Package 'microbiomeMarker'

March 7, 2025

Title microbiome biomarker analysis toolkit

Version 1.12.2

Description To date, a number of methods have been developed for microbiome marker discovery based on metagenomic profiles, e.g. LEfSe. However, all of these methods have its own advantages and disadvantages, and none of them is considered standard or universal. Moreover, different programs or softwares may be development using different programming languages, even in different operating systems. Here, we have developed an all-in-one R package microbiomeMarker that integrates commonly used differential analysis methods as well as three machine learning-based approaches, including Logistic regression, Random forest, and Support vector machine, to facilitate the identification of microbiome markers.

License GPL-3

biocViews Metagenomics, Microbiome, DifferentialExpression

URL https://github.com/yiluheihei/microbiomeMarker

BugReports https://github.com/yiluheihei/microbiomeMarker/issues

Depends R (>= 4.1.0)

Imports dplyr, phyloseq, magrittr, purrr, MASS, utils, ggplot2, tibble, rlang, stats, coin, ggtree, tidytree, methods, IRanges, tidyr, patchwork, ggsignif, metagenomeSeq, DESeq2, edgeR, BiocGenerics, Biostrings, yaml, biomformat, S4Vectors, Biobase, ComplexHeatmap, ANCOMBC, caret, limma, ALDEx2, multtest, plotROC, vegan, pROC, BiocParallel

Encoding UTF-8

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Suggests testthat, covr, glmnet, Matrix, kernlab, e1071, ranger, knitr, rmarkdown, BiocStyle, withr, microbiome

VignetteBuilder knitr

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/microbiomeMarker

git_branch RELEASE_3_20

git_last_commit c71d8a2

git_last_commit_date 2025-02-19

Repository Bioconductor 3.20

Date/Publication 2025-03-06

Author Yang Cao [aut, cre]

Maintainer Yang Cao <caoyang.name@gmail.com>

Contents

`microbiomeMarker-package` *microbiomeMarker: A package for microbiome biomarker discovery*

Description

The microboimeMarker package provides several methods to identify micribome biomarker, such as lefse, deseq2.

Author(s)

Maintainer: Yang Cao <caoyang.name@gmail.com>

See Also

Useful links:

- https://github.com/yiluheihei/microbiomeMarker
- Report bugs at https://github.com/yiluheihei/microbiomeMarker/issues

---

`abundances` *Extract taxa abundances*

Description

Extract taxa abundances from phyloseq objects.

Usage

```
abundances(object, transform = c("identity", "log10", "log10p"), norm = FALSE)

## S4 method for signature 'otu_table'
abundances(object, transform = c("identity", "log10", "log10p"), norm = FALSE)

## S4 method for signature 'phyloseq'
abundances(object, transform = c("identity", "log10", "log10p"), norm = FALSE)

## S4 method for signature 'microbiomeMarker'
abundances(object, transform = c("identity", "log10", "log10p"))
```

Arguments

object	otu_table, phyloseq, or microbiomeMarker.
transform	transformation to apply, the options inclulde: • "identity", return the original data without any transformation. • "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`. • "log10p", the transformation is `log10(1 + object)`.
norm	logical, indicating whether or not to return the normalized taxa abundances.

Value

abundance matrix with taxa in rows and samples in columns.

See Also

otu_table, phyloseq, microbiomeMarker,transform_abundances

Examples

```
data(caporaso)
abd <- abundances(caporaso)
```

---

aggregate_taxa *Aggregate Taxa*

Description

Summarize phyloseq data into a higher phylogenetic level.

Usage

```
aggregate_taxa(x, level, verbose = FALSE)
```

Arguments

x	phyloseq-class object
level	Summarization level (from `rank_names(pseq)`)
verbose	verbose

Details

This provides a convenient way to aggregate phyloseq OTUs (or other taxa) when the phylogenetic tree is missing. Calculates the sum of OTU abundances over all OTUs that map to the same higher-level group. Removes ambiguous levels from the taxonomy table. Returns a phyloseq object with the summarized abundances.

Value

Summarized phyloseq object

Author(s)

Contact: Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation('microbiome')

Examples

```
data(caporaso)
caporaso_phylum <- aggregate_taxa(caporaso, "Phylum")
```

---

assign-otu_table *Assign a new OTU table*

Description

Assign a new OTU table in microbiomeMarker object

Usage

```
## S4 replacement method for signature 'microbiomeMarker,otu_table'
otu_table(x) <- value

## S4 replacement method for signature 'microbiomeMarker,phyloseq'
otu_table(x) <- value

## S4 replacement method for signature 'microbiomeMarker,microbiomeMarker'
otu_table(x) <- value
```

Arguments

x	microbiomeMarker
value	otu_table, phyloseq, or microbiomeMarker

Value

a microbiomeMarker object.

compare_DA *Comparing the results of differential analysis methods by Empirical power and False Discovery Rate*

Description

Calculating power, false discovery rates, false positive rates and auc (area under the receiver operating characteristic (ROC) curve) for various DA methods.

Usage

```
compare_DA(
  ps,
  group,
  taxa_rank = "none",
  methods,
  args = list(),
  n_rep = 20,
  effect_size = 5,
  k = NULL,
  relative = TRUE,
  BPPARAM = BiocParallel::SnowParam(progressbar = TRUE)
)
```

Arguments

ps, group, taxa_rank	main arguments of all differential analysis methods. ps: a phyloseq::phyloseq object; group, character, the variable to set the group, must be one of the var of the sample metadata; taxa_rank: character, taxonomic rank, please not that **since the abundance table is spiked in the lowest level, only** taxa_rank = "none" **is allowed**.
methods	character vector, differential analysis methods to be compared, available methods are "aldex", "ancom", "ancombc", "deseq2", "edger", "lefse", "limma_voom", "metagenomeseq", "simple_stat".
args	named list, which used to set the extra arguments of the differential analysis methods, so the names must be contained in methods. For more see details below.
n_rep	integer, number of times to run the differential analyses.
effect_size	numeric, the effect size for the spike-ins. Default 5.
k	numeric vector of length 3, number of features to spike in each tertile (lower, mid, upper), e.g. k=c(5,10,15) means 5 features spiked in low abundance tertile, 10 features spiked in mid abundance tertile and 15 features spiked in high abundance tertile. Default NULL, which will spike 2 percent of the total amount of features in each tertile (a total of 6 percent), but minimum c(5,5,5).
relative	logical, whether rescale the total number of individuals observed for each sample to the original level after spike-in. Default TRUE.
BPPARAM	BiocParallel::BiocParallelParam instance defining the parallel back-end.

Details

To make this function support for different arguments for a certain DA method `args` allows list of list of list e.g. `args = list(lefse = list(list(norm = "CPM"), list(norm = "TSS")))`, which specify to compare the different norm arguments for lefse analysis.

For `taxa_rank`, only `taxa_rank = "none"` is supported, if this argument is not "none", it will be forced to "none" internally.

Value

an `compareDA` object, which contains a two-length list of:

- `metrics`: `data.frame`, FPR, AUC and spike detection rate for each run.
- `mm`: differential analysis results.

`confounder` *Confounder analysis*

Description

Confounding variables may mask the actual differential features. This function utilizes constrained correspondence analysis (CCA) to measure the confounding factors.

Usage

```
confounder(
  ps,
  target_var,
  norm = "none",
  confounders = NULL,
  permutations = 999,
  ...
)
```

Arguments

`ps`	a `phyloseq::phyloseq` object.
`target_var`	character, the variable of interest
`norm`	norm the methods used to normalize the microbial abundance data. See `normalize()` for more details.
`confounders`	the confounding variables to be measured, if `NULL`, all variables in the meta data will be analyzed.
`permutations`	the number of permutations, see `vegan::anova.cca()`.
`...`	extra arguments passed to `vegan::anova.cca()`.

