Package ‘pctax’

December 2, 2024

Type Package

Title Professional Comprehensive Omics Data Analysis

Version 0.1.3

Description Provides a comprehensive suite of tools for analyzing omics data. It includes functionalities for alpha diversity analysis, beta
diversity analysis, differential abundance analysis, community assembly analysis, visualization of phylogenetic tree, and
functional enrichment analysis. With a progressive approach, the package offers a range of analysis methods to explore and
understand the complex communities. It is designed to support researchers and practitioners in conducting in-depth and
professional omics data analysis.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 4.2.0)

LazyData true

Imports pcutils (>= 0.2.5), dplyr, ggplot2 (>= 3.2.0), vegan, magrittr, grDevices, RColorBrewer, ggrepel, reshape2, stats, tibble, utils, ggpubr, ggnewscale, ade4, scales

Suggests picante, httr, NST, permute, aplot, pheatmap, MASS, Rtsne, mixOmics, geosphere, phyloseq, phyloseqGraphTest, plotly, umap, Hmisc, minpack.lm, bbmle, snow, foreach, doSNOW, patchwork, tidytree, ggtree, ggtreeExtra, vctrs, zoo, ape, DESeq2, limma, ALDEx2, Mfuzz, edgeR, methods, randomForest, knitr, rmarkdown, MetaNet, showtext, jsonlite, prettydoc, readxl, clipr, zetadiv, ggforce

VignetteBuilder knitr

BugReports `https://github.com/Asa12138/pctax/issues`

URL `https://github.com/Asa12138/pctax`

ByteCompile true

biocViews Microbiome, Software, Visualization

NeedsCompilation no

Author Chen Peng [aut, cre] (<https://orcid.org/0000-0002-9449-7606>)

Maintainer Chen Peng <pengchen2001@zju.edu.cn>

Repository CRAN

Date/Publication 2024-12-02 10:00:02 UTC

Contents

`add_strip` *add strips for a tree plot*

Description

add strips for a tree plot

Usage

```
add_strip(trp, some_tax, flat_n = 5, strip_params = NULL)
```

Arguments

`trp`	tree plot from `ggtree`
`some_tax`	some tax you want to add strip
`flat_n`	flat the text when taxa number more than `flat_n`.
`strip_params`	parameters parse to `geom_strip`

Value

tree plot

Examples

```
data(otutab, package = "pcutils")
# run yourself
if (interactive()) {
  ann_tree(taxonomy, otutab) -> tree
  easy_tree(tree) -> p
  some_tax <- table(taxonomy$Phylum) %>%
    sort(decreasing = TRUE) %>%
    head(5) %>%
    names()
  add_strip(p, some_tax)
}
```

`add_tax` *Add taxonomy for a pc_otu object*

Description

Add taxonomy for a pc_otu object

Usage

```
add_tax(pc, taxonomy)
```

Arguments

`pc`	a pc_otu object
`taxonomy`	a taxomomy data.frame, look out the rownames of taxonomy and otutab should matched!

Value

pc_otu

Examples

```
data(otutab, package = "pcutils")
pc_tax1 <- pc_otu(otutab, metadata)
pc_tax1 <- add_tax(pc_tax1, taxonomy)
```

ALDEX *ALDEX*

Description

ALDEX

Usage

```
ALDEX(otutab, group_df)
```

Arguments

otutab	otutab
group_df	a dataframe with rowname same to dist and one group column

Value

diff

References

https://cloud.tencent.com/developer/article/1621879

Examples

```
if (requireNamespace("ALDEx2")) {
  data(otutab, package = "pcutils")
  ALDEX(otutab, metadata["Group"]) -> res
  res %>%
    dplyr::top_n(9, -glm.eBH) %>%
    .[, "tax"] -> sig
  data.frame(t(otutab[sig, ])) %>% pcutils::group_box(., "Group", metadata)
}
```

`all_ec_info` *all element cycle information.*

Description

all element cycle information.

Format

a list contains four tables.

ec_node chemicals

ec_link reactions

ec_gene genes

ec_path reactions labels

`all_sp_la_zh_name` *all species latin names and chinese names*

Description

all species latin names and chinese names.

Format

a dataframe.

latin_name latin name

chinese_name chinese name

`ann_tree` *Annotate a tree*

Description

Annotate a tree

Easy way to plot a phylogenetic tree

Usage

```
ann_tree(f_tax, otutab = NULL, level = ncol(f_tax))

easy_tree(
  tree,
  highlight = "Phylum",
  colorfill = "color",
  topN = NULL,
  pal = NULL,
  name_prefix = FALSE,
  basic_params = NULL,
  add_abundance = TRUE,
  color_name = "abundance",
  add_tiplab = TRUE,
  fontsize = NULL
)
```

Arguments

`f_tax`	taxonomy dataframe
`otutab`	otutab, rowname==rowname(taxonomy)
`level`	1~7
`tree`	result from `ann_tree`
`highlight`	highlight which level, one of `tree$level`
`colorfill`	"color" or "fill"
`topN`	topN to show
`pal`	color pal
`name_prefix`	keep the prefix like "k__" or "p__" in the label? Default: FALSE
`basic_params`	parameters parse to `ggtree`
`add_abundance`	logical
`color_name`	color name
`add_tiplab`	logical
`fontsize`	tip label fontsize

Value

a treedata

a ggplot

Examples

```
if (interactive()) {
  data(otutab, package = "pcutils")
  ann_tree(taxonomy, otutab) -> tree
  # run yourself
  easy_tree(tree, add_abundance = FALSE) -> p
  p
}
```

---

aor *Calculate Abundance-occupancy_relationship*

Description

Calculate Abundance-occupancy_relationship

Plot a AOR

Usage

```
aor(ototab, ...)

## S3 method for class 'data.frame'
aor(
  otutab,
  top_r = 0.7,
  ocup_n = ceiling(0.8 * ncol(otutab)),
  special_n = ceiling(0.1 * ncol(otutab)),
  ...
)

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

Arguments

`otutab`	otutab
`...`	add
`top_r`	percentage of top relative abundance
`ocup_n`	percentage of top occupied
`special_n`	how many occupancy define as specialists
`x`	AOR object

Value

AOR

ggplot

References

Barberán, A., Bates, S. T., Casamayor, E. & Fierer, N. (2012) Using network analysis to explore co-occurrence patterns in soil microbial communities.

Examples

```
data(otutab, package = "pcutils")
aor(otutab) -> AOR
plot(AOR)
```

`as.b_dist` *Transfer dist to b_dist*

Description

Transfer dist to b_dist

Plot dist

Plot b_dist

Usage

```
as.b_dist(dist, group_df = NULL)

## S3 method for class 'dist'
plot(x, group_df = NULL, ...)

## S3 method for class 'b_dist'
plot(x, mode = 1, c_group = "inter", ...)
```

Arguments

`dist`	a dist object
`group_df`	a dataframe with rowname same to dist and one group column
`x`	a b_dist
`...`	additional
`mode`	1~3
`c_group`	"inter" or "intra" or both to plot

Value

a b_dist with annotation by group

a pheatmap

a ggplot or pheatmap

Examples

```
data(otutab, package = "pcutils")
mat_dist(otutab) %>% as.b_dist(., group_df = metadata["Group"]) -> aa
plot(aa)
plot(aa, mode = 2)
```

as.dist.b_dist *Transfer b_dist to dist*

Description

Transfer b_dist to dist

Usage

```
## S3 method for class 'b_dist'
as.dist(m, diag = FALSE, upper = FALSE)
```

Arguments

`m`	a b_dist object
`diag`	logical value indicating whether the diagonal of the distance matrix should be printed by `print.dist`.
`upper`	logical value indicating whether the upper triangle of the distance matrix should be printed by `print.dist`.

Value

dist

a_diversity *Calculate a_diversity of otutab*

Description

Calculate a_diversity of otutab

Usage

```
a_diversity(otutab, ...)

