Package ‘FuncDiv’

March 8, 2023

Title Compute Contributional Diversity Metrics

Version 1.0.0

Description Compute alpha and beta contributional diversity metrics, which is intended for linking taxonomic and functional microbiome data. See 'GitHub' repository for the tutorial: <`https://github.com/gavinmdouglas/FuncDiv/wiki`>. Citation: Gavin M. Douglas, Sunu Kim, Morgan G. I. Langille, B. Jesse Shapiro (2023) <doi:10.1093/bioinformatics/btac809>.

License AGPL-3

Depends R (>= 3.5.0)

Imports ape, collapse, data.table, parallel, parallelDist, Rcpp, RcppParallel, RcppXPtrUtils, stats

Suggests testthat (>= 3.0.0)

LinkingTo Rcpp, RcppArmadillo, RcppParallel

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation yes

Author Gavin Douglas [aut, cre] (<`https://orcid.org/0000-0001-5164-6707`>)

Maintainer Gavin Douglas <`gavinmdouglas@gmail.com`>

Repository CRAN

Date/Publication 2023-03-08 19:20:02 UTC

R topics documented:

alpha_div_contrib *Main function for computing contributional* **alpha** *diversity*

Description

Based on joint taxa-function input data (i.e., contributional data), a dataframe will be returned for each specified metric, which will contain the metric values for all function and sample combinations.

Usage

```
alpha_div_contrib(
  metrics,
  func_tab = NULL,
  abun_tab = NULL,
  contrib_tab = NULL,
  in_tree = NULL,
  ncores = 1,
  replace_NA = FALSE,
  custom_metric_functions = NULL,
  samp_colname = "sample",
  func_colname = "function.",
  taxon_colname = "taxon",
  abun_colname = "taxon_abun"
)
```

Arguments

`metrics`	alpha diversity metrics to compute. Must either be names of functions in `FuncDiv_alpha_metrics`, or alternatively in `custom_metric_functions`, if specified.
`func_tab`	data.frame object containing function copy numbers, with rows as functions and columns as taxa. Required if `abun_tab` is specified, and is mutually exclusive with `contrib_tab`.
`abun_tab`	data.frame object containing taxonomic abundances across samples, with rows as taxa and columns as samples. Required if `func_tab` is specified, and is mutually exclusive with `contrib_tab`.
`contrib_tab`	data.frame object containing combined taxa abundances and function copy numbers across taxa. Must contain columns corresponding to the sample ids, function ids, taxa ids, and taxa abundances within samples. These column names are specified by the `samp_colname`, `func_colname`, `taxon_colname`, and `abun_colname`, respectively. Mutually exclusive with `abun_tab` and `func_tab`.
`in_tree`	phylo object to use if `faiths_pd` is specified.

`ncores`	integer indicating number of cores to use for parallelizable steps.
`replace_NA`	Boolean vector of length one, indicating whether all NA's in the output of all metrics should be converted to 0's. Note that this done automatically done for `richness` either way.
`custom_metric_functions`	List object containing custom alpha diversity metric functions. This overrides `FuncDiv_alpha_metrics` when specified. The list element names must correspond to at least the names indicated by the `metrics` parameter.
`samp_colname`	sample id column name of `contrib_tab` input data.frame.
`func_colname`	function id column name of `contrib_tab` input data.frame.
`taxon_colname`	taxon id column name of `contrib_tab` input data.frame.
`abun_colname`	taxonomic abundance (within each sample) column name of `contrib_tab` input data.frame.

Details

Input data can be either a separate function copy number and taxonomic abundance table, or a joint contributional table. By default, specified metrics must be one of `names(FuncDiv_alpha_metrics)`. However, custom alpha diversity metric functions can be specified with the `custom_metric_functions` parameter.

Note that the taxonomic abundances can be relative abundance, read counts, or transformed in another way. However, note that some default metrics are only compatible with count data (see `?FuncDiv_alpha_metrics`).

Value

a list, containing one dataframe for each specified alpha diversity metric. In each dataframe, rows are functions and samples are columns.

