Package 'EvoPhylo'

August 19, 2022

Version 0.3

Title Pre- And Postprocessing of Morphological Data from Relaxed Clock Bayesian Phylogenetics

Description Performs automated morphological character partitioning for phylogenetic analyses and analyze macroevolutionary parameter outputs from clock (time-calibrated) Bayesian inference analyses, following concepts introduced by Simões and Pierce (2021) <doi:10.1038/s41559-021-01532-x>.

Depends R (>= 4.0.0)

Imports ape (>= 1.16.2), dplyr (>= 1.0.8), cluster (>= 2.1.2), deeptime (>= 0.2.0), ggplot2 (>= 3.3.5), ggrepel (>= 0.9.1), ggtree (>= 3.1.5.902), patchwork, treeio (>= 1.16.2), Rtsne (>= 0.15), unglue (>= 0.1.0)

Suggests kableExtra, knitr, rmarkdown

Encoding UTF-8

LazyData false

License GPL (>= 2)

URL `https://github.com/tiago-simoes/EvoPhylo`,
`https://tiago-simoes.github.io/EvoPhylo/`

BugReports `https://github.com/tiago-simoes/EvoPhylo/issues`

VignetteBuilder knitr

RoxygenNote 7.2.1

NeedsCompilation no

Author Tiago Simoes [cre, aut] (<`https://orcid.org/0000-0003-4716-649X`>), Noah Greifer [aut] (<`https://orcid.org/0000-0003-3067-7154`>), Joelle Barido-Sottani [aut] (<`https://orcid.org/0000-0002-5220-5468`>), Stephanie Pierce [aut] (<`https://orcid.org/0000-0003-0717-1841`>)

Maintainer Tiago Simoes <`trsimoes87@gmail.com`>

Repository CRAN

Date/Publication 2022-08-19 13:30:02 UTC

R topics documented:

`characters` *A morphological phylogenetic data matrix*

Description

An example dataset of morphological characters for early tetrapodomorphs from Simões & Pierce (2021). This type of data would be used as input to `get_gower_dist`.

Usage

```
data("characters")
```

Format

A data frame with 178 observations (characters) on 43 columns (taxa).

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

`clockrate_dens_plot` *Plot clock rate distributions*

Description

Plots the distribution density of clock rates by clock and clade. The input must have a "clade" column.

Usage

```
clockrate_dens_plot(rate_table, clock = NULL,
                    stack = FALSE, nrow = 1,
                    scales = "fixed")
```

Arguments

`rate_table` A data frame of clock rates, such as from the output of `get_clockrate_table_MrBayes` with an extra "clade" column.

`clock` Which clock rates will be plotted. If unspecified, all clocks are plotted.

`stack` Whether to display stacked density plots (`TRUE`) or overlapping density plots (`FALSE`).

`nrow` When plotting rates for more than one clock, how many rows should be filled by the plots. This is passed to `facet_wrap`.

`scales` When plotting rates for more than one clock, whether the axis scales should be "fixed" (default) across clocks or allowed to vary ("free", "free_x", or "free_y"). This is passed to `facet_wrap`.

Details

The user must manually add clades to the rate table produced by `get_clockrate_table_MrBayes` before it can be used with this function. This can be doen manually with in R, such as by using a graphical user interface for editing data like the **DataEditR** package, or by writing the rate table to a spreadsheet and reading it back in after adding the clades. The example below uses a table that has had the clades added.

Value

A ggplot object, which can be modified using **ggplot2** functions.

See Also

vignette("rates-selection") for the use of this function as part of an analysis pipeline.

get_clockrate_table_MrBayes, geom_density

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

data("rate_table_clades_means3")

# Overlapping plots
clockrate_dens_plot(rate_table_clades_means3, stack = FALSE,
                    nrow = 1, scales = "fixed")

# Stacked density for all three clocks, changing the color
# palette to viridis using ggplot2 functions
clockrate_dens_plot(rate_table_clades_means3,
                    clock = 1:3, nrow = 1, stack = TRUE,
                    scales = "fixed") +
  ggplot2::scale_color_viridis_d() +
  ggplot2::scale_fill_viridis_d()
```

clockrate_reg_plot *Plot regression lines between sets of rates*

Description

Displays a scatterplot and fits regression line of one set of clock rates against another, optionally displaying their Pearson correlation coefficient (r) and R-squared values (R^2).

Usage

```
clockrate_reg_plot(rate_table, clock_x, clock_y,
                   method = "lm", show_lm = TRUE,
                   ...)
```

Arguments

rate_table A table of clock rates, such as from the output of get_clockrate_table_MrBayes.

clock_x, clock_y
The clock rates that should go on the x- and y-axes, respectively.

`method` The method (function) used fit the regression of one clock on the other. Check the `method` argument in the to `geom_smooth` function of **ggplot2** for all options. Default is `"lm"` for a linear regression model. `"glm"` and `"loess"` are alternative options.

`show_lm` Whether to display the Pearson correlation coefficient (r) and R-squared values (R^2) between two sets of clock rates.

`...` Other arguments passed to `geom_smooth`.

Details

`clockrate_reg_plot()` can only be used when multiple clocks are present in the clock rate table. Unlike `clockrate_summary` and `clockrate_dens_plot`, no "clade" column is required.

Value

A `ggplot` object, which can be modified using **ggplot2** functions.

See Also

`vignette("rates-selection")` for the use of this function as part of an analysis pipeline.

`geom_point`, `geom_smooth`

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

data("rate_table_clades_means3")

#Plot correlations between clocks 1 and 3
clockrate_reg_plot(rate_table_clades_means3,
                   clock_x = 1, clock_y = 3)

#Use arguments supplied to geom_smooth():
clockrate_reg_plot(rate_table_clades_means3,
                   clock_x = 1, clock_y = 3,
                   color = "red", se = FALSE)
```

`clockrate_summary` *Compute rate summary statistics across clades and clocks*

Description

Computes summary statistics for each clade and/or each clock partition. The input must have a "clade" column.

Usage

```
clockrate_summary(rate_table, file = NULL, digits = 3)
```

Arguments

`rate_table`	A data frame of clock rates, such as from the output of `get_clockrate_table_MrBayes` with an extra `"clade"` column.
`file`	An optional file path where the resulting table will be stored using `write.csv`.
`digits`	The number of digits to round the summary results to. Default is 3. See `round`.

Details

The user must manually add clades to the rate table produced by `get_clockrate_table_MrBayes` before it can be used with this function. This can be doen manually within R, such as by using a graphical user interface for editing data like the **DataEditR** package, or by writing the rate table to a spreadsheet and reading it back in after adding the clades. The example below uses a table that has had the clades added.

Value

A data frame containing a row for each clade and each clock with summary statistics (n, mean, standard deviation, minimum, 1st quartile, median, third quartile, maximum).

See Also

`vignette("rates-selection")` for the use of this function as part of an analysis pipeline.

