Package: RevGadgets (via r-universe)

November 4, 2024

Type Package

Title Visualization and Post-Processing of 'RevBayes' Analyses

Version 1.2.1

Maintainer Carrie Tribble <ctribble09@gmail.com>

Description Processes and visualizes the output of complex phylogenetic analyses from the 'RevBayes' phylogenetic graphical modeling software.

URL https://github.com/revbayes/RevGadgets, https://revbayes.github.io/tutorials/intro/revgadgets

BugReports https://github.com/revbayes/RevGadgets/issues

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 4.2.0)

Imports ape (>= 5.4), phytools (>= 0.7-70), dplyr (>= 1.0.0), ggtree (>= 3.6.1), tidytree (>= 0.3.4), treeio (>= 1.12.0), ggplot2 (>= 3.4.0), reshape (>= 0.8.8), methods (>= 4.1.0), tidyr (>= 1.1.0), tibble (>= 3.0.1), gginnards (>= 0.0.3), ggplotify (>= 0.0.5), ggpp, ggimage, png (>= 0.1-7), stats (>= 4.0.1), utils (>= 4.0.1), grDevices (>= 4.0.1), deeptime (>= 0.1.0), scales (>= 1.1.1)

Suggests testthat, knitr, rmarkdown, phangorn

Repository https://revbayes.r-universe.dev

RemoteUrl https://github.com/revbayes/revgadgets

RemoteRef HEAD

RemoteSha 2a7ce495aeeb10d9178b640267779ab82b58b122

Contents

`calculateShiftBayesFactor` *Bayes Factors in support of a shift in diversification rates over a given time interval.*

Description

This function computes the Bayes Factor in favor of a rate-shift between time t1 and t2 (t1 < t2). The default assumption (suitable to standard HSMRF and GMRF models) is that the prior probability of a shift is 0.5.

Usage

```
calculateShiftBayesFactor(
  rate_trace,
  time_trace,
  rate_name,
  time_name,
  t1,
  t2,
  prior_prob = 0.5,
  decrease = TRUE,
  return_2lnBF = TRUE
)
```

Arguments

`rate_trace`	(list; no default) The processed Rev output of the rate of interest through time for computation (output of readTrace()).
`time_trace`	(list; no default) The processed Rev output of the change/interval times of the rate of interest through time for computation (output of readTrace()).
`rate_name`	(character; no default) The name of the parameter (e.g. "speciation") for which Bayes Factor is to be calculated.
`time_name`	(character; no default) The name of the interval times (e.g. "interval_times) for the rate change times.
`t1`	(numeric; no default) Support will be assessed for a shift between time t1 and time t2 (t1 < t2).
`t2`	(numeric; no default) Support will be assessed for a shift between time t1 and time t2 (t1 < t2).
`prior_prob`	(numeric; 0.5) The prior probability of a shift over this interval (default of 0.5 applies to standard HSMRF- and GMRF-based models).
`decrease`	(logical; default TRUE) Should support be assessed for a decrease in the parameter (if TRUE) or an increase (if FALSE) between t1 and t2?
`return_2lnBF`	(logical; TRUE) Should the 2ln(BF) be returned (if TRUE) or simply the BF (if FALSE)?

Value

The Bayes Factor.

References

Kass and Raftery (1995) Bayes Factors. *JASA*, **90 (430)**, 773-795.

Examples

```
#' # download the example datasets to working directory
url_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_times.log"
dest_path_times <- "primates_EBD_speciation_times.log"
download.file(url_times, dest_path_times)

url_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_rates.log"
dest_path_rates <- "primates_EBD_speciation_rates.log"
download.file(url_rates, dest_path_rates)

# to run on your own data, change this to the path to your data file
speciation_time_file <- dest_path_times
speciation_rate_file <- dest_path_rates

speciation_times <- readTrace(speciation_time_file, burnin = 0.25)
speciation_rate <- readTrace(speciation_rate_file, burnin = 0.25)

calculateShiftBayesFactor(speciation_rate,
                          speciation_times,
                          "speciation",
                          "interval_times",
                          0.0,40.0,
                          decrease=FALSE)

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_times, dest_path_rates)
```

---

colFun *Color Function*

Description

Produce default RevGadgets colors

Usage

```
colFun(n)
```

Arguments

n	(integer; no default) Number of colors to return. Maximum of 12.

Details

Produces a vector of colors from the default RevGadgets colors of length given by n, maximum of 12 colors.

Value

Character vector of color hex codes.

Examples

```
my_colors <- colFun(2)
```

---

combineTraces *Combine traces*

Description

Combine traces into one trace file

Usage

```
combineTraces(traces, burnin = 0)
```

Arguments

traces	(list of data frames; no default) Name of a list of data frames, such as produced by readTrace().
burnin	(single numeric value; default = 0.0) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations to discard (if value provided is greater than 1) before combining the samples.

Details

Combines multiple traces from independent MCMC replicates into one trace file.

Value

combineTraces() returns a list of data frames of length 1, corresponding to the combination of the provided samples.

Examples

```
#' # download the example dataset to working directory
url_1 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_1.log"
dest_path_1 <- "primates_cytb_GTR_run_1.log"
download.file(url_1, dest_path_1)

url_2 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_2.log"
dest_path_2 <- "primates_cytb_GTR_run_2.log"
download.file(url_2, dest_path_2)

# to run on your own data, change this to the path to your data file
file_1 <- dest_path_1
file_2 <- dest_path_2

# read in the multiple trace files
multi_trace <- readTrace(path = c(file_1, file_2), burnin = 0.0)

# combine samples after discarding 10% burnin
combined_trace <- combineTraces(trace = multi_trace,
                                burnin = 0.1)

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_1, dest_path_2)
```

`densiTreeWithBranchData`

DensiTree-style plot with branch-specific data

Description

This function plots a distribution of trees (e.g obtained from an MCMC inference) with branch-specific rates or other data. The plot is similar to those produced by DensiTree, i.e all the trees are overlapped with each other. The data is expected to be given per node, and will be associated with the branch above its corresponding node. Its values are plotted as a color gradient.

Usage

```
densiTreeWithBranchData(
  tree_files = NULL,
  burnin = 0.1,
```

```
  trees = NULL,
  data = NULL,
  data_name = NULL,
  type = "cladogram",
  consensus = NULL,
  direction = "rightwards",
  scaleX = FALSE,
  width = 1,
  lty = 1,
  cex = 0.8,
  font = 3,
  tip.color = 1,
  adj = 0,
  srt = 0,
  keep_underscores = FALSE,
  label_offset = 0.01,
  scale_bar = TRUE,
  jitter = list(amount = 0, random = TRUE),
  color_gradient = c("red", "yellow", "green"),
  alpha = NULL,
  bias = 1,
  data_intervals = NULL,
  ...
)
```

Arguments

tree_files	vector of tree files in NEXUS format with data attached to the branches/nodes of the tree. All trees should have the same tip labels (the order can change). Either tree_files or both trees and data have to be specified.
burnin	fraction of samples to discard from the tree files as burn-in. Default 0.1.
trees	multiPhylo object or list of trees in phylo format. All trees should have the same tip labels (the order can change). Either tree_files or both trees and data have to be specified.
data	data to be plotted on the tree - expected to be a list of vectors in the same order as the trees, each vector in the order of the tips and nodes of the corresponding tree
data_name	Only used when reading from tree_files. Name of the data to be plotted, if multiple are present.
type	character string specifying the type of phylogeny. Options are "cladogram" (default) or "phylogram".
consensus	A tree or character vector which is used to define the order of the tip labels. If NULL will be calculated from the trees.
direction	a character string specifying the direction of the tree. Options are "rightwards" (default), "leftwards", "upwards" and "downwards".
scaleX	whether to scale trees to have identical heights. Default FALSE.

width	width of the tree edges.
lty	line type of the tree edges.
cex	a numeric value giving the factor scaling of the tip labels.
font	an integer specifying the type of font for the labels: 1 (plain text), 2 (bold), 3 (italic, the default), or 4 (bold italic).
tip.color	color of the tip labels.
adj	a numeric specifying the justification of the text strings of the tip labels: 0 (left-justification), 0.5 (centering), or 1 (right-justification).
srt	a numeric giving how much the labels are rotated in degrees.
keep_underscores	whether the underscores in tip labels should be written as spaces (the default) or left as they are (if TRUE).
label_offset	a numeric giving the space between the nodes and the tips of the phylogeny and their corresponding labels.
scale_bar	whether to add a scale bar to the plot. Default TRUE.
jitter	controls whether to shift trees. a list with two arguments: the amount of jitter and random or equally spaced (see details below)
color_gradient	range of colors to be used for the data, in order of increasing values. Defaults to red to yellow to green.
alpha	transparency parameter for tree colors. If NULL will be set based on the number of trees.
bias	bias applied to the color gradient. See colorRampPalette for more details.
data_intervals	value intervals used for the color gradient. Can be given as a vector of interval boundaries or min and max values.If NULL will be set based on the data.
...	further arguments to be passed to plot.

Details

If no consensus tree is provided, a consensus tree will be computed. This should avoid too many unnecessary crossings of edges. Trees should be rooted, other wise the output may not be visually pleasing. The `jitter` parameter controls whether to shift trees so that they are not exactly on top of each other. If `amount = 0`, no jitter is applied. If `random = TRUE`, the applied jitter is calculated as `runif(n, -amount, amount)`, otherwise `seq(-amount, amount, length=n)`, where `n` is the number of trees.

Value

No return value, produces plot in base R

References

This code is adapted from the densiTree function by Klaus Schliep <klaus.schliep@gmail.com>. densiTree is inspired from the DensiTree program by Remco Bouckaert.

Remco R. Bouckaert (2010) DensiTree: making sense of sets of phylogenetic trees *Bioinformatics*, **26 (10)**, 1372-1373.

Examples

```
# generate random trees & data
trees <- lapply(1:5, function(x) ape::rcoal(5))
data <- lapply(1:5, function(x) stats::runif(9, 1, 10))

# densiTree plot
densiTreeWithBranchData(trees = trees, data = data, width = 2)

# densiTree plot with different colors
densiTreeWithBranchData(trees = trees, data = data,
                        color_gradient = c("green", "blue"), width = 2)
```

---

dropTip *dropTip*

Description

Drop one or multiple tips from your tree

Usage

```
dropTip(tree, tips)
```

Arguments

tree	(list of lists of treedata objects; no default) Name of a list of lists of treedata objects, such as produced by readTrees().
tips	(character or numeric, no default) The tips(s) to drop. Either a single taxon name or node number or vector of such.

Details

Modifies a tree object (in RevGadget's format) by dropping one or more tips from the tree and from any associated data. Wrapper for treeio::drop.tip().

Value

returns a list of list of treedata objects, with the modified tips.

See Also

treeio: drop.tip and ape: drop.tip.

Examples

```
file <- system.file("extdata",
                    "sub_models/primates_cytb_GTR_MAP.tre",
                    package="RevGadgets")
tree <- readTrees(paths = file)
tree_dropped <- dropTip(tree, "Otolemur_crassicaudatus")
```

`geom_stepribbon` *plot geom stepribbon for diversification rates*

Description

Modified from `RmcdrPlugin.KMggplot2` step ribbon plots.