Examples

```
# First, simulate some (non-realistic) data.
set.seed(123)
test_tree <- ape::rtree(100)
test_abun <- data.frame(matrix(rnorm(500), nrow = 100, ncol = 5))
rownames(test_abun) <- test_tree$tip.label
colnames(test_abun) <- c("sample1", "sample2", "sample3", "sample4", "sample5")
test_abun[test_abun < 0] <- 0
test_func <- data.frame(matrix(sample(c(0L, 1L), 200, replace = TRUE),
                               nrow = 2, ncol = 100))
colnames(test_func) <- test_tree$tip.label
rownames(test_func) <- c("func1", "func2")

# Compute alpha diversity, based on (observed) richness, Faith's phylogenetic
# diversity, and the Gini-Simpson Index.
contrib_alpha <- alpha_div_contrib(metrics = c("richness", "faiths_pd", "gini_simpson_index"),
                                   func_tab = test_func,
                                   abun_tab = test_abun,
                                   in_tree = test_tree,
                                    ncores = 1)

# Print out computed Gini-Simpson Index values.
contrib_alpha$gini_simpson_index
```

`beta_div_contrib` *Main function for computing contributional* **beta** *diversity*

Description

Based on joint taxa-function input data (i.e., contributional data), the beta diversity (i.e., inter-sample distance or divergence) will be computed for the subset of taxa encoding each individual function separately. A large List object containing all these tables can be returned, or alternatively these tables will be written to the disk as plain-text files.

Usage

```
beta_div_contrib(
  metrics = NULL,
  func_tab = NULL,
  abun_tab = NULL,
  contrib_tab = NULL,
  in_tree = NULL,
  func_ids = NULL,
  return_objects = FALSE,
  write_outfiles = FALSE,
  outdir = NULL,
  ncores = 1,
  samp_colname = "sample",
  func_colname = "function.",
  taxon_colname = "taxon",
  abun_colname = "taxon_abun"
)
```

Arguments

`metrics`	beta diversity metrics to compute. Must be default metric computed by `parallelDist::parDist` or one of "weighted_unifrac", "unweighted_unifrac", or "jensen_shannon_div".
`func_tab`	data.frame object containing function copy numbers, with rows as functions and columns as taxa. Required if `abun_tab` is specified, and is mutually exclusive with `contrib_tab`.
`abun_tab`	data.frame object containing taxonomic abundances across samples, with rows as taxa and columns as samples. Required if `func_tab` is specified, and is mutually exclusive with `contrib_tab`.

`contrib_tab`	data.frame object containing combined taxa abundances and function copy numbers across taxa. Must contain columns corresponding to the sample ids, function ids, taxa ids, and taxa abundances within samples. These column names are specified by the `samp_colname`, `func_colname`, `taxon_colname`, and `abun_colname`, respectively.Mutually exclusive with `abun_tab` and `func_tab`.
`in_tree`	phylo object to use if `weighted_unifrac` or `unweighted_unifrac` are specified.
`func_ids`	character vector specifying subset of function ids to include for analysis. Will analyze all functions present if this is not specified.
`return_objects`	Boolean vector of length one, specifying whether function should return a list of all output distance tables (nested by metric name, and then by function id). Incompatible with `write_outfiles`.
`write_outfiles`	Boolean vector of length one, specifying whether function write all distance tables to plain-text files in the specified `outdir` location. Incompatible with `return_objects`.
`outdir`	character vector of length one, indicating where to save output files if `write_outfiles = TRUE`.
`ncores`	integer indicating number of cores to use for parallelizable steps.
`samp_colname`	sample id column name of `contrib_tab` input data.frame.
`func_colname`	function id column name of `contrib_tab` input data.frame.
`taxon_colname`	taxon id column name of `contrib_tab` input data.frame.
`abun_colname`	taxonomic abundance (within each sample) column name of `contrib_tab` input data.frame.

Details

Input data can be either a separate function copy number and taxonomic abundance table, or a joint contributional table. Metrics must be one of "weighted_unifrac", "unweighted_unifrac", "jensen_shannon_div", or a default metric available through the `parallelDist::parDist` function. See `?parallelDist::parDist` for a description of all default metrics.

The taxonomic abundances will be converted to relative abundances prior to computing inter-sample distances.

Value

differs depending on the `return_objects` and `write_outfiles` parameters.

If `return_objects = TRUE`, then a nested List will be returned. Each specific beta diversity metric will be the first level, and the functions are the second level (e.g., contrib_beta$binary$func2).