`get_clockrate_table_MrBayes`, `summary`

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

data("rate_table_clades_means3")

clockrate_summary(rate_table_clades_means3)
```

`clock_reshape` *Convert clock rate tables from wide to long format*

Description

Converts clock rate tables, such as those produced by `clockrate_summary` and imported back after including clade names, from wide to long format.

Usage

```
clock_reshape(rate_table)
```

Arguments

rate_table	A data frame of clock rates, such as from the output of get_clockrate_table_MrBayes with an extra "clade" column.

Details

This function will convert clock rate tables from wide to long format, with a new column "clock" containing the clock partition from where each rate estimate was obtained as a factor. The long format is necessary for downstream analyses of selection strength (mode), as similarly done by FBD_reshape for posterior parameter log files.

Value

A data frame containing a single "value" column (for all rate values) and one column for the "clock" variable (indicating to which clock partition each rate values refers to)

See Also

vignette("rates-selection") for the use of this function as part of an analysis pipeline.

get_clockrate_table_MrBayes, summary, clockrate_summary, FBD_reshape

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

## The example dataset rate_table_clades_means3 already
## has clades and 3 clock rate columns:
data("rate_table_clades_means3")

head(rate_table_clades_means3)

## Reshape a clock rate table with clade names to long format
rates_by_clade <- clock_reshape(rate_table_clades_means3)
```

---

cluster_to_nexus *Export character partitions to a Nexus file*

Description

Creates and exports a Nexus file with a list of characters and their respective partitions as inferred by the make_clusters function. The contents can be copied and pasted directly into a Mr. Bayes commands block for a partitioned clock Bayesian inference analysis.

Usage

```
cluster_to_nexus(cluster_df, file = NULL)
```

Arguments

cluster_df	A cluster_df object; the output of a call to make_clusters.
file	The path of the text file to be created containing the partitioning information in Nexus format. If NULL (the default), no file will be written and the output will be returned as a string. If "", the text will be printed to the console. Passed directly to the file argument of cat.

Value

The text as a string, returned invisibly if file is not NULL. Use cat on the resulting output to format it correctly (i.e., to turn "\n" into line breaks).

See Also

vignette("char-part") for the use of this function as part of an analysis pipeline.

make_clusters

Examples

```
# See vignette("char-part") for how to use this
# function as part of an analysis pipeline

# Load example phylogenetic data matrix
data("characters")

# Create distance matrix
Dmatrix <- get_gower_dist(characters)

# Find optimal partitioning scheme using PAM under k=3
# partitions
cluster_df <- make_clusters(Dmatrix, k = 3)

# Write to Nexus file and export to .txt file:
file <- tempfile(fileext = ".txt")

# You would set, e.g.,
# file <- "path/to/file.txt"

cluster_to_nexus(cluster_df, file = file)
```

---

combine_log *Combine and filter (.p) log files from Mr.Bayes*

Description

Imports parameter (.p) log files from Mr. Bayes and combines them into a single data frame. Samples can be dropped from the start of each log file (i.e., discarded as burn-in) and/or downsampled to reduce the size of the output object.

Usage

```
combine_log(path = ".", burnin = 0.25, downsample = 10000)
```

Arguments

`path`	The path to a folder containing (.p) log files or a character vector of log files to be read.
`burnin`	Either the number or a proportion of generations to drop from the beginning of each log file.
`downsample`	Either the number or the proportion of generations the user wants to keep after downsampling for the final (combined) log file. Generations will be dropped in approximately equally-spaced intervals.

Details

`combine_log()` imports log files produced by Mr.Bayes, ignoring the first row of the file (which contains an ID number). The files are appended together, optionally after removing burn-in generations from the beginning and/or by further filtering throughout the rest of each file. When `burnin` is greater than 0, the number or propotion of generations corresponding to the supplied value will be dropped from the beginning of each file as it is read in. For example, setting `burnin = .25` (the default) will drop the first 25% of generations from each file. When `downsample` is greater than 0, the file will be downsampled until the number or proportion of generations corresponding to the supplied value is reached. For example, if `downsample = 10000` generations (the default) for log files from 4 independent runs (i.e., 4 (.p) files), each log file will be downsampled to 2500 generations, and the final combined data frame will contain 10000 samples, selected in approximately equally spaced intervals from the original data.

The output can be supplied to `get_pwt_rates` and to `FBD_reshape`. The latter will convert the log data frame from my wide to long format, which is necessary to be used as input for downstream analyses using `FBD_summary`, `FBD_dens_plot`, `FBD_normality_plot`, `FBD_tests1`, or `FBD_tests2`.

Value

A data frame with columns corresponding to the columns in the supplied log files and rows containing the sampled parameter values. Examples of the kind of output produced can be accessed using `data("posterior1p")` and `data("posterior3p")`.

See Also

`vignette("fbd-params")` for the use of this function as part of an analysis pipeline.

`FBD_reshape`, which reshapes a combined parameter log file for use in some other package functions.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline
## Not run:
posterior <- combine_log("path/to/folder", burnin = .25,
                                  downsample = 10000)

## End(Not run)
```

drop.dummy.beast *Remove dummy tip from beast summary trees, accounting for metadata on the tips*

Description

This method is designed to remove the dummy tip added on offset trees once postprocessing is complete (for instance once the summary tree has been built using TreeAnnotator).

Usage

```
drop.dummy.beast(
  tree.file,
  output.file = NULL,
  dummy.name = "dummy",
  convert.heights = TRUE
)
```

Arguments

`tree.file` path to file containing the tree with dummy tip

`output.file` path to file to write converted tree. If `NULL` (default), the tree is simply returned.

`dummy.name` name of the added dummy tip, default `dummy`.

`convert.heights` whether height metadata should be converted to height - offset (required to plot e.g. HPD intervals correctly). Default TRUE.

Value

list of `tree` converted tree (as treedata) ; and `offset` age of the youngest tip in the final tree

See Also

`drop.dummy.mb()` for the same function using summary trees with a "dummy" extant from Mr. Bayes

Examples

```
# Analyze the trees with dummy tips - for instance, calculate the MCC summary tree
# Then remove the dummy tip from the MCC tree
final_tree = drop.dummy.beast(system.file("extdata", "ex_offset.MCC.tre", package = "EvoPhylo"))
```

`drop.dummy.mb` *Remove dummy tip from Mr. Bayes summary trees, accounting for metadata on the tips*

Description

This method is designed to remove the dummy tip added to a dataset before running with Mr. Bayes.

Usage

```
drop.dummy.mb(
  tree.file,
  output.file = NULL,
  dummy.name = "dummy",
  convert.ages = TRUE
)
```

Arguments

`tree.file`	path to file containing the tree with dummy tip
`output.file`	path to file to write converted tree. If `NULL` (default), the tree is simply returned.
`dummy.name`	name of the added dummy tip, default `dummy`.
`convert.ages`	whether height metadata should be converted to height - offset (required to plot e.g. HPD intervals correctly). Default TRUE.

Value

list of `tree` converted tree (as treedata) ; and `offset` age of the youngest tip in the final tree

See Also

`drop.dummy.beast()` for the same function using summary trees with a "dummy" extant from BEAST2

Examples

```
# Remove the dummy tip from the summary tree
final_tree = drop.dummy.mb(system.file("extdata", "tree_mb_dummy.tre", package = "EvoPhylo"))
```

FBD_dens_plot *Density plots for each FBD parameter*

Description

Produces a density or violin plot displaying the distribution of FBD parameter samples by time bin.

Usage

