Package: phylocomr (via r-universe)

September 6, 2024

Type Package

Title Interface to 'Phylocom'

Description Interface to 'Phylocom' (<https://phylodiversity.net/phylocom/>), a library for analysis of 'phylogenetic' community structure and character evolution. Includes low level methods for interacting with the three executables, as well as higher level interfaces for methods like 'aot', 'ecovolve', 'bladj', 'phylomatic', and more.

Version 0.3.4

URL https://docs.ropensci.org/phylocomr/, https://github.com/ropensci/phylocomr

BugReports https://github.com/ropensci/phylocomr/issues

License BSD_2_clause + file LICENSE

Encoding UTF-8

Language en-US

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Imports tibble, sys (>= 3.2)

Suggests testthat, knitr, rmarkdown, ape

RoxygenNote 7.2.3

X-schema.org-applicationCategory Biodiversity

X-schema.org-keywords phylogeny, Phylocom, phylodiversity, community structure, character evolution, species

X-schema.org-isPartOf https://ropensci.org

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

RemoteUrl https://github.com/ropensci/phylocomr

RemoteRef HEAD

RemoteSha a1937fabaa4d1dfbff7efc11b67ca097f7e9ee75

Contents

`phylocomr-package` *Phylocom interface*

Description

`phylocomr` gives you access to Phylocom, specifically the Phylocom C library (https://github.com/phylocom/phylocom/), licensed under BSD 2-clause (http://www.opensource.org/licenses/bsd-license.php)

Details

This package isn't doing system calls to a separately installed Phylocom instance - but actually includes Phylocom itself in the package.

Phylocom is usually used either on the command line or through the R package **picante**, which has duplicated some of the Phylocom functionality.

In terms of performance, some functionality will be faster here than in `picante`, but the maintainers of `picante` have re-written some Phylocom functionality in C/C++, so performance should be similar in those cases.

A note about files

As a convenience you can pass ages, sample and trait data.frame's, and phylogenies as strings, to `phylocomr` functions. However, `phylocomr` has to write these data.frame's/strings to disk (your computer's file system) to be able to run the Phylocom code on them. Internally, `phylocomr` is writing to a temporary file to run Phylocom code, and then the file is removed.

In addition, you can pass in files instead of data.frame's/strings. These are not themselves used. Instead, we read and write those files to temporary files. We do this for two reasons. First, Phylocom expects the files its using to be in the same directory, so if we control the file paths that becomes easier. Second, Phylocom is case sensitive, so we simply standardize all taxon names by lower casing all of them. We do this case manipulation on the temporary files so that your original data files are not modified.

Package API

- `ecovolve()`/`ph_ecovolve()` - interface to `ecovolve` executable, and a higher level interface
- `phylomatic()`/`ph_phylomatic()` - interface to `phylomatic` executable, and a higher level interface
- `phylocom()` - interface to `phylocom` executable
- `ph_aot()` - higher level interface to `aot`
- `ph_bladj()` - higher level interface to `bladj`
- `ph_comdist()`/`ph_comdistnt()` - higher level interface to comdist
- `ph_comstruct()` - higher level interface to comstruct
- `ph_comtrait()` - higher level interface to comtrait
- `ph_pd()` - higher level interface to Faith's phylogenetic diversity

Author(s)

Scott Chamberlain

Jeroen Ooms

`executables` *Executables*

Description

Executables

Usage

```
ecovolve(args = "--help", intern = FALSE)

phylocom(args = "help", intern = FALSE)

phylomatic(args = "--help", intern = FALSE)
```

Arguments

`args`	a character vector of arguments to command.
`intern`	capture output as character vector. Default: `FALSE`

Examples

```
ecovolve()
phylocom()
phylomatic()
```

phylocomr-inputs *Expected inputs*

Description

Expected inputs

Ages data

A file or data.frame, with 2 columns:

- (character) node taxonomic name
- (numeric) age estimate for the node

If a file path is used, the table must not have headers

Applies to:

- `ph_bladj()`

Sample data

A file or data.frame, with 3 columns, sorted by column 1, one row per taxon:

