Package 'nLTT'

October 12, 2016

Type Package

Title Calculate the NLTT Statistic

Version 1.3.1

Description Provides functions to calculate the normalised Lineage-Through-Time (nLTT) statistic, given two phylogenetic trees. The nLTT statistic measures the difference between two Lineage-Through-Time curves, where each curve is normalised both in time and in number of lineages.

License GPL-2

Imports ape, coda, deSolve

Suggests DDD, ggplot2, Hmisc, knitr, microbenchmark, plyr, reshape2, rmarkdown, TESS, testit, testthat, TreeSim

NeedsCompilation no

RoxygenNote 5.0.1

VignetteBuilder knitr

URL https://github.com/richelbilderbeek/nLTT

BugReports https://github.com/richelbilderbeek/nLTT/issues

Author Thijs Janzen [aut, cre], Richel Bilderbeek [aut]

Maintainer Thijs Janzen <thijsjanzen@gmail.com>

Repository CRAN

Date/Publication 2016-10-12 00:10:19

R topics documented:

`nLTT-package` *Package providing functions to visualize the normalized Lineage-Through-Time statistic, and calculate the difference between two nLTT curves*

Description

This package provides a function to visualize the normalized Lineage-Through-Time (nLTT) statistic, where the number of lineages relative to the maximum number of lineages in a phylogenetic tree is plotted against the relative time between the most common recent ancestor and the present. Furthermore the package provides a function to calculate the difference between two nLTT curves, including two different distance measurements.

Updates:
Version 1.3.1: Added walkthrough vignette, and updated several typos in the manual
Version 1.3: Version 1.3 adds a lot of extended functionality: firstly, we have added functions to calculate, and plot, the average nLTT across a number of phylogenies. Furthermore, we have added vignettes, and we have added a GitHub repository. On the GitHub repository the vignettes are separately accessible through the wiki. Lastly we have added an extra option to the nLTT functions, where the user can specify if the used trees are rooted, or not. Under de hood, some changes have been made as well, the majority of the code is now conforming to the lintR code conventions, and we have written formalized tests that check correctness of all code (code coverage 100

Version 1.2.1: updated comments and coding style to adhere to the general coding rules. Backwards compatibility has been favoured for the nLTT stat functions. ABC related functions are no longer backwards compatible (variable names have been changed to adhere to coding style).
Version 1.2: added an "exact" nLTT function. This function is faster for small trees, and provides an exact measurement of the nLTT function. Comparison between "old" and "exact" estimates show that these are highly correlated, although the "exact" values are slightly higher than the "old" values. The "exact" function should generally be preferred, unless dealing with extremely large trees (500+ tips) in which case the old function is much faster.

Version 1.2: updated the example for the ABC_SMC_nLTT function, prior generating and prior density functions are now more realistic

Version 1.1.1: fixed a minor bug in the ABC_SMC_nLTT function
Version 1.1.1: removed some intermediate output in ABC_SMC_nLTT function
Version 1.1: Made a universal nLTT function called "nLTTstat", with argument "distanceMethod", this serves as a more elegant wrapper for the functions "normLTTdiffABS" and "normLTTdiffSQ"
Version 1.1: Updated references in the manual

Details

Package:	nLTT
Type:	Package
Version:	1.3.1
Date:	2016-10-09
License:	GPL 2.0

Author(s)

Thijs Janzen

Maintainer: Thijs Janzen <thijsjanzen@gmail.com>

References

Janzen,T. Hoehna,S., Etienne,R.S. (2015) Approximate Bayesian Computation of diversification rates from molecular phylogenies: introducing a new efficient summary statistic, the nLTT. Methods in Ecology and Evolution. doi: 10.1111/2041-210X.12350

`abc_smc_nltt` *A function to perform Approximate Bayesian Computation within an Sequential Markov Chain (ABC-SMC), for diversification analysis of phylogenetic trees.*

Description

This function performs ABC-SMC as described in Toni 2009 for given diversification model, provided a phylogenetic tree. ABC-SMC is not limited to only using the normalized LTT as statistic.

Usage

```
abc_smc_nltt(
    tree, statistics, simulation_function, init_epsilon_values,
    prior_generating_function, prior_density_function,
    number_of_particles = 1000, sigma = 0.05, stop_rate = 1e-05
  )
```

Arguments

tree	an object of class "phylo"; the tree upon which we want to fit our diversification model
statistics	A vector containing functions that take a tree as an argument and return a single scalar value (the statistic).
simulation_function	A function that implements the diversification model and returns an object of class "phylo".
init_epsilon_values	A vector containing the initial threshold values for the summary statistics from the vector statistics.
prior_generating_function	Function to generate parameters from the prior distribution of these parameters (e.g. a function returning lambda and mu in case of the birth-death model)
prior_density_function	Function to calculate the prior probability of a set of parameters.
number_of_particles	Number of particles to be used per iteration of the ABC-SMC algorithm.
sigma	Standard deviation of the perturbance distribution (perturbance distribution is a gaussian with mean 0).
stop_rate	If the acceptance rate drops below stopRate, stop the ABC-SMC algorithm and assume convergence.