Value

a `data.frame` contains three variables: confounder, pseudo-F and p value.

Examples

```
data(caporaso)
confounder(caporaso, "SampleType", confounders = "ReportedAntibioticUsage")
```

`data-caporaso` *16S rRNA data from "Moving pictures of the human microbiome"*

Description

16S read counts and phylogenetic tree file of 34 Illumina samples derived from Moving Pictures of the Human Microbiome (Caporaso et al.) Group label: gut, left palm, right palm, and tongue - indicating different sampled body sites.

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

Source

Data was downloaded from https://www.microbiomeanalyst.ca

References

Caporaso, et al. Moving pictures of the human microbiome. Genome Biol 12, R50 (2011).
https://doi.org/10.1186/gb-2011-12-5-r50

`data-cid_ying` *16S rRNA data of 94 patients from CID 2012*

Description

Data from a cohort of 94 Bone Marrow Transplant patients previously published on in CID

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

Source

https://github.com/ying14/yingtools2/tree/master/data

References

Ying, et al. Intestinal Domination and the Risk of Bacteremia in Patients Undergoing Allogeneic Hematopoietic Stem Cell Transplantation, Clinical Infectious Diseases, Volume 55, Issue 7, 1 October 2012, Pages 905–914,

https://academic.oup.com/cid/article/55/7/905/428203

`data-ecam` *Data from Early Childhood Antibiotics and the Microbiome (ECAM) study*

Description

The data from a subset of the Early Childhood Antibiotics and the Microbiome (ECAM) study, which tracked the microbiome composition and development of 43 infants in the United States from birth to 2 years of age, identifying microbiome associations with antibiotic exposure, delivery mode, and diet.

Format

a `phyloseq::phyloseq` object

References

Bokulich, Nicholas A., et al. "Antibiotics, birth mode, and diet shape microbiome maturation during early life." Science translational medicine 8.343 (2016): 343ra82-343ra82.

https://github.com/FrederickHuangLin/ANCOM/tree/master/data

`data-enterotypes_arumugam`
Enterotypes data of 39 samples

Description

The data contains 22 European metagenomes from Danish, French, Italian, and Spanish individuals, and 13 Japanese and 4 American.

Format

a `phyloseq::phyloseq` object

Author(s)

Yang Cao

References

Arumugam, Manimozhiyan, et al. Enterotypes of the human gut microbiome. nature 473.7346 (2011): 174-180.

data-kostic_crc *Data from a study on colorectal cancer (kostic 2012)*

Description

The data from a study on colorectal cancer. Samples that had no DIAGNOSIS attribute assigned and with less than 500 reads (counts) were removed, and 191 samples remains (91 healthy and 86 Tumors).

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

References

Kostic et al. Genomic analysis identifies association of Fusobacterium with colorectal carcinoma. Genome research, 2012, 22(2), 292-298.

data-oxygen *Oxygen availability 16S dataset, of which taxa table has been summarized for python lefse input*

Description

A small subset of the HMP 16S dataset for finding biomarkers characterizing different level of oxygen availability in different bodysites

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

Source

http://huttenhower.sph.harvard.edu/webfm_send/129

data-pediatric_ibd *IBD stool samples*

Description

43 pediatric IBD stool samples obtained from the Integrative Human Microbiome Project Consortium (iHMP). Group label: CD and Controls.

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

Source

https://www.microbiomeanalyst.ca/MicrobiomeAnalyst/resources

data-spontaneous_colitis *This is a sample data from lefse python script, a 16S dataset for studying the characteristics of the fecal microbiota in a mouse model of spontaneous colitis.*

Description

The dataset contains 30 abundance profiles (obtained processing the 16S reads with RDP) belonging to 10 rag2 (control) and 20 truc (case) mice.

Format

a phyloseq::phyloseq object

Author(s)

Yang Cao

Source

http://www.huttenhower.org/webfm_send/73

`extract_posthoc_res` *Extract results from a posthoc test*

Description

This function extracts the results of posthoc test.

Usage

```
extract_posthoc_res(object, features = NULL)
```

Arguments

`object`	a `postHocTest` object.
`features`	either `NULL` extracts results of all features, or a character vector to specify the test resuts of which features are extracted.

Value

a `IRanges::SimpleDFrameList` object.

Examples

```
require(IRanges)
pht <- postHocTest(
    result = DataFrameList(
        featureA = DataFrame(
            comparisons = c("group2-group1",
                "group3-group1",
                "group3-group2"),
            diff_mean = runif(3),
            pvalue = rep(0.01, 3),
            ci_lower = rep(0.01, 3),
            ci_upper = rep(0.011, 3)
        ),
        featureB = DataFrame(
            comparisons = c("group2-group1",
                "group3-group1",
                "group3-group2"),
            diff_mean = runif(3),
            pvalue = rep(0.01, 3),
            ci_lower = rep(0.01, 3),
            ci_upper = rep(0.011, 3)
        )
    ),
    abundance = data.frame(
        featureA = runif(3),
        featureB = runif(3),
        group = c("group1", "group2", "grou3")
    )
)
extract_posthoc_res(pht, "featureA")[[1]]
```

get_treedata_phyloseq *Generate tree data from phyloseq object*

Description

Generate tree data from phyloseq object

Usage

```
get_treedata_phyloseq(ps, sep = "|")
```

Arguments

ps	a phyloseq::phyloseq object
sep	character, separate between different levels of taxa, default |

Value

a tidytree::treedata object

Author(s)

Yang Cao

import_dada2 *Import function to read the the output of dada2 as phyloseq object*

Description

Import the output of dada2 into phyloseq object

Usage

```
import_dada2(
  seq_tab,
  tax_tab = NULL,
  sam_tab = NULL,
  phy_tree = NULL,
  keep_taxa_rows = TRUE
)
```

Arguments

seq_tab	matrix-like, ASV table, the output of dada2::removeBimeraDenovo.
tax_tab	matrix, taxonomy table, the output of dada2::assignTaxonomy or dada2::addSpecies.
sam_tab	data.frame or phyloseq::sample_data, sample data
phy_tree	ape::phylo class or character represents the path of the tree file
keep_taxa_rows	logical, whether keep taxa in rows or not in the otu_table of the returned phyloseq object, default TRUE.

Details

The output of the dada2 pipeline is a feature table of amplicon sequence variants (an ASV table): A matrix with rows corresponding to samples and columns to ASVs, in which the value of each entry is the number of times that ASV was observed in that sample. This table is analogous to the traditional OTU table. Conveniently, taxa names are saved as ASV1, ASV2, ..., in the returned phyloseq object.

Value

`phyloseq::phyloseq` object hold the taxonomy info, sample metadata, number of reads per ASV.

Examples

```
seq_tab <- readRDS(system.file("extdata", "dada2_seqtab.rds",
    package = "microbiomeMarker"
))
tax_tab <- readRDS(system.file("extdata", "dada2_taxtab.rds",
    package = "microbiomeMarker"
))
sam_tab <- read.table(system.file("extdata", "dada2_samdata.txt",
    package = "microbiomeMarker"
), sep = "\t", header = TRUE, row.names = 1)
ps <- import_dada2(seq_tab = seq_tab, tax_tab = tax_tab, sam_tab = sam_tab)
ps
```

---

import_picrust2 *Import function to read the output of picrust2 as phyloseq object*

Description

Import the output of picrust2 into phyloseq object

Usage

```
import_picrust2(
  feature_tab,
  sam_tab = NULL,
  trait = c("PATHWAY", "COG", "EC", "KO", "PFAM", "TIGRFAM", "PHENO")
)
```

Arguments

`feature_tab`	character, file path of the prediction abundance table of functional feature.
`sam_tab`	character, file path of the sample meta data.
`trait`	character, options are picrust2 function traits (including "COG", "EC", "KO", "PFAM", "TIGRFAM", and "PHENO") and "PATHWAY".