Usage

```
geom_stepribbon(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`stat`	The statistical transformation to use on the data for this layer, either as a `ggproto` `Geom` subclass or as a string naming the stat stripped of the `stat_` prefix (e.g. `"count"` rather than `"stat_count"`)

position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders().
...	Other arguments passed on to layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like colour = "red" or size = 3. They may also be parameters to the paired geom/stat.

Details

geom_stepribbon is an extension of the geom_ribbon, and is optimized for Kaplan-Meier plots with pointwise confidence intervals or a confidence band.

See Also

geom_ribbon geom_stepribbon inherits from geom_ribbon. geom_stepribbon is modified from RcmdrPlugin.KMggplot2::geom_stepribbon.

Examples

```
huron <- data.frame(year = 1875:1972, level = as.vector(LakeHuron))

h <- ggplot2::ggplot(huron, ggplot2::aes(year))

h + geom_stepribbon(ggplot2::aes(ymin = level - 1, ymax = level + 1),
                    fill = "grey70") +
    ggplot2::geom_step(ggplot2::aes(y = level))

# contrast ggplot2::geom_ribbon with geom_stepribbon:
h + ggplot2::geom_ribbon(ggplot2::aes(ymin = level - 1, ymax = level + 1),
                         fill = "grey70") +
    ggplot2::geom_line(ggplot2::aes(y = level))
```

---

getMAP *get MAP*

Description

Calculates the Maximum a Posteriori estimate for the trace of a quantitative variable

Usage

```
getMAP(var)
```

Arguments

var	(numeric vector; no default) Vector of the samples from the trace of a quantitative variable

Details

Uses the SANN method of the optim() function to approximate the MAP estimate

Value

the MAP estimate

See Also

optim

Examples

```
# download the example dataset to working directory
url <-
  "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.log"
dest_path <- "primates_cytb_GTR.log"
download.file(url, dest_path)

# to run on your own data, change this to the path to your data file
file <- dest_path

trace <- readTrace(paths = file)
MAP <- getMAP(trace[[1]]$"pi[1]")

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path)
```

---

matchNodes *match Nodes*

Description

match Nodes

Usage

```
matchNodes(phy)
```

Arguments

phy	(tree in ape format; no default) Tree on which to match nodes

Value

a data frame that translates ape node numbers to RevBayes node numbers

Examples

```
treefile <- system.file("extdata", "bds/primates.tre", package="RevGadgets")
tree <- readTrees(treefile)
map <- matchNodes(tree[[1]][[1]]@phylo)
```

---

plotAncStatesMAP *plot Ancestral States MAP*

Description

Plots the MAP estimates of ancestral states. Can accommodate cladogenetic reconstructions by plotting on shoulders. Defaults to varying the symbols by color to indicate estimated ancestral state and varying the size of the symbol to indicate the posterior probability of that estimate, but symbol shape may also vary to accommodate black and white figures. For more details on the aesthetics options, see parameter details below. For data with many character states (such as chromosome counts), vary the size of the symbol by estimated ancestral state, and vary the posterior probability of that estimate by a color gradient. Text labels at nodes and tips are also available.

Usage

```
plotAncStatesMAP(
  t,
  cladogenetic = FALSE,
  tip_labels = TRUE,
  tip_labels_size = 2,
  tip_labels_offset = 1,
  tip_labels_italics = FALSE,
  tip_labels_formatted = FALSE,
  tip_labels_remove_underscore = TRUE,
  tip_labels_states = FALSE,
  tip_labels_states_size = 2,
  tip_labels_states_offset = 0.1,
  node_labels_as = NULL,
  node_labels_size = 2,
```

```
  node_labels_offset = 0.1,
  node_labels_centered = FALSE,
  node_size_as = "state_posterior",
  node_color_as = "state",
  node_shape_as = NULL,
  node_shape = 19,
  node_color = "default",
  node_size = c(2, 6),
  tip_states = TRUE,
  tip_states_size = node_size,
  tip_states_shape = node_shape,
  state_transparency = 0.75,
  tree_layout = "rectangular",
  timeline = FALSE,
  geo = timeline,
  geo_units = list("epochs", "periods"),
  time_bars = timeline,
  ...
)
```

Arguments

t	(treedata object; none) Output of processAncStates() function containing tree and ancestral states.
cladogenetic	(logical; FALSE) Plot shoulder states of cladogenetic analyses?
tip_labels	(logical; TRUE) Label taxa labels at tips?
tip_labels_size	(numeric; 2) Size of tip labels.
tip_labels_offset	(numeric; 1) Horizontal offset of tip labels from tree.
tip_labels_italics	(logical; FALSE) Italicize tip labels?
tip_labels_formatted	(logical; FALSE) Do the tip labels contain manually added formatting information? Will set parse = TRUE in geom_text() and associated functions to interpret formatting. See ?plotmath for more. Cannot be TRUE if tip_labels_italics = TRUE.
tip_labels_remove_underscore	(logical; TRUE) Remove underscores from tip labels?
tip_labels_states	(logical; FALSE) Optional plotting of text at tips in addition to taxa labels.
tip_labels_states_size	(numeric; 2) Size of state labels at tips. Ignored if tip_labels_states is FALSE.
tip_labels_states_offset	(numeric; 0.1) Horizontal offset of tip state labels. Ignored if tip_labels_states = NULL.

node_labels_as (character; NULL) Optional plotting of text at nodes. Possible values are "state" for the ancestral states , "state_posterior" for posterior probabilities of the estimated ancestral state, "node_posterior" or the posterior probability of the node on the tree, or NULL for not plotting any text at the nodes (default).

node_labels_size (numeric; 2) Size of node labels text. Ignored if node_labels_as = NULL.

node_labels_offset (numeric; 0.1) Horizontal offset of node labels from nodes. Ignored if node_labels_as = NULL.

node_labels_centered (logical; FALSE) Should node labels be centered over the nodes? Defaults to FALSE: adjusting node labels to the right of nodes and left of shoulders.

node_size_as (character; "state_posterior") How to vary size of node symbols. Options are "state_posterior" (default) for posterior probabilities of the estimated ancestral state, "node_posterior" or the posterior probability of the node on the tree, "state" for vary size by the ancestral state itself in cases where there are many character states (e.g. chromosome numbers; we do not recommend this option for characters with few states), or NULL for fixed symbol size.

node_color_as (character; "state") How to vary to color of node symbols. Options are "state" (default) to vary by estimated ancestral states, "state_posterior" for posterior probabilities of the estimated ancestral state, "node_posterior" or the posterior probability of the node on the tree, or NULL to set all as one color.

node_shape_as (character; NULL) Option to vary node symbol by shape. Options are NULL to keep shape constant or "state" to vary shape by ancestral state.

node_shape (integer; 19) Shape type for nodes. If node_shape_as = "state", provide a vector with length of the number of states. See ggplot2 documentation for details: https://ggplot2.tidyverse.org/articles/ggplot2-specs.html#point

node_color ("character"; "default") Colors for node symbols. Defaults to default RevGadgets colors. If node_color_as = "state', provide a vector of length of the character states. If your color vector is labeled with state labels, the legend will be displayed in the order of the labels. If node_color_as = "posterior", provide a vector of length 2 to generate a color gradient.

node_size (numeric; c(2, 6)) Range of sizes, or fixed size, for node symbols. If node_size_as = "state_posterior", "node_posterior", or "state", numeric vector of length two. If node_size_as = NULL, numeric vector of length one. Size regulates the area of the symbol, following ggplot2 best practices: https://ggplot2.tidyverse.org/reference/scale_size.html)

tip_states (logical; TRUE) Plot states of taxa at tips?

tip_states_size (numeric; node_size) Size for tip symbols. Defaults to the same size as node symbols.

tip_states_shape (integer; node_shape) Shape for tip symbols. Defaults to the same as node symbols.

state_transparency (integer; 0.75) Alpha (transparency) of state symbols- varies from 0 to 1.

tree_layout	(character; "rectangular") Tree shape layout, passed to ggtree(). Options are 'rectangular', 'slanted', 'ellipse', 'roundrect', 'fan', 'circular', 'inward_circular', 'radial', 'equal_angle', 'daylight' or 'ape'. When cladogenetic = TRUE, only "rectangular" and 'circular' are available.
timeline	(logical; FALSE) Plot time tree with labeled x-axis with timescale in MYA.
geo	(logical; timeline) Add a geological timeline? Defaults to the same as timeline.
geo_units	(list; list("epochs", "periods")) Which geological units to include in the geo timescale. May be "periods", "epochs", "stages", "eons", "eras", or a list of two of those units.
time_bars	(logical; timeline) Add vertical gray bars to indicate geological timeline units if geo == TRUE or regular time intervals (in MYA) if geo == FALSE.
...	(various) Additional arguments passed to ggtree::ggtree().

Value

A ggplot object

Examples

```
# Standard ancestral state reconstruction example with various aesthetics

# process file
file <- system.file("extdata",
                    "comp_method_disc/ase_freeK.tree",
                    package="RevGadgets")
example <- processAncStates(file,
                            state_labels = c("1" = "Awesome",
                                             "2" = "Beautiful",
                                             "3" = "Cool!"))

# have states vary by color and indicate state pp with size (default)
plotAncStatesMAP(t = example)

# have states vary by color and indicate state pp with size ,
# and add a timeline
plotAncStatesMAP(t = example, timeline = TRUE)

# have states vary by color and symbol, label nodes with pp of states
plotAncStatesMAP(t = example,  node_shape_as = "state",
                 node_size = 4, node_shape = c(15, 17,20),
                 node_size_as = NULL, node_labels_as = "state_posterior")

# black and white figure - state as symbols and state pp with text
plotAncStatesMAP(t = example, node_color_as = NULL,
                 node_shape_as = "state", node_shape =  c(15, 17,20),
                 node_size_as = NULL, node_size = 4,
                 node_labels_as = "state_posterior",
                 node_color = "grey", state_transparency = 1)
```

```
# default with circular tree
plotAncStatesMAP(t = example, tree_layout = "circular")


# Chromosome evolution example

# process file
file <- system.file("extdata",
                    "chromo/ChromEvol_simple_final.tree",
                    package="RevGadgets")
chromo_example <- processAncStates(file, labels_as_numbers = TRUE)

# plot
plotAncStatesMAP(t = chromo_example, node_color_as = "state_posterior",
                 node_size_as = "state", node_color = colFun(2),
                 tip_labels_offset = 0.005, node_labels_as = "state",
                 node_labels_offset = 0, tip_labels_states = TRUE,
                 tip_labels_states_offset = 0, tip_states = FALSE)

# DEC example (cladogenetic)

# process file
file <- system.file("extdata", "dec/simple.ase.tre", package="RevGadgets")
labs <- c("1" = "K", "2" = "O", "3" = "M", "4" = "H", "5" = "KO",
"6" = "KM", "7" = "OM", "8" = "KH", "9" = "OH", "10" = "MH", "11" = "KOM",
"12" = "KOH", "13" = "KMH", "14" = "OMH", "15" = "KOMH")
dec_example <- processAncStates(file, state_labels = labs)

# plot
plotAncStatesMAP(t = dec_example,
                 cladogenetic = TRUE,
                 tip_labels_offset = 0.5)
```

---

plotAncStatesPie *plot Ancestral States Pie*

Description

Plot character states and posterior probabilities as pies on nodes.

Usage

```
plotAncStatesPie(
  t,
  cladogenetic = FALSE,
  tip_labels = TRUE,
  tip_labels_size = 2,
  tip_labels_offset = 1,
```