```
FBD_dens_plot(posterior, parameter, type = "density",
              stack = FALSE, color = "red")
```

Arguments

posterior	A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value. Such data frame can be imported using `combine_log` followed by `FBD_reshape`.
parameter	A string containing the name of an FBD parameter in the data frame; abbreviations allowed.
type	The type of plot; either `"density"` for a density plot or `"violin"` for violin plots. Abbreviations allowed.
stack	When `type = "density"`, whether to produce stacked densities (`TRUE`) or overlapping densities (`FALSE`, the default). Ignored otherwise.
color	When `type = "violin"`, the color of the plotted densities.

Details

Density plots are produced using `ggplot2::stat_density`, and violin plots are produced using `ggplot2::geom_violin`. On violin plots, a horizontal line indicates the median (of the density), and the black dot indicates the mean.

Value

A `ggplot` object, which can be modified using **ggplot2** functions.

Note

When setting `type = "violin"`, a warning may appear saying something like "In regularize.values(x, y, ties, missing(ties), na.rm = na.rm) : collapsing to unique 'x' values". This warning can be ignored.

See Also

`vignette("fbd-params")` for the use of this function as part of an analysis pipeline.

`ggplot2::stat_density`, `ggplot2::geom_violin` for the underlying functions to produce the plots.

combine_log for producing a single data frame of FBD parameter posterior samples from multiple log files.

FBD_reshape for converting a single data frame of FBD parameter estimates, such as those imported using combine_log, from wide to long format.

FBD_summary, FBD_normality_plot, FBD_tests1, and FBD_tests2 for other functions used to summarize and display the distributions of the parameters.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

posterior3p_long <- FBD_reshape(posterior3p)

FBD_dens_plot(posterior3p_long, parameter = "net_speciation",
              type = "density", stack = FALSE)

FBD_dens_plot(posterior3p_long, parameter = "net_speciation",
              type = "density", stack = TRUE)

FBD_dens_plot(posterior3p_long, parameter = "net_speciation",
              type = "violin", color = "red")
```

FBD_normality_plot *Inspect FBD parameter distributions visually*

Description

Produces plots of the distributions of fossilized birth–death process (FBD) parameters to facilitate the assessment of the assumptions of normality within time bins and homogeneity of variance across time bins.

Usage

```
FBD_normality_plot(posterior)
```

Arguments

posterior	A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value. Such data frame can be imported using combine_log followed by FBD_reshape.

Details

The plots produced include density plots for each parameter within each time bin (residualized to have a mean of zero), scaled so that the top of the density is at a value of one (in *black*). Superimposed onto these densitys are the densities of a normal distribution with the same mean and variance (and scaled by the same amount) (in *red*). Deviations between the normal density in *red* and the density of the parameters in *black* indiciate deviations from normality. The standard deviation of each parameter is also displayed for each time bin to facilitate assessing homogenity of variance.

Value

A `ggplot` object, which can be modified using **ggplot2** functions.

See Also

`vignette("fbd-params")` for the use of this function as part of an analysis pipeline.

`combine_log` for producing a single data set of parameter posterior samples from individual parameter log files.

`FBD_reshape` for converting posterior parameter table from wide to long format.

`FBD_tests1` for statistical tests of normality and homogeneity of variance.

`FBD_tests2` for tests of differences in parameter means.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

posterior3p_long <- FBD_reshape(posterior3p)

FBD_normality_plot(posterior3p_long)
```

`FBD_reshape` *Convert an FBD posterior parameter table from wide to long format*

Description

Converts FBD posterior parameter table, such as those imported using `combine_log`, from wide to long format.

Usage

```
FBD_reshape(posterior, variables = NULL, log.type = c("MrBayes", "BEAST2"))
```

Arguments

`posterior` Single posterior parameter sample dataset with skyline FBD parameters produced with `combine_log`.

`variables` Names of FBD rate variables in the log. If NULL (default), will attempt to auto-detect the names and log type.

`log.type` Name of the software which produced the log (currently supported: MrBayes or BEAST2). Has to be set if `variables` is not NULL.

Details

The posterior parameters log files produced by Bayesian evolutionary analyses using skyline birth-death tree models, including the skyline FBD model, result into two or more estimates for each FBD parameter, one for each time bin. This function will convert a table of parameters with skyline FBD parameters from wide to long format, with one row per generation per time bin and a new column "Time_bin" containing the respective time bins as a factor. The long format is necessary for downstream analyses using `FBD_summary`, `FBD_dens_plot`, `FBD_normality_plot`, `FBD_tests1`, or `FBD_tests2`, as similarly done by `clock_reshape` for clock rate tables.

The format of the log files can either be specified using the `variables` and `log.type` or auto-detected by the function. The "posterior" data frame can be obtained by reading in a log file directly (e.g. using the `read.table` function) or by combining several output log files from Mr. Bayes using `combine_log`.

Value

A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value.

See Also

`vignette("fbd-params")` for the use of this function as part of an analysis pipeline.

`combine_log`, `reshape`

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

head(posterior3p)

## Reshape FBD table to long format
posterior3p_long <- FBD_reshape(posterior3p)

head(posterior3p_long)
```

FBD_summary *Summarize FBD posterior parameter estimates*

Description

Produces numerical summaries of each fossilized birth–death process (FBD) posterior parameter by time bin.

Usage

```
FBD_summary(posterior, file = NULL, digits = 3)
```

Arguments

posterior	A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value. Such data frame can be imported using `combine_log` followed by `FBD_reshape`.
file	An optional file path where the resulting table will be stored using `write.csv`.
digits	The number of digitis to round the summary results to. Default is 3. See `round`.

Value

A data frame with a row for each paramater and time bin, and columns for different summary statistics. These include the number of data points (`n`) and the mean, standard deviation (`sd`), minimum value (`min`), first quartile (`Q1`), median, third quartile (`Q3`), and maximum value (`max`). When `file` is not `NULL`, a .csv file containing this data frame will be saved to the filepath specified in `file` and the output will be returned invisibly.

See Also

`vignette("fbd-params")` for the use of this function as part of an analysis pipeline.

`combine_log` for producing a single data set of parameter posterior samples from individual parameter log files.

`FBD_reshape` for converting posterior parameter table from wide to long format.

`FBD_dens_plot`, `FBD_normality_plot`, `FBD_tests1`, and `FBD_tests2` for other functions used to summarize and display the distributions of the parameters.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

posterior3p_long <- FBD_reshape(posterior3p)

FBD_summary(posterior3p_long)
```

FBD_tests1 — *Test assumptions of normality and homoscedasticity for FBD posterior parameters*

Description

Produces tests of normality (within time bin, ignoring time bin, and pooling within-time bin values) and homoscedasticity (homogeneity of variances) for each fossilized birth–death process (FBD) parameter in the posterior parameter log file.

Usage

```
FBD_tests1(posterior, downsample = TRUE)
```

Arguments

`posterior` A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value. Such data frame can be imported using `combine_log` followed by `FBD_reshape`.

`downsample` Whether to downsample the observations to ensure Shapiro-Wilk normality tests can be run. If `TRUE`, observations will be dropped so that no more than 5000 observations are used for the tests on the full dataset, as required by `shapiro.test`. They will be dropped in evenly spaced intervals. If `FALSE` and there are more than 5000 observations for any test, that test will not be run.

Details

`FBD_tests1()` performs several tests on the posterior distributions of parameter values within and across time bins. It produces the Shapiro-Wilk test for normality using `shapiro.test` and the Bartlett and Fligner tests for homogeneity of variance using `bartlett.test` and `fligner.test`, respectively. Note that these tests are likely to be significant even if the observations are approximately normally distributed or have approximately equal variance; therefore, they should be supplemented with visual inspection using `FBD_normality_plot`.

Value

A list containing the results of the three tests with the following elements:

`shapiro` A list with an element for each parameter. Each element is a data frame with a row for each time bin and the test statistic and p-value for the Shapiro-Wilk test for normality. In addition, there will be a row for an overall test, combining all observations ignoring time bin, and a test of the residuals, which combines the group-mean-centered observations (equivalent to the residuals in a regression of the parameter on time bin).

`bartlett` A data frame of the Bartlett test for homogeneity of variance across time bins with a row for each parameter and the test statistic and p-value for the test.

`fligner` A data frame of the Fligner test for homogeneity of variance across time bins with a row for each parameter and the test statistic and p-value for the test.

See Also

vignette("fbd-params") for the use of this function as part of an analysis pipeline.

combine_log for producing a single data set of parameter posterior samples from individual parameter log files.

FBD_reshape for converting posterior parameter table from wide to long format.

FBD_normality_plot for visual assessments.

FBD_tests2 for tests of differences between parameter means.

shapiro.test, bartlett.test, and fligner.test for the statistical tests used.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

posterior3p_long <- FBD_reshape(posterior3p)

FBD_tests1(posterior3p_long)
```

FBD_tests2 *Test for differences in FBD parameter values*

Description

FBD_tests2() performs t-tests and Mann-Whitney U-tests to compare the average value of fossilized birth–death process (FBD) parameters between time bins.