Value

A matrix with n columns, where n is the number of parameters you are trying to estimate.

Author(s)

Thijs Janzen

References

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., & Stumpf, M.P.H. (2009). Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface, 6(31), 187-202.

Examples

```
## Not run:

prior_gen <- function() {
  return ( rexp(n=2, rate=0.1) )
}

prior_dens <- function(val) {
  return ( dexp( val[1], rate = 0.1) * dexp( val[2], rate = 0.1) )
}
```

```
require(TESS)

treeSim <- function(params) {
  t <- TESS.sim.age(n=1, lambda = params[1], mu = params[2], age = 10)[[1]]
  return (t)
}

obs <- treeSim(c(0.5,0.1))

statWrapper <- function(tree1) {
  return ( nLTTstat_exact(tree1, obs, "abs"))
}

stats <- c(statWrapper)

results <-  abc.smc.nltt(
  obs, stats, treeSim, init_epsilon_values = 0.2,
  prior_generating_function = prior_gen,
  prior_density_function = prior_dens,
  number_of_particles = 1000, sigma = 0.05, stop_rate = 1e-5
)


## End(Not run) # end of dontrun
```

`exampleTrees` *example trees to test the functionality of the package*

Description

100 phylogenetic trees of class `phylo`, generated using the sim.globalBiDe.age function from the `TESS` package, with lambda = 0.3, mu = 0.1, age = 10.

Usage

```
data(exampleTrees)
```

Format

A list containing objects of class `phylo`.

Examples

```
data(exampleTrees);
obs <- exampleTrees[[1]];
nltt_plot(obs);
```

get_average_nltt_matrix *Get the average nLTT from a collection of phylogenies*

Description

Get the average nLTT from a collection of phylogenies

Usage

```
get_average_nltt_matrix(phylogenies, dt = 0.001)
```

Arguments

phylogenies	the phylogenies, supplied as either a list or a multiPhylo object, where the phylogenies are of type 'phylo'
dt	The timestep resolution, where 1/dt is the number of points evaluated

Value

A matrix of timepoints with the average number of (normalized) lineages through (normalized) time

Author(s)

Richel Bilderbeek

Examples

```
get_average_nltt_matrix(c(ape::rcoal(10), ape::rcoal(20)))
```

get_nltt_values *Collects the nLTT values of all phylogenies in the melted/uncast/long form*

Description

Collects the nLTT values of all phylogenies in the melted/uncast/long form

Usage

```
get_nltt_values(phylogenies, dt)
```

Arguments

phylogenies	the phylogenies, supplied as either a list or a multiPhylo object, where the phylogenies are of type 'phylo'
dt	The timestep resolution, where 1/dt is the number of points evaluated

Value

A dataframe of timepoints with the nLTT value of each phylogeny in time

Author(s)

Richel Bilderbeek

Examples

```
library(ape)
library(ggplot2)
library(nLTT)

# Create some random phylogenies
phylogeny1 <- rcoal(10)
phylogeny2 <- rcoal(20)
phylogeny3 <- rcoal(30)
phylogeny4 <- rcoal(40)
phylogeny5 <- rcoal(50)
phylogeny6 <- rcoal(60)
phylogeny7 <- rcoal(70)
phylogenies <- c(phylogeny1, phylogeny2, phylogeny3,
  phylogeny4, phylogeny5, phylogeny6, phylogeny7
)

# Obtain the nLTT values
dt <- 0.2
nltt_values <- get_nltt_values(phylogenies, dt = dt)

# Check properties of nltt_values
testit::assert(names(nltt_values) == c("id", "t", "nltt"))
nltt_values_per_phylogeny <- (1 + (1 / dt))
n_phylogenies <- length(phylogenies)
testit::assert(nrow(nltt_values)
  == nltt_values_per_phylogeny * n_phylogenies
)

# Plot the phylognies, where the individual nLTT values are visible
qplot(t, nltt, data = nltt_values, geom = "point",
  ylim = c(0,1),
  main = "Average nLTT plot of phylogenies", color = id, size = I(0.1)
) + stat_summary(
  fun.data = "mean_cl_boot", color = "red", geom = "smooth"
)
```

```
# Plot the phylognies, where the individual nLTT values are omitted
qplot(
  t, nltt, data = nltt_values, geom = "blank", ylim = c(0,1),
  main = "Average nLTT plot of phylogenies"
) + stat_summary(
  fun.data = "mean_cl_boot", color = "red", geom = "smooth"
)
```

