Demonstration of nLTTstat functions

Richel Bilderbeek

2016-09-14

Both `nLTTstat` and `nLTTstat_exact` calculate the nLTT statistic value. The nLTT statistic is a measure of mismatch between two phylogenies.

In this vignette:

- an introduction of the data we work on
- a demo of how to use the function itself

General setup

Load all libraries:

```
library(nLTT) # nolint
```

Use the heigest warning level:

```
options(warn = 2)
```

Set the random number generator seed:

```
set.seed(42)
```

Setting a random number generator seed makes the results reproducible.

The data we work on will be:

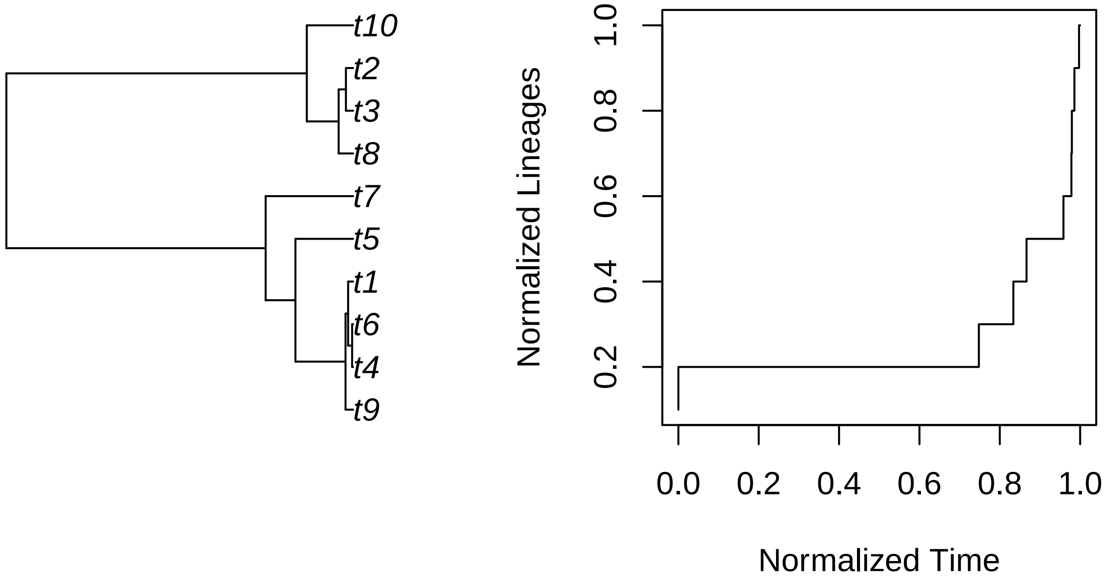
- a focal phylogeny, called 'the species tree'
- multiple other phylogenies to be compared to the species tree, called 'the posterior'

First we construct the focal species tree:

```
species_tree <- ape::rcoal(n = 10)
```

Here the phylogeny and nLTT plot are displayed:

```
par(mfrow = c(1, 2))
ape::plot.phylo(species_tree)
nLTT::nltt_plot(species_tree)
```



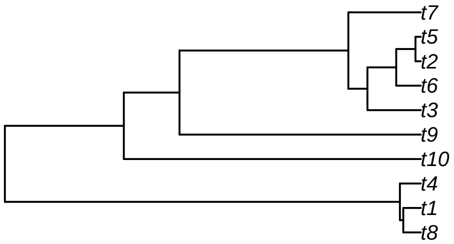
```
par(mfrow = c(1, 1))
```

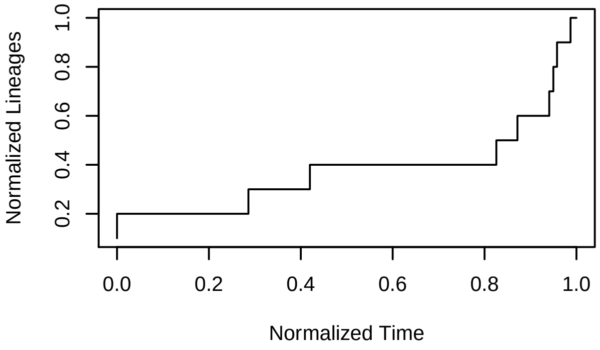
Then we collect four possible posterior species trees:

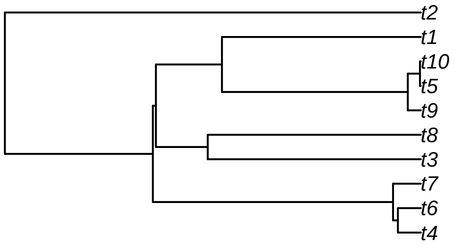
```
posteriors <- c(
  ape::rcoal(n = 10),
  ape::rcoal(n = 10),
  species_tree,
  ape::rcoal(n = 10)
)
```