Usage

```
FBD_tests2(posterior, p.adjust.method = "fdr")
```

Arguments

posterior
A data frame of posterior parameter estimates containing a single "Time_bin" column and one column for each FBD parameter value. Such data frame can be imported using combine_log followed by FBD_reshape.

p.adjust.method
The method use to adjust the p-values for multiple testing. See p.adjust for details and options. Default if "fdr" for the Benjamini-Hochberg false discovery rate correction.

Details

pairwise.t.test and pairwise.wilcox.test are used to calculate, respectively, the t-test and Mann-Whitney U-tests statistics and p-values. Because the power of these tests depends on the number of posterior samples, it can be helpful to examine the distributions of FBD parameter posteriors using FBD_dens_plot instead of relying heavily on the tests.

Value

A list with an element for each test, each of which contains a list of test results for each parameter. The results are in the form of a data frame containing the sample sizes and unadjusted and adjusted p-values for each comparison.

See Also

vignette("fbd-params") for the use of this function as part of an analysis pipeline.

combine_log for producing a single data set of parameter posterior samples from individual parameter log files.

FBD_reshape for converting posterior parameter table from wide to long format.

FBD_dens_plot, FBD_normality_plot, FBD_tests1, and FBD_tests2 for other functions used to summarize and display the distributions of the parameter posteriors.

pairwise.t.test and pairwise.wilcox.test for the tests used.

Examples

```
# See vignette("fbd-params") for how to use this
# function as part of an analysis pipeline

data("posterior3p")

posterior3p_long <- FBD_reshape(posterior3p)

FBD_tests2(posterior3p_long)
```

---

get_clockrate_table_BEAST2

Extract evolutionary rates from Bayesian clock trees produced by BEAST2

Description

BEAST2 stores the rates for each clock in a separate file. All trees need to be loaded using treeio::read.beast.

Usage

```
get_clockrate_table_BEAST2(..., summary = "median", drop_dummy = NULL)
```

Arguments

...	treedata objects containing the summary trees with associated data on the rates for each separate clock.
summary	summary metric used for the rates. Currently supported: "mean" or "median", default "median".

drop_dummy   if not NULL, will drop the dummy extant tip with the given label from the BEAST2 summary trees prior to extracting the clock rates (when present). Default is NULL.

Value

A data frame with a column containing the node identifier (node) and one column containing the clock rates for each tree provided, in the same order as the trees.

See Also

get_clockrate_table_MrBayes() for the equivalent function for MrBayes output files.

clockrate_summary() for summarizing and examining properties of the resulting rate table. Note that clade membership for each node must be customized (manually added) before these functions can be used, since this is tree and dataset dependent.

Examples

```
#Import all clock summary trees produced by BEAST2 from your local directory
## Not run:
tree_clock1 = treeio::read.beast("tree_file_clock1.tre")
tree_clock2 = treeio::read.beast("tree_file_clock2.tre")

## End(Not run)

#Or use the example BEAST2 multiple clock trees that accompany EvoPhylo.
data(tree_clock1)
data(tree_clock2)

# obtain the rate table from BEAST2 trees
rate_table = get_clockrate_table_BEAST2(tree_clock1, tree_clock2, summary = "mean")
```

---

get_clockrate_table_MrBayes

Extract evolutionary rates from a Bayesian clock tree produced by Mr. Bayes

---

Description

Extract evolutionary rate summary statistics for each node from a Bayesian clock summary tree produced by Mr. Bayes and stores them in a data frame.

Usage

```
get_clockrate_table_MrBayes(tree, summary = "median",
                      drop_dummy = NULL)
```

Arguments

tree	An S4 class object of type treedata; a Bayesian clock tree imported using treeio::read.mrbayes for Mr. Bayes summary trees.
summary	The name of the rate summary. Should be one of "mean" or "median".
drop_dummy	if not NULL, will drop the dummy extant tip with the given label from the Mr. Bayes summary tree prior to extracting the clock rates (when present). Default is NULL.

Value

A data frame with a column containing the node identifier (node) and one column for each relaxed clock partition in the tree object containing clock rates.

See Also

vignette("rates-selection") for the use of this function as part of an analysis pipeline. get_clockrate_table_BEAST2 for the equivalent function for BEAST2 output files. clockrate_summary for summarizing and examining properties of the resulting rate table. Note that clade membership for each node must be customized (manually added) before these functions can be used, since this is tree and dataset dependent.

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

## Import summary tree with three clock partitions produced by
## Mr. Bayes (.t or .tre files) from your local directory
## Not run:
tree3p <- treeio::read.mrbayes("Tree3p.t")

## End(Not run)

#Or use the example Mr.Bayes multi-clock tree file (\code{tree3p})
data("tree3p")

# obtain the rate table from MrBayes tree
rate_table <- get_clockrate_table_MrBayes(tree3p)

head(rate_table)
```

get_gower_dist *Compute Gower distances between characters*

Description

Computes Gower distance between characters from a phylogenetic data matrix.

Usage

```
get_gower_dist(x, numeric = FALSE)
```

Arguments

x	A phylogenetic data matrix in Nexus (.nex) format, or in any other data frame or matrix format with a column for each character and terminal taxa as rows, which will be read using `ape::read.nexus.data`. The data cannot include polymorphisms.
numeric	Whether to treat the values contained in the `x` as numeric or categorical. If `FALSE` (default), features will be considered categorical; if `TRUE`, they will be considered numeric.

Value

The Gower distance matrix.

Author(s)

This function uses code adapted from `StatMatch::gower.dist()` written by Marcello D'Orazio.

See Also

`vignette("char-part")` for the use of this function as part of an analysis pipeline.

Examples

```
# See vignette("char-part") for how to use this
# function as part of an analysis pipeline

# Load example phylogenetic data matrix
data("characters")

# Create distance matrix
Dmatrix <- get_gower_dist(characters)

# Reading data matrix as numeric data
Dmatrix <- get_gower_dist(characters, numeric = TRUE)
```

get_pwt_rates *Conduct pairwise t-tests between node rates and clock base rate*

Description

Produces a data frame containing the results of 1-sample t-tests for the mean of posterior clock rates against each node's absolute clock rate.

Usage