`get_phylogeny_nltt_matrix` *Extract the nLTT matrix from a phylogeny*

Description

Extract the nLTT matrix from a phylogeny

Usage

```
get_phylogeny_nltt_matrix(phylogeny)
```

Arguments

`phylogeny`	A phylogeny of type phylo

Value

a matrix

Author(s)

Richel Bilderbeek

`mcmc_nltt` *Code to perform Metropolis-Hastings MCMC for a diversification model, given a phylogenetic tree. This function was used in the MEE paper to calculate the likelihood reference estimates.*

Description

This function performs Metropolis-Hastings MCMC, where the user provides a likelihood function and a phylogenetic tree.

Usage

```
mcmc_nltt(phy, likelihood_function, parameters, logtransforms,
  iterations, burnin = round(iterations / 3), thinning = 1, sigma=1)
```

Arguments

phy	an object of class "phylo"; the tree upon which we want to fit our diversification model
likelihood_function	Function that calculates the likelihood of our diversification model, given the tree. Function should me of the format function(parameters,phy).
parameters	Initial parameters to start the chain.
logtransforms	Whether to perform jumps on logtransformed parameters (TRUE) or not (FALSE)
iterations	Length of the chain
burnin	Length of the burnin, default is 30
thinning	Size of thinning, default = 1
sigma	Standard deviation of the jumping distribution, which is N(0, sigma).

Value

An MCMC object, as used by the package "coda".

Author(s)

Sebastian Hoehna & Thijs Janzen

Examples

```
## MCMC examples are typically very slow ####
## Not run:

require(TESS);

obs <- TESS.sim.age(n = 1, lambda = 0.5, mu = 0.1, age = 10)[[1]];

LL_BD <- function(params, phy) {
 lnl <- tess.likelihood(phy, lambda = params[1], mu = params[2],
    samplingProbability = 1, log = TRUE);
 prior1 <- dunif( params[1], 0, 100, log = TRUE)
 prior2 <- dunif( params[2], 0, 100, log = TRUE);
 return(lnl + prior1 + prior2);
}

require(coda);

mcmc_out <- mcmc_nltt(obs, LL_BD, c(0.5, 0.1), c(TRUE, TRUE),
iterations = 1000, burnin = 100, thinning = 10, sigma = 1)
plot(mcmc_out);

## End(Not run)
```

nLTTstat *Calculate the difference between two normalized Lineage-Through-Time curves, given two phylogenetic trees.*

Description

This function takes two ultrametric phylogenetic trees, calculates the normalized Lineage-Through-Time statistic for both trees and then calculates the difference between the two statistics.

Usage

```
nLTTstat(tree1, tree2, distance_method = "abs")
```

Arguments

tree1	an object of class "phylo"
tree2	an object of class "phylo"
distance_method	Chosen measurement of distance between the two nLTT curves, options are (case sensitive): - "abs": use the absolute distance - "squ": use the squared distance;

Value

The difference between the two nLTT statistics

Author(s)

Thijs Janzen

Examples

```
data(exampleTrees)
nltt_plot(exampleTrees[[1]])
nltt_lines(exampleTrees[[2]], lty=2)
nLTTstat(exampleTrees[[1]], exampleTrees[[2]], distance_method = "abs")
```

nLTTstat_exact *Calculate the exact difference between two normalized Lineage-Through-Time curves, given two phylogenetic trees.*

Description

This function takes two ultrametric phylogenetic trees, calculates the normalized Lineage-Through-Time statistic for both trees and then calculates the exact difference between the two statistics. Whereas the function `nLTTstat` uses an approximation to calculate the difference (which is faster for large trees), the function `nLTTstat_exact` calculates the exact difference, and should generally be preferred. Although the estimates are highly similar, `nLTTstat_exact` tends to return slightly higher values.

Usage

```
nLTTstat_exact(tree1, tree2, distance_method = "abs", ignore_stem = TRUE)
```

Arguments

`tree1`	an object of class `"phylo"`
`tree2`	an object of class `"phylo"`
`distance_method`	Chosen measurement of distance between the two nLTT curves, options are (case sensitive): - "abs": use the absolute distance. - "squ": use the squared distance
`ignore_stem`	a boolean whether to ignore the stem length

Value

The exact difference between the two nLTT statistics

Author(s)