- (character) sample plot/quadrat/trap/etc. name (no spaces, must begin with a letter, not a number or symbol)
- (integer) abundance (leave as 1 for presence/absence data)
- (character) species code (same as in the phylogeny, must begin with a letter, not a number or symbol)

If a file path is used, the table must not have headers, and must be tab-delimited

Applies to:

- `ph_comtrait()`
- `ph_comstruct()`
- `ph_comdist()`
- `ph_pd()`
- `ph_rao()`

Traits data

A tab-delimited file with the first line as `type<TAB>n<TAB>n<TAB>... [up to the number of traits]`, for example: `type<TAB>3<TAB>3<TAB>3<TAB>0`

where n indicates the type of trait in each of the four columns. Types:

- `0`: binary (only one binary trait may be included, and it must be in the first column) 1 for unordered multistate (no algorithms currently implemented)

- 2: ordered multistate (currently treated as continuous)
- 3: continuous

Optional: The second line can start with the word name (lower case only) and then list the names of the traits in order. These will appear in the full output file

Subsequent lines should have the taxon name, which must be identical to its appearance in phylo, and the data columns separated by tabs. For example: `sp1<TAB>1<TAB>1<TAB>1<TAB>0`

- OR -

A data.frame, where the first column called `name` has each taxon, followed by any number of columns with traits. The first column name must be `name`, and the following columns should be named using the name of the trait.

Applies to:

- `ph_comtrait()`
- `ph_aot()`

Phylogenies

Phylocom expects trees in Newick format. The basic Newick format used by Phylocom is: `((A,B),C);`. See the Phylocom manual (https://phylodiversity.net/phylocom/) for more details on what they expect.

Applies to: all functions except `ph_phylomatic()` and `ph_ecovolve()`

`ph_aot` *aot*

Description

AOT conducts univariate and bivariate tests of phylogenetic signal and trait correlations, respectively, and node-level analyses of trait means and diversification.

Usage

```
ph_aot(
  traits,
  phylo,
  randomizations = 999,
  trait_contrasts = 1,
  ebl_unstconst = FALSE
)
```

Arguments

`traits` (data.frame/character) trait data.frame or path to traits file. required. See Details.

`phylo` (character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.

`randomizations` (numeric) number of randomizations. Default: 999

`trait_contrasts` (numeric) Specify which trait should be used as ’x’ variable for contrasts. Default: 1

`ebl_unstconst` (logical) Use equal branch lengths and unstandardized contrasts. Default: `FALSE`

Details

See phylocomr-inputs for expected input formats

Value

a list of data.frames

Taxon name case

In the `traits` table, if you’re passing in a file, the names in the first column must be all lowercase; if not, we’ll lowercase them for you. If you pass in a data.frame, we’ll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

Examples

```
## Not run:
traits_file <- system.file("examples/traits_aot", package = "phylocomr")
phylo_file <- system.file("examples/phylo_aot", package = "phylocomr")

# from data.frame
traitsdf_file <- system.file("examples/traits_aot_df",
  package = "phylocomr")
traits <- read.table(text = readLines(traitsdf_file), header = TRUE,
  stringsAsFactors = FALSE)
phylo_str <- readLines(phylo_file)
(res <- ph_aot(traits, phylo = phylo_str))

# from files
traits_str <- paste0(readLines(traits_file), collapse = "\n")
traits_file2 <- tempfile()
cat(traits_str, file = traits_file2, sep = '\n')
phylo_file2 <- tempfile()
cat(phylo_str, file = phylo_file2, sep = '\n')
(res <- ph_aot(traits_file2, phylo_file2))

## End(Not run)
```

ph_bladj *bladj*

Description

Bladj take a phylogeny and fixes the root node at a specified age, and fixes other nodes you might have age estimates for. It then sets all other branch lengths by placing the nodes evenly between dated nodes, and between dated nodes and terminals (beginning with the longest 'chains').

Usage

```
ph_bladj(ages, phylo)
```

Arguments

`ages` (data.frame/character) ages data.frame, or path to an ages file. required. column names do not matter, and are discarded anyway. the first column must be the node names, and the second column the node ages. See Details.

`phylo` (character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.