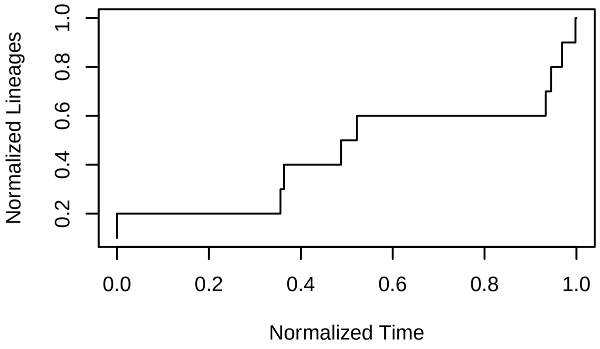
Here these phylogenies and nLTT plots are displayed:

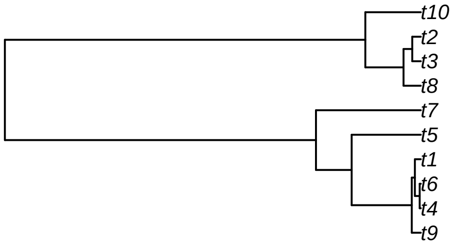
```
par(mfrow = c(4, 2))
for (p in posteriors) {
  ape::plot.phylo(p, cex = 1)
  nLTT::nltt_plot(p)
}
```

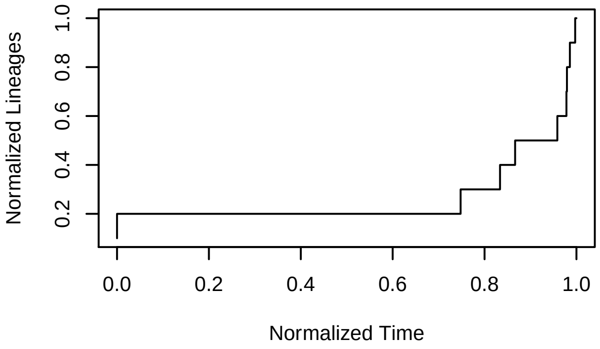

t7
t5
t2
t6
t3
t9
t10
t4
t1
t8

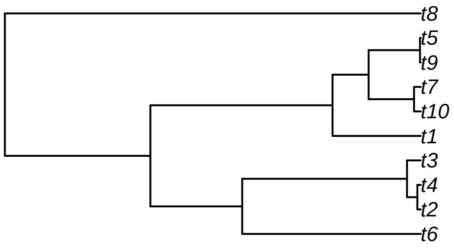

Normalized Lineages
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Normalized Time

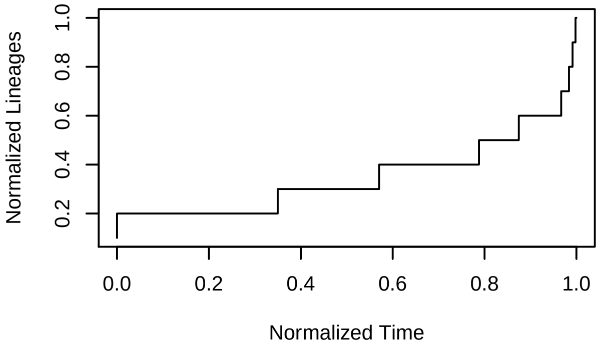

t2
t1
t10
t5
t9
t8
t3
t7
t6
t4


Normalized Lineages
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Normalized Time


t10
t2
t3
t8
t7
t5
t1
t6
t4
t9


Normalized Lineages
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Normalized Time