## S3 method for class 'data.frame'
a_diversity(
  otutab,
  method = c("richness", "shannon"),
  tree = NULL,
  digits = 4,
  ...
)

## S3 method for class 'pc_otu'
```

```
a_diversity(otutab, method = "all", tbl = "otutab", ...)

## S3 method for class 'numeric'
a_diversity(otutab, ...)
```

Arguments

`otutab`	numeric
`...`	pass to `a_diversity.data.frame`
`method`	one of "all","richness","chao1","ace","gc","shannon","simpson","pd","pielou"
`tree`	a iphylo object match the rownames of otutab
`digits`	maintance how many digits
`tbl`	which table

Value

a a_res object

Examples

```
data(otutab, package = "pcutils")
a_diversity(otutab) -> a_res
plot(a_res, "Group", metadata)
```

`bbtt` *ggdotchart for diff analysis*

Description

ggdotchart for diff analysis

Usage

```
bbtt(res, pvalue = "glm.eBH", topN = 20)
```

Arguments

`res`	result of ALDEX or kwtest
`pvalue`	the name of pvaule
`topN`	topN

Value

ggplot

before_tree *Before df2tree check*

Description

Before df2tree check

Usage

```
before_tree(f_tax)
```

Arguments

`f_tax`	table

Value

table

Examples

```
wrong_taxdf <- data.frame(
  kingdom = c(rep(c("A", "B"), each = 4), "C", NA),
  "phylum" = c("A", "a", "b", "c", "c", "c", "d", NA, NA, "e")
)
before_tree(wrong_taxdf)
```

b_analyse *Beta_diversity Ordination: dimensionality reduction*

Description

Species abundance data can be preprocessed with Hellinger transformation or chord transformation data before PCA analysis. Because the Hellinger distance or chord distance with-without data is equal to $\sqrt{2}\sqrt{1 - Ochiai\ similarity}$, therefore, the sorting diagram (type 1 scale) of PCA analysis after Hellinger transformation or chord transformation with-without data is internal sample The distance between the squares is the Ochiai distance. $\sqrt{2}\sqrt{1 - Ochiai\ similarity}$ is a distance measure, which is also suitable for the analysis of species data. The processed data is then used for pca without norm.

Usage

```
b_analyse(otutab, ...)

## S3 method for class 'data.frame'
b_analyse(
  otutab,
  norm = TRUE,
  method = c("pca"),
  group = NULL,
  dist = "bray",
  ndim = 2,
  scale = FALSE,
  ...
)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`...`	add
`norm`	should normalized or not? (hellinger)
`method`	one of "pca","pcoa","ca","dca","nmds","plsda","tsne","umap","lda","all"
`group`	if needed, give a group vector
`dist`	if use pcoa or nmds, your can choose a dist method (default: bray) or input a distance matrix.
`ndim`	how many dimension be kept? (default:2). 3 for b_res_3d()
`scale`	scale, default: FALSE

Value

b_res object

References

https://www.jianshu.com/p/9694c0b6302d https://zhuanlan.zhihu.com/p/25501130

Examples

```
data(otutab, package = "pcutils")
b_analyse(otutab, method = "pca") -> b_res
plot(b_res, "Group", metadata)
```

b_NTI1 *Calculate beta_NTI*

Description

Calculate beta_NTI

Usage

```
b_NTI1(
  phylo,
  otutab,
  beta.reps = 9,
  weighted = TRUE,
  threads = 1,
  verbose = TRUE
)
```

Arguments

phylo	a phylo object
otutab	otutab
beta.reps	how many simulation performed?
weighted	logical
threads	use how many threads to calculate (default:4)
verbose	verbose

Value

a dist: b_NTI

b_res_3d *3D plot for b_res*

Description

3D plot for b_res

Usage

```
b_res_3d(b_res, Group, metadata = NULL, ...)
```

Arguments

`b_res`	a b_res object
`Group`	group vector for color
`metadata`	metadata contain Group
`...`	add

Value

plotly list

Examples

```
if (requireNamespace("plotly")) {
  data(otutab, package = "pcutils")
  b_analyse(otutab, method = "pca", ndim = 3) -> b_res
  b_res_3d(b_res, "Group", metadata)
}
```

`check_taxonkit` *Check taxonkit*

Description

Check taxonkit

Usage

```
check_taxonkit(print = TRUE)
```

Arguments

`print`	print

Value

taxonkit path

See Also

Other Rtaxonkit: `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

`convert_taxon_name` *Convert taxon names between Chinese and Latin*

Description

Convert taxon names between Chinese and Latin

Usage

```
convert_taxon_name(input_names, mode = "latin_to_chinese", fuzzy = FALSE)
```

Arguments

`input_names`	input names
`mode`	conversion mode, "latin_to_chinese" or "chinese_to_latin"
`fuzzy`	whether to use fuzzy matching, default is FALSE

Value

character vector of converted names

Examples

```
convert_taxon_name(c("Escherichia coli", "Clostridioides difficile"))
```

`cor_net` *Correlation network, species-interaction network for omics*

Description

Correlation network, species-interaction network for omics

Usage

```
cor_net()
```

Value

No value

`df2tree` *From a dataframe to construct a phylo*

Description

NOTE: this function will do `before_tree` first.

Usage

```
df2tree(data, edge_df = FALSE)
```

Arguments

`data`	dataframe
`edge_df`	if the data is edge_df ?

Value

phylo object

Examples

```
data(otutab, package = "pcutils")
df2tree(taxonomy) -> tax_tree
print(tax_tree)
# check all nodes matched!
if (requireNamespace("picante")) {
  picante::match.phylo.comm(tax_tree, t(otutab)) -> nn
  nrow(nn$comm) == nrow(t(otutab))
}
```

`df2tree1` *From a dataframe to construct a phylo (save nwk)*

Description

NOTE: this function will transfer all space to `_`

Usage

```
df2tree1(taxa)
```

Arguments

`taxa`	dataframe

Value

phylo object

Examples

```
data(otutab, package = "pcutils")
df2tree(taxonomy) -> tax_tree
print(tax_tree)
```

---

diff_da *Difference analysis*

Description

Difference analysis

Usage

```
diff_da(
  otutab,
  group_df,
  ctrl = NULL,
  method = "deseq2",
  log = TRUE,
  add_mini = NULL,
  ...
)
```

Arguments

otutab	otutab
group_df	a dataframe with rowname same to dist and one group column
ctrl	the control group, one level of groups
method	one of "deseq2","edger","limma","t.test","wilcox.test"
log	do log transfer for limma?
add_mini	add_mini when calculate the logFC. e.g (10+0.1)/(0+0.1), default `0.5*min(abundance)`
...	other parameters

Value

a dataframe

Examples

```
if (requireNamespace("limma")) {
  data(otutab, package = "pcutils")
  diff_da(otutab, metadata["Group"], method = "limma") -> res
  volcano_p(res)
  volcano_p(res, mode = 2)
}
```