```
  tip_labels_italics = FALSE,
  tip_labels_formatted = FALSE,
  tip_labels_remove_underscore = TRUE,
  tip_labels_states = FALSE,
  tip_labels_states_size = 2,
  tip_labels_states_offset = 0.1,
  node_labels_as = NULL,
  node_labels_size = 2,
  node_labels_offset = 0.1,
  pie_colors = "default",
  node_pie_size = 1,
  shoulder_pie_size = node_pie_size,
  tip_pies = TRUE,
  tip_pie_size = 0.5,
  node_pie_nudge_x = 0,
  node_pie_nudge_y = 0,
  tip_pie_nudge_x = node_pie_nudge_x,
  tip_pie_nudge_y = node_pie_nudge_y,
  shoulder_pie_nudge_x = node_pie_nudge_x,
  shoulder_pie_nudge_y = node_pie_nudge_y,
  state_transparency = 0.75,
  timeline = FALSE,
  geo = timeline,
  geo_units = list("epochs", "periods"),
  time_bars = timeline,
  ...
)
```

Arguments

t	(treedata object; none) Output of processAncStates() function containing tree and ancestral states.
cladogenetic	(logical; FALSE) Plot shoulder pies of cladogenetic analyses?
tip_labels	(logical; TRUE) Label taxa labels at tips?
tip_labels_size	(numeric; 2) Size of tip labels.
tip_labels_offset	(numeric; 1) Horizontal offset of tip labels from tree.
tip_labels_italics	(logical; FALSE) Italicize tip labels?
tip_labels_formatted	(logical; FALSE) Do the tip labels contain manually added formatting information? Will set parse = TRUE in geom_text() and associated functions to interpret formatting. See ?plotmath for more. Cannot be TRUE if tip_labels_italics = TRUE.
tip_labels_remove_underscore	(logical; TRUE) Remove underscores from tip labels?

`tip_labels_states`
: (logical; FALSE) Optional plotting of text at tips in addition to taxa labels. Will plot the MAP state label.

`tip_labels_states_size`
: (numeric; 2) Size of state labels at tips. Ignored if tip_labels_states is FALSE.

`tip_labels_states_offset`
: (numeric; 0.1) Horizontal offset of tip state labels. Ignored if tip_labels_states = NULL.

`node_labels_as`
: (character; NULL) Optional plotting of text at nodes. Possible values are "state" for the MAP ancestral states, "node_posterior" for the posterior probability of the node on the tree, "state_posterior" for the posterior probability of the MAP, or NULL for not plotting any text at the nodes (default).

`node_labels_size`
: (numeric; 2) Size of node labels text. Ignored if node_labels_as = NULL.

`node_labels_offset`
: (numeric; 0.1) Horizontal offset of node labels from nodes. Ignored if node_labels_as = NULL.

`pie_colors`
: ("character"; "default") Colors for states in pies. If "default", plots the default RevGadgets colors. Provide a character vector of hex codes or other R-readable colors the same length of the number of character states. Names of the vector should correspond to state labels.

`node_pie_size`
: (numeric; 1) Size (diameter) of the pies at nodes.

`shoulder_pie_size`
: (numeric; node_pie_size) Size (diameter) of the pies at shoulders for cladogenetic plots.

`tip_pies`
: (logical; TRUE) Plot pies tips?

`tip_pie_size`
: (numeric; 0.5) Size (diameter) of the pies at tips.

`node_pie_nudge_x`
: (numeric; 0) If pies aren't centered, adjust by nudging

`node_pie_nudge_y`
: (numeric; 0) If pies aren't centered, adjust by nudging

`tip_pie_nudge_x`
: (numeric; node_pie_nudge_x) If pies aren't centered, adjust by nudging

`tip_pie_nudge_y`
: (numeric; node_pie_nudge_y) If pies aren't centered, adjust by nudging

`shoulder_pie_nudge_x`
: (numeric; node_pie_nudge_x) If pies aren't centered, adjust by nudging

`shoulder_pie_nudge_y`
: (numeric; node_pie_nudge_y) If pies aren't centered, adjust by nudging

`state_transparency`
: (integer; 0.75) Alpha (transparency) of state symbols- varies from 0 to 1.

`timeline`
: (logical; FALSE) Plot tree with labeled x-axis with timescale in MYA.

`geo`
: (logical; timeline) Add a geological timeline? Defaults to the same as timeline.

geo_units	(list; list("epochs", "periods")) Which geological units to include in the geo timescale. May be "periods", "epochs", "stages", "eons", "eras", or a list of two of those units.
time_bars	(logical; timeline) Add vertical gray bars to indicate geological timeline units if geo == TRUE or regular time intervals (in MYA) if geo == FALSE.
...	(various) Additional arguments passed to ggtree::ggtree().

Value

A ggplot object

Examples

```
# Standard ancestral state reconstruction example

# process file and assign state labels
file <- system.file("extdata",
                    "comp_method_disc/ase_freeK.tree",
                    package="RevGadgets")
example <- processAncStates(file,
                            state_labels = c("1" = "Awesome",
                                             "2" = "Beautiful",
                                             "3" = "Cool!"))
# plot (this may take a while)
plotAncStatesPie(t = example)

# DEC Biogeographic range evolution example (with timeline)

# process file
file <- system.file("extdata", "dec/simple.ase.tre", package="RevGadgets")

# labels that correspond to each region/ possible combination of regions
labs <- c("1" = "K", "2" = "O", "3" = "M", "4" = "H", "5" = "KO",
          "6" = "KM", "7" = "OM", "8" = "KH", "9" = "OH", "10" = "MH",
          "11" = "KOM", "12" = "KOH", "13" = "KMH", "14" = "OMH",
          "15" = "KOMH")
dec_example  <- processAncStates(file , state_labels = labs)
# Use the state_labels in returned tidytree object to define color palette
# These state_labels may be a subset of the labels you provided
# (not all possible regions may be sampled in the dataset)
colors <- colorRampPalette(colFun(12))(length(dec_example@state_labels))
names(colors) <- dec_example@state_labels

# plot
plotAncStatesPie(t = dec_example, pie_colors = colors, tip_labels_size = 3,
        cladogenetic = TRUE, tip_labels_offset = 0.25, timeline = TRUE,
        geo = FALSE) +
        ggplot2::theme(legend.position = c(0.1, 0.75))
```

plotDiversityOBDP *Plot Diversity Distribution from OBDP Analysis*

Description

Plots the probability distribution of the number of lineages through time inferred with the Occurrence Birth Death Process #'

Usage

```
plotDiversityOBDP(
  Kt_mean,
  xlab = "Time",
  ylab = "Number of lineages",
  xticks_n_breaks = 5,
  col_Hidden = "dodgerblue3",
  col_LTT = "gray25",
  col_Total = "forestgreen",
  col_Hidden_interval = "dodgerblue2",
  col_Total_interval = "darkolivegreen4",
 palette_Hidden = c("transparent", "dodgerblue2", "dodgerblue3", "dodgerblue4", "black"),
  palette_Total = c("transparent", "green4", "forestgreen", "black"),
  line_size = 0.7,
  interval_line_size = 0.5,
  show_Hidden = TRUE,
  show_LTT = TRUE,
  show_Total = TRUE,
  show_intervals = TRUE,
  show_densities = TRUE,
  show_expectations = TRUE,
  use_interpolate = TRUE
)
```

Arguments

Kt_mean	(data.frame; no default) The processed data.frame (output of readOBDP()).
xlab	(character; "Time") The label of the x-axis.
ylab	(character; "Number of lineages") The label of the y-axis.
xticks_n_breaks	(numeric; 5) An integer guiding the number of major breaks.
col_Hidden	(character; "dodgerblue3") The color of the hidden lineages plot line.
col_LTT	(character; "gray25") The color of the LTT plot line.
col_Total	(character; "forestgreen") The color of the total lineages plot line.
col_Hidden_interval	(character; "dodgerblue2") The color of the credible interval lines around the hidden lineages plot.

col_Total_interval
: (character; "darkolivegreen4") The color of the credible interval lines around the total lineages plot.

palette_Hidden
: (character; c("transparent", "dodgerblue2", "dodgerblue3", "dodgerblue4", "black")) The palette of the hidden lineages plot distribution.

palette_Total
: (character; c("transparent", "green4", "forestgreen", "black")) The palette of the total lineages plot distribution.

line_size
: (numeric; 0.7) The width of the lineage plot line.

interval_line_size
: (numeric; 0.5) The width of the credible interval.

show_Hidden
: (boolean; TRUE) Whether to show the plot for hidden lineages.

show_LTT
: (boolean; TRUE) Whether to show the plot for observed lineages.

show_Total
: (boolean; TRUE) Whether to show the plot for total lineages.

show_intervals
: (boolean; TRUE) Whether to show the credible intervals.

show_densities
: (boolean; TRUE) Whether to show the diversity densities.

show_expectations
: (boolean; TRUE) Whether to show the diversity expectations.

use_interpolate
: (boolean; TRUE) Whether to interpolate densities.

Value

A ggplot object

Examples

```
## Not run:
# first run readOBDP()
start_time_trace_file <-
     system.file("extdata", "obdp/start_time_trace.p", package="RevGadgets")
popSize_distribution_matrices_file <-
     system.file("extdata", "obdp/Kt_trace.p", package="RevGadgets")
trees_trace_file <-
     system.file("extdata", "obdp/mcmc_OBDP_trees.p", package="RevGadgets")

Kt_mean <- readOBDP( start_time_trace_file=start_time_trace_file,
                  popSize_distribution_matrices_file=popSize_distribution_matrices_file,
                     trees_trace_file=trees_trace_file )

# then get the customized ggplot object with plotDiversityOBDP()
p <- plotDiversityOBDP( Kt_mean,
                        xlab="Time (My)",
                        ylab="Number of lineages",
                        xticks_n_breaks=21,
                        col_Hidden="dodgerblue3",
                        col_LTT="gray25",
                        col_Total="forestgreen",
                        col_Hidden_interval="dodgerblue2",
```

```
                          col_Total_interval="darkolivegreen4",
                          palette_Hidden=c("transparent", "dodgerblue2", "dodgerblue3",
                                           "dodgerblue4", "black"),
                         palette_Total=c("transparent", "green4", "forestgreen", "black"),
                          line_size=0.7,
                          interval_line_size=0.5,
                          show_Hidden=TRUE,
                          show_LTT=TRUE,
                          show_Total=TRUE,
                          show_intervals=TRUE,
                          show_densities=TRUE,
                          show_expectations=TRUE,
                          use_interpolate=TRUE )

# basic plot
p

# option: add a stratigraphic scale
library(deeptime)
library(ggplot2)
q <- gggeo_scale(p, dat="periods", height=unit(1.3, "line"), abbrv=F, size=4.5, neg=T)
r <- gggeo_scale(q, dat="epochs", height=unit(1.1, "line"), abbrv=F, size=3.5, neg=T,
                 skip=c("Paleocene", "Pliocene", "Pleistocene", "Holocene"))
s <- gggeo_scale(r, dat="stages", height=unit(1, "line"), abbrv=T, size=2.5, neg=T)
s

## End(Not run)
```

`plotDivRates` *Plot Diversification Rates*

Description

Plots the output of an episodic diversification rate analysis

Usage

```
plotDivRates(rates, facet = TRUE)
```

Arguments

`rates`	(list of dataframes; no default) A list of dataframes, such as produced by processDivRates(), containing the data on rates and interval times for each type of rate to be plotted (e.g. speciation rate, etc.).
`facet`	(logical; TRUE) plot rates in separate facets.

Details

Plots the output of episodic diversification rate analyses. Takes as input the output of processDivRates() and plotting parameters. For now, only variable names (under "item") that contain the word "rate" are included in the plot.

The return object can be manipulated. For example, you can change the axis labels, the color palette, whether the axes are to be linked, or the overall plotting style/theme, just as with any ggplot object.

Value

A ggplot object

Examples

```
# download the example datasets to working directory

url_ex_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_times.log"
dest_path_ex_times <- "primates_EBD_extinction_times.log"
download.file(url_ex_times, dest_path_ex_times)

url_ex_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_rates.log"
dest_path_ex_rates <- "primates_EBD_extinction_rates.log"
download.file(url_ex_rates, dest_path_ex_rates)

url_sp_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_times.log"
dest_path_sp_times <- "primates_EBD_speciation_times.log"
download.file(url_sp_times, dest_path_sp_times)

url_sp_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_rates.log"
dest_path_sp_rates <- "primates_EBD_speciation_rates.log"
download.file(url_sp_rates, dest_path_sp_rates)

# to run on your own data, change this to the path to your data file
speciation_time_file <- dest_path_sp_times
speciation_rate_file <- dest_path_sp_rates
extinction_time_file <- dest_path_ex_times
extinction_rate_file <- dest_path_ex_rates

rates <- processDivRates(speciation_time_log = speciation_time_file,
                         speciation_rate_log = speciation_rate_file,
                         extinction_time_log = extinction_time_file,
                         extinction_rate_log = extinction_rate_file,
                         burnin = 0.25)