```
get_pwt_rates(rate_table, posterior)
```

Arguments

`rate_table` A data frame containing a single "value" column (for all rate values) and one column for the "clock" variable (indicating to which clock partition each rate values refers to), such as from the output of `get_clockrate_table_MrBayes` with an extra `clade` column added, and followed by `clock_reshape`.

`posterior` A data frame of posterior parameter estimates including a "clockrate" column indicating the base of the clock rate estimate for each generation that will be used for pairwise t-tests. Such data frame can be imported using `combine_log` (no need to reshape from wide to long). See the `posterior1p` or `posterior3p` datasets for an examples of how the input file should look.

Details

`get_pwt_rates()` first transforms relative clock rates to absolute rate values for each node and each clock, by multiplying these by the mean posterior clock rate base value. Then, for each node and clock, a one-sample t-test is performed with the null hypothesis that the mean of the posterior clockrates is equal to that node and clock's absolute clock rate.

Value

A long data frame with one row per node per clock and the following columns:

`clade` The name of the clade, taken from the "clade" column of `rate_table`

`nodes` The node number, taken from the "node" column of `rate_table`

`clock` The clock partition number

`relative rate` The relative mean clock rate per node, taken from the "rates" columns of `rate_table`

`absolute rate (mean)`
The absolute mean clock rate per node; the relative clock rate multiplied by the mean of the posterior clock rates

`null` The absolute clock rate used as the null value in the t-test

`p.value` The p-value of the test comparing the mean ofthe posterior clockrates to each absolute clockrate

See Also

`vignette("rates-selection")` for the use of this function as part of an analysis pipeline.

`combine_log`

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

# Load example rate table and posterior data sets
data("rate_table_clades_means3")
data("posterior3p")

get_pwt_rates(rate_table_clades_means3, posterior3p)
```

get_sil_widths *Calculate silhouette widths index for various numbers of partitions*

Description

Computes silhouette widths index for several possible numbers of clusters(partitions) k, which determines how well an object falls within their cluster compared to other clusters. The best number of clusters k is the one with the highest silhouette width.

Usage

```
get_sil_widths(dist_mat, max.k = 10)

## S3 method for class 'sil_width_df'
plot(x, ...)
```

Arguments

dist_mat	A Gower distance matrix, the output of a call to get_gower_dist.
max.k	The maximum number of clusters(partitions) to search across.
x	A sil_width_df object; the output of a call to get_sil_widths().
...	Further arguments passed to ggplot2::geom_line to control the appearance of the plot.

Details

get_sil_widths calls cluster::pam on the supplied Gower distance matrix with each number of clusters (partitions) up to max.k and stores the average silhouette widths across the clustered characters. When plot = TRUE, a plot of the sillhouette widths against the number of clusters is produced, though this can also be produced seperately on the resulting data frame using plot.sil_width_df(). The number of clusters with the greatest silhouette width should be selected for use in the final clustering specification.

Value

For `get_sil_widths()`, it produces a data frame, inheriting from class `"sil_width_df"`, with two columns: `k` is the number of clusters, and `sil_width` is the silhouette widths for each number of clusters. If `plot = TRUE`, the output is returned invisibly.

For `plot()` on a `get_sil_widths()` object, it produces a `ggplot` object that can be manipulated using **ggplot2** syntax (e.g., to change the `theme` or labels).

See Also

`vignette("char-part")` for the use of this function as part of an analysis pipeline.

`get_gower_dist`, `cluster::pam`

Examples

```
# See vignette("char-part") for how to use this
# function as part of an analysis pipeline

data("characters")

#Reading example file as categorical data
Dmatrix <- get_gower_dist(characters)

#Get silhouette widths for k=7
sw <- get_sil_widths(Dmatrix, max.k = 7)

sw

plot(sw, color = "red", size =2)
```

`make_clusters` *Estimate and plot character partitions*

Description

Determines cluster (partition) membership for phylogenetic morphological characters from the supplied Gower distance matrix and requested number of clusters using partitioning around medoids (PAM, or K-medoids). For further and independently testing the quality of the chosen partitioning scheme, users may also poduce graphic clustering (tSNEs), coloring data points according to PAM clusters, to verify PAM clustering results.

Usage

```
make_clusters(dist_mat, k, tsne = FALSE,
              tsne_dim = 2, tsne_theta = 0,
              ...)

## S3 method for class 'cluster_df'
plot(x, seed = NA, nrow = 1,
              ...)
```

Arguments

`dist_mat` A Gower distance matrix, the output of a call to `get_gower_dist`.

`k` The desired number of clusters (or character partitions), the output from `get_sil_widths`.

`tsne` Whether to perform Barnes-Hut t-distributed stochastic neighbor embedding (tSNE) to produce a multi-dimensional representation of the distance matrix using `Rtsne::Rtsne`. The number of dimensions is controlled by the `tsne_dim` argument. See Details. Default is `FALSE`.

`tsne_dim` When `tsne = TRUE`, the number of dimensions for the tSNE multidimensional scaling plots. This is passed to the `dims` argument of `Rtsne::Rtsne`. Default is 2.

`tsne_theta` When `tsne = TRUE`, a parameter controlling the speed/accuracy trade-off (increase for faster but less accurate results). This is passed to the `theta` argument of `Rtsne::Rtsne`. Default is 0 for exact tSNE.

`...` For `make_clusters()`, other arguments passed to `Rtsne::Rtsne` when `tsne = TRUE`.

For `plot()`, when plotting a `cluster_df` object, other arguments passed to `ggrepel::geom_text_repel` to control display of the observation labels.

`x` For `plot()`, a `cluster_df` object; the output of a call to `make_clusters()`.

`seed` For `plot()`, the seed used to control the placement of the labels and the jittering of the points. Jittering only occurs when `tsne = FALSE` in the call to `make_clusters()`. Using a non-`NA` seed ensure replicability across uses.

`nrow` For `plot()`, when `tsne = TRUE` in the call to `make_clusters()` and `tsne_dim` is greater than 2, the number of rows used to display the resulting 2-dimensional plots. Default is 1 for side-by-side plots.

Details

`make_clusters` calls `cluster::pam` on the supplied Gower distance matrix with the specified number of clusters to determine cluster membership for each character. PAM is analogous to K-means, but it has its clusters centered around medoids instead of centered around centroids, which are less prone to the impact from outliers and heterogeneous cluster sizes. PAM also has the advantage over k-means of utilizing Gower distance matrices instead of Euclidean distance matrices only.

When `tsne = TRUE`, a Barnes-Hut t-distributed stochastic neighbor embedding is used to compute a multi-dimensional embedding of the distance matrix, coloring data points according to the PAM-defined clusters, as estimated by the function `make_clusters`. This graphic clustering allows users to independently test the quality of the chosen partitioning scheme from PAM, and can help in visualizing the resulting clusters. `Rtsne::Rtsne` is used to do this. The resulting dimensions will be included in the output; see Value below.

`plot()` plots all morphological characters in a scatterplot with points colored based on cluster membership. When `tsne = TRUE` in the call to `make_clusters()`, the x- and y-axes will correspond to requested tSNE dimensions. With more than 2 dimensions, several plots will be produced, one for each pair of tSNE dimensions. These are displayed together using `patchwork::plot_layout`. When `tsne = FALSE`, the points will be arrange horizontally by cluster membership and randomly placed vertically.

Value

A data frame, inheriting from class "cluster_df", with a row for each character with its number (character_number) and cluster membership (cluster). When tsne = TRUE, additional columns will be included, one for each requested tSNE dimension, labeled tSNE_Dim1, tSNE_Dim2, etc., containing the values on the dimensions computed using Rtsne().

The pam fit resulting from cluster::pam is returned in the "pam.fit" attribute of the outut object.