Details

PICRUSt2 is a software for predicting abundances of functional profiles based on marker gene sequencing data. The functional profiles can be predicted from the taxonomic profiles using PICRUSt2. "Function" usually refers to gene families such as KEGG orthologs and Enzyme Classification numbers, but predictions can be made for any arbitrary trait.

In the `phyloseq object`, the predicted function abundance profile is stored in `otu_table` slot. And the functional trait is saved in `tax_table` slot, if the descriptions of function features is not added to the predicted table, `tax_table` will have only one rank `Picrust_trait` to represent the function feature id, or if the desciptions are added one more rank `Picrust_description` will be added to represent the description of function feature.

Value

`phyloseq::phyloseq` object.

Examples

```
sam_tab <- system.file(
    "extdata", "picrust2_metadata.tsv",
    package = "microbiomeMarker")
feature_tab <- system.file(
    "extdata", "path_abun_unstrat_descrip.tsv.gz",
    package = "microbiomeMarker")
ps <- import_picrust2(feature_tab, sam_tab, trait = "PATHWAY")
ps
```

---

import_qiime2 *Import function to read the the output of dada2 as phyloseq object*

Description

Import the qiime2 artifacts, including feature table, taxonomic table, phylogenetic tree, representative sequence and sample metadata into phyloseq object.

Usage

```
import_qiime2(
  otu_qza,
  taxa_qza = NULL,
  sam_tab = NULL,
  refseq_qza = NULL,
  tree_qza = NULL
)
```

Arguments

`otu_qza`	character, file path of the feature table from qiime2.
`taxa_qza`	character, file path of the taxonomic table from qiime2, default `NULL`.
`sam_tab`	character, file path of the sample metadata in tsv format, default `NULL`.
`refseq_qza`	character, file path of the representative sequences from qiime2, default `NULL`.
`tree_qza`	character, file path of the phylogenetic tree from qiime2, default `NULL`.

Value

phyloseq::phyloseq object.

Examples

```
otuqza_file <- system.file(
    "extdata", "table.qza",
    package = "microbiomeMarker"
)
taxaqza_file <- system.file(
    "extdata", "taxonomy.qza",
    package = "microbiomeMarker"
)
sample_file <- system.file(
    "extdata", "sample-metadata.tsv",
    package = "microbiomeMarker"
)
treeqza_file <- system.file(
    "extdata", "tree.qza",
    package = "microbiomeMarker"
)
ps <- import_qiime2(
    otu_qza = otuqza_file, taxa_qza = taxaqza_file,
    sam_tab = sample_file, tree_qza = treeqza_file
)
ps
```

---

marker_table *Build or access the marker_table*

Description

This is the recommended function for both building and accessing microbiome marker table (marker_table).

Usage

```
marker_table(object)

## S4 method for signature 'data.frame'
marker_table(object)

## S4 method for signature 'microbiomeMarker'
marker_table(object)
```

Arguments

object	an object among the set of classes defined by the microbiomeMarker package that contain marker_table

Value

a marker_table object.

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.05,
    p_adjust = "fdr"
)
marker_table(mm)
```

---

marker_table-class *The S4 class for storing microbiome marker information*

Description

This Class is inherit from `data.frame`. Rows represent the microbiome markers and variables represents feature of the marker.

Fields

`names,row.names` a character vector, inherited from the input data.frame

`.data` a list, each element corresponding the each column of the input data.frame

`.S3Class` character, the S3 class `marker_table` inherited from: `"data.frame"`

Author(s)

Yang Cao

---

marker_table<- *Assign marker_table to* `object`

Description

This function replace the `marker_table` slot of `object` with `value`.

Usage

```
marker_table(object) <- value
```

Arguments

`object`	a `microbiomeMarker` object to modify.
`value`	new value to replace the `marker_table` slot of `object`. Either a `marker_table-class`, a `data.frame` that can be coerced into `marker_table-class`.

Value

a `microbiomeMarker` object.

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.1,
    p_adjust = "fdr"
)
mm_marker <- marker_table(mm)
mm_marker
marker_table(mm) <- mm_marker[1:2, ]
marker_table(mm)
```

`microbiomeMarker` *Build microbiomeMarker-class objects*

Description

This the constructor to build the `microbiomeMarker` object, don't use the `new()` constructor.

Usage

```
microbiomeMarker(
  marker_table = NULL,
  norm_method = NULL,
  diff_method = NULL,
  ...
)
```

Arguments

`marker_table`	a `marker_table` object differtial analysis.
`norm_method`	character, method used to normalize the input `phyloseq` object.
`diff_method`	character, method used for microbiome marker identification.
`...`	arguments passed to `phyloseq::phyloseq()`

Value

a `microbiomeMarker` object.

See Also

`phyloseq::phyloseq()`

Examples

```
microbiomeMarker(
    marker_table = marker_table(data.frame(
        feature = c("speciesA", "speciesB"),
        enrich_group = c("groupA", "groupB"),
        ef_logFC = c(-2, 2),
        pvalue = c(0.01, 0.01),
        padj = c(0.01, 0.01),
        row.names = c("marker1", "marker2")
    )),
    norm_method = "TSS",
    diff_method = "DESeq2",
    otu_table = otu_table(matrix(
        c(4, 1, 1, 4),
        nrow = 2, byrow = TRUE,
        dimnames = list(c("speciesA", "speciesB"), c("sample1", "sample2"))
    ),
    taxa_are_rows = TRUE
    ),
    tax_table = tax_table(matrix(
        c("speciesA", "speciesB"),
        nrow = 2,
        dimnames = list(c("speciesA", "speciesB"), "Species")
    )),
    sam_data = sample_data(data.frame(
        group = c("groupA", "groupB"),
        row.names = c("sample1", "sample2")
    ))
)
```

---

microbiomeMarker-class

The main class for microbiomeMarker data

Description

`microbiomeMarker-class` is inherited from the `phyloseq::phyloseq` by adding a custom slot `microbiome_marker` to save the differential analysis results. And it provides a seamless interface with **phyloseq**, which makes **microbiomeMarker** simple and easy to use. For more details on see the document of `phyloseq::phyloseq`.

Usage

```
## S4 method for signature 'microbiomeMarker'
show(object)
```

Arguments

`object`	a `microbiomeMarker-class` object

Value

a `microbiomeMarker` object.

Slots

`marker_table` a data.frame, a `marker_table` object.

`norm_method` character, method used to normalize the input `phyloseq` object.

`diff_method` character, method used for marker identification.

See Also

`phyloseq::phyloseq`, `marker_table`, `summarize_taxa()`

---

nmarker *Get the number of microbiome markers*

Description

Get the number of microbiome markers

Usage

```
nmarker(object)

## S4 method for signature 'microbiomeMarker'
nmarker(object)

## S4 method for signature 'marker_table'
nmarker(object)
```

Arguments

`object`	a `microbiomeMarker` or `marker_table` object

Value

an integer, the number of microbiome markers

Examples

```
mt <- marker_table(data.frame(
    feature = c("speciesA", "speciesB"),
    enrich_group = c("groupA", "groupB"),
    ef_logFC = c(-2, 2),
    pvalue = c(0.01, 0.01),
    padj = c(0.01, 0.01),
    row.names = c("marker1", "marker2")
))
nmarker(mt)
```

`normalize,phyloseq-method`

Normalize the microbial abundance data

Description

It is critical to normalize the feature table to eliminate any bias due to differences in the sampling sequencing depth.This function implements six widely-used normalization methods for microbial compositional data.