# then plot results:
p <- plotDivRates(rates = rates);p
```

```
# change the x-axis
p <- p + ggplot2::xlab("Thousands of years ago");p

# change the colors
p <- p + ggplot2::scale_fill_manual(values = c("red",
                                               "green",
                                               "yellow",
                                               "purple")) +
  ggplot2::scale_color_manual(values = c("red",
                                         "green",
                                         "yellow",
                                         "purple"));p

# let's say we don't want to plot relative-extinction rate,
# and use the same y-axis for all three rates
rates <- rates[!grepl("relative-extinction", rates$item),]
p2 <- plotDivRates(rates)
p2 <- p2 + ggplot2::facet_wrap(ggplot2::vars(item), scale = "fixed");p2

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_sp_times, dest_path_ex_times,
            dest_path_sp_rates, dest_path_ex_rates)
```

---

plotFBDTree *Plot FBD tree*

Description

Plots a single FBD tree, such as an MCC or MAP tree.

Usage

```
plotFBDTree(
  tree,
  timeline = FALSE,
  geo = timeline,
  geo_units = list("epochs", "periods"),
  time_bars = timeline,
  node_age_bars = TRUE,
  tip_age_bars = TRUE,
  age_bars_color = "blue",
  age_bars_colored_by = NULL,
  age_bars_width = 1.5,
  node_labels = NULL,
```

```
  node_labels_color = "black",
  node_labels_size = 3,
  node_labels_offset = 0,
  tip_labels = TRUE,
  tip_labels_italics = FALSE,
  tip_labels_formatted = FALSE,
  tip_labels_remove_underscore = TRUE,
  tip_labels_color = "black",
  tip_labels_size = 3,
  tip_labels_offset = 0,
  node_pp = FALSE,
  node_pp_shape = 16,
  node_pp_color = "black",
  node_pp_size = "variable",
  branch_color = "black",
  color_branch_by = NULL,
  line_width = 1,
  label_sampled_ancs = FALSE,
  ...
)
```

Arguments

tree	(list of lists of treedata objects; no default) Name of a list of lists of treedata objects, such as produced by readTrees(). This object should only contain only one summary tree from one trace file. If it contains multiple trees or multiple traces, only the first will be used.
timeline	(logical; FALSE) Plot time tree with labeled x-axis with timescale in MYA. #'
geo	(logical; timeline) Add a geological timeline? Defaults to the same as timeline.
geo_units	(list; list("epochs", "periods")) Which geological units to include in the geo timescale. May be "periods", "epochs", "stages", "eons", "eras", or a list of two of those units.
time_bars	(logical; timeline) Add vertical gray bars to indicate geological timeline units if geo == TRUE or regular time intervals (in MYA) if geo == FALSE.
node_age_bars	(logical; TRUE) Plot time tree with node age bars?
tip_age_bars	(logical; FALSE) Plot node age bars for the tips as well? Useful for plotting serial sampled analyses or fossilized birth-death analyses, or any cases where some tip ages are estimated.
age_bars_color	(character; "blue") Color for node/tip age bars. If age_bars_colored_by specifies a variable (not NULL), you must provide two colors, low and high values for a gradient. Colors must be either R valid color names or valid hex codes.
age_bars_colored_by	(character; NULL) Specify column to color node/tip age bars by, such as "posterior". If null, all age bars plotted the same color, specified by age_bars_color
age_bars_width	(numeric; 1.5) Change line width for age bars

node_labels (character; NULL) Plot text labels at nodes, specified by the name of the corresponding column in the tidytree object. If NULL, no text is plotted.

node_labels_color (character; "black") Color to plot node_labels, either as a valid R color name or a valid hex code.

node_labels_size (numeric; 3) Size of node labels

node_labels_offset (numeric; 0) Horizontal offset of node labels from nodes.

tip_labels (logical; TRUE) Plot tip labels?

tip_labels_italics (logical; FALSE) Plot tip labels in italics?

tip_labels_formatted (logical; FALSE) Do the tip labels contain manually added formatting information? Will set parse = TRUE in geom_text() and associated functions to interpret formatting. See ?plotmath for more. Cannot be TRUE if tip_labels_italics = TRUE.

tip_labels_remove_underscore (logical; FALSE) Should underscores be replaced by spaces in tip labels?

tip_labels_color (character; "black") Color to plot tip labels, either as a valid R color name or a valid hex code.

tip_labels_size (numeric; 3) Size of tip labels

tip_labels_offset (numeric; 1) Horizontal offset of tip labels from tree.

node_pp (logical; FALSE) Plot posterior probabilities as symbols at nodes? Specify symbol aesthetics with node_pp_shape, node_pp_color, and node_pp_size.

node_pp_shape (integer; 1) Integer corresponding to point shape (value between 0-25). See ggplot2 documentation for details: https://ggplot2.tidyverse.org/articles/ggplot2-specs.html#point

node_pp_color (character; "black") Color for node_pp symbols, either as valid R color name(s) or hex code(s). Can be a single character string specifying a single color, or a vector of length two specifying two colors to form a gradient. In this case, posterior probabilities will be indicated by color along the specified gradient.

node_pp_size (numeric or character; 1) Size for node_pp symbols. If numeric, the size will be fixed at the specified value. If a character, it should specify "variable", indicating that size should be scaled by the posterior value. Size regulates the area of the shape, following ggplot2 best practices: https://ggplot2.tidyverse.org/reference/scale_size.html)

branch_color (character; "black") A single character string specifying the color (R color name or hex code) for all branches OR a vector of length 2 specifying two colors for a gradient, used to color the branches according to the variable specified in color_branch_by. If only 1 color is provided and you specify color_branch_by, default colors will be chosen (low = "#005ac8", high = "#fa7850").

color_branch_by
: (character; NULL) Optional name of one quantitative variable in the treedata object to color branches, such as a rate.

line_width
: (numeric; 1) Change line width for branches

label_sampled_ancs
: (logical; FALSE) Label any sampled ancestors? Will inherent tip labels aesthetics for size and color.

...
: (various) Additional arguments passed to ggtree::ggtree().

Details

Plots a single tree, such as an MCC or MAP tree, with special features for plotting the output of fossilized birth-death analyses.

Value

returns a single plot object.

Examples

```
file <- system.file("extdata", "fbd/bears.mcc.tre", package="RevGadgets")
tree <- readTrees(paths = file)
plotFBDTree(tree = tree, timeline = TRUE, tip_labels_italics = FALSE,
            tip_labels_remove_underscore = TRUE,
            node_age_bars = TRUE, age_bars_colored_by = "posterior",
            age_bars_color = rev(colFun(2))) +
  ggplot2::theme(legend.position=c(.25, .85))
```

---

plotHiSSE *plotHiSSE*

Description

plotHiSSE

Usage

```
plotHiSSE(rates)
```

Arguments

rates
: (data.frame; no default) a data frame containing columns "value", "rate", "hidden_state", "observed_state" (such as the output of processSSE())

Value

a ggplot object

Examples

```
# download the example dataset to working directory

url <- "https://revbayes.github.io/tutorials/intro/data/primates_HiSSE_2.log"
dest_path <- "primates_HiSSE_2.log"
download.file(url, dest_path)

# to run on your own data, change this to the path to your data file
hisse_file <- dest_path

pdata <- processSSE(hisse_file)
p <- plotHiSSE(pdata);p

# change colors:
p + ggplot2::scale_fill_manual(values = c("red","green"))

# change x-axis label
p + ggplot2::xlab("Rate (events/Ma)")

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path)
```

---

plotMassExtinctions *Plot Mass Extinction Support*

Description

Plots the support (as 2ln Bayes factors) for mass extinctions.

Usage

```
plotMassExtinctions(
  mass_extinction_trace,
  mass_extinction_times,
  mass_extinction_name,
  prior_prob,
  return_2lnBF = TRUE
)
```

Arguments

`mass_extinction_trace`
: (list; no default) The processed Rev output of the mass extinction probabilities (output of readTrace()).

`mass_extinction_times`
: (numeric; no default) Vector of the fixed grid of times at which mass extinctions were allowed to occur.

`mass_extinction_name`
: (character; no default) The name of the mass extinction probability parameter (e.g. "mass_extinction_probabilities") for which support is to be calculated/plotted.

`prior_prob`
: (numeric; no default) The per-interval prior probability of a mass extinction (one minus the p parameter in RevBayes' dnReversibleJumpMixture()).

`return_2lnBF`
: (logical; TRUE) Should the 2ln(BF) be returned (if TRUE) or simply the BF (if FALSE)?

Details

Works only for analyses with a fixed grid where mass extinctions may occur.

The return object can be manipulated. For example, you can change the axis labels, the color palette, whether the axes are to be linked, or the overall plotting style/theme, just as with any ggplot object.

Value

A ggplot object

References

Kass and Raftery (1995) Bayes Factors. *JASA*, **90 (430)**, 773-795.

Examples

```
# download the example dataset to working directory
url <-
 "https://revbayes.github.io/tutorials/intro/data/crocs_mass_extinction_probabilities.log"
dest_path <- "crocs_mass_extinction_probabilities.log"
download.file(url, dest_path)

# to run on your own data, change this to the path to your data file
mass_extinction_probability_file <- dest_path

mass_extinction_probabilities <-
              readTrace(mass_extinction_probability_file,burnin = 0.25)

# prior probability of mass extinction at any time
prior_n_expected <- 0.1
n_intervals <- 100
prior_prob <- prior_n_expected/(n_intervals-1)

# times when mass extinctions were allowed
tree_age <- 243.5
interval_times <- tree_age * seq(1/n_intervals,(n_intervals-1) /
                                                  n_intervals,1/n_intervals)
```

```
# then plot results:
p <- plotMassExtinctions(mass_extinction_trace=mass_extinction_probabilities,
                         mass_extinction_times=interval_times,
                         mass_extinction_name="mass_extinction_probabilities"
                         ,prior_prob);p

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path)
```

---

plotMuSSE *plotMuSSE*

Description

plotMuSSE

Usage

```
plotMuSSE(rates)
```

Arguments

rates	(data.frame; no default) a data frame containing columns "value", "rate", "hidden_state", "observed_state" (such as the output of processSSE())

Value

a ggplot object

Examples

```
# download the example dataset to working directory

url <-
  "https://revbayes.github.io/tutorials/intro/data/primates_BiSSE_activity_period.log"
dest_path <- "primates_BiSSE_activity_period.log"
download.file(url, dest_path)

# to run on your own data, change this to the path to your data file
bisse_file <- dest_path

pdata <- processSSE(bisse_file)
p <- plotMuSSE(pdata);p

# change colors:
```

```
p + ggplot2::scale_fill_manual(values = c("red","green"))

# change x-axis label
p + ggplot2::xlab("Rate (events/Ma)")

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path)
```

`plotPopSizes` *Plot Population Sizes*

Description

Plots the output of a coalescent demographic analysis.

Usage

```
plotPopSizes(
  df,
  plot_CIs = TRUE,
  add = FALSE,
  existing_plot = NULL,
  col = "#00883a"
)
```

Arguments

`df`	(data frame) such as produced by processPopSizes(), containing the data on population sizes and corresponding grid points (points in time for population size evaluation)
`plot_CIs`	(boolean; default: TRUE) specifies whether the credible intervals should be plotted.
`add`	(boolean; default: FALSE) specifies whether the new plot should be added to an existing ggplot2 object. If TRUE, the existing_plot has to be given.
`existing_plot`	(ggplot2 object; default: NULL) a ggplot2 object to which the new plot should be added.
`col`	(string; default: "#00883a") color for the trajectories

Details

Plots the output of coalescent demographic analyses. Takes as input the output of processPopSizes() and plotting parameters.

The return object can be manipulated. For example, you can change the axis labels, the color palette, whether the axes are to be linked, or the overall plotting style/theme, just as with any ggplot object.

Value

a ggplot object

Examples