`download_taxonkit_dataset` *Download taxonkit dataset*

Description

Download taxonkit dataset

Usage

```
download_taxonkit_dataset(make_sure = FALSE, taxdump_tar_gz = NULL)
```

Arguments

`make_sure`	make sure to do this
`taxdump_tar_gz`	your download taxdump_tar_gz file from https://ftp.ncbi.nih.gov/pub/taxonomy/taxdump.tar.gz

Value

No value

See Also

Other Rtaxonkit: `check_taxonkit()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

`envfitt` *Envfit test for RDA result*

Description

Envfit test for RDA result

Usage

```
envfitt(phy.rda, env, ...)
```

Arguments

`phy.rda`	a rda result
`env`	environmental factors
`...`	add

Value

g_test object

See Also

`envfit`

Examples

```
data(otutab, package = "pcutils")
env <- metadata[, 6:10]
# RDA
myRDA(otutab, env) -> phy.rda
envfitt(phy.rda, env) -> envfit_res
plot(envfit_res)
```

`geo_sim` *Lm for sample similarity and geographical distance*

Description

Lm for sample similarity and geographical distance

Usage

```
geo_sim(otutab, geo, method = "bray", spe_nwk = NULL, ...)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`geo`	a two-columns dataframe, first is latitude, second is longitude
`method`	Dissimilarity index, partial match to "bray", "euclidean"...see `vegdist`;`unifrac`
`spe_nwk`	a phylo tree if use unifrac...
`...`	additional

Value

a ggplot

References

Graco-Roza, C. et al. (2022) Distance decay 2.0 - A global synthesis of taxonomic and functional turnover in ecological communities. Glob Ecol Biogeogr 31, 1399–1421.

Examples

```
if (requireNamespace("geosphere")) {
  library(ggplot2)
  data(otutab, package = "pcutils")
  metadata[, c("lat", "long")] -> geo
  geo_sim(otutab, geo) -> geo_res
}
```

get_all_sp_la_zh_name *get all species Latin and Chinese name from the CCTCC database*

Description

get all species Latin and Chinese name from the CCTCC database

Usage

```
get_all_sp_la_zh_name(
  download_dir = "~/Documents/",
  each_verbose = FALSE,
  max_requests = 50,
  max_id = 30609,
  failure_ids = NULL
)
```

Arguments

download_dir	default
each_verbose	each_verbose
max_requests	default 50
max_id	default 30609, try to make sure on the website
failure_ids	failure_ids

Value

No value

get_diff_type *Get mean and type*

Description

Get mean and type

Usage

```
get_diff_type(otutab, group_df)
```

Arguments

`otutab`	otutab
`group_df`	a dataframe with rowname same to dist and one group column

Value

No value

gp_dis_density *Group inter-intra density*

Description

Group inter-intra density

Usage

```
gp_dis_density(otutab, group)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`group`	group vector

Value

ggplot

Examples

```
data(otutab, package = "pcutils")
gp_dis_density(otutab, metadata["Group"])
```

grap_p_test *Performs graph-based permutation tests*

Description

Performs graph-based permutation tests

Usage

```
grap_p_test(otutab, metadata, group = "Group", nperm = 999, ...)
```

Arguments

otutab	an otutab data.frame, samples are columns, taxs are rows.
metadata	metadata
group	one group name in columns of metadata
nperm	numbers of permutations to perform
...	additional

Value

ggplot

Examples

```
if (requireNamespace("phyloseqGraphTest") && requireNamespace("phyloseq")) {
  data(otutab, package = "pcutils")
  grap_p_test(otutab, metadata, "Group")
}
```

install_taxonkit *Install taxonkit*

Description

Install taxonkit

Usage

```
install_taxonkit(make_sure = FALSE, taxonkit_tar_gz = NULL)
```

Arguments

make_sure	make sure to do this
taxonkit_tar_gz	your download taxonkit_tar_gz file from https://github.com/shenwei356/taxonkit/releases/

Value

No value

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

`kwtest` *KW test*

Description

KW test

Usage

```
kwtest(otutab, group_df, method = "kruskal.test")
```

Arguments

`otutab`	otutab
`group_df`	a dataframe with rowname same to dist and one group column
`method`	"kruskal.test", see `compare_means`

Value

res

Examples

```
data(otutab, package = "pcutils")
kwtest(otutab, metadata["Group"]) -> res
bbtt(res, pvalue = "p.format")
```

`load_N_data` *Load N-cycle data*

Description

Load N-cycle data

Usage

```
load_N_data()
```

Value

list

References

Tu, Q., Lin, L., Cheng, L., Deng, Y. & He, Z. (2019) NCycDB: a curated integrative database for fast and accurate metagenomic profiling of nitrogen cycling genes. Bioinformatics 35, 1040–1048. Kuypers, M. M. M., Marchant, H. K. & Kartal, B. (2018) The microbial nitrogen-cycling network. Nat Rev Microbiol 16, 263–276.

`mat_dist` *Calculate distance for otutab*

Description

Calculate distance for otutab

Usage

```
mat_dist(otutab, method = "bray", spe_nwk = NULL)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`method`	Dissimilarity index, partial match to "bray", "euclidean"...see `vegdist`;`unifrac`
`spe_nwk`	a phylo tree if use unifrac...

Value

dist

Examples

```
data(otutab, package = "pcutils")
mat_dist(otutab)
```

micro_sbatch *Microbiome sbatch*

Description

Microbiome sbatch

Usage

```
micro_sbatch(
  work_dir = "/share/home/jianglab/pengchen/work/asthma/",
  step = "fastp",
  all_sample_num = 40,
  array = 1,
  partition = "cpu",
  cpus_per_task = 1,
  mem_per_cpu = "2G"
)
```

Arguments

`work_dir`	work_dir
`step`	"fastp","rm_human","megahit","prodigal","salmon-quant",...
`all_sample_num`	all sample number
`array`	array number
`partition`	partition
`cpus_per_task`	cpus_per_task
`mem_per_cpu`	mem_per_cpu, "2G"

Value

No value

multi_bar *Difference analysis*

Description

Difference analysis

Usage

```
multi_bar(
  otutab,
  group_df,
  mode = 1,
  text_df = NULL,
  text_x = NULL,
  text_angle = -90,
  errorbar = "bottom"
)
```

Arguments

`otutab`	otutab
`group_df`	a dataframe with rowname same to dist and one group column
`mode`	1~2
`text_df`	text_df
`text_x`	text_x
`text_angle`	text_angle
`errorbar`	top, bottom, none

Value

a data.frame

Examples

```
data(otutab, package = "pcutils")
multi_bar(otutab[1:10, ], metadata["Group"])
```

`myRDA` *RDA*

Description

RDA

Usage

```
myRDA(
  otutab,
  env,
  norm = TRUE,
  scale = FALSE,
  choose_var = FALSE,
  direction = "forward",
  nperm = 499,
  verbose = TRUE,
  method = "rda",
  dist = "bray"
)

myCCA(
  otutab,
  env,
  norm = TRUE,
  scale = FALSE,
  choose_var = FALSE,
  nperm = 499,
  verbose = TRUE
)

myCAP(
  otutab,
  env,
  norm = TRUE,
  scale = FALSE,
  choose_var = FALSE,
  nperm = 499,
  verbose = TRUE,
  dist = "bray"
)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`env`	environmental factors
`norm`	should normalize? (default:TRUE)
`scale`	should scale species? (default:FALSE)
`choose_var`	should choose variables? use forward step
`direction`	The direction of the stepwise selection, "both", "forward" or "backward", default is "forward"
`nperm`	number of permutation
`verbose`	verbose
`method`	"rda", "cca", "cap", "dbrda"
`dist`	The name of the dissimilarity (or distance) index for "cap" or "dbrda", for `vegdist`