If `write_outfiles` then a character vector will be returned, indicating where the output tables were written.

Examples

```
# First, simulate some (non-realistic) data.
set.seed(123)
```

```
test_tree <- ape::rtree(100)
test_abun <- data.frame(matrix(rnorm(500), nrow = 100, ncol = 5))
rownames(test_abun) <- test_tree$tip.label
colnames(test_abun) <- c("sample1", "sample2", "sample3", "sample4", "sample5")
test_abun[test_abun < 0] <- 0
test_func <- data.frame(matrix(sample(c(0L, 1L), 200, replace = TRUE),
                               nrow = 2, ncol = 100))
colnames(test_func) <- test_tree$tip.label
rownames(test_func) <- c("func1", "func2")

# Compute beta diversity, based on Weighted UniFrac and Jaccard distances
# (i.e., "binary").
contrib_beta <- beta_div_contrib(metrics = c("weighted_unifrac", "binary"),
                                 func_tab = test_func,
                                 abun_tab = test_abun,
                                 in_tree = test_tree,
                                 return_objects = TRUE,
                                 ncores = 1)

# Parse beta diversity distance list value for a specific function (func2) and
# distance metric (Jaccard).
contrib_beta$binary$func2
```

compute_alpha_div *Convenience function for running default alpha diversity metrics on a single vector input*

Description

This is a simple wrapper for `FuncDiv_alpha_metrics`, and you can see more details with `?FuncDiv_alpha_metrics`.

Usage

```
compute_alpha_div(x, metric, ...)
```

Arguments

`x`	input vector. Either class numeric (representing abundance of categories [e.g., microbes]) or character (indicating which taxa are present, which is required for `faiths_pd`).
`metric`	alpha diversity metric to compute. Must be one of `names(FuncDiv_alpha_metrics)`.
`...`	included so that functions with single arguments will not throw errors if `tree` is included (and ignored). This should be a phylo object to use in case of `faiths_pd`.

Details

These functions all have a single input: a numeric vector containing taxa abundances within a given sample. The exception is for `faiths_pd`, which expects a character vector of taxa labels that are present, as well as a tree (phylo object), which must contain all these specified taxa labels as tip labels.

Value

numeric vector with alpha diversity value.

Examples

```
# Most metrics just require an input vector of abundances.
test_abun <- c(0, NA, 1, 2, 10, 4)
compute_alpha_div(x = test_abun, metric = "richness")

# Note that the input for computing Faith's PD is different.
# Get a randomly generated tree:
test_tree <- ape::rtree(n = 50)
test_present_tips <- c('t1', 't2', 't3')
compute_alpha_div(x = test_present_tips, metric = "faiths_pd", tree = test_tree)
```

`contrib_to_multitab` *Utility function to convert from contributional to multi-table input objects*

Description

Converts from contributional-type table (i.e., a single, long table with joint taxa/function information) to separate taxa abundance and function copy number tables.

Usage

```
contrib_to_multitab(
  contrib_tab,
  samp_colname = "sample",
  func_colname = "function.",
  abun_colname = "taxon_abun",
  taxon_colname = "taxon",
  copy.num_colname = "genome_function_count"
)
```

Arguments

`contrib_tab`	data.frame object containing combined taxa abundances and function copy numbers across taxa. Must contain columns corresponding to the sample ids, function ids, taxa ids, and taxa abundances within samples. These column names are specified by the `samp_colname`, `func_colname`, `taxon_colname`, `abun_colname`, and `copy.num_colname`, respectively.
`samp_colname`	sample id column name of `contrib_tab` input data.frame.
`func_colname`	function id column name of `contrib_tab` input data.frame.
`abun_colname`	taxonomic abundance (within each sample) column name of `contrib_tab` input data.frame.
`taxon_colname`	taxon id column name of `contrib_tab` input data.frame.
`copy.num_colname`	function copy number column name of `contrib_tab` input data.frame.

Value

list with taxon abundance (`taxon_abun`) and function copy number (`function_copy_num`) data.frames as separate elements.

`FuncDiv_alpha_metrics` *List object containing the functions to compute the default alpha diversity metrics*

Description

These functions are used by the `alpha_div_contrib` function to compute contributional diversity, but can be used for any arbitrary input vector as well to compute standard alpha diversity.