```
df <- dplyr::tibble("time" = c(0.0, 1.0, 2.0, 3.0, 4.0),
                    "value" = c(1.0, 1.5, 2.0, 1.5, 1.5),
                    "upper" = c(3.5, 7.0, 6.5, 5.0, 5.0),
                    "lower" = c(0.5, 0.1, 0.5, 0.5, 0.8))

plotPopSizes(df)
```

---

plotPostPredStats *plot Posterior Predictive Statistics*

Description

Plots the posterior predictive statistics data

Usage

```
plotPostPredStats(
  data,
  prob = c(0.9, 0.95),
  col = NULL,
  side = "both",
  type = "strict",
  PPES = FALSE,
  ...
)
```

Arguments

data	(list of data frames; no default) A list of data frames of the empirical and simulated values, such as the output of processPostPredStats.R
prob	(vector of numerics; default c(0.9, 0.95)) The posterior-predictive intervals to shade.
col	(vector of colors; default NULL) The colors for each quantile. Defaults to blue and red.
side	(character; default "both") Whether the plotted/colored intervals are on "both" sides, the "left" side, or the "right" side of the distribution.
type	(character; default "strict") Whether equal values are considered as less extreme as the observed data ("strict") or half of the equal values are considered to be higher and half to be lower ("midpoint")
PPES	(boolean; default FALSE) Whether we provide the posterior predictive effect size (PPES).
...	Additional arguments are passed to stats::density().

Details

Produces one ggplot object per metric. Intended to plot the results of the RevBayes tutorial: Assessing Phylogenetic Reliability Using RevBayes and P3 Model adequacy testing using posterior prediction (Data Version).

Each plot shows the rejection region for the provided quantiles, as well as a p-value for the observed statistic. If side="left" (or "right"), then the p-value is the fraction of simulated statistics that are less than (or greater than) or equal to the observed statistic. If side="both", then the p-value is calculated by first fitting a KDE to the samples, then computing the fraction of simulated statistics with density lower than the density of he observed statistic; in this sense, the "both" option computes the size of HPD defined by the observed statistic.

Value

A list of ggplot objects, where each plot contains a density distribution of the predicted values and a dashed line of the empirical value. The blue shaded region of the density plot corresponds to the 5% two-sided quantile and the orange corresponds to the 2% two-sided quantile.

Examples

```
# download the example datasets to working directory

url_emp <-
   "https://revbayes.github.io/tutorials/intro/data/empirical_data_pps_example.csv"
dest_path_emp <- "empirical_data_pps_example.csv"
download.file(url_emp, dest_path_emp)

url_sim <-
   "https://revbayes.github.io/tutorials/intro/data/simulated_data_pps_example.csv"
dest_path_sim <- "simulated_data_pps_example.csv"
download.file(url_sim, dest_path_sim)

# to run on your own data, change this to the path to your data file
file_sim <- dest_path_sim
file_emp <- dest_path_emp

t <- processPostPredStats(path_sim = file_sim,
                          path_emp = file_emp)
plots <- plotPostPredStats(data = t)
plots[[1]]

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_sim, dest_path_emp)
```

plotTrace *Plot trace*

Description

Plots the posterior distributions of variables from trace file.

Usage

```
plotTrace(trace, color = "default", vars = NULL, match = NULL)
```

Arguments

trace	(list of data frames; no default) Name of a list of data frames, such as produced by readTrace(). If the readTrace() output contains multiple traces (such as from multiple runs), summarizeTrace() will provide summaries for each trace individually, as well as the combined trace.
color	("character"; "default") Colors for parameters. Defaults to default RevGadgets colors. For non-default colors, provide a named vector of length of the number of parameters.
vars	(character or character vector; NULL) The specific name(s) of the variable(s) to be summarized.
match	(character; NULL) A string to match to a group of parameters. For example, match = "er" will plot the variables "er[1]", "er[2]", "er[3]", etc.. match will only work if your search string is followed by brackets in one or more of the column names of the provided trace file. match = "er" will only return the exchangeability parameters, but will not plot "Posterior".

Details

Plots the posterior distributions of continuous variables from one or multiple traces (as in, from multiple runs). Shaded regions under the curve represent the 95% credible interval. If multiple traces are provided, plotTrace() will plot each run independently as well as plot the combined output. Note that for variables ith very different distributions, overlaying the plots may result in illegible figures. In these cases, we recommend plotting each parameter separately.

Value

plotTrace() returns a list of the length of provided trace object, plus one combined trace. Each element of the list contains a ggplot object with plots of the provided parameters. These plots may be modified in typical ggplot fashion.

Examples

```
# example with quantitative parameters

# download the example dataset to working directory
url_gtr <-
   "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.log"
dest_path_gtr <- "primates_cytb_GTR.log"
download.file(url_gtr, dest_path_gtr)

# to run on your own data, change this to the path to your data file
file <- dest_path_gtr

one_trace <- readTrace(paths = file)
plots <- plotTrace(trace = one_trace,
                    vars = c("pi[1]","pi[2]","pi[3]","pi[4]"))
plots[[1]]

# add custom colors
plots <- plotTrace(trace = one_trace,
                   vars = c("pi[3]","pi[4]","pi[1]","pi[2]"),
                   color = c("pi[1]" = "green",
                             "pi[2]"= "red",
                             "pi[3]"= "blue",
                             "pi[4]"= "orange"))
plots[[1]]

# make the same plot, using match
plots <- plotTrace(trace = one_trace, match = "pi")
plots[[1]]

#' # remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_gtr)

# plot some qualitative variables

# download the example dataset to working directory
url_rj <- "https://revbayes.github.io/tutorials/intro/data/freeK_RJ.log"
dest_path_rj <- "freeK_RJ.log"
download.file(url_rj, dest_path_rj)

file <- dest_path_rj
trace <- readTrace(path = file)

plots <- plotTrace(trace = trace,
                   vars = c("prob_rate_12", "prob_rate_13",
                            "prob_rate_31", "prob_rate_32"))
plots[[1]]
```

```
# with custom colors
plots <- plotTrace(trace = trace,
                   vars = c("prob_rate_12", "prob_rate_13",
                            "prob_rate_31", "prob_rate_32"),
                   color = c("prob_rate_12" = "green",
                             "prob_rate_13" = "red",
                             "prob_rate_31"= "blue",
                             "prob_rate_32" = "orange"))
plots[[1]]

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_rj)
```

---

plotTree *Plot tree*

Description

Plots a single tree, such as an MCC or MAP tree.

Usage

```
plotTree(
  tree,
  timeline = FALSE,
  geo_units = list("epochs", "periods"),
  geo = timeline,
  time_bars = timeline,
  node_age_bars = FALSE,
  age_bars_color = "blue",
  age_bars_colored_by = NULL,
  age_bars_width = 1.5,
  node_labels = NULL,
  node_labels_color = "black",
  node_labels_size = 3,
  node_labels_offset = 0,
  tip_labels = TRUE,
  tip_labels_italics = FALSE,
  tip_labels_formatted = FALSE,
  tip_labels_remove_underscore = TRUE,
  tip_labels_color = "black",
  tip_labels_size = 3,
  tip_labels_offset = 0,
```

```
  node_pp = FALSE,
  node_pp_shape = 16,
  node_pp_color = "black",
  node_pp_size = "variable",
  branch_color = "black",
  color_branch_by = NULL,
  line_width = 1,
  tree_layout = "rectangular",
  ...
)
```

Arguments

tree	(list of lists of treedata objects; no default) Name of a list of lists of treedata objects, such as produced by readTrees(). This object should only contain only one summary tree from one trace file. If it contains multiple trees or multiple traces, only the first will be used.
timeline	(logical; FALSE) Plot time tree with labeled x-axis with timescale in MYA.
geo_units	(list; list("epochs", "periods")) Which geological units to include in the geo timescale. May be "periods", "epochs", "stages", "eons", "eras", or a list of two of those units.
geo	(logical; timeline) Add a geological timeline? Defaults to the same as timeline.
time_bars	(logical; timeline) Add vertical gray bars to indicate geological timeline units if geo == TRUE or regular time intervals (in MYA) if geo == FALSE.
node_age_bars	(logical; FALSE) Plot time tree with node age bars?
age_bars_color	(character; "blue") Color for node age bars. If age_bars_colored_by specifies a variable (not NULL), you must provide two colors, low and high values for a gradient. Colors must be either R valid color names or valid hex codes.
age_bars_colored_by	(character; NULL) Specify column to color node age bars by, such as "posterior". If null, all node age bars plotted the same color, specified by age_bars_color
age_bars_width	(numeric; 1.5) Change line width for age bars
node_labels	(character; NULL) Plot text labels at nodes, specified by the name of the corresponding column in the tidytree object. If NULL, no text is plotted.
node_labels_color	(character; "black") Color to plot node_labels, either as a valid R color name or a valid hex code.
node_labels_size	(numeric; 3) Size of node labels
node_labels_offset	(numeric; 0) Horizontal offset of node labels from nodes.
tip_labels	(logical; TRUE) Plot tip labels?
tip_labels_italics	(logical; FALSE) Plot tip labels in italics?

`tip_labels_formatted`
: (logical; FALSE) Do the tip labels contain manually added formatting information? Will set parse = TRUE in geom_text() and associated functions to interpret formatting. See ?plotmath for more. Cannot be TRUE if tip_labels_italics = TRUE.

`tip_labels_remove_underscore`
: (logical; TRUE) Remove underscores in tip labels?

`tip_labels_color`
: (character; "black") Color to plot tip labels, either as a valid R color name or a valid hex code.

`tip_labels_size`
: (numeric; 3) Size of tip labels

`tip_labels_offset`
: (numeric; 1) Horizontal offset of tip labels from tree.

`node_pp`
: (logical; FALSE) Plot posterior probabilities as symbols at nodes? Specify symbol aesthetics with node_pp_shape, node_pp_color, and node_pp_size.

`node_pp_shape`
: (integer; 1) Integer corresponding to point shape (value between 0-25). See ggplot2 documentation for details: https://ggplot2.tidyverse.org/articles/ggplot2-specs.html#point

`node_pp_color`
: (character; "black") Color for node_pp symbols, either as valid R color name(s) or hex code(s). Can be a single character string specifying a single color, or a vector of length two specifying two colors to form a gradient. In this case, posterior probabilities will be indicated by color along the specified gradient.

`node_pp_size`
: (numeric or character; 1) Size for node_pp symbols. If numeric, the size will be fixed at the specified value. If a character, it should specify "variable", indicating that size should be scaled by the posterior value. Size regulates the area of the shape, following ggplot2 best practices: https://ggplot2.tidyverse.org/reference/scale_size.html)

`branch_color`
: (character; "black") A single character string specifying the color (R color name or hex code) for all branches OR a vector of length 2 specifying two colors for a gradient, used to color the branches according to the variable specified in color_branch_by. If only 1 color is provided and you specify color_branch_by, default colors will be chosen (low = "#005ac8", high = "#fa7850").

`color_branch_by`
: (character; NULL) Optional name of one quantitative variable in the treedata object to color branches, such as a rate.

`line_width`
: (numeric; 1) Change line width for branches

`tree_layout`
: (character; "rectangular") Tree shape layout, passed to ggtree(). Options are 'rectangular', 'cladogram', 'slanted', 'ellipse', 'roundrect', 'fan', 'circular', 'inward_circular', 'radial', 'equal_angle', 'daylight', or 'ape'.

`...`
: (various) Additional arguments passed to ggtree::ggtree().

Details

Plots a single tree, such as an MCC or MAP tree, with optionally labeled posterior probabilities at nodes, a timescale plotted on the x - axis, and 95% CI for node ages.

Value

returns a single plot object.

Examples

```
# Example of standard tree plot

file <- system.file("extdata",
                    "sub_models/primates_cytb_GTR_MAP.tre",
                    package="RevGadgets")
tree <- readTrees(paths = file)
# Reroot tree before plotting
tree_rooted <- rerootPhylo(tree = tree, outgroup = "Galeopterus_variegatus")
# Plot
p <- plotTree(tree = tree_rooted, node_labels = "posterior");p