Note

When using plot() on a cluster_df object, warnings may appear from ggrepel saying something along the lines of "unlabeled data points (too many overlaps). Consider increasing max.overlaps". See ggrepel::geom_text_repel for details; the max.overlaps argument can be supplied to plot() to increase the maximum number of element overlap in the plot. Alternatively, users can increase the size of the plot when exporting it, as it will increase the plot area and reduce the number of elements overlap. This warning can generally be ignored, though.

See Also

vignette("char-part") for the use of this function as part of an analysis pipeline.

get_gower_dist, get_sil_widths, cluster_to_nexus

cluster::pam, Rtsne::Rtsne

Examples

```
# See vignette("char-part") for how to use this
# function as part of an analysis pipeline

data("characters")

# Reading example file as categorical data
Dmatrix <- get_gower_dist(characters)

sil_widths <- get_sil_widths(Dmatrix, max.k = 7)

sil_widths
# 3 clusters yields the highest silhouette width

# Create clusters with PAM under k=3 partitions
cluster_df <- make_clusters(Dmatrix, k = 3)

# Simple plot of clusters
plot(cluster_df, seed = 12345)

# Create clusters with PAM under k=3 partitions and perform
# tSNE (3 dimensions; default is 2)
cluster_df_tsne <- make_clusters(Dmatrix, k = 3, tsne = TRUE,
                                 tsne_dim = 2)

# Plot clusters, plots divided into 2 rows, and increasing
# overlap of text labels (default = 10)
```

```
plot(cluster_df_tsne, nrow = 2, max.overlaps = 20)
```

---

offset.to.dummy *Convert trees produced by a BEAST2 FBD analysis with offset to trees with correct ages.*

Description

This method adds a dummy tip at the present (t = 0) to fully extinct trees with offsets, in order to have correct ages (otherwise the most recent tip is assumed to be at 0). This is a workaround to get the proper ages of the trees into other tools such as TreeAnnotator.

Usage

```
offset.to.dummy(trees.file, log.file, output.file = NULL, dummy.name = "dummy")
```

Arguments

`trees.file`	path to BEAST2 output file containing posterior trees
`log.file`	path to BEAST2 trace log file containing offset values
`output.file`	path to file to write converted trees. If `NULL` (default), trees are simply returned.
`dummy.name`	name of the added dummy tip, default `dummy`.

Details

NB: Any metadata present on the tips will be discarded. If you want to keep metadata (such as clock rate values), use `offset.to.dummy.metadata` instead.

Value

list of converted trees (as treedata)

See Also

`offset.to.dummy.metadata()` (slower version, keeping metadata)

Examples

```
# Convert trees with offset to trees with dummy tip
trees_file = system.file("extdata", "ex_offset.trees", package = "EvoPhylo")
log_file = system.file("extdata", "ex_offset.log", package = "EvoPhylo")
converted_trees = offset.to.dummy.metadata(trees_file, log_file)

# Do something with the converted trees - for instance, calculate the MCC summary tree
# Then remove the dummy tip from the MCC tree
final_tree = drop.dummy.beast(system.file("extdata", "ex_offset.MCC.tre", package = "EvoPhylo"))
```

`offset.to.dummy.metadata`

Convert trees produced by a BEAST2 FBD analysis with offset to trees with correct ages, accounting for possible metadata on the tips.

Description

This method adds a dummy tip at the present (t = 0) to fully extinct trees with offsets, in order to have correct ages (otherwise the most recent tip is assumed to be at 0). This is a workaround to get the proper ages of the trees into other tools such as TreeAnnotator.

Usage

```
offset.to.dummy.metadata(
  trees.file,
  log.file,
  output.file = NULL,
  dummy.name = "dummy"
)
```

Arguments

`trees.file`	path to BEAST2 output file containing posterior trees
`log.file`	path to BEAST2 trace log file containing offset values
`output.file`	path to file to write converted trees. If `NULL` (default), trees are simply returned.
`dummy.name`	name of the added dummy tip, default `dummy`.

Value

list of converted trees (as treedata)

See Also

`offset.to.dummy()` (faster version discarding metadata)

Examples

```
# Convert trees with offset to trees with dummy tip
trees_file = system.file("extdata", "ex_offset.trees", package = "EvoPhylo")
log_file = system.file("extdata", "ex_offset.log", package = "EvoPhylo")
converted_trees = offset.to.dummy.metadata(trees_file, log_file)

# Do something with the converted trees - for instance, calculate the MCC summary tree
# Then remove the dummy tip from the MCC tree
final_tree = drop.dummy.beast(system.file("extdata", "ex_offset.MCC.tre", package = "EvoPhylo"))
```

`plot_treerates_sgn` *Plot Bayesian evolutionary tree with rate thresholds for selection mode*

Description

Plots the summary Bayesian evolutionary tree with branches, according to user-defined thresholds (in units of standard deviations) used to infer the strength and mode of selection.

Usage

```
plot_treerates_sgn(tree, posterior, clock = 1,
       summary = "mean", threshold = c("1 SD", "2 SD"),
       drop.dummyextant = TRUE,
       low = "blue", mid = "gray90", high = "red",
       branch_size = 2, tip_size = 2,
       xlim = NULL, nbreaks = 10, geo_size=list(2, 3),
       geo_skip = c("Quaternary", "Holocene", "Late Pleistocene"))
```

Arguments

`tree` A `tidytree` object; the output of a call to `treeio::read.mrbayes`.

`posterior` A data frame of posterior parameter estimates including a "clockrate" column indicating the base of the clock rate estimate for each generation that will be used for pairwise t-tests. Such data frame can be imported using `combine_log` (no need to reshape from wide to long). See the `posterior1p` or `posterior3p` datasets for an examples of how the input file should look.

`clock` The clock number to plot, (i.e., the number in "rate<clockmodel>Brlens{#}_mean"). Ignored if only one clock is available.

`summary` The rate summary to plot, (i.e., the value in "rate<clockmodel>Brlens1_{summary}"). Only "mean" and "median" are allowed. Default is "mean".

`threshold` A vector of threshold values. Default is to display thresholds of ±1 relative standard deviation (SD) of the relative posterior clock rates. Should be specified as a number of standard deviations (e.g., `"1 SD"`) or the confidence level for a confidence interal around the mean relative posterior clockrate (e.g., `"95%"`). Multiple values are allowed to produce a plot with multiple thresholds. Set to `NULL` to omit thresholds.

`drop.dummyextant`
`logical`; whether to drop the "Dummyextant" tip (if present) from the tree before plotting the tree. Default is `TRUE`.

`low, mid, high` Colors passed to `scale_color_steps2` to control the colors of the branches based on which thresholds are exceeded. When no thresholds are supplied, use `mid` to control the color of the tree.

`branch_size` The thickness of the lines that form the tree.

`tip_size` The font size for the tips of the tree.

`xlim` The x-axis limits. Should be two negative numbers (though the axis labels will be in absolute value, i.e., Ma).

`nbreaks` The number of interval breaks in the geological timescale.

`geo_size` The font size for the labels in the geological scale. The first value in `list()` is the font size for geological epochs and the second value is for geological periods. Passed directly to the `size` argument of `deeptime::coord_geo`.

`geo_skip` A vector of interval names indicating which intervals should not be labeled. Passed directly to the `skip` argument of `deeptime::coord_geo`.