Usage

```
FuncDiv_alpha_metrics
```

Format

An object of class `list` of length 15.

Details

The metrics were primarily taken from definitions provided by `scikit-bio` Python package, as well as the `vegan` and `picante` R packages. The functions are provided as elements of this list, so that it is more convenient to call them programatically. All available alpha diversity metrics can be seen by typing `names(FuncDiv_alpha_metrics)`. The code to compute each metric can be inspected for each function, for instance, for richness, by typing: `FuncDiv_alpha_metrics$richness`.

These functions all have a single input: a numeric vector containing taxa abundances within a given sample. The exception is for `faiths_pd`, which expects a character vector of taxa labels that are present, as well as a tree (phylo object), which must contain all these specified taxa labels as tip labels.

Note that not all these metrics are appropriate for relative abundance data. In particular, these metrics expect count data (e.g., read counts) corresponding to the number of occurrences of each category (e.g., each microbe): `menhinicks_richness`, `mcintoshs_evenness`, `mcintoshs_dominance`, `margalefs_richness`, and `fishers_alpha`.

Value

numeric vector with alpha diversity value.

Examples

```
# Most metrics just require an input vector of abundances.
test_abun <- c(0, NA, 1, 2, 10, 4)
FuncDiv_alpha_metrics[["richness"]](test_abun)

# Note that the input for computing Faith's PD is different.
# Get a randomly generated tree:
test_tree <- ape::rtree(n = 50)
test_present_tips <- c('t1', 't2', 't3')
FuncDiv_alpha_metrics[["faiths_pd"]](test_present_tips, test_tree)
```

`func_abun_crossproduct` *Utility function to get community-wide function abundance table*

Description

Takes in table of function copy numbers across taxa and table of taxa abundances across samples. I.e., it represents the multiplication of the function copy numbers by the abundances of the taxa within each sample.

Usage

```
func_abun_crossproduct(func_tab, abun_tab)
```

Arguments

`func_tab`	data.frame object containing function copy numbers, with rows as functions and columns as taxa.
`abun_tab`	data.frame object containing taxonomic abundances across samples, with rows as taxa and columns as samples.

Value

data.frame representing the *unnormalized* community-wide abundances of functions across samples.

multitab_to_contrib *Utility function to convert from multi-table objects to contributional table*

Description

Converts from separate taxa abundance and function copy number table input style to contributional-type table (i.e., a single, long table with joint taxa/function information).

Usage

```
multitab_to_contrib(
  func_tab,
  abun_tab,
  ncores = 1,
  samp_colname = "sample",
  func_colname = "function.",
  abun_colname = "taxon_abun",
  taxon_colname = "taxon",
  copy.num_colname = "genome_function_count"
)
```

Arguments

`func_tab`	data.frame object containing function copy numbers, with rows as functions and columns as taxa.
`abun_tab`	data.frame object containing taxonomic abundances across samples, with rows as taxa and columns as samples.
`ncores`	integer specifying number of cores to use for parallizable steps.
`samp_colname`	sample id column name of `contrib_tab` output data.frame.
`func_colname`	function id column name of `contrib_tab` output data.frame.
`abun_colname`	taxonomic abundance (within each sample) column name of `contrib_tab` output data.frame.
`taxon_colname`	taxon id column name of `contrib_tab` output data.frame.
`copy.num_colname`	function copy number (within each taxa) column name of `contrib_tab` output data.frame.

Value

data.frame in contributional format (i.e., single, long-format version of both input tables).

`subset_func_and_abun_tables` *Utility function to subset function copy number and taxonomic abundance tables*

Description

The input tables will be returned except subset to the same taxa ids. Any functions and / or samples that are totally absent after this step will be dropped.

Usage

```
subset_func_and_abun_tables(func_table, abun_table, func_ids = NULL)
```

Arguments

`func_table`	data.frame object containing function copy numbers, with rows as functions and columns as taxa.
`abun_table`	data.frame object containing taxonomic abundances across samples, with rows as taxa and columns as samples.
`func_ids`	optional character vector of function ids to retain (all other rows of `func_tab` will be removed).

Value

list containing subsetted function and abundance data.frames as separate elements.

Index