Package ‘canaper’

October 12, 2022

Title  Categorical Analysis of Neo- And Paleo-Endemism

Version  1.0.0

Description  Provides functions to analyze the spatial distribution of biodiversity, in particular categorical analysis of neo- and paleo-endemism (CANAPE) as described in Mishler et al (2014) <doi:10.1038/ncomms5473>. 'canaper' conducts statistical tests to determine the types of endemism that occur in a study area while accounting for the evolutionary relationships of species.

License  MIT + file LICENSE

Encoding  UTF-8

LazyData  true

RoxygenNote  7.2.1

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

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

Language  en-US

Imports  ape, assertr, assertthat, dplyr, future.apply, phyloregion, progressr, purrr, stats, tibble, vegan

Suggests  rmarkdown, knitr, future, tictoc, patchwork, tidyverse, testthat (>= 3.0.0), stringr, magrittr, covr, picante, withr, fs, readr, usethis, scales, colorBlindness

Config/testthat/edition  3

Depends  R (>= 4.1.0)

VignetteBuilder  knitr

NeedsCompilation  no

Author  Joel H. Nitta [aut, cre] (<https://orcid.org/0000-0003-4719-7472>),
        Shawn W. Laffan [ctb, dtc],
        Brent D. Mishler [ctb, dtc],
        Wataru Iwasaki [ctb] (<https://orcid.org/0000-0002-9169-9245>),
        Klaus Schliep [rev] (<https://orcid.org/0000-0003-2941-0161>)
## Description

Dataset of Australian *Acacia* from Mishler et al. 2014 (Nat. Comm.)

## Usage

```r
acacia
```

## Format

A list with two elements:

- **phy** Phylogeny of Australian *Acacia* (list of class "phylo"). Tip labels are specific epithet, except for the outgroup, which includes genus and specific epithet. Includes 508 ingroup taxa (genus *Acacia*) and two outgroup taxa.
Community matrix of Australian *Acacia* (dataframe). Column names are specific epithet of each species. Row names are centroids of 50km grid cells in Australian Albers equal area EPSG:3577 projection. 3037 rows (sites) x 506 columns (species). Data are counts, i.e., the number of times a species was observed in a grid cell.

References


Examples

```r
# Example phylogeny
acacia$phy
# Example community
acacia$comm[1:5, 1:5]
```

biod_example  

**Biodiverse example data**

**Description**

Fictional data for testing purposes from Biodiverse.

**Usage**

biod_example

**Format**

A list with two elements:

- **phy** Phylogeny with 31 tips
- **comm** Community matrix with 127 sites and 31 species. Data are counts, i.e., the number of times a species was observed in a grid cell.

**Details**

Corresponds to the community matrix ("groups" object) and phylogeny from the Biodiverse example_project.bps. # nolint

**References**

Examples

# Example phylogeny
biod_example$phy
# Example community
biod_example$comm[1:5, 1:5]

biod_results  Output from Biodiverse

Description
Output of analyzing test data with Biodiverse.

Usage
biod_results

Format
A tibble (dataframe) with 127 rows and 7 columns. Columns include:

site  Site name; corresponds to row names of biod_example$comm
pd_biodiv  Phylogenetic diversity (PD; PD_P in Biodiverse)
pd_alt_biodiv  Alternative PD (PD measured on tree with all branchlengths equal; PHYLO_RPD_NULL2 in Biodiverse)
rpd_biodiv  Relative PD (PD divided by alternative PD; PHYLO_RPD_NULL2 in Biodiverse)
pe_biodiv  Phylogenetic endemism (PE; PE_WE_P in Biodiverse)
pe_alt_biodiv  Alternative PE (PE measured on tree with all branchlengths equal; PHYLO_RPE_NULL2 in Biodiverse)
rpe_biodiv  Relative PE (PE divided by alternative PD; PHYLO_RPE2 in Biodiverse)

Details
The example_project.bps # nolint example data set was used as input, which corresponds to the biod_example dataset in this package.
For a description of all Biodiverse metrics, see the Biodiverse wiki. # nolint

References

Examples
biod_results
cpr_classify_endem

Classify phylogenetic endemism

Description

Given the results of `cpr_rand_test()`, classifies phylogenetic endemism according to CANAPE scheme of Mishler et al. 2014.