For rarefying, reads in the different samples are randomly removed until the same predefined number has been reached, to assure all samples have the same library size. Rarefying normalization method is the standard in microbial ecology. Please note that the authors of phyloseq do not advocate using this rarefying a normalization procedure, despite its recent popularity

TSS simply transforms the feature table into relative abundance by dividing the number of total reads of each sample.

CSS is based on the assumption that the count distributions in each sample are equivalent for low abundant genes up to a certain threshold. Only the segment of each sample's count distribution that is relatively invariant across samples is scaled by CSS

RLE assumes most features are not differential and uses the relative abundances to calculate the normalization factor.

TMM calculates the normalization factor using a robust statistics based on the assumption that most features are not differential and should, in average, be equal between the samples. The TMM scaling factor is calculated as the weighted mean of log-ratios between each pair of samples, after excluding the highest count OTUs and OTUs with the largest log-fold change.

In CLR, the log-ratios are computed relative to the geometric mean of all features.

`norm_cpm`: This normalization method is from the original LEfSe algorithm, recommended when very low values are present (as shown in the LEfSe galaxy).

Usage

```
## S4 method for signature 'phyloseq'
normalize(object, method = "TSS", ...)

## S4 method for signature 'otu_table'
normalize(object, method = "TSS", ...)

## S4 method for signature 'data.frame'
normalize(object, method = "TSS", ...)

## S4 method for signature 'matrix'
normalize(object, method = "TSS", ...)

norm_rarefy(
  object,
  size = min(sample_sums(object)),
  rng_seed = FALSE,
  replace = TRUE,
  trim_otus = TRUE,
```

```
  verbose = TRUE
)

norm_tss(object)

norm_css(object, sl = 1000)

norm_rle(
  object,
  locfunc = stats::median,
  type = c("poscounts", "ratio"),
  geo_means = NULL,
  control_genes = NULL
)

norm_tmm(
  object,
  ref_column = NULL,
  logratio_trim = 0.3,
  sum_trim = 0.05,
  do_weighting = TRUE,
  Acutoff = -1e+10
)

norm_clr(object)

norm_cpm(object)
```

Arguments

object	a phyloseq::phyloseq or phyloseq::otu_table
method	the methods used to normalize the microbial abundance data. Options includes: • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size. • "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference. • "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference. • "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold. • "CLR": centered log-ratio normalization. • "CPM": pre-sample normalization of the sum of the values to 1e+06.
...	other arguments passed to the corresponding normalization methods.

size, rng_seed, replace, trim_otus, verbose	extra arguments passed to phyloseq::rarefy_even_depth().
sl	The value to scale.
locfunc	a function to compute a location for a sample. By default, the median is used.
type	method for estimation: either "ratio"or "poscounts" (recommend).
geo_means	default NULL, which means the geometric means of the counts are used. A vector of geometric means from another count matrix can be provided for a "frozen" size factor calculation.
control_genes	default NULL, which means all taxa are used for size factor estimation, numeric or logical index vector specifying the taxa used for size factor estimation (e.g. core taxa).
ref_column	column to use as reference
logratio_trim	amount of trim to use on log-ratios
sum_trim	amount of trim to use on the combined absolute levels ("A" values)
do_weighting	whether to compute the weights or not
Acutoff	cutoff on "A" values to use before trimming

Value

the same class with object.

See Also

edgeR::calcNormFactors(),DESeq2::estimateSizeFactorsForMatrix(),metagenomeSeq::cumNorm()

phyloseq::rarefy_even_depth()

metagenomeSeq::calcNormFactors()

DESeq2::estimateSizeFactorsForMatrix()

edgeR::calcNormFactors()

Examples

```
data(caporaso)
normalize(caporaso, "TSS")
```

---

phyloseq2DESeq2 *Convert* phyloseq-class *object to* DESeqDataSet-class *object*

Description

This function convert [phyloseq::phyloseq-class] to [DESeq2::DESeqDataSet-class], which can then be tested us

Usage

```
phyloseq2DESeq2(ps, design, ...)
```

Arguments

ps	the [phyloseq::phyloseq-class] object to convert, which must have a [phyloseq::sample_data() component.
design	a formula or matrix, the formula expresses how the counts for each gene depend on the variables in colData. Many R formula are valid, including designs with multiple variables, e.g., ~ group + condition. This argument is passed to DESeq2::DESeqDataSetFromMatrix().
...	additional arguments passed to DESeq2::DESeqDataSetFromMatrix(), Most users will not need to pass any additional arguments here.

Value

a DESeq2::DESeqDataSet object.

See Also

DESeq2::DESeqDataSetFromMatrix(),DESeq2::DESeq()

Examples

```
data(caporaso)
phyloseq2DESeq2(caporaso, ~SampleType)
```

---

phyloseq2edgeR *Convert phyloseq data to edgeR* DGEList *object*

Description

This function convert phyloseq::phyloseq object to edgeR::DGEList object, can then can be used to perform differential analysis using the methods in **edgeR**.

Usage

```
phyloseq2edgeR(ps, ...)
```

Arguments

ps	a phyloseq::phyloseq object.
...	optional, additional named arguments passed to edgeR::DGEList(). Most users will not need to pass any additional arguments here.

Value

A edgeR::DGEList object.

Examples

```
data(caporaso)
dge <- phyloseq2edgeR(caporaso)
```

phyloseq2metagenomeSeq *Convert phyloseq data to MetagenomeSeq* MRexperiment *object*

Description

The phyloseq data is converted to the relevant metagenomeSeq::MRexperiment object, which can then be tested in the zero-inflated mixture model framework in the metagenomeSeq package.

Usage

```
phyloseq2metagenomeSeq(ps, ...)

otu_table2metagenomeSeq(ps, ...)
```

Arguments

ps	phyloseq::phyloseq object for phyloseq2metagenomeSeq(), or phyloseq::otu_table object for otu_table2metagenomeseq().
...	optional, additional named arguments passed to metagenomeSeq::newMRexperiment(). Most users will not need to pass any additional arguments here.

Value

A metagenomeSeq::MRexperiment object.

See Also

metagenomeSeq::fitTimeSeries(), metagenomeSeq::fitLogNormal(),metagenomeSeq::fitZig(), metagenomeSeq::MRtable(),metagenomeSeq::MRfulltable()

Examples

```
data(caporaso)
phyloseq2metagenomeSeq(caporaso)
```

plot.compareDA *Plotting DA comparing result*

Description

Plotting DA comparing result

Usage

```
## S3 method for class 'compareDA'
plot(x, sort = c("score", "auc", "fpr", "power"), ...)
```

Arguments

x	an compareDA object, output from compare_DA().
sort	character string specifying sort method. Possibilities are "score" which is calculated as $(auc - 0.5) * power - fdr$, "auc" for area under the ROC curve, "fpr" for false positive rate, "power" for empirical power.
...	extra arguments, just ignore it.

Value

a ggplot2::ggplot object containing 4 subplots: "auc", "fdr", "power"and "score" plot.

---

plot_abundance *plot the abundances of markers*

Description

plot the abundances of markers

Usage

```
plot_abundance(mm, label_level = 1, max_label_len = 60, markers = NULL, group)
```

Arguments

mm	a microbiomeMarker object
label_level	integer, number of label levels to be displayed, default 1, 0 means display the full name of the feature
max_label_len	integer, maximum number of characters of feature label, default 60
markers	character vector, markers to display, default NULL, indicating plot all markers.
group	character, the variable to set the group

Value

a ggplot2::ggplot object.