Details

Plots the phylogentic tree contained in `tree` using `ggtree::ggtree`. Branches undergoing accelerating evolutionary rates (e.g., >`"1 SD"`, `"3 SD"`, or `"5 SD"` relative to the background rate) for each morphological partition suggest directional (or positive) selection for that morphological partition in that branch of the tree. Branches undergoing decelerating evolutionary rates (e.g., <`"1 SD"`, `"3 SD"`, or `"5 SD"` relative to the background rate) for each morphological partition suggest stabilizing selection for that morphological partition in that branch of the tree. For details methods and rationale, see Simões & Pierce (2021).

Value

A `ggtree` object, which inherits from `ggplot`.

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

See Also

`vignette("rates-selection")` for the use of this function as part of an analysis pipeline.

`ggtree::ggtree`, `deeptime::coord_geo`

Examples

```
# See vignette("rates-selection") for how to use this
# function as part of an analysis pipeline

# Load example tree and posterior
data("tree3p")
data("posterior3p")

plot_treerates_sgn(
  tree3p, posterior3p,
  # show rates for clock partition 1
  clock = 1,
  # set summary stats to get from summary tree nodes
  summary = "mean",
  # set size for tree elements
  branch_size = 1.5, tip_size = 3,
  # set limits, breaks, and size for geological scale
  xlim = c(-450, -260), nbreaks = 8, geo_size = list(3, 3),
  # set rate threshold for selection strength
  threshold = c("1 SD", "2 SD"))
```

posterior1p *Posterior parameter samples (single clock)*

Description

An example dataset of posterior parameter samples resulting from a clock-based Bayesian inference analysis using the skyline fossilized birth–death process (FBD) tree model with Mr. Bayes after combining all parameter (.p) files into a single data frame with the combine_log function. This particular example was produced by analyzing the data set with a single morphological partition from Simões & Pierce (2021).

Usage

```
data("posterior1p")
```

Format

A data frame with 4000 observations on several variables estimated for each generation during analysis:

Gen A numeric vector for the generation number

LnL A numeric vector for the natural log likelihood of the cold chain

LnPr A numeric vector for the natural log likelihood of the priors

TH A numeric vector for the total tree height (sum of all branch durations, as chronological units)

TL A numeric vector for total tree length (sum of all branch lengths, as accumulated substitutions/changes)

prop_ancfossil A numeric vector indicating the proportion of fossils recovered as ancestors

sigma A numeric vector for the standard deviation of the lognormal distribution governing how much rates vary across characters.

net_speciation_1, net_speciation_2, net_speciation_3, net_speciation_4 A numeric vector for net speciation estimates for each time bin

relative_extinction_1, relative_extinction_2, relative_extinction_3, relative_extinction_4 A numeric vector for relative extinction estimates for each time bin

relative_fossilization_1, relative_fossilization_2, relative_fossilization_3, relative_fossilization_4 A numeric vector for relative fossilization estimates for each time bin

tk02var A numeric vector for the variance on the base of the clock rate

clockrate A numeric vector for the base of the clock rate

Details

Datasets like this one can be produced from parameter log (.p) files using `combine_log`. The number of variables depends on parameter set up, but for clock analyses with Mr. Bayes, will typically include the ones above, possibly also including `alpha`, which contains the shape of the gamma distribution governing how much rates vary across characters. When using the traditional FBD model rather than the skyline FBD model used to produce this dataset, there will be only one column for each of `net_speciation`, `relative_extinction` and `relative_fossilization`. When using more than one morphological partition, different columns may be present; see `posterior3p` for an example with 3 partitions.

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

See Also

`posterior3p` for an example dataset of posterior parameter samples resulting from an analysis with 3 partitions rather than 1.

`posterior3p` *Posterior parameter samples (3 clock partions)*

Description

An example dataset of posterior parameter samples resulting from a clock-based Bayesian inference analysis using the skyline fossilized birth–death process (FBD) tree model with Mr. Bayes after combining all parameter (.p) files into a single data frame with the `combine_log` function. This particular example was produced by analyzing the data set with three morphological partitions from Simões & Pierce (2021).

Usage

```
data("posterior3p")
```

Format

A data frame with 4000 observations on several variables estimated for each generation during analysis. The number of variables depends on parameter set up, but for clock analyses with Mr. Bayes, will typically include the following:

`Gen` A numeric vector for the generation number

`LnL` A numeric vector for the natural log likelihood of the cold chain

`LnPr` A numeric vector for the natural log likelihood of the priors

`TH.all.` A numeric vector for the total tree height (sum of all branch durations, as chronological units)

`TL.all.` A numeric vector for total tree length (sum of all branch lengths, as accumulated substitutions/changes)

`prop_ancfossil.all.` A numeric vector indicating the proportion of fossils recovered as ancestors

`sigma.1.`, `sigma.2.`, `sigma.3.` A numeric vector for the standard deviation of the lognormal distribution governing how much rates vary across characters for each data partition

`m.1.`, `m.2.`, `m.3.` A numeric vector for the rate multiplier parameter for each data partition

`net_speciation_1.all.`, `net_speciation_2.all.`, `net_speciation_3.all.`, `net_speciation_4.all.`
A numeric vector for net speciation estimates for each time bin

`relative_extinction_1.all.`, `relative_extinction_2.all.`, `relative_extinction_3.all.`, `relative_extinctio`
A numeric vector for relative extinction estimates for each time bin

`relative_fossilization_1.all.`, `relative_fossilization_2.all.`, `relative_fossilization_3.all.`, `relative_`
A numeric vector for relative fossilization estimates for each time bin

`tk02var.1.`, `tk02var.2.`, `tk02var.3.` A numeric vector for the variance on the base of the clock rate for each clock partition

`clockrate.all.` A numeric vector for the base of the clock rate

Details

Datasets like this one can be produced from parameter log (.p) files using `combine_log`. The number of variables depends on parameter set up, but for clock analyses with Mr. Bayes, will typically include the ones above, possibly also including an `alpha` for each partition, which contains the shape of the gamma distribution governing how much rates vary across characters (when shape of the distribution is unlinked across partitions). When using the traditional FBD model rather than the skyline FBD model used to produce this dataset, there will be only one column for each of `net_speciation`, `relative_extinction` and `relative_fossilization`. When using a single morphological partition, different columns may be present; see `posterior1p` for an example with just one partition.

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

See Also

`posterior1p` for an example dataset of posterior parameter samples resulting from an analysis with 1 partition rather than 3.

post_trees *Multiple phylogenetic clock trees*

Description

Multiple clock Bayesian phylogenetic tree, imported as an S4 class object using `treeio::read.beast()`.

Usage

```
data("post_trees")
```

Format

A `tidytree` object.

Details

Example tree file for function `write.beast.treedata`.

See Also

`write.beast.treedata` for using this file in context.

rate_table_clades_means1 *Mean clock rates by node and clade (single clock)*

Description

A data set containing the mean clock rates for a tree with 1 clock partition, such as the output of `get_clockrate_table_MrBayes` but with an additional "clade" column added, which is required for use in `clockrate_summary` and `clockrate_dens_plot`.

Usage

```
data("rate_table_clades_means1")
```

Format

A data frame with 79 observations on the following 3 variables.

`clade` A character vector containing the clade names for each corresponding node

`nodes` A numeric vector for the node numbers in the summary tree

`rates` A numeric vector containing the mean posterior clock rate for each node

Details

`rate_table_clades_means1` was created by running `get_clockrate_table_MrBayes(tree1p)` and then adding a "clade" column. It can be produced by using the following procedure:

1) Import tree file:

```
data("tree1p")
```

2) Produce clock rate table with, for instance, mean rate values from each branch in the tree:

```
rate_table <- get_clockrate_table_MrBayes(tree1p, summary = "mean")

write.csv(rate_table, file = "rate_table.csv", row.names = FALSE)
```

3) Now, manually add clades using, e.g., Excel:

3.1) Manually edit rate_table.csv, adding a "clade" column. This introduces customized clade names to individual nodes in the tree.