Usage

cpr_classify_endem(df)

Arguments

- **df**: Input data frame. Must have the following columns:
  - `pe_obs_p_upper`: Upper p-value comparing observed phylogenetic endemism to random values
  - `pe_alt_obs_p_upper`: Upper p-value comparing observed phylogenetic endemism on alternate tree to random values
  - `rpe_obs_p_upper`: Upper p-value comparing observed relative phylogenetic endemism to random values

Details


Value

Object of class data.frame with column `endem_type` (character) added. Values of `endem_type` type include `paleo` (paleoendemic), `neo` (neoendemic), `not significant` (what it says), `mixed` (mixed endemism), and `super` (super-endemic; both `pe_obs` and `pe_obs_alt` are highly significant).

References


Examples

define seed(12345)
data(phylocom)
rand_test <- cpr_rand_test(
  phylocom$comm, phylocom$phy,
  null_model = "curveball", metrics = c("pe", "rpe"), n_reps = 10
)
cpr_classify_endem(rand_test)
cpr_classify_signif  

Classify statistical significance

Description

Given the results of cpr_rand_test(), classifies statistical significance of a biodiversity metric. The null hypothesis is that observed value does not lie in the extreme of the random values.

Usage

cpr_classify_signif(df, metric, one_sided = FALSE, upper = FALSE)

Arguments

df  
Input data frame.

metric  
Character vector of length 1; selected metric to classify significance. May choose from pd (phylogenetic diversity), rpd (relative phylogenetic diversity), pe (phylogenetic endemism), rpe (relative phylogenetic endemism) (case-sensitive).

one_sided  
Logical vector of length 1; is the null hypothesis one-sided? If TRUE, values will be classified as significant if they are in either the top 5% or bottom 5%. If FALSE, values will be classified as significant if they are in the top 2.5% or bottom 2.5%, combined.

upper  
Logical vector of length 1; only applies if one_sided is TRUE. If TRUE, values in the top 5% will be classified as significant. If FALSE, values in the bottom 5% will be classified as significant.

Details

For metrics like pe, you probably want to consider a one-sided hypothesis testing values in the upper extreme (i.e., we are interested in areas that have higher than expected endemism). For this, you would set one_sided = TRUE, upper = TRUE. For metrics like pd, you probably want to consider a two-sided hypothesis (i.e., we are interested in areas that are either more diverse or less than diverse than expected at random). For this, set one_sided = FALSE.

Value

Object of class data.frame with column added for statistical significance of the selected metric. The new column name is the name of the metric with _signif appended. The new column is a character that may contain the following values, depending on the null hypothesis:

- \(< 0.01, < 0.025, > 0.975, > 0.99, not significant\) (two-sided)
- \(< 0.01, < 0.05, > 0.99, > 0.95, not significant\) (one-sided)
Examples

```r
set.seed(12345)
data(phylocom)
rand_test <- cpr_rand_test(
  phylocom$comm, phylocom$phy,
  null_model = "curveball", metrics = "pd", n_reps = 50
)
cpr_classify_signif(rand_test, "pd")
```

---

cpr_endem_cols

**CVD-friendly color palette for plotting results of CANAPE**

### Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Should be distinguishable to people with color vision deficiency (CVD).

### Usage

```r
cpr_endem_cols
```

### Format

An object of class character of length 5.

### Details

Color scheme:

- paleo: dark blue
- neo: red
- not significant: beige
- mixed: green
- super: yellow

### See Also

Other colors: `cpr_endem_cols_2`, `cpr_endem_cols_3`, `cpr_endem_cols_4`, `cpr_make_pal()`, `cpr_signif_cols_2`, `cpr_signif_cols`, `mishler_endem_cols`, `mishler_signif_cols`

### Examples

```r
cpr_endem_cols
scales::show_col(cpr_endem_cols)
```
cpr_endem_cols_2

CVD-friendly color palette for plotting results of CANAPE, version 2

Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Should be distinguishable to people with color vision deficiency (CVD).

Usage

cpr_endem_cols_2

Format

An object of class character of length 5.

Details

Color scheme:

- paleo: dark blue
- neo: red
- not significant: light grey
- mixed: green
- super: yellow

See Also

Other colors: `cpr_endem_cols_3`, `cpr_endem_cols_4`, `cpr_endem_cols`, `cpr_make_pal()`, `cpr_signif_cols_2`, `cpr_signif_cols`, `mishler_endem_cols`, `mishler_signif_cols`

Examples

cpr_endem_cols_2
scales::show_col(cpr_endem_cols_2)
CVD-friendly color palette for plotting results of CANAPE, version 3

Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Should be distinguishable to people with color vision deficiency (CVD).

Usage

cpr_endem_cols_3

Format

An object of class character of length 5.

