  
 color = "transparent"  
) + ggplot2::scale\_x\_continuous(  
 limits = c(0, 1)  
) + ggplot2::scale\_y\_continuous(  
 limits = c(0, 1)  
) + ggplot2::stat\_summary(  
 fun.data = "mean\_cl\_boot", size = 0.5, lty = "dashed",  
 color = I("black"), geom = "smooth"  
) + ggplot2::geom\_step(  
 data = true\_nltt\_values,  
 ggplot2::aes(t, nltt)  
)



How do we interpret these results? There is a big difference between the 'true' nLTT and the inferred posterior its average nLTT, thus, as expected, BEAST2 does worse at inferring phylogenies when its assumptions (that speciation is instantanous) are violated.

Note that the error is not quantified only visualized. TODO: Issue #21

We cannot draw concusions yet:

* the number of branches is too little
* the number of replicates is too little

In the next examples, we add replicates to the data.

## References

* [1] Janzen, Thijs, Sebastian Höhna, and Rampal S. Etienne. "Approximate Bayesian Computation of diversification rates from molecular phylogenies: introducing a new efficient summary statistic, the nLTT." Methods in Ecology and Evolution 6.5 (2015): 566-575.
* [2] Pybus, Oliver G., and Paul H. Harvey. "Testing macro–evolutionary models using incomplete molecular phylogenies." Proceedings of the Royal Society of London B: Biological Sciences 267.1459 (2000): 2267-2272.