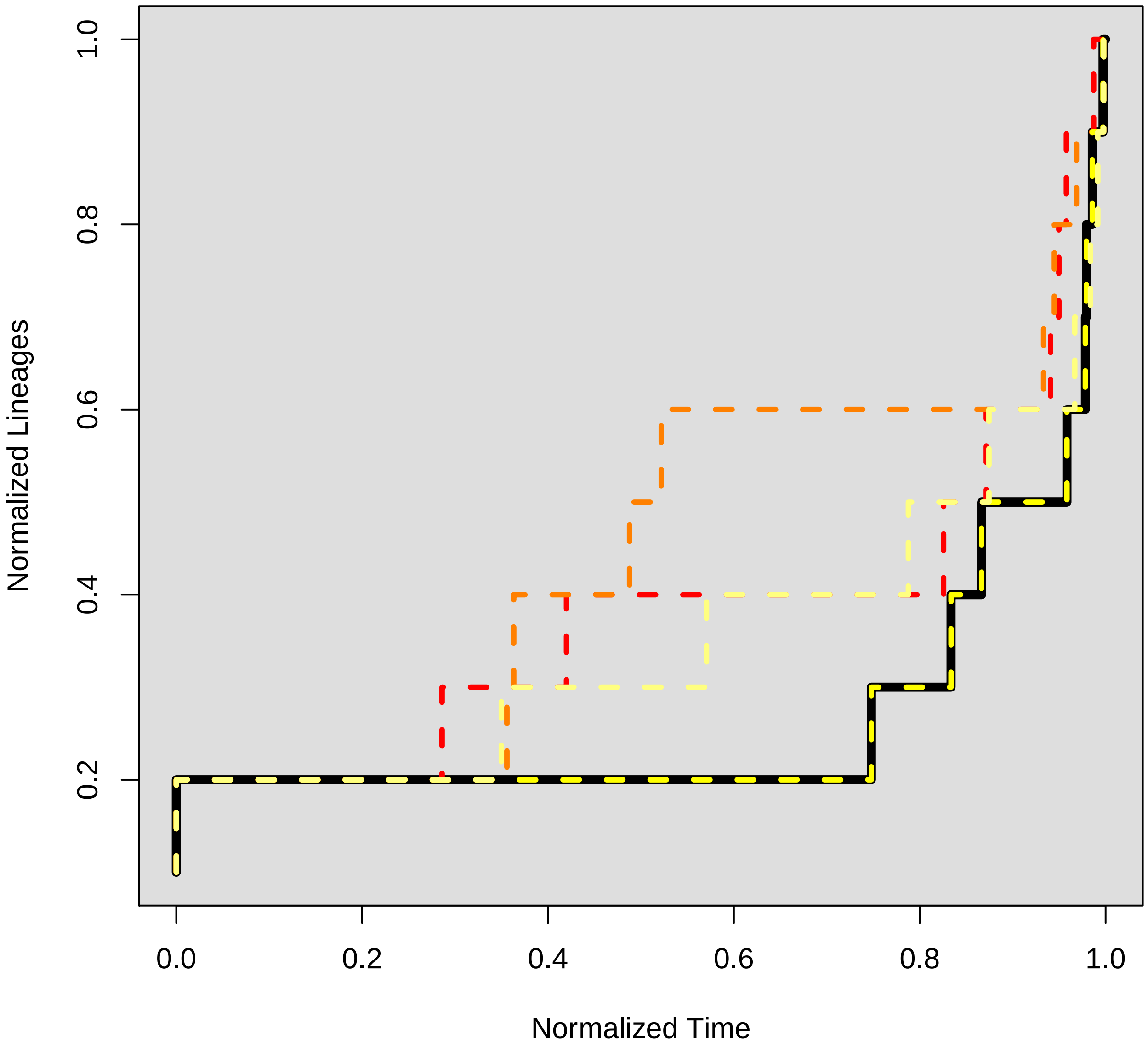

t8
t5
t9
t7
t10
t1
t3
t4
t2
t6


Normalized Lineages
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Normalized Time

```
par(mfrow = c(1, 1))
```

Here we display the nLTT plots of both the original species tree (thicker lines) and its posteriors:

```
# Set the plot area
nLTT::nltt_plot(species_tree)
# Set the background to light grey
rect(par("usr")[1],par("usr")[3],par("usr")[2],par("usr")[4], col = colors()[240])
# Draw the species tree
nLTT::nltt_lines(species_tree, lwd = 5)
# Draw the posteriors
col_index <- 1
for (p in posteriors) {
  nLTT::nltt_lines(p, lwd = 3, lty = 2, col = grDevices::heat.colors(n = 4)[col_index])
  col_index <- col_index + 1
}
```



The nLTT statistic

The nLTT package supplies two version of calculating the nLTT statistic: exact or approximated.

First, we calculate both nLTT statistic between original species tree and each posterior tree:

```
nltt_stats_exact  <- rep(x = 0, times = length(posteriors))
nltt_stats_approx <- rep(x = 0, times = length(posteriors))
i <- 1
for (p in posteriors) {
  nltt_stats_exact[i]  <- nLTTstat_exact(species_tree, p)
  nltt_stats_approx[i] <- nLTTstat(species_tree, p)
  i <- i + 1
}
```

One of the posterior tree is a copy of the input phylogeny. This allows us to check our algorithm: the final nLTT statistic should be zero for that posterior tree.