Details

Color scheme:

- paleo: dark blue
- neo: red
- not significant: light grey
- mixed: green
- super: orange

See Also

Other colors: cpr_endem_cols_2, cpr_endem_cols_4, cpr_endem_cols, cpr_make_pal(), cpr_signif_cols_2, cpr_signif_cols, mishler_endem_cols, mishler_signif_cols

Examples

cpr_endem_cols_3
scales::show_col(cpr_endem_cols_3)
Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Should be distinguishable to people with color vision deficiency (CVD).

Usage

cpr_endem_cols_4

Format

An object of class character of length 5.

Details

Color scheme:

- paleo: light blue
- neo: red
- not significant: light grey
- mixed: green
- super: yellow

See Also

Other colors: cpr_endem_cols_2, cpr_endem_cols_3, cpr_endem_cols, cpr_make_pal(), cpr_signif_cols_2, cpr_signif_cols, mishler_endem_cols, mishler_signif_cols

Examples

cpr_endem_cols_4
scales::show_col(cpr_endem_cols_4)
**cpr_iter_sim**

Simulate the number of iterations needed to generate a random community that is sufficiently different from the original community.

**Description**

For randomization algorithms that involve swapping (iterations), there is no way to know *a-priori* how many iterations are needed to sufficiently "mix" the community data matrix. `cpr_iter_sim()` records the percentage similarity between the original matrix and a matrix that has been randomized with successive swapping iterations, at each iteration.

**Usage**

```r

cpr_iter_sim(
  comm,
  null_model = "curveball",
  n_iterations = 100,
  thin = 1,
  seed = NULL
)
```

**Arguments**

- `comm`: Dataframe or matrix; input community data with sites (communities) as rows and species as columns. Values of each cell are the presence/absence (0 or 1) or number of individuals (abundance) of each species in each site.
- `null_model`: Character vector of length 1 or object of class `commsim`; either the name of the model to use for generating random communities (null model), or a custom null model. For full list of available predefined null models, see the help file of `vegan::commsim()`, or run `vegan::make.commsim()`. An object of class `commsim` can be generated with `vegan::commsim()`.
- `n_iterations`: Numeric vector of length 1; maximum number of iterations to conduct.
- `thin`: Numeric vector of length 1; frequency to record percentage similarity between original matrix and randomized matrix. Results will be recorded every `thin` iterations (see Details).
- `seed`: Integer vector of length 1 or NULL; random seed that will be used in a call to `set.seed()` before randomizing the matrix. Default (NULL) will not change the random generator state.

**Details**

The user should inspect the results to determine at what number of iterations the original matrix and randomized matrix reach maximum dissimilarity (see Examples). This number will strongly depend on the size and structure of the original matrix. Large matrices with many zeros will likely take more iterations, and even then still retain relatively high similarity between the original matrix and the randomized matrix.
Available memory may be quickly exhausted if many (e.g., tens or hundreds of thousands, or more) of iterations are used with no thinning on large matrices; use thin to only record a portion of the results and save on memory.

Of course, `cpr_iter_sim()` only makes sense for randomization algorithms that use iterations. Only presence/absence information is used to calculate percentage similarity between community matrices.

Value

Tibble (dataframe) with the following columns:

- iteration: Number of iterations used to generate random community
- similarity: Percentage similarity between original community and random community

Examples

```r
# Simulate generation of a random community with maximum of 10,000
# iterations, recording similarity every 100 iterations
(res <- cpr_iter_sim(
  comm = biod_example$comm,
  null_model = "swap",
  n_iterations = 10000,
  thin = 100,
  seed = 123
))

# Plot reveals that ca. 1000 iterations are sufficient to
# completely mix random community
plot(res$iteration, res$similarity, type = "l")
```

---

cpr_make_pal

*Make a palette of colors for plotting CANAPE results*

Description

Palettes can also be loaded by name directly (see "Other colors" below).

Usage

`cpr_make_pal(name, type)`

Arguments

- `name` Character vector of length 1; name of palette to select. Must be one of `misher2014`, `canaper1`, `canaper2`, `canaper3`, or `canaper4`
- `type` Character vector of length 1; type of palette to select. Must be one of `endem` (endemism) or `signif` (`p`-rank significance)
Details

Palettes with the name ‘mishler2014’ correspond to colors used in Mishler et al. 2014. This color scheme has been widely used to show CANAPE results, but some colors may not be distinguishable to people with color vision deficiency (CVD).