Details

See phylocomr-inputs for expected input formats

Value

newick string with attributes for where ages and phylo files used are stored

Common Errors

A few issues to be aware of:

- the ages table must have a row for the root node with an age estimate. bladj will not work without that. We attempt to check this but can only check it if you pass in a phylo object; there's no easy way to parse a newick string without requiring ape
- bladj is case sensitive. internally we lowercase all tip and node labels and taxon names in your ages file to avoid any case sensitivity problems

Examples

```
## Not run:
ages_file <- system.file("examples/ages", package = "phylocomr")
phylo_file <- system.file("examples/phylo_bladj", package = "phylocomr")

# from data.frame
ages_df <- data.frame(
  a = c('malpighiales','eudicots','ericales_to_asterales','plantaginaceae',
        'malvids', 'poales'),
  b = c(81, 20, 56, 76, 47, 71)
)
phylo_str <- readLines(phylo_file)
(res <- ph_bladj(ages = ages_df, phylo = phylo_str))
if (requireNamespace("ape")) {
  library(ape)
  plot(read.tree(text = res))
}

# from files
ages_file2 <- file.path(tempdir(), "ages")
write.table(ages_df, file = ages_file2, row.names = FALSE,
  col.names = FALSE, quote = FALSE)
phylo_file2 <- tempfile()
cat(phylo_str, file = phylo_file2, sep = '\n')
(res <- ph_bladj(ages_file2, phylo_file2))
if (requireNamespace("ape")) {
  library(ape)
  plot(read.tree(text = res))
}

# using a ape phylo phylogeny object
x <- read.tree(text = phylo_str)
if (requireNamespace("ape")) {
  library(ape)
  plot(x)
}

(res <- ph_bladj(ages_file2, x))
if (requireNamespace("ape")) {
  library(ape)
  tree <- read.tree(text = res)
  plot(tree)
}

## End(Not run)
```

ph_comdist *comdist*

Description

Outputs the phylogenetic distance between samples, based on phylogenetic distances of taxa in one sample to the taxa in the other

Usage

```
ph_comdist(
    sample,
    phylo,
    rand_test = FALSE,
    null_model = 0,
    randomizations = 999,
    abundance = TRUE
  )

  ph_comdistnt(
    sample,
    phylo,
    rand_test = FALSE,
    null_model = 0,
    randomizations = 999,
    abundance = TRUE
  )
```

Arguments

`sample`	(data.frame/character) sample data.frame or path to a sample file
`phylo`	(character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.
`rand_test`	(logical) do you want to use null models? Default: `FALSE`
`null_model`	(integer) which null model to use. See Details.
`randomizations`	(numeric) number of randomizations. Default: 999
`abundance`	(logical) If `TRUE` (default) computed accounting for abundance. Otherwise, uses presence-absence.

Details

See phylocomr-inputs for expected input formats

Value

data.frame or a list of data.frame's if use null models

Null models

- 0 - Phylogeny shuffle: This null model shuffles species labels across the entire phylogeny. This randomizes phylogenetic relationships among species.
- 1 - Species in each sample become random draws from sample pool: This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species actually occurring in at least one sample. Thus, species in the phylogeny that are not actually observed to occur in a sample will not be included in the null communities

- 2 - Species in each sample become random draws from phylogeny pool: This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species in the phylogeny pool. All species in the phylogeny will have equal probability of being included in the null communities. By changing the phylogeny, different species pools can be simulated. For example, the phylogeny could include the species present in some larger region.
- 3 - Independent swap: The independent swap algorithm (Gotelli and Entsminger, 2003); also known as 'SIM9' (Gotelli, 2000) creates swapped versions of the sample/species matrix.