3.2) Save the edited rate table with a different name to differentiate from the original output (e.g., rate_table_clades_means.csv).

4) Read the file back in:

```
rate_table_clades_means1 <- read.csv("rate_table_clades_means.csv")

head(rate_table_clades_means1)
```

See Also

`tree1p` for the tree from which the clock rates were extracted.

`get_clockrate_table_MrBayes` for extracting a clock rate table from a tree.

`clockrate_summary`, `clockrate_dens_plot`, and `clockrate_reg_plot` for examples of using a clockrate table.

`rate_table_clades_means3` *Mean clock rates by node and clade (3 clock partitions)*

Description

A data set containing the mean clock rates for a tree with 3 clock partitions, such as the output of `get_clockrate_table_MrBayes` but with an additional "clade" column added, which is required for use in `clockrate_summary` and `clockrate_dens_plot`.

Usage

```
data("rate_table_clades_means3")
```

Format

A data frame with 79 observations on the following 5 variables.

`clade` A character vector containing the clade names for each corresponding node

`nodes` A numeric vector for the node numbers in the summary tree

`rates1` A numeric vector containing the mean posterior clock rate for each node for the first partition

`rates2` A numeric vector containing the mean posterior clock rate for each node for the second partition

`rates3` A numeric vector containing the mean posterior clock rate for each node for the third partition

Details

`rate_table_clades_means3` was created by running `get_clockrate_table_MrBayes(tree3p)` and then adding a "clade" column. It can be produced by using the following procedure:

1) Import tree file:

```
data("tree3p")
```

2) Produce clock rate table with, for instance, mean rate values from each branch in the tree:

```
rate_table <- get_clockrate_table_MrBayes(tree3p, summary = "mean")

write.csv(rate_table, file = "rate_table.csv", row.names = FALSE)
```

3) Now, manually add clades using, e.g., Excel:

3.1) Manually edit rate_table.csv, adding a "clade" column. This introduces customized clade names to individual nodes in the tree.

3.2) Save the edited rate table with a different name to differentiate from the original output (e.g., rate_table_clades_means.csv).

4) Read the file back in:

```
rate_table_clades_means3 <- read.csv("rate_table_clades_means.csv")

head(rate_table_clades_means3)
```

See Also

`tree3p` for the tree from which the clock rates were extracted.

`get_clockrate_table_MrBayes` for extracting a clock rate table from a tree.

`clockrate_summary`, `clockrate_dens_plot`, and `clockrate_reg_plot` for examples of using a clockrate table.

tree1p *Phylogenetic tree with a single clock partition*

Description

A clock Bayesian phylogenetic tree, imported as an S4 class object using `treeio::read.mrbayes()`.

Usage

```
data("tree1p")
```

Format

A `tidytree` object.

Details

This example tree file was produced by analyzing the data set with a single morphological partition from Simões & Pierce (2021).

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

See Also

`tree3p` for another tree object with 3 clock partitions.

`get_clockrate_table_MrBayes` for extratcing the poserior clockrates from a tree object.

tree3p *Phylogenetic tree with 3 clock partitions*

Description

A clock Bayesian phylogenetic tree, imported as an S4 class object using `treeio::read.mrbayes()`.

Usage

```
data("tree3p")
```

Format

A `tidytree` object.

Details

This example tree file was produced by analyzing the data set with 3 morphological clock partitions from Simões & Pierce (2021).

References

Simões, T. R. and S. E. Pierce (2021). Sustained High Rates of Morphological Evolution During the Rise of Tetrapods. *Nature Ecology & Evolution* 5: 1403–1414.

See Also

`tree1p` for another tree object with a single clock partition.

`get_clockrate_table_MrBayes` for extratcing the poserior clockrates from a tree object.

`tree_clock1` *BEAST2 phylogenetic tree with clock rates from partition 1*

Description

A clock Bayesian phylogenetic tree with clock rates from a single clock partition (partition 1 here), imported as an S4 class object using `treeio::read.beast()`.

Usage

```
data("tree_clock1")
```

Format

A `tidytree` object.

Details

This example tree file was produced by analyzing the data set with a single morphological partition from

See Also

`tree_clock2` for another BEAST2 tree object with clock rates from partition 2 for this same dataset.

`tree3p` for another tree object with 3 clock partitions from Mr.Bayes.

`tree1p` for another tree object with a single clock from Mr.Bayes.

`get_clockrate_table_BEAST2` for extratcing the poserior clock rates from BEAST2 tree objects.

tree_clock2 *BEAST2 phylogenetic tree with clock rates from partition 2*

Description

A clock Bayesian phylogenetic tree with clock rates from a single clock partition (partition 2 here), imported as an S4 class object using `treeio::read.beast()`.

Usage

```
data("tree_clock2")
```

Format

A `tidytree` object.

Details

This example tree file was produced by analyzing the data set with a single morphological partition from

See Also

`tree_clock1` for another BEAST2 tree object with clock rates from partition 1 for this same dataset.

`tree3p` for another tree object with 3 clock partitions from Mr.Bayes.

`tree1p` for another tree object with a single clock from Mr.Bayes.

`get_clockrate_table_BEAST2` for extratcing the poserior clock rates from BEAST2 tree objects.

write.beast.treedata *Export multiple treedata objects (S4 class tree files) to BEAST NEXUS file*

Description

This function was adopted and modified from treeio::write.beast to export a list of trees instead of a single tree.

Usage

```
write.beast.treedata(treedata, file = "",
                     translate = TRUE, tree.name = "STATE")
```

Arguments

treedata	An S4 class object of type `treedata` containing multiple trees; e.g. a Bayesian clock tree distribution imported using `treeio::read.beast` or `treeio::read.mrbayes`.
file	Output file. If `file = ""`, prints the output content on screen.
translate	Whether to translate taxa labels.
tree.name	Name of the trees, default `"STATE"`.

Value

Writes object type `treedata` containing multiple trees to a file or file content on screen

Examples

```
#Load file with multiple trees
## Not run:
trees_file = system.file("extdata", "ex_offset.trees", package = "EvoPhylo")
posterior_trees_offset = treeio::read.beast(trees_file)

#Write multiple trees to screen
write.beast.treedata(posterior_trees_offset)

## End(Not run)
```

write_partitioned_alignments

Write character partitions as separate Nexus files (for use in BEAUti)

Description

Write character partitions as separate Nexus files (for use in BEAUti)

Usage

```
write_partitioned_alignments(characters, cluster_df, file)
```

Arguments

characters	character data matrix as matrix or data frame (with taxa as columns and characters as rows)
cluster_df	cluster partitions as outputted by `make.clusters`
file	path to save the alignments. If `file = "example.nex"`, alignments will be saved to files `"example_part1.nex"`, `"example_part2.nex"`, etc.

Value

no return value

Examples

```
# Load example phylogenetic data matrix
data("characters")

# Create distance matrix
Dmatrix <- get_gower_dist(characters)

# Find optimal partitioning scheme using PAM under k=3 partitions
cluster_df <- make_clusters(Dmatrix, k = 3)

# Write to Nexus files
## Not run: write_partitioned_alignments(characters, cluster_df, "example.nex")
```

Index