Palettes with the name ‘canaper1’ or ‘canaper2’ are based on the Okabe-Ito palette (endemism; Okabe & Ito 2002) or RColorBrewer (significance) and should be CVD-friendly.

Names of colors correspond to either endemism type or p-rank output by `cpr_classify_endem()` or `cpr_classify_signif()`, respectively. Not all names are all available for both types.

Value

Named character vector; color hex codes

References


See Also

Other colors: `cpr_endem_cols_2`, `cpr_endem_cols_3`, `cpr_endem_cols_4`, `cpr_endem_cols`, `cpr_signif_cols_2`, `cpr_signif_cols`, `mishler_endem_cols`, `mishler_signif_cols`

Examples

```r
  cpr_make_pal("mishler2014", "endem")
  cpr_make_pal("canaper1", "endem")
  scales::show_col(cpr_make_pal("mishler2014", "endem"))
```

Description

Note that binary null models return a binary matrix, even if an abundance matrix was used as input.

Usage

```r
  cpr_rand_comm(comm, null_model, n_iterations = 1, thin = 1, seed = NULL)
```
Arguments

comm  Dataframe or matrix; input community data with sites (communities) as rows and species as columns. Values of each cell are the presence/absence (0 or 1) or number of individuals (abundance) of each species in each site.

null_model  Character vector of length 1 or object of class commsim; either the name of the model to use for generating random communities (null model), or a custom null model. For full list of available predefined null models, see the help file of vegan::commsim(), or run vegan::make.commsim(). An object of class commsim can be generated with vegan::commsim() (see Examples).

n_iterations  Numeric vector of length 1; number of iterations for sequential null models. Ignored by non-sequential null models.

thin  Numeric vector of length 1; thinning parameter used by some null models in vegan (e.g., quasiswap); ignored for other models.

seed  Integer vector of length 1 or NULL; random seed that will be used in a call to set.seed() before randomizing the matrix. Default (NULL) will not change the random generator state.

Value

Matrix

Examples

set.seed(12345)

# Check list of available pre-defined null models in vegan
vegan::make.commsim()

# Binary null model produces binary output
data(phylocom)
cpr_rand_comm(phylocom$comm, "swap", 100)

# Quantitative null model produces quantitative output
cpr_rand_comm(phylocom$comm, "swap_count", 100)

# How to use a custom null model
# 1. Define a randomizing function, e.g. re-sample the matrix while
#    preserving total number of presences (same as the "r00" model)
randomizer <- function(x, n, ...) {
  array(replicate(n, sample(x)), c(dim(x), n))
}

# 2. Generate a commsim object
cs_object <- vegan::commsim(  
  "r00_model",  
  fun = randomizer, binary = TRUE,  
  isSeq = FALSE, mode = "integer"
)
# 3. Generate the null community

cpr_rand_test(cpr_rand_test(phylocom$comm, cs_object, 100))

---

**Description**

The observed value of the biodiversity metric(s) will be calculated for the input community data, then compared against a set of random communities. Various statistics are calculated from the comparison (see Value below).

**Usage**

```r
cpr_rand_test(
  comm,
  phy,
  null_model,
  n_reps = 100,
  n_iterations = 10000,
  thin = 1,
  metrics = c("pd", "rpd", "pe", "rpe"),
  site_col = "site",
  tbl_out = tibble::is_tibble(comm),
  quiet = FALSE
)
```

**Arguments**

- **comm**: Dataframe, tibble, or matrix; input community data with sites (communities) as rows and species as columns. Either presence-absence data (values only 0s or 1s) or abundance data (values >= 0) accepted, but calculations do not use abundance-weighting, so results from abundance data will be the same as if converted to presence-absence before analysis.

- **phy**: List of class phylo; input phylogeny.

- **null_model**: Character vector of length 1 or object of class commsim; either the name of the model to use for generating random communities (null model), or a custom null model. For full list of available predefined null models, see the help file of vegan::commsim(), or run vegan::make.commsim(). An object of class commsim can be generated with vegan::commsim() (see Examples in cpr_rand_comm()).

- **n_reps**: Numeric vector of length 1; number of random communities to replicate.

- **n_iterations**: Numeric vector of length 1; number of iterations to use for sequential null models; ignored for non-sequential models.