# Plot unladderized tree
p <- plotTree(tree = tree_rooted,
              node_labels = "posterior",
              ladderize = FALSE);p

# We can add a scale bar:
p + ggtree::geom_treescale(x = -0.35, y = -1)

# Example of coloring branches by rate
file <- system.file("extdata",
                    "relaxed_ou/relaxed_OU_MAP.tre",
                    package="RevGadgets")
tree <- readTrees(paths = file)
p <- plotTree(tree = tree,
              node_age_bars = FALSE,
              node_pp = FALSE,
              tip_labels_remove_underscore = TRUE,
              tip_labels_italics = FALSE,
              color_branch_by = "branch_thetas",
              line_width = 1.7) +
       ggplot2::theme(legend.position=c(.1, .9));p
```

---

plotTreeFull *Plot Full tree*

Description

Plots a tree, such as an MCC or MAP tree

Usage

```
plotTreeFull(
  tree,
  timeline,
  geo,
  geo_units,
  time_bars,
  node_age_bars,
  tip_age_bars,
  age_bars_color,
  age_bars_colored_by,
  age_bars_width,
  node_labels,
  node_labels_color,
  node_labels_size,
  node_labels_offset,
  tip_labels,
  tip_labels_italics,
  tip_labels_formatted,
  tip_labels_remove_underscore,
  tip_labels_color,
  tip_labels_size,
  tip_labels_offset,
  label_sampled_ancs,
  node_pp,
  node_pp_shape,
  node_pp_color,
  node_pp_size,
  branch_color,
  color_branch_by,
  line_width,
  tree_layout,
  ...
)
```

Arguments

tree	(list of lists of treedata objects; no default) Name of a list of lists of treedata objects, such as produced by readTrees(). This object should only contain only one summary tree from one trace file. If it contains multiple trees or multiple traces, only the first will be used.
timeline	(logical; FALSE) Plot time tree with labeled x-axis with timescale in MYA. #'
geo	(logical; timeline) Add a geological timeline? Defaults to the same as timeline.
geo_units	(list; list("epochs", "periods")) Which geological units to include in the geo timescale. May be "periods", "epochs", "stages", "eons", "eras", or a list of two of those units.

time_bars (logical; timeline) Add vertical gray bars to indicate geological timeline units if geo == TRUE or regular time intervals (in MYA) if geo == FALSE.

node_age_bars (logical; TRUE) Plot time tree with node age bars?

tip_age_bars (logical; FALSE) Plot node age bars for the tips as well? Useful for plotting serial sampled analyses or fossilized birth-death analyses, or any cases where some tip ages are estimated.

age_bars_color (character; "blue") Color for node/tip age bars. If age_bars_colored_by specifies a variable (not NULL), you must provide two colors, low and high values for a gradient. Colors must be either R valid color names or valid hex codes.

age_bars_colored_by
(character; NULL) Specify column to color node/tip age bars by, such as "posterior". If null, all age bars plotted the same color, specified by age_bars_color

age_bars_width (numeric; 1) Change line width for age bars

node_labels (character; NULL) Plot text labels at nodes, specified by the name of the corresponding column in the tidytree object. If NULL, no text is plotted.

node_labels_color
(character; "black") Color to plot node_labels, either as a valid R color name or a valid hex code.

node_labels_size
(numeric; 3) Size of node labels

node_labels_offset
(numeric; 0) Horizontal offset of node labels from nodes.

tip_labels (logical; TRUE) Plot tip labels?

tip_labels_italics
(logical; FALSE) Plot tip labels in italics?

tip_labels_formatted
(logical; FALSE) Do the tip labels contain manually added formatting information? Will set parse = TRUE in geom_text() and associated functions to interpret formatting. See ?plotmath for more. Cannot be TRUE if tip_labels_italics = TRUE.

tip_labels_remove_underscore
(logical; FALSE) Should underscores be replaced by spaces in tip labels?

tip_labels_color
(character; "black") Color to plot tip labels, either as a valid R color name or a valid hex code.

tip_labels_size
(numeric; 3) Size of tip labels

tip_labels_offset
(numeric; 1) Horizontal offset of tip labels from tree.

label_sampled_ancs
(logical; FALSE) Label any sampled ancestors? Will inherent tip labels aesthetics for size and color.

added in from plotTree

node_pp (logical; FALSE) Plot posterior probabilities as symbols at nodes? Specify symbol aesthetics with node_pp_shape, node_pp_color, and node_pp_size.

node_pp_shape	(integer; 1) Integer corresponding to point shape (value between 0-25). See ggplot2 documentation for details: https://ggplot2.tidyverse.org/articles/ggplot2-specs.html#point
node_pp_color	(character; "black") Color for node_pp symbols, either as valid R color name(s) or hex code(s). Can be a single character string specifying a single color, or a vector of length two specifying two colors to form a gradient. In this case, posterior probabilities will be indicated by color along the specified gradient.
node_pp_size	(numeric or character; 1) Size for node_pp symbols. If numeric, the size will be fixed at the specified value. If a character, it should specify "variable", indicating that size should be scaled by the posterior value. Size regulates the area of the shape, following ggplot2 best practices: https://ggplot2.tidyverse.org/reference/scale_size.html)
branch_color	(character; "black") A single character string specifying the color (R color name or hex code) for all branches OR a vector of length 2 specifying two colors for a gradient, used to color the branches according to the variable specified in color_branch_by. If only 1 color is provided and you specify color_branch_by, default colors will be chosen (low = "#005ac8", high = "#fa7850").
color_branch_by	(character; NULL) Optional name of one quantitative variable in the treedata object to color branches, such as a rate.
line_width	(numeric; 1) Change line width for branches
tree_layout	(character; "rectangular") Tree shape layout, passed to ggtree(). Options are 'rectangular', 'cladogram', 'slanted', 'ellipse', 'roundrect', 'fan', 'circular', 'inward_circular', 'radial', 'equal_angle', 'daylight' or 'ape'.
...	(various) Additional arguments passed to ggtree::ggtree().

Details

Plots a single tree, such as an MCC or MAP tree, with the full set of functionality to be called by plotTree() and plotFBDTree()

Value

returns a single plot object.

See Also

called by plotTree and plotFBDTree

---

posteriorSamplesToParametricPrior

Priors from MCMC samples

Description

Turn posterior samples collected by MCMC into a parametric prior distribution.

Usage

```
posteriorSamplesToParametricPrior(
  samples,
  distribution,
  variance_inflation_factor = 2
)
```

Arguments

samples	(numeric vector; no default) MCMC samples for a single parameter.
distribution	(character; no default) The distribution to fit. Options are gamma for strictly positive parameters and normal for unbounded parameters.
variance_inflation_factor	(single numeric value; default = 2.0) Makes the prior variance larger than the variance of the posterior

Details

The distributions are fit by the method of moments. The function allows inflating the prior variance relative to the posterior being supplied.

Value

Numeric vector of parameters with names (to avoid rate/scale and var/sd confusion).

Examples

```
# download the example datasets to working directory

url_ex_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_times.log"
dest_path_ex_times <- "primates_EBD_extinction_times.log"
download.file(url_ex_times, dest_path_ex_times)

url_ex_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_rates.log"
dest_path_ex_rates <- "primates_EBD_extinction_rates.log"
download.file(url_ex_rates, dest_path_ex_rates)

url_sp_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_times.log"
dest_path_sp_times <- "primates_EBD_speciation_times.log"
download.file(url_sp_times, dest_path_sp_times)

url_sp_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_rates.log"
dest_path_sp_rates <- "primates_EBD_speciation_rates.log"
download.file(url_sp_rates, dest_path_sp_rates)
```

```
# to run on your own data, change this to the path to your data file
speciation_time_file <- dest_path_sp_times
speciation_rate_file <- dest_path_sp_rates
extinction_time_file <- dest_path_ex_times
extinction_rate_file <- dest_path_ex_rates

primates <- processDivRates(speciation_time_log = speciation_time_file,
                            speciation_rate_log = speciation_rate_file,
                            extinction_time_log = extinction_time_file,
                            extinction_rate_log = extinction_rate_file,
                            burnin = 0.25)

speciation_rates <-
       dplyr::pull(primates[which(primates$item == "speciation rate"),],
                   "value")
speciation_1_gamma_prior <-
          posteriorSamplesToParametricPrior(speciation_rates,"gamma")

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_sp_times, dest_path_ex_times,
            dest_path_sp_rates, dest_path_ex_rates)
```

`processAncStates` *Process Ancestral States*

Description

Process data for ancestral states plotting

Usage

```
processAncStates(
  path,
  state_labels = NULL,
  labels_as_numbers = FALSE,
  missing_to_NA = TRUE
)
```

Arguments

`path`	(character string; no default) File path to annotated tree.
`state_labels`	(character vector; NULL) Vector of labels for ancestral states named with the current state labels in annotated tree file (as characters).

labels_as_numbers
: (logical; FALSE) Should the state labels be treated as integers (for example, as chromosome numbers)?

missing_to_NA
: (logical; TRUE) Should missing data, coded as "?", be coded to NA? If TRUE, the state will not be plotted. If FALSE, it will be considered an additional state when plotting.

Value

A treedata object

Examples

```
# standard ancestral state estimation example
file <- system.file("extdata",
                    "comp_method_disc/ase_freeK.tree",
                    package="RevGadgets")
example <- processAncStates(file,
                            state_labels = c("1" = "Awesome",
                                             "2" = "Beautiful",
                                             "3" = "Cool!"))

#chromosome evolution example
file <- system.file("extdata",
                    "chromo/ChromEvol_simple_final.tree",
                    package="RevGadgets")
chromo_example <- processAncStates(file, labels_as_numbers = TRUE)
```

---

processBranchData *processBranchData*

Description

processBranchData

Usage

```
processBranchData(
  tree,
  dat,
  burnin = 0.25,
  parnames = c("avg_lambda", "avg_mu", "num_shifts"),
  summary = "median",
  net_div = FALSE
)
```

Arguments

tree	(treedata object; no default) a phylogenetic tree in the treedata format, or a list of lists of a single tree data object, such as the output of readTrees().
dat	(data.frame or list; no default) a data frame, or a list (of length 1) of a data frame, with branch specific data, such as the output of readTrace().
burnin	(numeric; 0.25) fraction of the markov-chain to discard
parnames	(character vector; c("avg_lambda", "avg_mu", "num_shifts")) Names of parameters to process
summary	(character; "median") function to summarize the continuous parameter. Typically mean or median
net_div	(logical; FALSE) Calculate net diversification?

Value

a treedata file with attached branch-specific data

Examples

```
# download the example dataset to working directory
url_rates <-
  "https://revbayes.github.io/tutorials/intro/data/primates_BDS_rates.log"
dest_path_rates <- "primates_BDS_rates.log"
download.file(url_rates, dest_path_rates)

url_tree <-
  "https://revbayes.github.io/tutorials/divrate/data/primates_tree.nex"
dest_path_tree <- "primates_tree.nex"
download.file(url_tree, dest_path_tree)

# to run on your own data, change this to the path to your data file
treefile <- dest_path_tree
logfile <- dest_path_rates

branch_data <- readTrace(logfile)
tree <- readTrees(paths = treefile)

annotated_tree <- processBranchData(tree, branch_data, summary = "median")

# you can plot this output
p <- plotTree(tree = annotated_tree,
              node_age_bars = FALSE,
              node_pp = FALSE,
              tip_labels = FALSE,
              color_branch_by = "avg_lambda",
              line_width = 0.8) +
     ggplot2::theme(legend.position=c(.1, .9));p
# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
```

```
file.remove(dest_path_tree, dest_path_rates)
```

`processDivRates` *Process Diversification Rates*

Description

Processing the output of a episodic diversification rate analysis with mass-extinction events.

Usage