Value

rda/cca

See Also

`vegdist`;`unifrac`

Examples

```
data(otutab, package = "pcutils")
env <- metadata[, 6:10]
# RDA
myRDA(otutab, env) -> phy.rda
RDA_plot(phy.rda, "Group", metadata)
```

`name_or_id2df` *Transfer taxon name or taxid to the lineage dataframe*

Description

Transfer taxon name or taxid to the lineage dataframe

Usage

```
name_or_id2df(
  name_or_id,
  mode = "name",
  add_prefix = TRUE,
  fill_miss_rank = TRUE,
  data_dir = NULL
)
```

Arguments

`name_or_id`	name or taxid
`mode`	"id" or "name"
`add_prefix`	add_prefix
`fill_miss_rank`	fill_miss_rank
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

dataframe

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

Examples

```
## Not run:
name_or_id2df(c("Homo sapiens", "Akkermansia muciniphila ATCC BAA-835"))

## End(Not run)
```

`ncm` *Sloan Neutral Model*

Description

Sloan Neutral Model

Plot ncm_res

Usage

```
ncm(otutab, model = "nls")

## S3 method for class 'ncm_res'
plot(
  x,
  mycols = c(Above = "#069870", Below = "#e29e02", In = "#1e353a"),
  text_position = NULL,
  pie_text_params = list(size = 2.5),
  ...
)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`model`	fit method, one of "nls","mle"
`x`	a ncm_res object
`mycols`	mycols
`text_position`	text_position
`pie_text_params`	pie text parameters
`...`	add

Value

ncm_res

ggplot

References

Sloan, W. TRUE. et al. (2006) Quantifying the roles of immigration and chance in shaping prokaryote community structure. Environmental Microbiology 8, 732–740.

Examples

```
if (requireNamespace("Hmisc") && requireNamespace("minpack.lm")) {
  data(otutab, package = "pcutils")
  ncm(otutab) -> ncm_res
  plot(ncm_res)
}
```

`nst` *Calculate NST for each group*

Description

Calculate NST for each group

Usage

```
nst(otutab, group_df, threads = 1, file = NULL, rep = 20, save = FALSE)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`group_df`	a dataframe with rowname and one group column
`threads`	default:4
`file`	filename to save
`rep`	repeat numbers: suggest 999
`save`	save the file

Value

a b_dist object, dis is MSTij

References

Ning, D., Deng, Y., Tiedje, J. M. & Zhou, J. (2019) A general framework for quantitatively assessing ecological stochasticity. Proceedings of the National Academy of Sciences 116, 16892–16898.

Examples

```
if (requireNamespace("NST")) {
  library(ggplot2)
  data(otutab, package = "pcutils")
  nst(otutab, metadata["Group"]) -> nst_res
  plot(nst_res, c_group = "intra") + geom_hline(yintercept = 0.5, lty = 2) + ylab("NST")
}
```

---

nti_rc *Calculate b_NTI and RC_bray for each group*

Description

Calculate b_NTI and RC_bray for each group

Plot NTI_RC object

Usage

```
nti_rc(
  otutab,
  phylo,
  group_df,
  threads = 1,
  file = NULL,
  rep = 20,
  save = FALSE
)

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

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`phylo`	a phylo object
`group_df`	a dataframe with rowname and one group column
`threads`	default:4
`file`	filename to save
`rep`	repeat numbers: suggest 999
`save`	save the file
`x`	NTI_RC object
`...`	pass to `stackplot`

Value

a b_dist object, dis is MSTij

ggplot

References

Ning, D., Deng, Y., Tiedje, J. M. & Zhou, J. (2019) A general framework for quantitatively assessing ecological stochasticity. Proceedings of the National Academy of Sciences 116, 16892–16898.

Examples

```
if (requireNamespace("NST") && requireNamespace("pctax")) {
  data(otutab, package = "pcutils")
  pctax::df2tree(taxonomy) -> phylo
  nti_rc(otutab, phylo, metadata["Group"]) -> nti_res
  plot(nti_res)
}
```

---

pc_otu *Create a pc_otu class object*

Description

Create a pc_otu class object

Usage

```
pc_otu(otutab = data.frame(), metadata = data.frame(), taxonomy = NULL, ...)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`metadata`	a metadata data.frame, samples are rows
`taxonomy`	a taxomomy data.frame, look out the rowname of taxonomy and otutab should matched!
`...`	add

Value

pc_otu

Examples

```
data(otutab, package = "pcutils")
pc_tax1 <- pc_otu(otutab, metadata)
print(pc_tax1)
```

`pc_tax1` *test data (pc_otu class) for pc_tax package.*

Description

an otutab, metadata and a taxonomy table.

Format

a pc_otu contains an otutab, metadata and a taxonomy table.

tbls contians otutable rawdata

metas contians metadata

otus contians taxomomy table

`pc_valid` *Judge pc_otu is valid or not*

Description

Judge pc_otu is valid or not

Usage

```
pc_valid(pc)
```

Arguments

`pc`	a pc_otu object

Value

logical

`permanova` *Permanova between a otutab and a variable*

Description

Permanova between a otutab and a variable

Usage

```
permanova(
  otutab,
  envs,
  norm = TRUE,
  each = TRUE,
  method = "adonis",
  dist = "bray",
  nperm = 999,
  ...
)
```

Arguments

`otutab`	an otutab data.frame, samples are columns, taxs are rows.
`envs`	factors need to test
`norm`	should normalize?(default:TRUE)
`each`	test factor one by one, rather than whole
`method`	adonis/mrpp/anosim/mantel
`dist`	if use pcoa or nmds, your can choose a dist method (default: bray)
`nperm`	numbers of permutations to perform
`...`	additional

Value

a g_test object with these columns

`group`	the test group or factor
`r`	relationship
`r2`	model R-square
`p_value`	model test p_value
`sig`	whether significant

References

https://blog.csdn.net/qq_42458954/article/details/110390488

Examples

```
data(otutab, package = "pcutils")
permanova(otutab, metadata[, c(2:10)]) -> adonis_res
print(adonis_res)
plot(adonis_res)
```

`plot.a_res` *Plot a_res object*

Description

Plot a_res object

Usage

```
## S3 method for class 'a_res'
plot(x, group, metadata, ...)
```

Arguments

`x`	a a_res object
`group`	one of colname of metadata
`metadata`	metadata
`...`	addditional parameters for `group_box` or `my_lm`

Value

patchwork object,you can change theme with &

See Also

`a_diversity`

`plot.b_res` *Plot a b_res*

Description

Plot a b_res

Usage

```
## S3 method for class 'b_res'
plot(
  x,
  Group,
  metadata = NULL,
  Group2 = NULL,
  mode = 1,
  bi = FALSE,
  Topn = 10,
  rate = 1,
  margin = FALSE,
  margin_label = TRUE,
  permanova_res = NULL,
  text_param = list(),
  box_margin = TRUE,
  box_param = list(),
  pal = NULL,
  sample_label = TRUE,
  stat_ellipse = TRUE,
  coord_fix = FALSE,
  bi_text_size = 3,
  ...
)
```

Arguments

`x`	a b_res object
`Group`	group vector for color
`metadata`	metadata contain Group
`Group2`	mapping point shape
`mode`	plot mode:1~3
`bi`	plot variables segments?
`Topn`	how many variables to show?
`rate`	segments length rate
`margin`	plot the margin boxplot?
`margin_label`	margin_label, TRUE
`permanova_res`	permanova result
`text_param`	text_param for `annotate`
`box_margin`	margin plot box or density?
`box_param`	box_param for `group_box`
`pal`	colors for group
`sample_label`	plot the labels of samples?