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.01,
    p_adjust = "none"
)
plot_abundance(mm, group = "Enterotype")
```

plot_cladogram *plot cladogram of micobiomeMaker results*

Description

plot cladogram of micobiomeMaker results

Usage

```
plot_cladogram(
  mm,
  color,
  only_marker = FALSE,
  branch_size = 0.2,
  alpha = 0.2,
  node_size_scale = 1,
  node_size_offset = 1,
  clade_label_level = 4,
  clade_label_font_size = 4,
  annotation_shape = 22,
  annotation_shape_size = 5,
  group_legend_param = list(),
  marker_legend_param = list()
)
```

Arguments

mm	a microbiomeMarker object
color	a color vector, used to highlight the clades of microbiome biomarker. The values will be matched in order (usually alphabetical) with the groups. If this is a named vector, then the colors will be matched based on the names instead.
only_marker	logical, whether show all the features or only markers in the cladogram, default FALSE.
branch_size	numeric, size of branch, default 0.2
alpha	alpha parameter for shading, default 0.2
node_size_scale	the parameter 'a' controlling node size: node_size=a*log(relative_abundance) + b
node_size_offset	the parameter 'b' controlling node size: node_size=a*log(relative_abundance) + b
clade_label_level	max level of taxa used to label the clade, other level of taxa will be shown on the side.
clade_label_font_size	font size of the clade label, default 4.
annotation_shape	shape used for annotation, default 22

annotation_shape_size
: size used for annotation shape, default 5

group_legend_param, marker_legend_param
: a list specifying extra parameters of group legend and marker legend, such as direction (the direction of the guide), nrow (the desired number of rows of legends). See ggplot2::guide_legend() for more details.

Value

a ggtree object

Author(s)

Chenhao Li, Guangchuang Yu, Chenghao Zhu, Yang Cao

References

This function is modified from clada.anno from microbiomeViz.

See Also

ggtree::ggtree()

Examples

```
data(kostic_crc)
kostic_crc_small <- phyloseq::subset_taxa(
    kostic_crc,
    Phylum %in% c("Firmicutes")
)
mm_lefse <- run_lefse(
    kostic_crc_small,
    wilcoxon_cutoff = 0.01,
    group = "DIAGNOSIS",
    kw_cutoff = 0.01,
    multigrp_strat = TRUE,
    lda_cutoff = 4
)
plot_cladogram(mm_lefse, color = c("darkgreen", "red"))
```

---

plot_ef_bar *bar and dot plot of effect size of microbiomeMarker data*

Description

bar and dot plot of effect size microbiomeMarker data. This function returns a ggplot2 object that can be saved or further customized using **ggplot2** package.

Usage

```
plot_ef_bar(mm, label_level = 1, max_label_len = 60, markers = NULL)

plot_ef_dot(mm, label_level = 1, max_label_len = 60, markers = NULL)
```

Arguments

mm	a microbiomeMarker object
label_level	integer, number of label levels to be displayed, default 1, 0 means display the full name of the feature
max_label_len	integer, maximum number of characters of feature label, default 60
markers	character vector, markers to display, default NULL, indicating plot all markers.

Value

a ggplot project

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.01,
    p_adjust = "none"
)
plot_ef_bar(mm)
```

---

plot_heatmap *Heatmap of microbiome marker*

Description

Display the microbiome marker using heatmap, in which rows represents the marker and columns represents the samples.

Usage

```
plot_heatmap(
  mm,
  transform = c("log10", "log10p", "identity"),
  cluster_marker = FALSE,
  cluster_sample = FALSE,
  markers = NULL,
  label_level = 1,
  max_label_len = 60,
  sample_label = FALSE,
  scale_by_row = FALSE,
  annotation_col = NULL,
  group,
  ...
)
```

Arguments

mm	a microbiomeMarker object
transform	transformation to apply, for more details see transform_abundances(): • "identity", return the original data without any transformation. • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
cluster_marker, cluster_sample	logical, controls whether to perform clustering in markers (rows) and samples (cols), default FALSE.
markers	character vector, markers to display, default NULL, indicating plot all markers.
label_level	integer, number of label levels to be displayed, default 1, 0 means display the full name of the feature
max_label_len	integer, maximum number of characters of feature label, default 60
sample_label	logical, controls whether to show the sample labels in the heatmap, default FALSE.
scale_by_row	logical, controls whether to scale the heatmap by the row (marker) values, default FALSE.
annotation_col	assign colors for the top annotation using a named vector, passed to col in ComplexHeatmap::HeatmapAnnotation().
group	character, the variable to set the group
...	extra arguments passed to ComplexHeatmap::Heatmap().

Value

a ComplexHeatmap::Heatmap object.

See Also

transform_abundances,ComplexHeatmap::Heatmap()

Examples

```
data(kostic_crc)
kostic_crc_small <- phyloseq::subset_taxa(
    kostic_crc,
    Phylum %in% c("Firmicutes")
)
mm_lefse <- run_lefse(
    kostic_crc_small,
    wilcoxon_cutoff = 0.01,
    group = "DIAGNOSIS",
    kw_cutoff = 0.01,
    multigrp_strat = TRUE,
    lda_cutoff = 4
)
plot_heatmap(mm_lefse, group = "DIAGNOSIS")
```

plot_postHocTest postHocTest *plot*

Description

Visualize the result of post-hoc test using ggplot2

Usage

```
plot_postHocTest(pht, feature, step_increase = 0.12)
```

Arguments

pht	a postHocTest object
feature	character, to plot the post-toc test result of this feature
step_increase	numeric vector with the increase in fraction of total height for every additional comparison to minimize overlap, default 0.12.

Value

a ggplot object

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2", "Enterotype 1")
) %>%
    phyloseq::subset_taxa(Phylum == "Bacteroidetes")
pht <- run_posthoc_test(ps, group = "Enterotype")
plot_postHocTest(pht, feature = "p__Bacteroidetes|g__Alistipes")
```

plot_sl_roc *ROC curve of microbiome marker from supervised learning methods*

Description

Show the ROC curve of the microbiome marker calculated by run_sl.

Usage

```
plot_sl_roc(mm, group, nfolds = 3, nrepeats = 3, tune_length = 5, ...)
```

Arguments

mm	a microbiomeMarker object.
group, nfolds, nrepeats, tune_length, ...	same with the run_sl().

Value

a ggplot2::ggplot object.

See Also

run_sl()

Examples

```
data(enterotypes_arumugam)
# small example phyloseq object for test
ps_s <- phyloseq::subset_taxa(
    enterotypes_arumugam,
    Phylum %in% c("Firmicutes", "Bacteroidetes")
)

set.seed(2021)
mm <- run_sl(
    ps_s,
    group = "Gender",
    taxa_rank = "Genus",
    nfolds = 2,
    nrepeats = 1,
    top_n = 15,
    norm = "TSS",
    method = "LR",
)
plot_sl_roc(mm, group = "Gender")
```

---

postHocTest *Build postHocTest object*

Description

This function is used for create postHocTest object, and is only used for developers.

Usage

```
postHocTest(
  result,
  abundance,
  conf_level = 0.95,
  method = "tukey",
  method_str = paste("Posthoc multiple comparisons of means: ", method)
)
```

Arguments

result	a IRanges::SimpleDFrameList object.
abundance	data.frame.
conf_level	numeric, confidence level.
method	character, method for posthoc test.
method_str	character, illustrates which method is used for posthoc test.

Value

a postHocTest object.

Examples