```
processDivRates(
  speciation_time_log = "",
  speciation_rate_log = "",
  extinction_time_log = "",
  extinction_rate_log = "",
  fossilization_time_log = "",
  fossilization_rate_log = "",
  burnin = 0.25,
  probs = c(0.025, 0.975),
  summary = "median"
)
```

Arguments

`speciation_time_log`
: (vector of character strings or single character string; "") Path to speciation times log file(s)

`speciation_rate_log`
: (vector of character strings or single character string; "") Path to speciation rates log file(s)

`extinction_time_log`
: (vector of character strings or single character string; "") Path to extinction times log file(s)

`extinction_rate_log`
: (vector of character strings or single character string; "") Path to extinction rates log file(s)

`fossilization_time_log`
: (vector of character strings or single character string; "") Path to fossilization times log file(s)

`fossilization_rate_log`
: (vector of character strings or single character string; "") Path to fossilization rates log file(s)

`burnin`
: (single numeric value; default = 0) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations (if value provided is greater than 1). Passed to readTrace().

probs	(numeric vector; c(0.025, 0.975)) a vector of length two containing the upper and lower bounds for the confidence intervals.
summary	typically "mean" or "median"; the metric to summarize the posterior distribution. Defaults to "median"

Details

For processing the output of an episodic diversification rate analysis. processDivRates() assumes that the epochs are fixed rather than inferred. Additionally, it assumes that times correspond to rates such that the first rate parameter (i.e. speciation[1]) corresponds to the present. Conversely, the first time parameter (i.e. interval_times[1]) corresponds to the first time interval after the present, moving backwards in time. processDivRates() relies on readTrace and produces a list object that can be read by plotDivRates() to visualize the results. For now, only one log file per parameter type is accepted (i.e. log files from multiple runs must be combined before reading into the function).

Value

List object with processed rate and time parameters.

Examples

```
# download the example datasets to working directory

url_ex_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_times.log"
dest_path_ex_times <- "primates_EBD_extinction_times.log"
download.file(url_ex_times, dest_path_ex_times)

url_ex_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_extinction_rates.log"
dest_path_ex_rates <- "primates_EBD_extinction_rates.log"
download.file(url_ex_rates, dest_path_ex_rates)

url_sp_times <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_times.log"
dest_path_sp_times <- "primates_EBD_speciation_times.log"
download.file(url_sp_times, dest_path_sp_times)

url_sp_rates <-
   "https://revbayes.github.io/tutorials/intro/data/primates_EBD_speciation_rates.log"
dest_path_sp_rates <- "primates_EBD_speciation_rates.log"
download.file(url_sp_rates, dest_path_sp_rates)

# to run on your own data, change this to the path to your data file
speciation_time_file <- dest_path_sp_times
speciation_rate_file <- dest_path_sp_rates
extinction_time_file <- dest_path_ex_times
extinction_rate_file <- dest_path_ex_rates

rates <- processDivRates(speciation_time_log = speciation_time_file,
```

```
                        speciation_rate_log = speciation_rate_file,
                        extinction_time_log = extinction_time_file,
                        extinction_rate_log = extinction_rate_file,
                        burnin = 0.25)

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_sp_times, dest_path_ex_times,
            dest_path_sp_rates, dest_path_ex_rates)
```

`processPopSizes` *Process Population Sizes*

Description

Processing the output of a coalescent demographic analysis.

Usage

```
processPopSizes(
  population_size_log = "",
  interval_change_points_log = "",
  model = "constant",
  burnin = 0.25,
  probs = c(0.025, 0.975),
  summary = "median",
  num_grid_points = 100,
  spacing = "exponential",
  max_age = NULL,
  min_age = NULL,
  distribution = FALSE
)
```

Arguments

`population_size_log`
: (vector of character strings or single character string; "") Path to population sizes log file(s)

`interval_change_points_log`
: (vector of character strings or single character string; "") Path to interval change points log file(s). If not given, a constant process with only one population size is assumed.

`model`
: (string, default: "constant") The demographic model of the intervals. Can be "constant" or "linear".

`burnin` (single numeric value; default: 0.25) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations (if value provided is greater than 1).

`probs` (numeric vector; c(0.025, 0.975)) a vector of length two containing the upper and lower bounds for the confidence intervals.

`summary` (string, default: "median") the metric to summarize the posterior distribution, typically "mean" or "median".

`num_grid_points` (numeric; default: 100) defines the number of grid points through time for which to evaluate the demographic functions.

`spacing` (string, default: "exponential") The spacing of grid points. Can be "exponential" or "equal". Exponentially spaced grid points are dense towards the present and have larger distances towards the past.

`max_age` (numeric; default: NULL, i.e. not provided) defines the maximal age up to which the demographic functions should be evaluated. If not provided, it will either be automatically set to 1e5 (in case of a constant process) or to the maximal age provided with the interval_change_points_log.

`min_age` (numeric; default: NULL, i.e. not provided) defines the minimal age up to which the demographic functions should be evaluated. If not provided, it will either be automatically set to 1e2 (in case of a constant process) or to the minimal age provided with the interval_change_points_log. Can not be 0 in case of exponential spacing.

`distribution` (boolean; default: FALSE) specifies whether the summary data frame will be returned (distribution = FALSE) or a matrix with distributions of population size for each point on the grid and with the times of the grid points as row names (distribution = TRUE).

Details

For processing the output of a coalescent demographic analysis. processPopSizes() assumes that the the first size parameter (i.e. population_size[1]) corresponds to the present. processPopSizes() partly relies on readTrace and produces a list object that can be read by plotPopSizes() to visualize the results. For now, only one log file per parameter type is accepted (i.e. log files from multiple runs must be combined before reading into the function).

Value

List object with processed rate and, if applicable, time parameters (if distribution = FALSE). Matrix object with distributions of population size (if distribution = TRUE). If applicable, one row for each point on the grid, with the times of the grid points as row names.

`processPostPredStats` *process Posterior Predictive Statistics*

Description

Reads in and processes posterior-predictive statistics

Usage

```
processPostPredStats(path_sim, path_emp)
```

Arguments

`path_sim`	(character string; no default) Path to the .csv file containing the simulated data results
`path_emp`	(character string; no default) Path to the .csv file containing the empirical values

Value

A list of data frames

Examples

```
# download the example datasets to working directory

url_emp <-
   "https://revbayes.github.io/tutorials/intro/data/empirical_data_pps_example.csv"
dest_path_emp <- "empirical_data_pps_example.csv"
download.file(url_emp, dest_path_emp)

url_sim <-
   "https://revbayes.github.io/tutorials/intro/data/simulated_data_pps_example.csv"
dest_path_sim <- "simulated_data_pps_example.csv"
download.file(url_sim, dest_path_sim)

# to run on your own data, change this to the path to your data file
file_sim <- dest_path_sim
file_emp <- dest_path_emp

t <- processPostPredStats(path_sim = file_sim,
                          path_emp = file_emp)

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_sim, dest_path_emp)
```

processSSE *Title*

Description

Title

Usage

```
processSSE(
  path,
  speciation = "speciation",
  extinction = "extinction",
  speciation_hidden = "speciation_hidden",
  rates = c(speciation, extinction, "net-diversification"),
  ...
)
```

Arguments

path	(vector of character strings; no default) File path(s) to trace file.
speciation	(single character string; "speciation") RevBayes variable name
extinction	(single character string; "extinction") RevBayes variable name
speciation_hidden	(single character string; "speciation_hidden") RevBayes variable name
rates	(vector; c(speciation, extinction, "net-diversification")) names of rates to be included in plot
...	additional arguments passed to readTrace()

Value

a data frame

Examples

```
# download the example dataset to working directory

url <-
  "https://revbayes.github.io/tutorials/intro/data/primates_BiSSE_activity_period.log"
dest_path <- "primates_BiSSE_activity_period.log"
download.file(url, dest_path)

# to run on your own data, change this to the path to your data file
bisse_file <- dest_path

pdata <- processSSE(bisse_file)
```

```
# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path)
```

---

readOBDP *Read OBDP Outputs*

Description

Reads and formats the outputs of an analysis with the Occurrence Birth Death Process (MCMC parameter inference + diversity estimation)

Usage

```
readOBDP(
  start_time_trace_file,
  popSize_distribution_matrices_file,
  trees_trace_file
)
```

Arguments

`start_time_trace_file`
: (character; no default) Trace of the starting times along the MCMC chain.

`popSize_distribution_matrices_file`
: (character; no default) Kt matrices computed with 'fnInferAncestralPopSize' in RevBayes.

`trees_trace_file`
: (character; no default) Trace of the trees.

Value

A data.frame

Examples

```
## Not run:
# first run readOBDP()
start_time_trace_file <-
     system.file("extdata", "obdp/start_time_trace.p", package="RevGadgets")
popSize_distribution_matrices_file <-
     system.file("extdata", "obdp/Kt_trace.p", package="RevGadgets")
trees_trace_file <-
     system.file("extdata", "obdp/mcmc_OBDP_trees.p", package="RevGadgets")
```

```
Kt_mean <- readOBDP( start_time_trace_file=start_time_trace_file,
                  popSize_distribution_matrices_file=popSize_distribution_matrices_file,
                     trees_trace_file=trees_trace_file )

# then get the customized ggplot object with plotDiversityOBDP()
p <- plotDiversityOBDP( Kt_mean,
                        xlab="Time (My)",
                        ylab="Number of lineages",
                        xticks_n_breaks=21,
                        col_Hidden="dodgerblue3",
                        col_LTT="gray25",
                        col_Total="forestgreen",
                        col_Hidden_interval="dodgerblue2",
                        col_Total_interval="darkolivegreen4",
                        palette_Hidden=c("transparent", "dodgerblue2", "dodgerblue3",
                                         "dodgerblue4", "black"),
                       palette_Total=c("transparent", "green4", "forestgreen", "black"),
                        line_size=0.7,
                        interval_line_size=0.5,
                        show_Hidden=TRUE,
                        show_LTT=TRUE,
                        show_Total=TRUE,
                        show_intervals=TRUE,
                        show_densities=TRUE,
                        show_expectations=TRUE,
                        use_interpolate=TRUE )

# basic plot
p

# option: add a stratigraphic scale
library(deeptime)
library(ggplot2)
q <- gggeo_scale(p, dat="periods", height=unit(1.3, "line"), abbrv=F, size=4.5, neg=T)
r <- gggeo_scale(q, dat="epochs", height=unit(1.1, "line"), abbrv=F, size=3.5, neg=T,
                    skip=c("Paleocene", "Pliocene", "Pleistocene", "Holocene"))
s <- gggeo_scale(r, dat="stages", height=unit(1, "line"), abbrv=T, size=2.5, neg=T)
s

## End(Not run)
```

---

readTrace *Read trace*

Description

Reads in MCMC log files

Usage

```
readTrace(
  paths,
  format = "simple",
  delim = "\t",
  burnin = 0.1,
  check.names = FALSE,
  ...
)
```

Arguments

paths	(vector of character strings; no default) File path(s) to trace file.
format	(single character string; default = simple) Indicates type of MCMC trace, complex indicates cases where trace contains vectors of vectors/ matrices - mnStochasticVariable monitor will sometimes be of this type.
delim	(single character string; default = "\t") Delimiter of file.
burnin	(single numeric value; default = 0.1) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations (if value provided is greater than 1).
check.names	(logical; default = FALSE) Passed to utils::read.table(); indicates if utils::read.table() should check column names and replace syntactically invalid characters.
...	(various) Additional arguments passed to utils::read.table().

Details

Reads in one or multiple MCMC log files from the same analysis and discards a user-specified burn-in, compatible with multiple monitor types. If the trace contains vectors of vectors and the user does not specify format = "complex", readTrace() will read in those columns as factors rather than as numeric vectors.

Value

List of dataframes (of length 1 if only 1 log file provided).

Examples

```
# read and process a single trace file


# download the example dataset to working directory
url_gtr <-
   "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.log"
dest_path_gtr <- "primates_cytb_GTR.log"
download.file(url_gtr, dest_path_gtr)