`stat_ellipse`	plot the stat_ellipse?
`coord_fix`	fix the coordinates y/x ratio
`bi_text_size`	biplot text size
`...`	add

Value

a ggplot

See Also

`b_analyse`

`plot.g_test` *Plot g_test*

Description

Plot g_test

Usage

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

Arguments

`x`	a g_test object
`...`	add

Value

ggplot

See Also

`permanova`

`plot.pro_res` *Plot pro_res*

Description

Plot pro_res

Usage

```
## S3 method for class 'pro_res'
plot(x, group, metadata = NULL, pal = NULL, ...)
```

Arguments

`x`	pro_res
`group`	group
`metadata`	metadata
`pal`	pal
`...`	add

Value

a ggplot

`plot.time_cm` *Plot time_cm*

Description

Plot time_cm

Usage

```
## S3 method for class 'time_cm'
plot(x, mem_thr = 0.6, ...)
```

Arguments

`x`	time_cm
`mem_thr`	membership threshold
`...`	add

Value

ggplot

`plot_element_cycle` *Plot element cycle*

Description

Plot element cycle

Usage

```
plot_element_cycle(
  cycle = "Nitrogen cycle",
  anno_df = NULL,
  only_anno = FALSE,
  cell_fill = NA,
  cell_color = "orange",
  use_chinese = FALSE,
  chemical_size = 7,
  chemical_bold = TRUE,
  chemical_color = "black",
  chemical_label = TRUE,
  reaction_width = 1,
  reaction_arrow_size = 4,
  reaction_arrow_closed = TRUE,
  gene_or_ko = "gene",
  gene_size = 3,
  gene_x_offset = 0.3,
  gene_y_offset = 0.15,
  gene_label = TRUE,
  gene_color = NULL,
  gene_bold = TRUE,
  gene_italic = TRUE,
  gene_label_fill = "white"
)
```

Arguments

`cycle`	one of c("Carbon cycle","Nitrogen cycle","Phosphorus cycle","Sulfur cycle","Iron cycle")
`anno_df`	anno_df, columns should contains Gene or KO and Group
`only_anno`	only show genes in anno_df?
`cell_fill`	cell fill color
`cell_color`	cell border color
`use_chinese`	use chinese label?
`chemical_size`	chemical text size
`chemical_bold`	chemical text bold

`chemical_color`	chemical text color
`chemical_label`	chemical text in geom_label or geom_text?
`reaction_width`	reaction line width
`reaction_arrow_size`	reaction arrow size
`reaction_arrow_closed`	reaction arrow closed?
`gene_or_ko`	"gene" or "ko"
`gene_size`	gene text size
`gene_x_offset`	gene_x_offset
`gene_y_offset`	gene_y_offset
`gene_label`	gene text in geom_label or geom_text?
`gene_color`	gene text color
`gene_bold`	gene text blod?
`gene_italic`	gene text italic?
`gene_label_fill`	gene label fill color

Value

ggplot

Examples

```
if (requireNamespace("ggforce")) plot_element_cycle()
```

`plot_N_cycle` *Plot the N-cycling pathway and genes*

Description

Plot the N-cycling pathway and genes

Usage

```
plot_N_cycle(
  my_N_genes = NULL,
  just_diff = FALSE,
  path_col = NULL,
  type_col = c(up = "red", down = "blue", none = NA),
  fill_alpha = 0.5,
  arrow_size = 0.1,
  line_width = 1,
  title = "Nitrogen cycling",
  legend.position = c(0.85, 0.15)
)
```

Arguments

`my_N_genes`	dataframe, "Gene_families","type" should in colnames of my_N_genes
`just_diff`	logical, just plot the different genes?
`path_col`	colors of pathways
`type_col`	colors of types
`fill_alpha`	alpha, default 0.5
`arrow_size`	arrow_size, default 0.1
`line_width`	line_width, default 1
`title`	title, default "Nitrogen cycling"
`legend.position`	default c(0.85,0.15)

Value

ggplot

Examples

```
N_data <- load_N_data()
my_N_genes <- data.frame(
  `Gene_families` = sample(N_data$N_genes$Gene_families, 10, replace = FALSE),
  change = rnorm(10), check.names = FALSE
)
my_N_genes <- dplyr::mutate(my_N_genes,
  type = ifelse(change > 0, "up", ifelse(change < 0, "down", "none"))
)
plot_N_cycle(my_N_genes, just_diff = FALSE, fill_alpha = 0.2)
# ggsave(filename = "test.pdf", width = 14, height = 10)
```

`plot_two_tree` *Plot two trees in one plot*

Description

Plot two trees in one plot

Usage

```
plot_two_tree(
  tree1,
  tree2,
  edge_df = NULL,
  tree2_x = 10,
  filter_link = FALSE,
  tree1_param = list(),
  tree2_param = list(),
  line_param = list(),
  tree1_tip = FALSE,
  tip1_param = list(),
  tree2_tip = FALSE,
  tip2_param = list(),
  tree1_highlight = NULL,
  highlight1_param = list(),
  highlight1_scale = NULL,
  tree2_highlight = NULL,
  highlight2_param = list(),
  highlight2_scale = ggplot2::scale_fill_hue(na.value = NA)
)
```

Arguments

`tree1`	phylo object
`tree2`	phylo object
`edge_df`	dataframe with edge information, containing "from" and "to" columns
`tree2_x`	x position of tree2
`filter_link`	filter the link between tree1 and tree2
`tree1_param`	parameters for `geom_tree`
`tree2_param`	parameters for `geom_tree`
`line_param`	parameters for `geom_line`
`tree1_tip`	tree tip label
`tip1_param`	parameters for `geom_tiplab`
`tree2_tip`	tree tip label
`tip2_param`	parameters for `geom_tiplab`
`tree1_highlight`	tree1 highlight data.frame
`highlight1_param`	parameters for `geom_hilight`
`highlight1_scale`	scale_fill_ for highlight1
`tree2_highlight`	tree2 highlight data.frame
`highlight2_param`	parameters for `geom_hilight`
`highlight2_scale`	scale_fill_ for highlight2

Value

ggplot object

Examples

```
if (requireNamespace("ggtree")) {
  data(otutab, package = "pcutils")
  df2tree(taxonomy[1:50, ]) -> tax_tree
  df2tree(taxonomy[51:100, ]) -> tax_tree2
 link <- data.frame(from = sample(tax_tree$tip.label, 20), to = sample(tax_tree2$tip.label, 20))
  plot_two_tree(tax_tree, tax_tree2, link)
}
```

pre_fastp *Prepare the result from fastp (.json file)*

Description

Prepare the result from fastp (.json file)

Usage

```
pre_fastp(jsonfiles, prefix = c("Raw", "Clean"))
```

Arguments

jsonfiles	the directory contains .json file
prefix	default c("Raw","Clean"), for the before filtering and after filtering.