Here we put the results in a data frame and display it as a table:

```
nltt_stats <- data.frame(
  id = seq(1, length(nltt_stats_exact)),
  nltt_stat_exact = nltt_stats_exact,
  nltt_stat_approx = nltt_stats_approx
)
knitr::kable(nltt_stats)
```

id	nltt_stat_exact	nltt_stat_approx
1	0.1108467	0.1114308
2	0.1775316	0.1774595
3	0.0000000	0.0000000
4	0.0846390	0.0863506

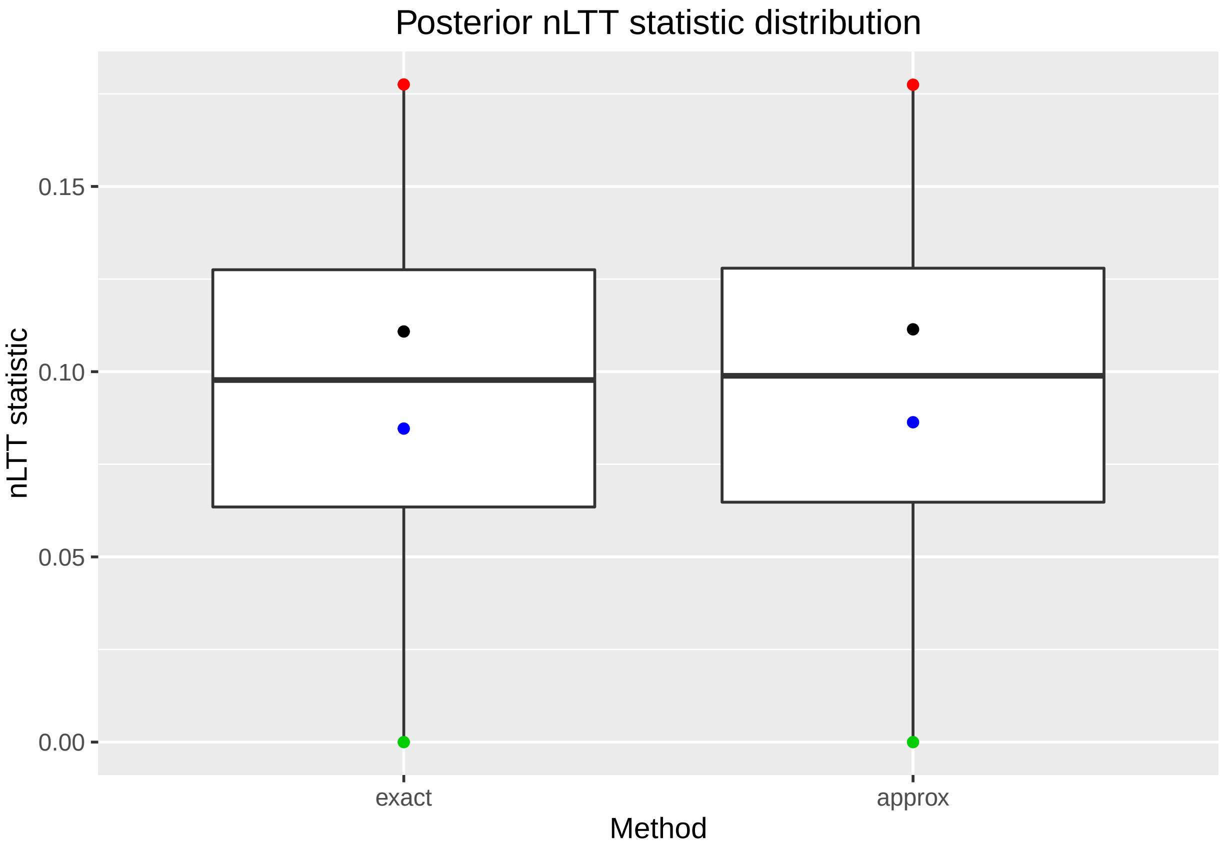
The date frame shows that both methods result in appoximately the same values.

Melting this data for the next plot:

```
df <- reshape2::melt(
  nltt_stats,
  id.vars = c("id"),
  measure.vars = c( "nltt_stat_exact", "nltt_stat_approx")
)
names(df) <- c("id", "method", "nltt_stat")
df$id <- as.factor(df$id)
df$method <- plyr::revalue(
  df$method,
  c("nltt_stat_exact" = "exact", "nltt_stat_approx" = "approx")
)
```

Plotting the methods in a boxplot:

```
ggplot2::ggplot(
  data = df,
  ggplot2::aes(x = df$method, y = df$nltt_stat)
) + ggplot2::geom_boxplot(
) + ggplot2::geom_point(color = df$id
) + ggplot2::scale_y_continuous(name = "nLTT statistic"
) + ggplot2::scale_x_discrete(name = "Method"
) + ggplot2::ggtitle("Posterior nLTT statistic distribution")
```


Posterior nLTT statistic distribution
nLTT statistic
exact
approx
Method