Taxon name case

In the `sample` table, if you're passing in a file, the names in the third column must be all lowercase; if not, we'll lowercase them for you. If you pass in a data.frame, we'll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

Examples

```
sfile <- system.file("examples/sample_comstruct", package = "phylocomr")
pfile <- system.file("examples/phylo_comstruct", package = "phylocomr")

# from data.frame
sampledf <- read.table(sfile, header = FALSE,
  stringsAsFactors = FALSE)
phylo_str <- readLines(pfile)
ph_comdist(sample = sampledf, phylo = phylo_str)
ph_comdistnt(sample = sampledf, phylo = phylo_str)
ph_comdist(sample = sampledf, phylo = phylo_str, rand_test = TRUE)
ph_comdistnt(sample = sampledf, phylo = phylo_str, rand_test = TRUE)

# from files
sample_str <- paste0(readLines(sfile), collapse = "\n")
sfile2 <- tempfile()
cat(sample_str, file = sfile2, sep = '\n')
pfile2 <- tempfile()
cat(phylo_str, file = pfile2, sep = '\n')
ph_comdist(sample = sfile2, phylo = pfile2)
ph_comdistnt(sample = sfile2, phylo = pfile2)
ph_comdist(sample = sfile2, phylo = pfile2, rand_test = TRUE)
ph_comdistnt(sample = sfile2, phylo = pfile2, rand_test = TRUE)
```

`ph_comstruct` *comstruct*

Description

Calculates mean phylogenetic distance (MPD) and mean nearest phylogenetic taxon distance (MNTD; aka MNND) for each sample, and compares them to MPD/MNTD values for randomly generated samples (null communities) or phylogenies.

Usage

```
ph_comstruct(
  sample,
  phylo,
  null_model = 0,
  randomizations = 999,
  swaps = 1000,
  abundance = TRUE
)
```

Arguments

`sample`	(data.frame/character) sample data.frame or path to a sample file
`phylo`	(character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.
`null_model`	(integer) which null model to use. See Details.
`randomizations`	(numeric) number of randomizations. Default: 999
`swaps`	(numeric) number of swaps. Default: 1000
`abundance`	(logical) If `TRUE` (default) computed accounting for abundance. Otherwise, uses presence-absence.

Details

See phylocomr-inputs for expected input formats

Value

data.frame

Null models

- 0 - Phylogeny shuffle: This null model shuffles species labels across the entire phylogeny. This randomizes phylogenetic relationships among species.
- 1 - Species in each sample become random draws from sample pool: This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species actually occurring in at least one sample. Thus, species in the phylogeny that are not actually observed to occur in a sample will not be included in the null communities
- 2 - Species in each sample become random draws from phylogeny pool: This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species in the phylogeny pool. All species in the phylogeny will have equal probability of being included in the null communities. By changing the phylogeny, different species pools can be simulated. For example, the phylogeny could include the species present in some larger region.
- 3 - Independent swap: The independent swap algorithm (Gotelli and Entsminger, 2003); also known as ‘SIM9’ (Gotelli, 2000) creates swapped versions of the sample/species matrix.

Taxon name case

In the sample table, if you're passing in a file, the names in the third column must be all lowercase; if not, we'll lowercase them for you. If you pass in a data.frame, we'll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

Examples

```
sfile <- system.file("examples/sample_comstruct", package = "phylocomr")
pfile <- system.file("examples/phylo_comstruct", package = "phylocomr")

# from data.frame
sampledf <- read.table(sfile, header = FALSE,
  stringsAsFactors = FALSE)
phylo_str <- readLines(pfile)
(res <- ph_comstruct(sample = sampledf, phylo = phylo_str))

# from files
sample_str <- paste0(readLines(sfile), collapse = "\n")
sfile2 <- tempfile()
cat(sample_str, file = sfile2, sep = '\n')
pfile2 <- tempfile()
cat(phylo_str, file = pfile2, sep = '\n')
(res <- ph_comstruct(sample = sfile2, phylo = pfile2))

# different null models
ph_comstruct(sample = sfile2, phylo = pfile2, null_model = 0)
ph_comstruct(sample = sfile2, phylo = pfile2, null_model = 1)
ph_comstruct(sample = sfile2, phylo = pfile2, null_model = 2)
# ph_comstruct(sample = sfile2, phylo = pfile2, null_model = 3)
```

ph_comtrait *comtrait*

Description

Calculate measures of trait dispersion within each community, and compare observed patterns to those expected under a null model.