Value

data.frame

pre_tax_table *Complete a taxonomy table*

Description

Complete a taxonomy table

Usage

```
pre_tax_table(
  tax_table,
  tax_levels = c("k", "p", "c", "o", "f", "g", "s", "st"),
 na_tax = "Unclassified|uncultured|Ambiguous|Unknown|unknown|metagenome|Unassig",
  ignore.case = TRUE,
  na_repalce = "Unknown"
)
```

Arguments

`tax_table`	taxonomy table
`tax_levels`	a vector whose length longer than `ncol(taxdf)`, use to be prefix. Default: c("k", "p", "c", "o", "f", "g","s", "st")
`na_tax`	grepl some words and turn to `na_repalce`, default: "Unclassified|uncultured|Ambiguous|Unknown|unkno
`ignore.case`	ignore.case for `na_tax`
`na_repalce`	defalut: Unknown

Value

a good taxonomy table

References

`MicrobiotaProcess`

Examples

```
taxmat <- matrix(sample("onelevel", 7 * 2, replace = TRUE), nrow = 2, ncol = 7) %>% as.data.frame()
colnames(taxmat) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species")
pre_tax_table(taxmat)
```

`print.pc_otu` *Print*

Description

Print

Usage

```
## S3 method for class 'pc_otu'
print(x, ...)
```

Arguments

`x`	pc_otu
`...`	add

Value

No value

`procrustes_analyse` *Procrustes Rotation of Two Configurations and PROTEST*

Description

Procrustes Rotation of Two Configurations and PROTEST

Usage

```
procrustes_analyse(b_res1, b_res2, nperm = 999, ...)
```

Arguments

`b_res1`	Target matrix
`b_res2`	Matrix to be rotated
`nperm`	numbers of permutations to perform
`...`	additional

Value

pro_res

Examples

```
data(otutab, package = "pcutils")
b_analyse(otutab, method = "pca") -> b_res1
b_analyse(otutab * abs(rnorm(10)), method = "pca") -> b_res2
pro_res <- procrustes_analyse(b_res1, b_res2)
plot(pro_res, "Group", metadata)
```

`rarefaction` *Rarefy a otutab*

Description

Rarefy a otutab

Usage

```
rarefaction(otutab, sample = NULL)
```

Arguments

`otutab`	otutab
`sample`	number

Value

a rarefied otutab

Examples

```
data(otutab, package = "pcutils")
rarefaction(otutab)
```

`rare_curve_sample` *Rare the sample*

Description

Rare the sample

Plot a rare curve

Usage

```
rare_curve_sample(otutab, rep = 30, count_cutoff = 1)

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

Arguments

`otutab`	otutab
`rep`	repeats number
`count_cutoff`	cutoff to be 0
`x`	AOR object
`...`	add

Value

ggplot

ggplot

Examples

```
data(otutab, package = "pcutils")
a <- rare_curve_sample(otutab)
plot(a)
```

`rare_curve_species` *Rare the species*

Description

Rare the species

Usage

```
rare_curve_species(
  otutab,
  step = 2000,
  method = "richness",
  mode = 2,
  reps = 3,
  threads = 1,
  verbose = TRUE
)
```

Arguments

`otutab`	otutab
`step`	default 2000
`method`	one of "richness","chao1","ace","gc","shannon","simpson","pd","pielou"
`mode`	1 for little table, 2 for big
`reps`	reps
`threads`	use how many threads to calculate (default:1)
`verbose`	verbose

Value

ggplot

Examples

```
data(otutab, package = "pcutils")
a <- rare_curve_species(otutab, mode = 1)
plot(a)
```

RCbray1 *Calculate RCbray-curtis*

Description

Calculate RCbray-curtis

Usage

```
RCbray1(
  otutab,
  reps = 9,
  threads = 1,
  classic_metric = TRUE,
  split_ties = TRUE
)
```

Arguments

`otutab`	otutab
`reps`	how many simulation performed?
`threads`	use how many threads to calculate (default:4)
`classic_metric`	standardizes the metric to range from -1 to 1
`split_ties`	adds half of the number of null observations that are equal to the observed number of shared species to the calculation- this is highly recommended

Details

Parallelized version of the Raup-Crick algorithm for "abundance" data (Stegen et al. 2013).

Value

a dist

Examples

```
if (requireNamespace("picante")) {
  data(otutab, package = "pcutils")
  df2tree(taxonomy) -> phylo
  b_NTI1(phylo, otutab) -> bnti_res
  RCbray1(otutab, reps = 9) -> rc_res

  data.frame(
    type = factor(c("Homo_S", "Heter_S", "Homo_D", "D_limit", "Undominated"),
      levels = c("Homo_S", "Heter_S", "Homo_D", "D_limit", "Undominated")
    ),
    number = c(
      sum(bnti_res < (-2)), sum(bnti_res > 2),
      sum((abs(bnti_res) < 2) & (abs(rc_res) < 0.95)),
      sum((abs(bnti_res) < 2) & (rc_res < (-0.95))),
      sum((abs(bnti_res) < 2) & (rc_res > 0.95))
    )
  ) -> com_pro
  pcutils::gghuan(com_pro, reorder = FALSE)
}
```

`RDA_plot` *Plot RDA res*

Description

Plot RDA res

Usage

```
RDA_plot(
  phy.rda,
  Group,
  metadata = NULL,
  Group2 = NULL,
  env_rate = 1,
  mode = 1,
  tri = FALSE,
  Topn = 10,
  rate = 1,
  margin = FALSE,
  box = TRUE,
  pal = NULL,
  sample_label = TRUE,
  stat_ellipse = TRUE,
  coord_fix = FALSE,
  bi_text_size = 3,
  env_text_param = NULL,
  ...
)
```

Arguments

`phy.rda`	rda/cca object
`Group`	group vector for color
`metadata`	metadata contain Group
`Group2`	mapping point shape
`env_rate`	default 1

`mode`	plot mode:1~3
`tri`	plot variables segments?
`Topn`	how many variables to show?
`rate`	segments length rate
`margin`	plot the margin boxplot?
`box`	margin plot box or density?
`pal`	colors for group
`sample_label`	plot the labels of samples?
`stat_ellipse`	plot the stat_ellipse?
`coord_fix`	fix the coordinates y/x ratio
`bi_text_size`	biplot text size
`env_text_param`	parameters pass to `geom_text`
`...`	add

Value

ggplot

See Also

`myRDA`

`suijisenlin` *RandomForest*

Description

RandomForest

Usage

```
suijisenlin(otutab, group_df, topN = 10)
```

Arguments

`otutab`	otutab
`group_df`	a dataframe with rowname same to dist and one group column
`topN`	default: 10

Value

diff

Examples

```
if (requireNamespace("randomForest")) {
  data(otutab, package = "pcutils")
  suijisenlin(otutab, metadata["Group"]) -> rf_res
}
```

---

summary.pc_otu *Summary pc_otu*

Description

Summary pc_otu

Usage

```
## S3 method for class 'pc_otu'
summary(object, ...)
```

Arguments

object	pc_otu
...	add

Value

No value

Examples

```
data("pc_tax1")
summary(pc_tax1)
```

---

taxonkit_filter *Filter TaxIDs based on Taxonomic Ranks*

Description

This function uses the "taxonkit filter" command to filter TaxIDs based on taxonomic ranks.

Usage

