Package ‘meconetcomp’
October 13, 2022

Type    Package
Title   Compare Microbial Networks of ‘trans_network’ Class of
        ‘microeco’ Package
Version 0.2.0
Author  Chi Liu [aut, cre],
        Minjie Yao [ctb],
        Xiangzhen Li [ctb]
Maintainer Chi Liu <liuchi0426@126.com>
Description Compare microbial co-
        occurrence networks created from ‘trans_network’ class of ‘microeco’ package <https://
        github.com/ChiLiubio/microeco>.
        This package is the extension of ‘trans_network’ class of ‘microeco’ package and especially use-
        ful when different networks are constructed and analyzed simultaneously.
URL    https://github.com/ChiLiubio/meconetcomp
Depends R (>= 3.5.0)
Imports R6, microeco (>= 0.12.0), magrittr, dplyr, igraph, reshape2,
        ggpubr
Suggests rgexf, ape, file2meco, agricolae
License GPL-3
LazyData true
Encoding UTF-8
NeedsCompilation no
Repository CRAN
Date/Publication 2022-10-05 15:20:02 UTC
RoxygenNote 7.2.1

R topics documented:
cal_module ................................................................. 2
cal_network_attr ....................................................... 3
cal_module

Assign modules to each network

Description

Calculating modularity of networks and assign the modules to nodes for each network.

Usage

```r
cal_module(
  network_list,
  undirected_method = "cluster_fast_greedy",
  directed_method = "cluster_optimal",
  ...
)
```

Arguments

- `network_list` a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.
- `undirected_method` default "cluster_fast_greedy"; the modularity algorithm for undirected network; see `cal_module` function of `trans_network` class for more algorithms.
- `directed_method` default 'cluster_optimal'; the modularity algorithm for directed network.
- `...` other parameters (except for method) passed to `cal_module` function of `trans_network` class.

Value

list, with module attribute in nodes of each network
**cal_network_attr**

**Examples**
```
data(soil_amp_network)
soil_amp_network <- cal_module(soil_amp_network)
```

---

**cal_network_attr**

*Calculate network topological property for each network*

**Description**

Calculate the topological properties of all the networks and merge the results into one table.

**Usage**
```
cal_network_attr(network_list)
```

**Arguments**
- `network_list` a list with multiple networks; all the networks should be trans_network object created from `trans_network` class of microeco package.

**Value**
- data.frame

**Examples**
```
data(soil_amp_network)
test <- cal_network_attr(soil_amp_network)
```

---

**edge_comp**

*Generate a microtable object with paired nodes distributions of edges across networks*

**Description**

Generate a microtable object with paired nodes distributions of edges across networks. Useful for the edge comparisons across different networks. The return `otu_table` in microtable object has the binary numbers in which 1 represents the presence of the edge in the corresponding network.

**Usage**
```
edge_comp(network_list)
```
edge_node_distance

Arguments

network_list a list with multiple networks; all the networks should be trans_network object created from trans_network class of microeco package.

Value

microtable object

Examples

data(soil_amp_network)
  test <- edge_comp(soil_amp_network)
  # test is a microtable object

Description

This class is a wrapper for a series of analysis on the distance values of paired nodes in edges across networks, including distance matrix conversion, the differential test and the visualization.

Methods

Public methods:

- edge_node_distance$new()
- edge_node_distance$cal_diff()
- edge_node_distance$plot()
- edge_node_distance$clone()

Method new():

Usage:

edge_node_distance$new(
  network_list, 
  dis_matrix = NULL, 
  label = "+", 
  with_module = FALSE, 
  module_thres = 2
)

Arguments:

network_list a list with multiple networks; all the networks should be trans_network object created from trans_network class of microeco package.

dis_matrix default NULL; the distance matrix of nodes, used for the value extraction; must be a symmetrical matrix with both colnames and rownames (i.e. feature names).
label default "+"; "+" or ";" or c("+", ";") the edge label used for the selection of edges.
with_module default FALSE; whether show the module classification of nodes in the result.
module_thres default 2; the threshold of the nodes number of modules remained when with_module
  = TRUE.