- **thin**: Numeric vector of length 1; thinning parameter used by some null models in vegan (e.g., quasiswap); ignored for other models.
metrics Character vector; names of biodiversity metrics to calculate. May include one or more of: pd, rpd, pe, rpe (case-sensitive).

site_col Character vector of length 1; name of column in comm that contains the site names; only used if comm is a tibble (object of class tbl_df).

tbl_out Logical vector of length 1; should the output be returned as a tibble? If FALSE, will return a dataframe. Defaults to TRUE if comm is a tibble.

quiet Logical vector of length 1; if TRUE, suppress all warnings and messages that would be emitted by this function.

Details

The biodiversity metrics (metrics) available for analysis include:

- pd: Phylogenetic diversity (Faith 1992)
- rpd: Relative phylogenetic diversity (Mishler et al 2014)
- pe: Phylogenetic endemism (Rosauer et al 2009)
- rpe: Relative phylogenetic endemism (Mishler et al 2014)

(pe and rpe are needed for CANAPE with cpr_classify_endem())

The choice of a randomization algorithm (null_model) is not trivial, and may strongly affect results. cpr_rand_test() uses null models provided by vegan; for a complete list, see the help file of vegan::commsim() or run vegan::make.commsim(). One frequently used null model is swap (Gotelli & Entsminger 2003), which randomizes the community matrix while preserving column and row sums (marginal sums). For a review of various null models, see Strona et al. (2018); swap is an "FF" model in the sense of Strona et al. (2018).

Instead of using one of the predefined null models in vegan::commsim(), it is also possible to define a custom null model; see Examples in cpr_rand_comm()

Note that the pre-defined models in vegan include binary models (designed for presence-absence data) and quantitative models (designed for abundance data). Although the binary models will accept abundance data, they treat it as binary and always return a binary (presence-absence) matrix. The PD and PE calculations in canaper are not abundance-weighted, so they return the same result regardless of whether the input is presence-absence or abundance. In that sense, binary null models are appropriate for cpr_rand_test(). The quantitative models could also be used for abundance data, but the output will be treated as binary anyways when calculating PD and PE. The effects of using binary vs. quantitative null models for cpr_rand_test() have not been investigated.

A minimum of 5 species and sites are required as input; fewer than that is likely cause the some randomization algorithms (e.g., swap) to enter an infinite loop. Besides, inferences on very small numbers of species and/or sites is not recommended generally.

The following rules apply to comm input:

- If dataframe or matrix, must include row names (site names) and column names (species names).
- If tibble, a single column (default, site) must be included with site names, and other columns must correspond to species names.
- Column names cannot start with a number and must be unique.
- Row names (site names) must be unique.
Values (other than site names) should only include integers >= 0; non-integer input will be converted to integer.

The results are identical regardless of whether the input for `comm` is abundance or presence-absence data (i.e., abundance weighting is not used).

### Value

**Dataframe.** For each of the biodiversity metrics, the following 9 columns will be produced:

- *obs*: Observed value
- *obs_c_lower*: Count of times observed value was lower than random values
- *obs_c_upper*: Count of times observed value was higher than random values
- *obs_p_lower*: Percentage of times observed value was lower than random values
- *obs_p_upper*: Percentage of times observed value was higher than random values
- *obs_q*: Count of the non-NA random values used for comparison
- *obs_z*: Standard effect size (z-score)
- *rand_mean*: Mean of the random values
- *rand_sd*: Standard deviation of the random values

So if you included `pd` in `metrics`, the output columns would include `pd_obs`, `pd_obs_c_lower`, etc...

### References


### Examples

```r
set.seed(12345)
data(phylocom)
# Returns a dataframe by defualt
cpr_rand_test(
    phylocom$comm, phylocom$phy,
    null_model = "curveball", metrics = "pd", n_reps = 10
)
# Tibbles may be preferable because of the large number of columns
```
cpr_rand_test(
  phylocom$comm, phylocom$phy,
  null_model = "curveball", tbl_out = TRUE, n_reps = 10
)

cpr_signif_cols_2

---

**cpr_signif_cols**

*CVD-friendly color palette for plotting results of randomization test*

**Description**

Character vector with names corresponding to significance levels and values corresponding to color codes.

**Usage**

`cpr_signif_cols`

**Format**

An object of class character of length 5.

**See Also**

Other colors: `cpr_endem_cols_2`, `cpr_endem_cols_3`, `cpr_endem_cols_4`, `cpr_endem_cols`, `cpr_make_pal()`, `cpr_signif_cols_2`, `mishler_endem_cols`, `mishler_signif_cols`

**Examples**

```r
library(scales)
show_col(cpr_signif_cols)
```

---

**cpr_signif_cols_2**

*CVD-friendly color palette for plotting results of randomization test, version 2*

**Description**

Character vector with names corresponding to significance levels and values corresponding to color codes, with "not significant" colored grey.