```
require(IRanges)
pht <- postHocTest(
    result = DataFrameList(
        featureA = DataFrame(
            comparisons = c("group2-group1",
                "group3-group1",
                "group3-group2"),
            diff_mean = runif(3),
            pvalue = rep(0.01, 3),
            ci_lower = rep(0.01, 3),
            ci_upper = rep(0.011, 3)
        ),
        featureB = DataFrame(
            comparisons = c("group2-group1",
                "group3-group1",
                "group3-group2"),
            diff_mean = runif(3),
            pvalue = rep(0.01, 3),
            ci_lower = rep(0.01, 3),
            ci_upper = rep(0.011, 3)
        )
    ),
    abundance = data.frame(
        featureA = runif(3),
        featureB = runif(3),
        group = c("group1", "group2", "grou3")
    )
)
pht
```

---

postHocTest-class *The postHocTest Class, represents the result of post-hoc test result among multiple groups*

Description

The postHocTest Class, represents the result of post-hoc test result among multiple groups

Usage

```
## S4 method for signature 'postHocTest'
show(object)
```

Arguments

object	a postHocTest-class object

Value

a postHocTest object.

Slots

result a IRanges::DataFrameList, each DataFrame consists of five variables:

- comparisons: character, specify which two groups to test (the group names are separated by "_)
- diff_mean: numeric, difference in mean abundances
- pvalue: numeric, p values
- ci_lower and ci_upper: numeric, lower and upper confidence interval of difference in mean abundances

abundance abundance of each feature in each group

conf_level confidence level

method method used for post-hoc test

method_str method illustration

Author(s)

Yang Cao

---

reexports *Objects exported from other packages*

Description

These objects are imported from other packages. Follow the links below to see their documentation.

magrittr %>%

phyloseq import_biom, import_mothur, import_qiime, nsamples, ntaxa, otu_table, sample_data, sample_names, tax_table, taxa_names

---

run_aldex *Perform differential analysis using ALDEx2*

Description

Perform differential analysis using ALDEx2

Usage

```
run_aldex(
  ps,
  group,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "none",
  norm_para = list(),
  method = c("t.test", "wilcox.test", "kruskal", "glm_anova"),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  mc_samples = 128,
  denom = c("all", "iqlr", "zero", "lvha"),
  paired = FALSE
)
```

Arguments

ps	a phyloseq::phyloseq object
group	character, the variable to set the group
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of phyloseq::rank_names(phyloseq), or "all" means to summarize the taxa by the top taxa ranks (summarize_taxa(ps, level = rank_names(ps)[1])), or "none" means perform differential analysis on the original taxa (taxa_names(phyloseq), e.g., OTU or ASV).
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation (default). • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include: • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size. • "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference. • "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference. • "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold. • "CLR": centered log-ratio normalization.

- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods

`method` test method, options include: "t.test" and "wilcox.test" for two groups comparison, "kruskal" and "glm_anova" for multiple groups comparison.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` cutoff of p value, default 0.05.

`mc_samples` integer, the number of Monte Carlo samples to use for underlying distributions estimation, 128 is usually sufficient.

`denom` character string, specifiy which features used to as the denominator for the geometric mean calculation. Options are:

- "all", with all features.
- "iqlr", accounts for data with systematic variation and centers the features on the set features that have variance that is between the lower and upper quartile of variance.
- "zero", a more extreme case where there are many non-zero features in one condition but many zeros in another. In this case the geometric mean of each group is calculated using the set of per-group non-zero features.
- "lvha", with house keeping features.

`paired` logical, whether to perform paired tests, only worked for method "t.test" and "wilcox.test".

Value

a `microbiomeMarker` object.

References

Fernandes, A.D., Reid, J.N., Macklaim, J.M. et al. Unifying the analysis of high-throughput sequencing datasets: characterizing RNA-seq, 16S rRNA gene sequencing and selective growth experiments by compositional data analysis. Microbiome 2, 15 (2014).

See Also

`ALDEx2::aldex()`

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
run_aldex(ps, group = "Enterotype")
```

run_ancom *Perform differential analysis using ANCOM*

Description

Perform significant test by comparing the pairwise log ratios between all features.

Usage

```
run_ancom(
  ps,
  group,
  confounders = character(0),
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "TSS",
  norm_para = list(),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  W_cutoff = 0.75
)
```

Arguments

ps	a phyloseq-class object.
group	character, the variable to set the group.
confounders	character vector, the confounding variables to be adjusted. default character(0), indicating no confounding variable.
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of phyloseq::rank_names(phyloseq), or "all" means to summarize the taxa by the top taxa ranks (summarize_taxa(ps, level = rank_names(ps)[1])), or "none" means perform differential analysis on the original taxa (taxa_names(phyloseq), e.g., OTU or ASV).
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation. • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include: • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.

- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` named `list`. other arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` significance level for each of the statistical tests, default 0.05.

`W_cutoff` lower bound for the proportion for the W-statistic, default 0.7.

Details

In an experiment with only two treatments, this tests the following hypothesis for feature i:

$$H_{0i} : E(log(\mu_i^1)) = E(log(\mu_i^2))$$

where μ_i^1 and μ_i^2 are the mean abundances for feature i in the two groups.

The developers of this method recommend the following significance tests if there are 2 groups, use non-parametric Wilcoxon rank sum test `stats::wilcox.test()`. If there are more than 2 groups, use nonparametric `stats::kruskal.test()` or one-way ANOVA `stats::aov()`.

Value

a microbiomeMarker object, in which the `slot` of `marker_table` contains four variables:

- `feature`, significantly different features.
- `enrich_group`, the class of the differential features enriched.
- `effect_size`, differential means for two groups, or F statistic for more than two groups.
- `W`, the W-statistic, number of features that a single feature is tested to be significantly different against.

Author(s)

Huang Lin, Yang Cao

References

Mandal et al. "Analysis of composition of microbiomes: a novel method for studying microbial composition", Microbial Ecology in Health & Disease, (2015), 26.

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
run_ancom(ps, group = "Enterotype")
```

`run_ancombc` *Differential analysis of compositions of microbiomes with bias correction (ANCOM-BC).*

Description

Differential abundance analysis for microbial absolute abundance data. This function is a wrapper of `ANCOMBC::ancombc()`.

Usage

```
run_ancombc(
  ps,
  group,
  confounders = character(0),
  contrast = NULL,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "none",
  norm_para = list(),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  prv_cut = 0.1,
  lib_cut = 0,
  struc_zero = FALSE,
  neg_lb = FALSE,
  tol = 1e-05,
  max_iter = 100,
  conserve = FALSE,
  pvalue_cutoff = 0.05
)
```

Arguments

`ps`	a `phyloseq::phyloseq` object, which consists of a feature table, a sample metadata and a taxonomy table.
`group`	the name of the group variable in metadata. Specifying `group` is required for detecting structural zeros and performing global test.
`confounders`	character vector, the confounding variables to be adjusted. default `character(0)`, indicating no confounding variable.
`contrast`	this parameter only used for two groups comparison while there are multiple groups. For more please see the following details.

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` named `list`. other arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`p_adjust` method to adjust p-values by. Default is "holm". Options include "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". See `stats::p.adjust()` for more details.

`prv_cut` a numerical fraction between 0 and 1. Taxa with prevalences less than `prv_cut` will be excluded in the analysis. Default is 0.10.

`lib_cut` a numerical threshold for filtering samples based on library sizes. Samples with library sizes less than `lib_cut` will be excluded in the analysis. Default is 0, i.e. do not filter any sample.

`struc_zero` whether to detect structural zeros. Default is FALSE.

`neg_lb` whether to classify a taxon as a structural zero in the corresponding study group using its asymptotic lower bound. Default is FALSE.

`tol` the iteration convergence tolerance for the E-M algorithm. Default is 1e-05.

`max_iter` the maximum number of iterations for the E-M algorithm. Default is 100.

`conserve` whether to use a conservative variance estimate of the test statistic. It is recommended if the sample size is small and/or the number of differentially abundant taxa is believed to be large. Default is FALSE.

`pvalue_cutoff` level of significance. Default is 0.05.

Details

`contrast` must be a two length character or `NULL` (default). It is only required to set manually for two groups comparison when there are multiple groups. The order determines the direction of comparison, the first element is used to specify the reference group (control). This means that, the first element is the denominator for the fold change, and the second element is used as baseline (numerator for fold change). Otherwise, users do required to concern this parameter (set as default `NULL`), and if there are two groups, the first level of groups will set as the reference group; if there are multiple groups, it will perform an ANOVA-like testing to find markers which difference in any of the groups.

Value

a `microbiomeMarker` object.

References

Lin, Huang, and Shyamal Das Peddada. "Analysis of compositions of microbiomes with bias correction." Nature communications 11.1 (2020): 1-11.

See Also

`ANCOMBC::ancombc`

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
if (requireNamespace("microbiome", quietly = TRUE)) {
    run_ancombc(ps, group = "Enterotype")
} else {
   message("The 'mirobiome' package is not installed, please install it to use this example")
}
```

---

run_deseq2 *Perform DESeq differential analysis*

Description

Differential expression analysis based on the Negative Binomial distribution using **DESeq2**.