Returns: data_table, stored in the object

Examples:
\dontrtest{
data(soil_amp_network)
data(soil_amp)
# filter useless features to speed up the calculation
node_names <- unique(unlist(lapply(soil_amp_network, function(x){colnames(x$data_abund)}))))
filter_soil_amp <- microeco::clone(soil_amp)
filter_soil_amp$otu_table <- filter_soil_amp$otu_table[node_names, ]
filter_soil_amp$tidy_dataset()
# obtain phylogenetic distance matrix
phylogenetic_distance <- as.matrix(cophenetic(filter_soil_amp$phylo_tree))
# choose the positive labels
t1 <- edge_node_distance$new(network_list = soil_amp_network,
  dis_matrix = phylogenetic_distance, label = "+")
}


Usage:
edge_node_distance$cal_diff(  
  method = c("anova", "KW", "KW_dunn", "wilcox", "t.test")[1],
  ...
)

Arguments:
method default "anova"; see the following available options:
  
  'anova' Duncan's multiple range test for anova
  'KW' KW: Kruskal-Wallis Rank Sum Test for all groups (>= 2)
  'KW_dunn' Dunn's Kruskal-Wallis Multiple Comparisons, see dunnTest function in FSA package
  'wilcox' Wilcoxon Rank Sum and Signed Rank Tests for all paired groups
  't.test' Student's t-Test for all paired groups
  ...
  parameters passed to cal_diff function of trans_alpha class of microeco package.

Returns: res_diff in object. See the Return of cal_diff function in trans_alpha class of microeco package.

Examples:
\dontrtest{
t1$cal_diff(method = "wilcox")
}

Method plot(): Plot the distance.

Usage:
edge_node_distance$plot(…)

Arguments:
… parameters pass to plot_alpha function of trans_alpha class of microeco package.

Returns: ggplot.

Examples:
\donttest{
t1$plot(boxplot_add = "none", add_sig = TRUE)
}

Method clone(): The objects of this class are cloneable with this method.

Usage:
edge_node_distance$clone(deep = FALSE)

Arguments:
deep Whether to make a deep clone.

Examples

## ------------------------------------------------
## Method `edge_node_distance$new`
## ------------------------------------------------
data(soil_amp_network)
data(soil_amp)
# filter useless features to speed up the calculation
node_names <- unique(unlist(lapply(soil_amp_network, function(x){colnames(x$data_abund)}))
filter_soil_amp <- microeco::clone(soil_amp)
filter_soil_amp$otu_table <- filter_soil_amp$otu_table[, node_names]
filter_soil_amp$tidy_dataset()
# obtain phylogenetic distance matrix
phylogenetic_distance <- as.matrix(cophenetic(filter_soil_amp$phylo_tree))
# choose the positive labels
t1 <- edge_node_distance$new(network_list = soil_amp_network, dis_matrix = phylogenetic_distance, label = "+")

## ------------------------------------------------
## Method `edge_node_distance$cal_diff`
## ------------------------------------------------
t1$cal_diff(method = "wilcox")

## ------------------------------------------------
## Method `edge_node_distance$plot`
## ------------------------------------------------
edge_tax_comp

```
t1$plot(boxplot_add = "none", add_sig = TRUE)
```

---

**edge_tax_comp**  
*Taxonomic sum of linked nodes in edges across networks*

**Description**

Taxonomic sum of linked nodes in edges across networks.

**Usage**

```
edge_tax_comp(network_list, taxrank = "Phylum", label = "+", rel = TRUE)
```

**Arguments**

- `network_list`: a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.
- `taxrank`: default "Phylum"; Which taxonomic level is used for the sum of nodes in edges.
- `label`: default "+"; "+" or "-" or `c("+", ":-")`; the edge label used for the selection of edges for the sum.
- `rel`: default TRUE; TRUE represents using ratio, the denominator is the number of selected edges; FALSE represents the absolute number of the sum of edges.

**Value**

data.frame

**Examples**

```
data(soil_amp_network)
test <- edge_tax_comp(soil_amp_network)
# test is a microtable object
```
**get_edge_table**

Get edge property table for each network in the list with multiple networks.

**Usage**

```r
get_edge_table(network_list)
```

**Arguments**

- `network_list` 
  a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.