Usage

```
ph_comtrait(
  sample,
  traits,
  binary = NULL,
  metric = "variance",
  null_model = 0,
  randomizations = 999,
  abundance = TRUE
)
```

Arguments

`sample` (data.frame/character) sample data.frame or path to a sample file

`traits` (data.frame/character) traits data.frame or path to a traits file

`binary` (logical) a logical vector indicating what columns are to be treated as binary characters - all others are treated as continuous

`metric` (integer) metric to calculate. One of variance, mpd, mntd, or range (converted to phylocom integer format internally)

`null_model` (integer) which null model to use. See Details.

`randomizations` (numeric) number of randomizations. Default: 999

`abundance` (logical) If `TRUE` (default) computed accounting for abundance. Otherwise, uses presence-absence.

Details

See phylocomr-inputs for expected input formats

If you give a data.frame to `traits` parameter it expects data.frame like

- species - the taxon labels matching the sample data to `sample` parameter
- col1,col2,col3,etc. - any number of trait columns - column names do not matter

When giving a data.frame to `traits` make sure to pass in a binary vector for what traits are to be treated as binary.

Value

data.frame of the form:

- trait - Trait name
- sample - Sample name
- ntaxa - Number of taxa in sample
- mean - Mean value of trait in sample
- metric - Observed metric in sample
- meanrndmetric - Mean value of metric in null models
- sdrndmetric - Standard deviation of metric in null models
- sesmetric - Standardized effect size of metric
- ranklow - Number of randomizations with metric lower than observed
- rankhigh - Number of randomizations with metric higher than observed
- runs - Number of randomizations

Null models

- 0 - This null model shuffles trait values across species.
- 1 - Species in each sample become random draws from sample pool. This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species actually occurring in at least one sample
- 2 - Species in each sample become random draws from traits data. This null model maintains the species richness of each sample, but the identities of the species occurring in each sample are randomized. For each sample, species are drawn without replacement from the list of all species with trait values. This function is redundant since by definition the sample and trait species must match, but is included for consistency with the comstruct function.
- 3 - Independent swap: Same as for ph_comdist and ph_comstruct

Taxon name case

In the `sample` and `trait` tables, if you're passing in a file, the names in the third and first columns, respectively, must be all lowercase; if not, we'll lowercase them for you. If you pass in a data.frame, we'll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

Examples

```
## Not run:
sfile <- system.file("examples/sample_comstruct", package = "phylocomr")
tfile <- system.file("examples/traits_aot", package = "phylocomr")

# from files
sample_str <- paste0(readLines(sfile), collapse = "\n")
sfile2 <- tempfile()
cat(sample_str, file = sfile2, sep = '\n')

traits_str <- paste0(readLines(tfile), collapse = "\n")
tfile2 <- tempfile()
cat(traits_str, file = tfile2, sep = '\n')

ph_comtrait(sample = sfile2, traits = tfile2)

# from data.frame
sampledf <- read.table(sfile, header = FALSE,
  stringsAsFactors = FALSE)
traitsdf_file <- system.file("examples/traits_aot_df",
  package = "phylocomr")
traitsdf <- read.table(text = readLines(traitsdf_file), header = TRUE,
  stringsAsFactors = FALSE)
ph_comtrait(sample = sampledf, traits = traitsdf,
  binary = c(FALSE, FALSE, FALSE, TRUE))

## End(Not run)
```

ph_ecovolve *ecovolve*

Description

Ecovolve generates a phylogeny via a random birth and death process, generates a traits file with five randomly evolving, in-dependent traits, and a sample file with a single sample unit ('alive') containing all extant members of the phylogeny.

Usage

```
ph_ecovolve(
  speciation = 0.05,
  extinction = 0.01,
  time_units = 100,
  out_mode = 3,
  prob_env = "3211000000",
  extant_lineages = FALSE,
  only_extant = FALSE,
  taper_change = NULL,
  competition = FALSE
)
```