# to run on your own data, change this to the path to your data file
file_single <- dest_path_gtr
```

```
one_trace <- readTrace(paths = file_single)

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_gtr)

# read and process multiple trace files, such as from multiple runs of
# the same analysis

# download the example dataset to working directory
url_1 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_1.log"
dest_path_1 <- "primates_cytb_GTR_run_1.log"
download.file(url_1, dest_path_1)

url_2 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_2.log"
dest_path_2 <- "primates_cytb_GTR_run_2.log"
download.file(url_2, dest_path_2)

# to run on your own data, change this to the path to your data file
file_1 <- dest_path_1
file_2 <- dest_path_2

# read in the multiple trace files
multi_trace <- readTrace(path = c(file_1, file_2), burnin = 0.0)

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_1, dest_path_2)
```

`readTrees` *Read trees*

Description

Reads in a tree file containing one or multiple trees

Usage

```
readTrees(paths, tree_name = "psi", burnin = 0, n_cores = 1L, verbose = TRUE)
```

Arguments

`paths`	(vector of character strings; no default) File path(s) to tree(s).

tree_name	(character string; default psi) Name of the tree variable.
burnin	(single numeric value; default = 0.1) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations (if value provided is greater than 1).
n_cores	(integer; default 1) Number of cores for parallelizing.
verbose	(logical; default true) Display a status bar?

Details

Reads in a tree file in either nexus or newick format, and containing a single tree or multiple trees (as in the results of a Bayesian analysis). For reading in annotated tree files of continuous character evolution, the parameter must be considered a node parameter rather than branch parameter. Set isNodeParameter = TRUE in the extended newick monitor (mnExtNewick)

Value

A list (across runs) of lists (across samples) of treedata objects.

Examples

```
# read in a single nexus file

# download the example dataset to working directory
url_nex <-
 "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_MAP.tre"
dest_path_nex <- "primates_cytb_GTR_MAP.tre"
download.file(url_nex, dest_path_nex)

# to run on your own data, change this to the path to your data file
file <- dest_path_nex
tree_single_old <- readTrees(paths = file)

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_nex)

# read in a single newick string

# download the example dataset to working directory
url_new <-
 "https://revbayes.github.io/tutorials/intro/data/primates.tre"
dest_path_new <- "primates.tre"
download.file(url_new, dest_path_new)

# to run on your own data, change this to the path to your data file
file_new <- dest_path_new
tree_new <- readTrees(paths = file_new)

# remove file
```

```
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_new)


# read in a tree trace (may take a few seconds)

# download the example dataset to working directory
url_multi <-
 "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.trees"
dest_path_multi <- "primates_cytb_GTR.trees"
download.file(url_multi, dest_path_multi)

# to run on your own data, change this to the path to your data file
file_multi <- dest_path_multi
tree_multi <- readTrees(paths = file_multi)

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_multi)
```

`removeBurnin` *Remove Burnin*

Description

Removes burnin from MCMC trace

Usage

```
removeBurnin(trace, burnin)
```

Arguments

`trace`	(list of data frames; no default) Name of a list of data frames, such as produced by readTrace().
`burnin`	(single numeric value; 0.1) Fraction of generations to discard (if value provided is between 0 and 1) or number of generations (if value provided is greater than 1).

Details

Removes burnin from an MCMC trace, such as the output of readTrace(). If multiple traces are provided, this function will remove the burnin from each.

Value

List of dataframes (of length 1 if only 1 log file provided).

Examples

```
# download the example dataset to working directory
url_gtr <-
    "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.log"
dest_path_gtr <- "primates_cytb_GTR.log"
download.file(url_gtr, dest_path_gtr)

# to run on your own data, change this to the path to your data file
file_single <- dest_path_gtr

one_trace <- readTrace(paths = file_single)
one_trace_burnin <- removeBurnin(trace = one_trace, burnin = 0.1)

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_gtr)
```

---

rerootPhylo *Reroot Phylo*

Description

Reroots a phylogeny given an outgroup taxon or clade

Usage

```
rerootPhylo(tree, outgroup)
```

Arguments

tree	(list of lists of treedata objects; no default) Name of a list of lists of treedata objects, such as produced by readTrees().
outgroup	(character, no default) Name of the outgroup(s). Either a single taxon name or a character vector of length two to specify a clade; in this case the root will be placed at the midpoint of the branch subtending the two taxa's MRCA. Modified from phytools::reroot().

Details

Modifies a tree object by rerooting using a specified outgroup taxon or clade. Places the root at the midpoint of the branch subtending the outgroup. If the input contains multiple trees, all trees will be rerooted.

Value

returns a list of list of treedata objects, with the trees rooted.

See Also

phytools: reroot.

Examples

```
file <- system.file("extdata",
                    "sub_models/primates_cytb_GTR_MAP.tre",
                    package="RevGadgets")
tree <- readTrees(paths = file)
# root with one taxon
tree_rooted <- rerootPhylo(tree = tree, outgroup = "Galeopterus_variegatus")
# root with clade, specified by two taxa
tree_rooted <- rerootPhylo(tree = tree,
                           outgroup = c("Varecia_variegata_variegata",
                                        "Propithecus_coquereli"))
```

RevGadgets *RevGadgets*

Description

This package provides functions to process and plot the output of RevBayes analyses.

---

setMRFGlobalScaleHyperpriorNShifts

Sets a global scale parameter for a GMRF or HSMRF model given a prior mean number of effective shifts.

Description

This function finds the global scale parameter value that produces the desired prior mean number of "effective" rate shifts. Given a specified magnitude for an effective shift, shift_size, an effective shift occurs when two adjacent values are more than shift_size-fold apart from each other. That is, an effective shift is the event that rate[i+1]/rate[i] > shift_size or rate[i+1]/rate[i] < 1/shift_size.

Usage

```
setMRFGlobalScaleHyperpriorNShifts(
  n_episodes,
  model,
  prior_n_shifts = log(2),
  shift_size = 2
)
```

Arguments

n_episodes	(numeric; no default) The number of episodes in the random field (the parameter vector will be this long).
model	(character; no default) What model should the global scale parameter be set for? Options are "GMRF" and "HSMRF" for first-order models (also allowable: "GMRF1" and "HSMRF1") and "GMRF2" and HSMRF2" for second-order models.
prior_n_shifts	(numeric; log(2)) The desired prior mean number of shifts.
shift_size	(numeric; 2) The magnitude of change that defines an effective shift (measured as a fold-change).

Details

Finding these values for a HSMRF model can take several seconds for large values of n_episodes because of the required numerical integration.

Value

The hyperprior.

References

Magee et al. (2019) Locally adaptive Bayesian birth-death model successfully detects slow and rapid rate shifts. doi: https://doi.org/10.1101/853960

Examples

```
# Get global scale for a HSMRF model with 100 episodes.
gs <- setMRFGlobalScaleHyperpriorNShifts(100, "HSMRF")

# Plot a draw from this HSMRF distribution

trajectory <- simulateMRF(n_episodes = 100,
                          model = "HSMRF",
                          global_scale_hyperprior = gs)

plot(1:100,
     rev(trajectory),
     type = "l",
     xlab = "time",
     ylab = "speciation rate")
```

`simulateMRF` *Simulates a single Markov random field trajectory.*

Description

This function simulates a draw from a HSMRF or GMRF distribution given a user-specified global scale parameter. The MRF can be taken to be on the log-scale (such as for a birth rate) or the real-scale. The first value must be specified

Usage

```
simulateMRF(
  n_episodes,
  model,
  global_scale_hyperprior,
  initial_value = NULL,
  exponentiate = TRUE
)
```

Arguments

`n_episodes`	(numeric; no default) The number of episodes in the random field (the parameter vector will be this long).
`model`	(character; no default) What model should the global scale parameter be set for? Options are "GMRF" and "HSMRF".
`global_scale_hyperprior`	(numeric; no default) The hyperprior on the global scale parameter.
`initial_value`	(numeric; NULL) The first value in the MRF. If no value is specified, the field is assumed to start at 0 (if exponentiate=FALSE) or 1 (if exponentiate=TRUE).
`exponentiate`	(logical; TRUE) If TRUE, the MRF model is taken to be on the log-scale and the values are returned on the real-scale (note this means that the specified initial value will be the log of the true initial value). If FALSE, the model is taken to be on the real scale.

Value

A vector drawn from the specified MRF model on the specified (log- or real-) scale.

References

Magee et al. (2020) Locally adaptive Bayesian birth-death model successfully detects slow and rapid rate shifts. *PLoS Computational Biology*, **16 (10)**: e1007999.

Faulkner, James R., and Vladimir N. Minin. Locally adaptive smoothing with Markov random fields and shrinkage priors. *Bayesian analysis*, **13 (1)**, 225.

Examples

```
# Simulate a 100-episode HSMRF model for a speciation-rate through time
trajectory <- simulateMRF(n_episodes = 100,
                          model = "HSMRF",
                          global_scale_hyperprior = 0.0021)
plot(1:100,
     rev(trajectory),
     type = "l",
     xlab = "time",
     ylab = "speciation rate")
```

`summarizeTrace` *Summarize trace*

Description

Summarizes trace file(s) that have been read into memory

Usage

```
summarizeTrace(trace, vars)
```

Arguments

`trace`	(list of data frames; no default) Name of a list of data frames, such as produced by readTrace(). If the readTrace() output contains multiple traces (such as from multiple runs), summarizeTrace() will provide summaries for each trace individually, as well as the combined trace.
`vars`	(character or character vector; no default) The name of the variable(s) to be summarized.

Details

Summarizes a trace file for continuous or discrete characters by computing the mean and 95% credible interval for quantitative character and the 95% credible set for discrete characters.

Value

summarizeTrace() returns a list of the length of provided variables. For quantitative variables, it returns the mean and 95 For discrete variables, it returns the 95 associated probabilities.

Examples

```
# continuous character only example, one run

# download the example dataset to working directory
url_gtr <-
    "https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR.log"
dest_path_gtr <- "primates_cytb_GTR.log"
download.file(url_gtr, dest_path_gtr)

# to run on your own data, change this to the path to your data file
file_single <- dest_path_gtr

one_trace <- readTrace(paths = file_single)
trace_sum <- summarizeTrace(trace = one_trace,
                            vars = c("pi[1]","pi[2]","pi[3]","pi[4]"))
trace_sum[["pi[1]"]]

# remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_gtr)

# continuous character example, multiple runs

#' # download the example dataset to working directory
url_1 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_1.log"
dest_path_1 <- "primates_cytb_GTR_run_1.log"
download.file(url_1, dest_path_1)

url_2 <-
"https://revbayes.github.io/tutorials/intro/data/primates_cytb_GTR_run_2.log"
dest_path_2 <- "primates_cytb_GTR_run_2.log"
download.file(url_2, dest_path_2)

# to run on your own data, change this to the path to your data file
file_1 <- dest_path_1
file_2 <- dest_path_2

# read in the multiple trace files
multi_trace <- readTrace(path = c(file_1, file_2), burnin = 0.0)

trace_sum_multi <- summarizeTrace(trace = multi_trace,
                                  vars = c("pi[1]","pi[2]","pi[3]","pi[4]"))
trace_sum_multi[["pi[1]"]]

# remove files
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_1, dest_path_2)
```

```
# discrete character example

# download the example dataset to working directory
url_rj <- "https://revbayes.github.io/tutorials/intro/data/freeK_RJ.log"
dest_path_rj <- "freeK_RJ.log"
download.file(url_rj, dest_path_rj)

file <- dest_path_rj
trace <- readTrace(path = file)

trace_sum_discrete <- summarizeTrace(trace = trace,
                                     vars = c("prob_rate_12",
                                              "prob_rate_13",
                                              "prob_rate_31",
                                              "prob_rate_32"))
trace_sum_discrete[["prob_rate_12"]]

#' # remove file
# WARNING: only run for example dataset!
# otherwise you might delete your data!
file.remove(dest_path_rj)
```

Index