**Usage**

`cpr_signif_cols_2`

**Format**

An object of class character of length 5.
See Also

Other colors: cpr_endem_cols_2, cpr_endem_cols_3, cpr_endem_cols_4, cpr_endem_cols, 
cpr_make_pal(), cpr_signif_cols, mishler_endem_cols, mishler_signif_cols

Examples

cpr_signif_cols_2
scales::show_col(cpr_signif_cols_2)

mishler_endem_cols  
Original color palette for plotting results of CANAPE

Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Original palette used by Mishler et al. (2014). May not be distinguishable to people with color vision deficiency (CVD).

Usage

mishler_endem_cols

Format

An object of class character of length 5.

Details

Color scheme:

- paleo: blue
- neo: red
- not significant: beige
- mixed: light purple
- super: dark purple

References


See Also

Other colors: cpr_endem_cols_2, cpr_endem_cols_3, cpr_endem_cols_4, cpr_endem_cols, 
cpr_make_pal(), cpr_signif_cols_2, cpr_signif_cols, mishler_signif_cols
Examples

mishler_endem_cols
scales::show_col(mishler_endem_cols)

mishler_signif_cols  Original color palette for plotting results of CANAPE

Description

Character vector with names corresponding to endemism types and values corresponding to color codes. Original palette used by Mishler et al. (2014). May not be distinguishable to people with color vision deficiency (CVD).

Usage

mishler_signif_cols

Format

An object of class character of length 5.

References


See Also

Other colors: cpr_endem_cols_2, cpr_endem_cols_3, cpr_endem_cols_4, cpr_endem_cols, cpr_make_pal(), cpr_signif_cols_2, cpr_signif_cols, mishler_endem_cols

Examples

mishler_signif_cols
scales::show_col(mishler_signif_cols)
**phylocom**

---

**Phylocom example data**

**Description**

Fictional data for testing purposes from *Phylocom* (Webb et al. 2008)

**Usage**

`phylocom`

**Format**

A list with three elements:

- `phylo`  Phylogeny with 32 tips
- `sample` Community matrix with 6 sites (rows) and 25 species (columns).
- `traits` Trait data; a data.frame with 32 species (rows) and 4 traits (columns)

**Details**

Obtained via the `picante` package (Kembel et al. 2010)

**References**


**Examples**

```r
# Example phylogeny
phylocom$phy

# Example community
phylocom$comm
```
Index

* colors
  - cpr_endem_cols, 7
  - cpr_endem_cols_2, 8
  - cpr_endem_cols_3, 9
  - cpr_endem_cols_4, 10
  - cpr_make_pal, 12
  - cpr_signif_cols, 18
  - cpr_signif_cols_2, 18
  - mishler_endem_cols, 19
  - mishler_signif_cols, 20

* datasets
  - acacia, 2
  - biod_example, 3
  - biod_results, 4
  - cpr_endem_cols, 7
  - cpr_endem_cols_2, 8
  - cpr_endem_cols_3, 9
  - cpr_endem_cols_4, 10
  - cpr_signif_cols, 18
  - cpr_signif_cols_2, 18
  - mishler_endem_cols, 19
  - mishler_signif_cols, 20
  - phylocom, 21

acacia, 2
biod_example, 3, 4
biod_results, 4

cpr_classify_endem, 5, 16
  - cpr_classify_endem(), 13
  - cpr_classify_signif, 6
  - cpr_classify_signif(), 13
  - cpr_endem_cols, 7–10, 13, 18–20
  - cpr_endem_cols_2, 7–10, 13, 18–20
  - cpr_endem_cols_3, 7, 8, 9, 10, 13, 18–20
  - cpr_endem_cols_4, 7–9, 10, 13, 18–20
  - cpr_iter_sim, 11
  - cpr_iter_sim(), 11, 12
  - cpr_make_pal, 7–10, 12, 18–20

cpr_rand_comm, 13, 15, 16
  - cpr_rand_test, 15
  - cpr_rand_test(), 5, 6
  - cpr_signif_cols, 7–10, 13, 18, 19, 20
  - cpr_signif_cols_2, 7–10, 13, 18, 19, 20
  - mishler_endem_cols, 7–10, 13, 18, 19, 20
  - mishler_signif_cols, 7–10, 13, 18, 19, 20

  phylocom, 21

vegan::commsim(), 11, 14–16
vegan::make.commsim(), 11, 14–16