Arguments

`speciation`	(numeric) Probability of speciation per unit time. Default: 0.05
`extinction`	(numeric) Probability of extinction per unit time. Default: 0.01
`time_units`	(integer) Time units to simulate over. Default: 100
`out_mode`	(integer) Output mode (2 = LTT; 3 = newick). Default: 3
`prob_env`	(character) Probability envelope for character change. must be a string of 10 integers. Default: 3211000000
`extant_lineages`	(logical) Stop simulation after this number of extant lineages. Default: `FALSE`
`only_extant`	(logical) Output phylogeny pruned only for extant taxa. Default: `FALSE`
`taper_change`	(numeric/integer) Taper character change by `e^(-time/F)`. This produces more conservatism in traits (see Kraft et al., 2007). Default: `NULL`, not passed
`competition`	(logical) Simulate competition, with trait proximity increasing extinction. Default: `FALSE`

Value

a list with three elements:

- phylogeny - a phylogeny as a newick string. In the case of `out_mode = 2` gives a Lineage Through Time data.frame instead of a newick phylogeny

- sample - a data.frame with three columns, "sample" (all "alive"), "abundance" (all 1's), "name" (the species code). In the case of `out_mode` = 2 gives an empty data.frame
- traits - a data.frame with first column with spcies code ("name"), then 5 randomly evolved and independent traits. In the case of `out_mode` = 2 gives an empty data.frame

Clean up

Two files, "ecovolve.sample" and "ecovolve.traits" are written to the current working directory when this function runs - we read these files in, then delete the files via unlink

Failure behavior

Function occasionally fails with error "call to 'ecovolve' failed with status 8. only 1 taxon; > 1 required" - this just means that only 1 taxon was created in the random process, so the function can't proceed

Examples

```
## Not run:
# ph_ecovolve(speciation = 0.05)
# ph_ecovolve(speciation = 0.1)
# ph_ecovolve(extinction = 0.005)
# ph_ecovolve(time_units = 50)
# ph_ecovolve(out_mode = 2)
# ph_ecovolve(extant_lineages = TRUE)
# ph_ecovolve(extant_lineages = FALSE)
# ph_ecovolve(only_extant = FALSE)
# ph_ecovolve(only_extant = TRUE, speciation = 0.1)
# ph_ecovolve(taper_change = 2)
# ph_ecovolve(taper_change = 10)
# ph_ecovolve(taper_change = 500)

if (requireNamespace("ape")) {
  # library(ape)
  # x <- ph_ecovolve(speciation = 0.05)
  # plot(read.tree(text = x$phylogeny))
}


## End(Not run)
```

---

ph_pd *pd - Faith's index of phylogenetic diversity*

Description

Calculates Faith's (1992) index of phylogenetic diversity (PD) for each sample in the phylo.

Usage

```
ph_pd(sample, phylo)
```

Arguments

`sample` (data.frame/character) sample data.frame or path to a sample file. required

`phylo` (character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.

Details

See phylocomr-inputs for expected input formats

Value

A single data.frame, with the colums:

- sample - community name/label
- ntaxa - number of taxa
- pd - Faith's phylogenetic diversity
- treebl - tree BL
- proptreebl - proportion tree BL

Taxon name case

In the `sample` table, if you're passing in a file, the names in the third column must be all lowercase; if not, we'll lowercase them for you. If you pass in a data.frame, we'll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

See Also

Other phylogenetic-diversity: `ph_rao()`

Examples

```
sfile <- system.file("examples/sample_comstruct", package = "phylocomr")
pfile <- system.file("examples/phylo_comstruct", package = "phylocomr")

# from data.frame
sampledf <- read.table(sfile, header = FALSE,
  stringsAsFactors = FALSE)
phylo_str <- readLines(pfile)
ph_pd(sample = sampledf, phylo = phylo_str)

# from files
sample_str <- paste0(readLines(sfile), collapse = "\n")
sfile2 <- tempfile()
cat(sample_str, file = sfile2, sep = '\n')
```

```
pfile2 <- tempfile()
phylo_str <- readLines(pfile)
cat(phylo_str, file = pfile2, sep = '\n')

ph_pd(sample = sfile2, phylo = pfile2)
```

ph_phylomatic *phylomatic*

Description

Phylomatic is a tool for extracting a phylogeny from a master phylogeny using only a user-supplied list of taxa.