Usage

```
run_deseq2(
  ps,
  group,
  confounders = character(0),
  contrast = NULL,
  taxa_rank = "all",
  norm = "RLE",
```

```
  norm_para = list(),
  transform = c("identity", "log10", "log10p"),
  fitType = c("parametric", "local", "mean", "glmGamPoi"),
  sfType = "poscounts",
  betaPrior = FALSE,
  modelMatrixType,
  useT = FALSE,
  minmu = ifelse(fitType == "glmGamPoi", 1e-06, 0.5),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  ...
)
```

Arguments

ps	a phyloseq::phyloseq object.
group	character, the variable to set the group, must be one of the var of the sample metadata.
confounders	character vector, the confounding variables to be adjusted. default character(0), indicating no confounding variable.
contrast	this parameter only used for two groups comparison while there are multiple groups. For more please see the following details.
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of phyloseq::rank_names(phyloseq), or "all" means to summarize the taxa by the top taxa ranks (summarize_taxa(ps, level = rank_names(ps)[1])), or "none" means perform differential analysis on the original taxa (taxa_names(phyloseq), e.g., OTU or ASV).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include: • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference. • "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference. • "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
norm_para	arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation (default). • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object).

- "log10p", the transformation is `log10(1 + object)`.

`fitType`, `sfType`, `betaPrior`, `modelMatrixType`, `useT`, `minmu`
: these seven parameters are inherited form `DESeq2::DESeq()`.
  - `fitType`, either "parametric", "local", "mean", or "glmGamPoi" for the type of fitting of dispersions to the mean intensity.
  - `sfType`, either "ratio", "poscounts", or "iterate" for the type of size factor estimation. We recommend to use "poscounts".
  - `betaPrior`, whether or not to put a zero-mean normal prior on the non-intercept coefficients.
  - `modelMatrixType`, either "standard" or "expanded", which describe how the model matrix,
  - `useT`, logical, where Wald statistics are assumed to follow a standard Normal.
  - `minmu`, lower bound on the estimated count for fitting gene-wise dispersion.

  For more details, see `DESeq2::DESeq()`. Most users will not need to set this arguments (just use the defaults).

`p_adjust`
: method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff`
: pvalue_cutoff numeric, p value cutoff, default 0.05.

`...`
: extra parameters passed to `DESeq2::DESeq()`.

Details

Note: DESeq2 requires the input is raw counts (un-normalized counts), as only the counts values allow assessing the measurement precision correctly. For more details see the vignette of DESeq2 (`vignette("DESeq2")`).

Thus, this function only supports "none", "rarefy", "RLE", "CSS", and "TMM" normalization methods. We strongly recommend using the "RLE" method (default normalization method in the DESeq2 package). The other normalization methods are used for expert users and comparisons among different normalization methods.

For two groups comparison, this function utilizes the Wald test (defined by `DESeq2::nbinomWaldTest()`) for hypothesis testing. A Wald test statistic is computed along with a probability (p-value) that a test statistic at least as extreme as the observed value were selected at random. `contrasts` are used to specify which two groups to compare. The order of the names determines the direction of fold change that is reported.

Likelihood ratio test (LRT) is used to identify the genes that significantly changed across all the different levels for multiple groups comparisons. The LRT identified the significant features by comparing the full model to the reduced model. It is testing whether a feature removed in the reduced model explains a significant variation in the data.

`contrast` must be a two length character or `NULL` (default). It is only required to set manually for two groups comparison when there are multiple groups. The order determines the direction of comparison, the first element is used to specify the reference group (control). This means that, the first element is the denominator for the fold change, and the second element is used as baseline (numerator for fold change). Otherwise, users do required to concern this parameter (set as default `NULL`), and if there are two groups, the first level of groups will set as the reference group; if there are multiple groups, it will perform an ANOVA-like testing to find markers which difference in any of the groups.

Value

a `microbiomeMarker` object.

References

Love, Michael I., Wolfgang Huber, and Simon Anders. "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." Genome biology 15.12 (2014): 1-21.

See Also

DESeq2::results(),DESeq2::DESeq()

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")) %>%
    phyloseq::subset_taxa(Phylum %in% c("Firmicutes"))
run_deseq2(ps, group = "Enterotype")
```

---

run_edger *Perform differential analysis using edgeR*

Description

Differential expression analysis based on the Negative Binomial distribution using **edgeR**.

Usage

```
run_edger(
  ps,
  group,
  confounders = character(0),
  contrast = NULL,
  taxa_rank = "all",
  method = c("LRT", "QLFT"),
  transform = c("identity", "log10", "log10p"),
  norm = "TMM",
  norm_para = list(),
  disp_para = list(),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  ...
)
```

Arguments

ps	ps a phyloseq::phyloseq object.
group	character, the variable to set the group, must be one of the var of the sample metadata.
confounders	character vector, the confounding variables to be adjusted. default character(0), indicating no confounding variable.
contrast	this parameter only used for two groups comparison while there are multiple groups. For more please see the following details.

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).

`method` character, used for differential analysis, please see details below for more info.

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`disp_para` additional arguments passed to `edgeR::estimateDisp()` used for dispersions estimation. Most users will not need to pass any additional arguments here.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05

`...` extra arguments passed to the model. See `edgeR::glmQLFit()` and `edgeR::glmFit()` for more details.

Details

Note that edgeR is designed to work with actual counts. This means that transformation is not required in any way before inputting them to edgeR.

There are two test methods for differential analysis in **edgeR**, likelihood ratio test (LRT) and quasi-likelihood F-test (QLFT). The QLFT method is recommended as it allows stricter error rate control by accounting for the uncertainty in dispersion estimation.

`contrast` must be a two length character or `NULL` (default). It is only required to set manually for two groups comparison when there are multiple groups. The order determines the direction of comparison, the first element is used to specify the reference group (control). This means that, the first element is the denominator for the fold change, and the second element is used as baseline (numerator for fold change). Otherwise, users do required to concern this parameter (set as default `NULL`), and if there are two groups, the first level of groups will set as the reference group; if there are multiple groups, it will perform an ANOVA-like testing to find markers which difference in any of the groups.

Value

a `microbiomeMarker` object.

Author(s)

Yang Cao

References

Robinson, Mark D., and Alicia Oshlack. "A scaling normalization method for differential expression analysis of RNA-seq data." Genome biology 11.3 (2010): 1-9.

Robinson, Mark D., Davis J. McCarthy, and Gordon K. Smyth. "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics 26.1 (2010): 139-140.

See Also

`edgeR::glmFit()`,`edgeR::glmQLFit()`,`edgeR::estimateDisp()` ,`normalize()`

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
run_edger(ps, group = "Enterotype")
```

---

run_lefse *Liner discriminant analysis (LDA) effect size (LEFSe) analysis*

Description

Perform Metagenomic LEFSe analysis based on phyloseq object.