**Value**

list, with `res_edge_table` in each network

**Examples**

```r
data(soil_amp_network)
soil_amp_network <- get_edge_table(soil_amp_network)
```

---

**get_node_table**

Get node property table for each network in the list with multiple networks.

**Usage**

```r
get_node_table(network_list, ...)
```

**Arguments**

- `network_list` 
  a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.

- `...` 
  parameter passed to `get_node_table` function of `trans_network` class.

**Value**

list, with `res_node_table` in each network
Generate a microtable object with node distributions across networks.

**node_comp**

**Description**

Generate a microtable object with node distributions across networks. Useful for the node information comparisons across different networks.

**Usage**

```r
node_comp(network_list, property = "name")
```

**Arguments**

- `network_list`: a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.
- `property`: default "name"; a colname of `res_node_table` in each network; the default "name" represents using node presence/absence information in the `otu_table` of final output, in which 1 represents presence of the node in the corresponding network; For other options (such as degree), the results in the output `otu_table` are the actual values of `res_node_table`.

**Value**

microtable object

**Examples**

```r
data(soil_amp_network)
soil_amp_network <- get_node_table(soil_amp_network, node_roles = FALSE)

data(soil_amp_network)
test <- node_comp(soil_amp_network)
# test is a microtable object
```
soil_amp

**Description**


**Usage**

```r
data(soil_amp)
```

soil_amp_network

**Description**

The soil_amp_network data is a list storing three trans_network objects created based on soil_amp data. Three networks are created for IW, CW and TW groups, respectively.

**Usage**

```r
data(soil_amp_network)
```

soil_measure_diversity

**Description**

The soil_measure_diversity data is a table storing all the abiotic factors and functional diversity based on the metagenomic sequencing and MetaCyc pathway analysis.

**Usage**

```r
data(soil_measure_diversity)
```
### stool_met

**The stool_met data**

The stool_met data is the metagenomic species abundance dataset of stool samples selected from R ExperimentHub package. It has 198 samples, collected from the people with alcohol drinking habit, and 92 species.

**Usage**

```r
data(stool_met)
```

### subnet_property

**Calculate properties of sub-networks selected according to features in samples**

Extracting sub-network according to the presence of features in each sample across networks and calculate the sub-network properties.

**Usage**

```r
subnet_property(network_list)
```

**Arguments**

- `network_list` a list with multiple networks; all the networks should be trans_network object created from `trans_network` class of `microeco` package.

**Value**

- `data.frame`

**Examples**

```r
data(soil_amp_network)
test <- subnet_property(soil_amp_network)
```
### subset_network

*Extract subset of network according to the edge intersection of networks*

#### Description

Extracting a network according to the edge intersection of networks.

#### Usage

```r
subset_network(network_list, venn = NULL, name = NULL)
```

#### Arguments

- `network_list` a list with multiple networks; all the networks should be `trans_network` object created from `trans_network` class of `microeco` package.
- `venn` default `NULL`; a microtable object which must be converted by `trans_comm` function of `trans_venn` class.
- `name` default `NULL`; integer or character; must be a number or one of colnames of the `otu_table` in the input `venn` parameter.

#### Value

`trans_network` object, with only the extracted edges in the network

#### Examples

```r
data(soil_amp_network)
# first obtain edge distribution
tmp <- edge_comp(soil_amp_network)
# obtain edge intersection using trans_venn class
tmp1 <- microeco::trans_venn$new(tmp)
# convert intersection result to microtable object
tmp2 <- tmp1$trans_comm()
# extract the intersection of all the three networks ("IW", "TW" and "CW")
test <- subset_network(soil_amp_network, venn = tmp2, name = "IW&TW&CW")
# test is a trans_network object
```
Index

* **data.frame**
  soil_measure_diversity, 10

* **list**
  soil_amp_network, 10

* **microtable**
  soil_amp, 10
  stool_met, 11

  cal_module, 2
  cal_network_attr, 3

  edge_comp, 3
  edge_node_distance, 4
  edge_tax_comp, 7

  get_edge_table, 8
  get_node_table, 8

  node_comp, 9

  soil_amp, 10
  soil_amp_network, 10
  soil_measure_diversity, 10
  stool_met, 11
  subnet_property, 11
  subset_network, 12

  trans_network, 2–4, 7–9, 11, 12