```
taxonkit_filter(
  file_path,
  black_list = NULL,
  discard_noranks = FALSE,
  discard_root = FALSE,
  equal_to = NULL,
  higher_than = NULL,
  lower_than = NULL,
  rank_file = NULL,
  root_taxid = NULL,
  save_predictable_norank = FALSE,
  taxid_field = NULL,
  text = FALSE,
  data_dir = NULL
)
```

Arguments

`file_path`	The path to the input file containing TaxIDs. Or file text (text=TRUE)
`black_list`	A character vector specifying the ranks to discard.
`discard_noranks`	Logical value indicating whether to discard all ranks without order (default is FALSE).
`discard_root`	Logical value indicating whether to discard the root taxid (default is FALSE).
`equal_to`	A character vector specifying the ranks for which TaxIDs should be equal to.
`higher_than`	The rank above which the TaxIDs should be (exclusive).
`lower_than`	The rank below which the TaxIDs should be (exclusive).
`rank_file`	The path to a user-defined ordered taxonomic ranks file.
`root_taxid`	The root taxid (default is 1).
`save_predictable_norank`	Logical value indicating whether to save some special ranks without order when using lower_than (default is FALSE).
`taxid_field`	The field index of the taxid in the input file (default is 1).
`text`	logical
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

A character vector containing the output of the "taxonkit filter" command.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

Examples

```
## Not run:
taxids2 <- system.file("extdata/taxids2.txt", package = "pctax")
taxonkit_filter(taxids2, lower_than = "genus")

## End(Not run)
```

---

taxonkit_lca *Compute Lowest Common Ancestor (LCA) of TaxIDs*

Description

This function uses the "taxonkit lca" command to compute the Lowest Common Ancestor (LCA) of TaxIDs.

Usage

```
taxonkit_lca(
  file_path,
  buffer_size = "1M",
  separator = " ",
  skip_deleted = FALSE,
  skip_unfound = FALSE,
  taxids_field = NULL,
  text = FALSE,
  data_dir = NULL
)
```

Arguments

`file_path`	The path to the input file containing TaxIDs. Or file text (text=TRUE)
`buffer_size`	The size of the line buffer (supported units: K, M, G).
`separator`	The separator for TaxIDs.
`skip_deleted`	Whether to skip deleted TaxIDs and compute with the remaining ones.
`skip_unfound`	Whether to skip unfound TaxIDs and compute with the remaining ones.
`taxids_field`	The field index of TaxIDs. Input data should be tab-separated (default 1).
`text`	logical
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

A character vector containing the computed LCAs.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

Examples

```
## Not run:
taxonkit_lca("239934, 239935, 349741", text = TRUE, separator = ", ")

## End(Not run)
```

`taxonkit_lineage` *Retrieve Taxonomic Lineage using taxonkit*

Description

Retrieve Taxonomic Lineage using taxonkit

Usage

```
taxonkit_lineage(
  file_path,
  delimiter = ";",
  no_lineage = FALSE,
  show_lineage_ranks = FALSE,
  show_lineage_taxids = FALSE,
  show_name = FALSE,
  show_rank = FALSE,
  show_status_code = FALSE,
  taxid_field = 1,
  text = FALSE,
  data_dir = NULL
)
```

Arguments

`file_path`	The path to the input file with taxonomic IDs. Or file text (text=TRUE)
`delimiter`	The field delimiter in the lineage (default ";").
`no_lineage`	Logical, indicating whether to exclude lineage information (default: FALSE).
`show_lineage_ranks`	Logical, indicating whether to append ranks of all levels in the lineage (default: FALSE).
`show_lineage_taxids`	Logical, indicating whether to append lineage consisting of taxids (default: FALSE).
`show_name`	Logical, indicating whether to append scientific name (default: FALSE).

`show_rank`	Logical, indicating whether to append rank of taxids (default: FALSE).
`show_status_code`	Logical, indicating whether to show status code before lineage (default: FALSE).
`taxid_field`	The field index of taxid. Input data should be tab-separated (default: 1).
`text`	logical,
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

A character vector containing the taxonomic lineage information.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_list()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

Examples

```
## Not run:
taxonkit_lineage("9606\n63221", show_name = TRUE, show_rank = TRUE, text = TRUE)

## End(Not run)
```

`taxonkit_list` *Taxonkit list*

Description

This function uses Taxonkit to perform the "list" operation, which retrieves information about taxa based on their TaxIDs.

Usage

```
taxonkit_list(
  ids,
  indent = "  ",
  json = FALSE,
  show_name = FALSE,
  show_rank = FALSE,
  data_dir = NULL
)
```

Arguments

`ids`	A character vector of TaxIDs to retrieve information for.
`indent`	The indentation string to use for pretty-printing the output. Default is " ".
`json`	Logical value indicating whether to output the result in JSON format. Default is FALSE.
`show_name`	Logical value indicating whether to show the scientific names of taxa. Default is FALSE.
`show_rank`	Logical value indicating whether to show the ranks of taxa. Default is FALSE.
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

The output of the Taxonkit list operation.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_name2taxid()`, `taxonkit_reformat()`

Examples

```
## Not run:
taxonkit_list(ids = c(9605), indent = "-", show_name = TRUE, show_rank = TRUE)

## End(Not run)
```

`taxonkit_name2taxid` *Convert Taxonomic Names to TaxIDs*

Description

This function uses the "taxonkit taxonkit_name2taxid" command to convert taxonomic names to corresponding taxonomic IDs (TaxIDs).

Usage

```
taxonkit_name2taxid(
  file_path,
  name_field = NULL,
  sci_name = FALSE,
  show_rank = FALSE,
  text = FALSE,
  data_dir = NULL
)
```

Arguments

`file_path`	The path to the input file containing taxonomic names. Or file text (text=TRUE)
`name_field`	The field index of the taxonomic name in the input file (default is 1).
`sci_name`	Logical value indicating whether to search only for scientific names (default is FALSE).
`show_rank`	Logical value indicating whether to show the taxonomic rank in the output (default is FALSE).
`text`	Logical
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

A character vector containing the output of the "taxonkit_name2taxid" command.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_reformat()`

Examples

```
## Not run:
names <- system.file("extdata/name.txt", package = "pctax")
taxonkit_name2taxid(names, name_field = 1, sci_name = FALSE, show_rank = FALSE)
"Homo sapiens" %>% taxonkit_name2taxid(text = TRUE)

## End(Not run)
```

`taxonkit_reformat` *Reformat Taxonomic Lineage using taxonkit*

Description

Reformat Taxonomic Lineage using taxonkit

Usage