Thijs Janzen

Examples

```
data(exampleTrees)
nltt_plot(exampleTrees[[1]])
nltt_lines(exampleTrees[[2]], lty = 2)
nLTTstat_exact(
  exampleTrees[[1]],
  exampleTrees[[2]],
  distance_method = "abs",
  ignore_stem = TRUE
)
```

nltts_plot *Get the average nLTT from a collection of phylogenies*

Description

Get the average nLTT from a collection of phylogenies

Usage

```
nltts_plot(phylogenies, dt = 0.001, plot_nltts = FALSE,
  xlab = "Normalized Time", ylab = "Normalized Lineages", replot = FALSE,
  ...)
```

Arguments

phylogenies	the phylogenies, where the phylogenies are of type 'phylo'
dt	The timestep resolution, where 1/dt is the number of points evaluated
plot_nltts	Also plot each nLLT line
xlab	Label on the x axis
ylab	Label on the y axis
replot	If false, start a clean plot. If true, plot the new data over the current
...	Plotting options

Value

Nothing

Author(s)

Richel Bilderbeek

Examples

```
nltts_plot(c(ape::rcoal(10), ape::rcoal(10)))
nltts_plot(c(ape::rcoal(10), ape::rcoal(20)), dt = 0.1)
```

`nltt_diff` *Calculates the exact difference between the lineage through time curves of tree1 & tree2 (normalized in time and for the number of lineages)*

Description

Calculates the exact difference between the lineage through time curves of tree1 & tree2 (normalized in time and for the number of lineages)

Usage

```
nltt_diff(tree1, tree2, distance_method = "abs")
```

Arguments

`tree1`	(phylo) First phylogenetic tree
`tree2`	(phylo) Second phylogenetic tree
`distance_method`	(string) absolute, or squared distance?

Value

(scalar) normalized Lineage-Through-Time difference between tree1 & tree2

Author(s)

Thijs Janzen

`nltt_diff_exact` *Calculates the exact, difference between the lineage through time curves of tree1 & tree2 (normalized in time and for the number of lineages)*

Description

Calculates the exact, difference between the lineage through time curves of tree1 & tree2 (normalized in time and for the number of lineages)

Usage

```
nltt_diff_exact(tree1, tree2, distance_method = "abs", ignore_stem = TRUE)
```

Arguments

tree1	(phylo) First phylogenetic tree
tree2	(phylo) Second phylogenetic tree
distance_method	(string) absolute, or squared distance?
ignore_stem	(logical) Should the phylogeny its stem be ignored?

Value

(scalar) normalized Lineage-Through-Time difference between tree1 & tree2

Author(s)

Thijs Janzen

---

nltt_lines *Normalized version of the ape function ltt.lines.*

Description

This is a modified version of the ape function ltt.lines: add the normalized Lineage-Through-Time statistic of a phylogenetic tree to an already existing plot

Usage

```
nltt_lines(phy, ...)
```

Arguments

phy	an object of class "phylo"
...	further graphical arguments that can be passed to lines()

Author(s)

Thijs Janzen

Examples

```
data(exampleTrees)
nltt_plot(exampleTrees[[1]])
nltt_lines(exampleTrees[[2]], lty=2)
```

nltt_plot *Normalized version of the ape function ltt.plot*

Description

This function uses a modified version of the ltt.plot function from "ape" to plot the normalized number of lineages through normalized time, where the number of lineages is normalized by dividing by the number of tips of the tree, and the time is normalized by the total time between the most common recent ancestor and the present, such that t(MRCA) = 0 & t(present) = 1.

Usage

```
nltt_plot(phy, xlab = "Normalized Time", ylab = "Normalized Lineages", ...)
```

Arguments

phy	an object of class "phylo"
xlab	a character string (or a variable of mode character) giving the label for the x-axis (default is "Normalized Time").
ylab	a character string (or a variable of mode character) giving the label for the y-axis (default is "Normalized Lineages").
...	further graphical arguments that can be passed to plot()

Author(s)

Thijs Janzen

Examples

```
data(exampleTrees)
nltt_plot(exampleTrees[[1]])
```

stretch_nltt_matrix *Stretch matrix 'm' with a timestep resolution of 'dt'*

Description

Stretch matrix 'm' with a timestep resolution of 'dt'

Usage

```
stretch_nltt_matrix(m, dt, step_type)
```

Arguments

m	A matrix of 2 columns and at least 2 rows
dt	The resultion, a value e <0,1]
step_type	can be 'lower' or 'upper'

Value

The stretched matrix

Author(s)

Richel Bilderbeek

Examples

```
m <- matrix( c(c(0.0, 1.0), c(0.5, 1.0)), ncol = 2, nrow = 2)
expected <- matrix(
  c(
    c(0.0, 0.5, 1.0),  # Timepoints
    c(0.5, 0.5, 1.0)   # Values
  ),
  ncol = 2, nrow = 3
)
result <- stretch_nltt_matrix(m = m, dt = 0.5, step_type = "lower")
testit::assert(identical(result, expected))
```

Index