Usage

```
ph_phylomatic(taxa, phylo, tabular = FALSE, lowercase = FALSE, nodes = FALSE)
```

Arguments

taxa	(character) all taxa as a character vector (will be written to a temp file if provided) - OR a path to taxa file. Required. See Details.
phylo	(character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** phylo object. required.
tabular	(logical) Output a tabular representation of phylogeny. Default: FALSE
lowercase	(logical) Convert all chars in taxa file to lowercase. Default: FALSE
nodes	(logical) label all nodes with default names. Default: FALSE

Details

The taxa character vector must have each element of the form family/genus/genus_epithet. If a file is passed in, each line should have a family/genus/genus_epithet string - make sure only one per line, and a newline (i.e., press ENTER) at the end of each line

References

Phylomatic is also available as a web service (https://github.com/camwebb/phylomatic-ws) - but is based on a different code base (https://github.com/camwebb/phylomatic-ws) See Webb and Donoghue (2005) doi:10.1111/j.14718286.2004.00829.x for more information on the goals of Phylomatic.

Examples

```
## Not run:
taxa_file <- system.file("examples/taxa", package = "phylocomr")
phylo_file <- system.file("examples/phylo", package = "phylocomr")

# from strings
(taxa_str <- readLines(taxa_file))
(phylo_str <- readLines(phylo_file))
(tree <- ph_phylomatic(taxa = taxa_str, phylo = phylo_str))

# from files
taxa_file2 <- tempfile()
cat(taxa_str, file = taxa_file2, sep = '\n')
phylo_file2 <- tempfile()
cat(phylo_str, file = phylo_file2, sep = '\n')
(tree <- ph_phylomatic(taxa = taxa_file2, phylo = phylo_file2))

if (requireNamespace("ape")) {
  library(ape)
  plot(read.tree(text = tree))
}

## End(Not run)
```

`ph_rao` *rao - Rao's quadratic entropy*

Description

A measure of within- and among-community diversity taking species dissimilarity (phylogenetic dissimilarity) into account

Usage

```
ph_rao(sample, phylo)
```

Arguments

`sample`	(data.frame/character) sample data.frame or path to a sample file
`phylo`	(character/phylo) One of: phylogeny as a newick string (will be written to a temp file) - OR path to file with a newick string - OR a an **ape** `phylo` object. required.

Details

See phylocomr-inputs for expected input formats

Value

A list of 6 data.frame's: **Diversity components**:

- overall alpha (within-site)
- beta (among-site)
- total diversity
- Fst statistic of differentiation for diversity and phylogenetic diversity

Within-community diversity:

- Plot - Plot name
- NSpp - Number of species
- NIndiv - Number of individuals
- PropIndiv - Proportion of all individuals found in this plot
- D - Diversity (= Simpson's diversity)
- Dp - Phylogenetic diversity (= Diversity weighted by interspecific phylogenetic distances)

The remaining 4 tables compare each community pairwise:

- among_community_diversity_d - Among-community diversities
- among_community_diversity_h - Among-community diversities excluding within-community diversity
- among_community_phylogenetic_diversity_dp - Among-community phylogenetic diversities
- among_community_phylogenetic_diversity_hp - Among-community phylogenetic diversities excluding within-community diversity

Taxon name case

In the `sample` table, if you're passing in a file, the names in the third column must be all lowercase; if not, we'll lowercase them for you. If you pass in a data.frame, we'll lowercase them for your. All phylo tip/node labels are also lowercased to avoid any casing problems

See Also

Other phylogenetic-diversity: `ph_pd()`

Examples

```
sfile <- system.file("examples/sample_comstruct", package = "phylocomr")
pfile <- system.file("examples/phylo_comstruct", package = "phylocomr")

# sample from data.frame, phylogeny from a string
sampledf <- read.table(sfile, header = FALSE,
  stringsAsFactors = FALSE)
phylo_str <- readLines(pfile)

ph_rao(sample = sampledf, phylo = phylo_str)
```

```
# both from files
sample_str <- paste0(readLines(sfile), collapse = "\n")
sfile2 <- tempfile()
cat(sample_str, file = sfile2, sep = '\n')
pfile2 <- tempfile()
phylo_str <- readLines(pfile)
cat(phylo_str, file = pfile2, sep = '\n')

ph_rao(sample = sfile2, phylo = pfile2)
```

Index