```
taxonkit_reformat(
  file_path,
  delimiter = NULL,
  add_prefix = FALSE,
  prefix_kingdom = "K__",
  prefix_phylum = "p__",
  prefix_class = "c__",
  prefix_order = "o__",
  prefix_family = "f__",
  prefix_genus = "g__",
  prefix_species = "s__",
  prefix_subspecies = "t__",
  prefix_strain = "T__",
  fill_miss_rank = FALSE,
  format_string = "",
  miss_rank_repl_prefix = "unclassified ",
  miss_rank_repl = "",
  miss_taxid_repl = "",
  output_ambiguous_result = FALSE,
  lineage_field = 2,
  taxid_field = NULL,
  pseudo_strain = FALSE,
  trim = FALSE,
  text = FALSE,
  data_dir = NULL
)
```

Arguments

`file_path`	The path to the input file with taxonomic lineages. Or file text (text=TRUE)
`delimiter`	The field delimiter in the input lineage (default ";").
`add_prefix`	Logical, indicating whether to add prefixes for all ranks (default: FALSE).
`prefix_kingdom`	The prefix for kingdom, used along with –add-prefix (default: "K__").
`prefix_phylum`	The prefix for phylum, used along with –add-prefix (default: "p__").
`prefix_class`	The prefix for class, used along with –add-prefix (default: "c__").
`prefix_order`	The prefix for order, used along with –add-prefix (default: "o__").
`prefix_family`	The prefix for family, used along with –add-prefix (default: "f__").
`prefix_genus`	The prefix for genus, used along with –add-prefix (default: "g__").
`prefix_species`	The prefix for species, used along with –add-prefix (default: "s__").
`prefix_subspecies`	The prefix for subspecies, used along with –add-prefix (default: "t__").
`prefix_strain`	The prefix for strain, used along with –add-prefix (default: "T__").
`fill_miss_rank`	Logical, indicating whether to fill missing rank with lineage information of the next higher rank (default: FALSE).
`format_string`	The output format string with placeholders for each rank.
`miss_rank_repl_prefix`	The prefix for estimated taxon level for missing rank (default: "unclassified ").
`miss_rank_repl`	The replacement string for missing rank.
`miss_taxid_repl`	The replacement string for missing taxid.
`output_ambiguous_result`	Logical, indicating whether to output one of the ambiguous result (default: FALSE).

`lineage_field`	The field index of lineage. Input data should be tab-separated (default: 2).
`taxid_field`	The field index of taxid. Input data should be tab-separated. It overrides -i/–lineage-field.
`pseudo_strain`	Logical, indicating whether to use the node with lowest rank as strain name (default: FALSE).
`trim`	Logical, indicating whether to not fill missing rank lower than current rank (default: FALSE).
`text`	logical
`data_dir`	directory containing nodes.dmp and names.dmp (default "/Users/asa/.taxonkit")

Value

A character vector containing the reformatted taxonomic lineages.

See Also

Other Rtaxonkit: `check_taxonkit()`, `download_taxonkit_dataset()`, `install_taxonkit()`, `name_or_id2df()`, `taxonkit_filter()`, `taxonkit_lca()`, `taxonkit_lineage()`, `taxonkit_list()`, `taxonkit_name2taxid()`

Examples

```
## Not run:
# Use taxid
taxids2 <- system.file("extdata/taxids2.txt", package = "pctax")
reformatted_lineages <- taxonkit_reformat(taxids2,
  add_prefix = TRUE, taxid_field = 1, fill_miss_rank = TRUE
)
reformatted_lineages
taxonomy <- strsplit2(reformatted_lineages, "\t")
taxonomy <- strsplit2(taxonomy$V2, ";")

# Use lineage result
taxonkit_lineage("9606\n63221", show_name = TRUE, show_rank = TRUE, text = TRUE) %>%
  taxonkit_reformat(text = TRUE)

## End(Not run)
```

---

tax_lca *Calculate the lowest common ancestor (LCA) of a set of taxa*

Description

Calculate the lowest common ancestor (LCA) of a set of taxa

Usage

```
tax_lca(df)
```

Arguments

`df`	a data frame with taxonomic information, with columns representing taxonomic levels

Value

character

Examples

```
df <- data.frame(
  A = c("a", "a", "a", "a"),
  B = c("x", "x", "y", "y"),
  C = c("1", "1", "2", "3"),
  stringsAsFactors = FALSE
)
tax_lca(df)
```

`time_by_cm` *Time series analysis*

Description

Time series analysis

Usage

```
time_by_cm(otu_time, n_cluster = 6, min.std = 0)
```

Arguments

`otu_time`	otutab hebing by a time variable
`n_cluster`	number of clusters
`min.std`	min.std

Value

time_cm

Examples

```
if (interactive()) {
  data(otutab, package = "pcutils")
  otu_time <- pcutils::hebing(otutab, metadata$Group)
  time_by_cm(otu_time, n_cluster = 4) -> time_cm_res
  plot(time_cm_res)
}
```

volcano_p *Volcano plot for difference analysis*

Description

Volcano plot for difference analysis

Usage

```
volcano_p(
  res,
  logfc = 1,
  adjp = 0.05,
  text = TRUE,
  repel = TRUE,
  mode = 1,
  number = FALSE
)
```

Arguments

`res`	result of `diff_da` which have colnames: tax, log2FoldChange, padj, compare, sig
`logfc`	log_fold_change threshold
`adjp`	adjust_p_value threshold
`text`	text, TRUE
`repel`	repel, TRUE
`mode`	1:normal; 2:multi_contrast
`number`	show the tax number

Value

ggplot

See Also

`diff_da`

z_diversity *Calculate Zeta Diversity*

Description

This function calculates Zeta diversity for each group in the provided otutab.

This function plots the Zeta diversity results obtained from the z_diversity function.

Usage

```
z_diversity(otutab, group_df = NULL, zetadiv_params = list())

## S3 method for class 'zeta_res'
plot(x, lm_model = c("exp", "pl")[1], ribbon = FALSE, text = TRUE, ...)
```

Arguments

`otutab`	A matrix or data frame containing OTU (Operational Taxonomic Unit) counts.
`group_df`	A data frame containing group information.
`zetadiv_params`	Additional parameters to be passed to the Zeta.decline.mc function from the zetadiv package.
`x`	Zeta diversity results obtained from z_diversity function.
`lm_model`	The linear model to be used for fitting ('exp' or 'pl').
`ribbon`	Logical, whether to add a ribbon to the plot for standard deviation.
`text`	Logical, whether to add R-squared and p-value text annotations.
`...`	Additional arguments to be passed to ggplot2 functions.

Value

zeta_res

A ggplot object.

Examples

```
if (requireNamespace("zetadiv")) {
  data(otutab, package = "pcutils")
  zeta_result <- z_diversity(otutab, metadata["Group"], zetadiv_params = list(sam = 10))
  plot(zeta_result, lm_model = "exp", text = TRUE)
}
```

z_diversity_decay *Calculate Zeta Diversity with Distance*

Description

This function calculates Zeta diversity for each group in the provided otutab.

Usage

```
z_diversity_decay(otutab, xy_df, group_df = NULL, zetadiv_params = list())

## S3 method for class 'zeta_decay'
plot(x, ribbon = TRUE, ...)
```

Arguments

`otutab`	A matrix or data frame containing OTU (Operational Taxonomic Unit) counts.
`xy_df`	Site coordinates.
`group_df`	A data frame containing group information.
`zetadiv_params`	Additional parameters to be passed to the Zeta.ddecay function from the zetadiv package.
`x`	Zeta diversity results obtained from z_diversity_decay function.
`ribbon`	Logical, whether to add a ribbon to the plot for standard deviation.
`...`	Additional arguments to be passed to ggplot2 functions.

Value

zeta_decay

A ggplot object.

Examples

```
if (requireNamespace("zetadiv")) {
  data(otutab, package = "pcutils")
  zeta_decay_result <- z_diversity_decay(otutab, metadata[, c("lat", "long")],
    metadata["Group"],
    zetadiv_params = list(sam = 10)
  )
  plot(zeta_decay_result)
}
```

Index