Usage

```
run_lefse(
  ps,
  group,
  subgroup = NULL,
  taxa_rank = "all",
```

```
  transform = c("identity", "log10", "log10p"),
  norm = "CPM",
  norm_para = list(),
  kw_cutoff = 0.05,
  lda_cutoff = 2,
  bootstrap_n = 30,
  bootstrap_fraction = 2/3,
  wilcoxon_cutoff = 0.05,
  multigrp_strat = FALSE,
  strict = c("0", "1", "2"),
  sample_min = 10,
  only_same_subgrp = FALSE,
  curv = FALSE
)
```

Arguments

ps	a phyloseq-class object
group	character, the column name to set the group
subgroup	character, the column name to set the subgroup
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of phyloseq::rank_names(phyloseq), or "all" means to summarize the taxa by the top taxa ranks (summarize_taxa(ps, level = rank_names(ps)[1])), or "none" means perform differential analysis on the original taxa (taxa_names(phyloseq), e.g., OTU or ASV).
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation (default). • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include: • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size. • "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference. • "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference. • "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold. • "CLR": centered log-ratio normalization.

- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` named `list`. other arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`kw_cutoff` numeric, p value cutoff of kw test, default 0.05

`lda_cutoff` numeric, lda score cutoff, default 2

`bootstrap_n` integer, the number of bootstrap iteration for LDA, default 30

`bootstrap_fraction` numeric, the subsampling fraction value for each bootstrap iteration, default `2/3`

`wilcoxon_cutoff` numeric, p value cutoff of wilcoxon test, default 0.05

`multigrp_strat` logical, for multiple group tasks, whether the test is performed in a one-against one (more strict) or in a one-against all setting, default `FALSE`.

`strict` multiple testing options, 0 for no correction (default), 1 for independent comparisons, 2 for independent comparison.

`sample_min` integer, minimum number of samples per subclass for performing wilcoxon test, default 10

`only_same_subgrp` logical, whether perform the wilcoxon test only among the subgroups with the same name, default `FALSE`

`curv` logical, whether perform the wilcoxon test using the Curtis's approach, defalt `FALSE`

Value

a microbiomeMarker object, in which the `slot` of `marker_table` contains four variables:

- `feature`, significantly different features.
- `enrich_group`, the class of the differential features enriched.
- `lda`, logarithmic LDA score (effect size)
- `pvalue`, p value of kw test.

Author(s)

Yang Cao

References

Segata, Nicola, et al. Metagenomic biomarker discovery and explanation. Genome biology 12.6 (2011): R60.

See Also

normalize

Examples

```
data(kostic_crc)
kostic_crc_small <- phyloseq::subset_taxa(
    kostic_crc,
    Phylum == "Firmicutes"
)
mm_lefse <- run_lefse(
    kostic_crc_small,
    wilcoxon_cutoff = 0.01,
    group = "DIAGNOSIS",
    kw_cutoff = 0.01,
    multigrp_strat = TRUE,
    lda_cutoff = 4
)
```

---

run_limma_voom *Differential analysis using limma-voom*

Description

Differential analysis using limma-voom

Usage

```
run_limma_voom(
  ps,
  group,
  confounders = character(0),
  contrast = NULL,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "none",
  norm_para = list(),
  voom_span = 0.5,
  p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  ...
)
```

Arguments

ps	ps a phyloseq::phyloseq object.
group	character, the variable to set the group, must be one of the var of the sample metadata.
confounders	character vector, the confounding variables to be adjusted. default character(0), indicating no confounding variable.
contrast	this parameter only used for two groups comparison while there are multiple groups. For more please see the following details.

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.

`norm_para` arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`voom_span` width of the smoothing window used for the lowess mean-variance trend for `limma::voom()`. Expressed as a proportion between 0 and 1.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` cutoff of p value, default 0.05.

`...` extra arguments passed to `limma::eBayes()`.

Details

`contrast` must be a two length character or `NULL` (default). It is only required to set manually for two groups comparison when there are multiple groups. The order determines the direction of comparison, the first element is used to specify the reference group (control). This means that, the first element is the denominator for the fold change, and the second element is used as baseline (numerator for fold change). Otherwise, users do required to concern this parameter (set as default `NULL`), and if there are two groups, the first level of groups will set as the reference group; if there are multiple groups, it will perform an ANOVA-like testing to find markers which difference in any of the groups.

Value

a `microbiomeMarker` object.

References

Law, C. W., Chen, Y., Shi, W., & Smyth, G. K. (2014). voom: Precision weights unlock linear model analysis tools for RNA-seq read counts. Genome biology, 15(2), 1-17.

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.01,
    p_adjust = "none"
)
mm
```

`run_marker` *Find makers (differentially expressed metagenomic features)*

Description

`run_marker` is a wrapper of all differential analysis functions.

Usage

```
run_marker(
  ps,
  group,
  da_method = c("lefse", "simple_t", "simple_welch", "simple_white", "simple_kruskal",
    "simple_anova", "edger", "deseq2", "metagenomeseq", "ancom", "ancombc", "aldex",
    "limma_voom", "sl_lr", "sl_rf", "sl_svm"),
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "none",
  norm_para = list(),
  p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  ...
)
```

Arguments

ps	a `phyloseq::phyloseq` object
group	character, the variable to set the group
da_method	character to specify the differential analysis method. The options include: • "lefse", linear discriminant analysis (LDA) effect size (LEfSe) method, for more details see `run_lefse()`. • "simple_t", "simple_welch", "simple_white", "simple_kruskal", and "simple_anova", simple statistic methods; "simple_t", "simple_welch" and "simple_white" for two groups comparison; "simple_kruskal", and "simple_anova" for multiple groups comparison. For more details see `run_simple_stat()`.

- "edger", see `run_edger()`.
- "deseq2", see `run_deseq2()`.
- "metagenomeseq", differential expression analysis based on the Zero-inflated Log-Normal mixture model or Zero-inflated Gaussian mixture model using metagenomeSeq, see `run_metagenomeseq()`.
- "ancom", see `run_ancom()`.
- "ancombc", differential analysis of compositions of microbiomes with bias correction, see `run_ancombc()`.
- "aldex", see `run_aldex()`.
- "limma_voom", see `run_limma_voom()`.
- "sl_lr", "sl_rf", and "sl_svm", there supervised leaning (SL) methods: logistic regression (lr), random forest (rf), or support vector machine (svm). For more details see `run_sl()`.

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05.

`...` extra arguments passed to the corresponding differential analysis functions, e.g. `run_lefse()`.

Details

This function is only a wrapper of all differential analysis functions, We recommend to use the corresponding function, since it has a better default arguments setting.

Value

a `microbiomeMarker` object.

See Also

`run_lefse()`,`run_simple_stat()`,`run_test_two_groups()`, `run_test_multiple_groups()`,`run_edger()`,`run_des`
`run_metagenomeseq`,`run_ancom()`,`run_ancombc()`,`run_aldex()`, `run_limma_voom()`,`run_sl()`

`run_metagenomeseq` *metagenomeSeq differential analysis*

Description

Differential expression analysis based on the Zero-inflated Log-Normal mixture model or Zero-inflated Gaussian mixture model using metagenomeSeq.

Usage

```
run_metagenomeseq(
  ps,
  group,
  confounders = character(0),
  contrast = NULL,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "CSS",
  norm_para = list(),
  method = c("ZILN", "ZIG"),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  ...
)
```

Arguments

`ps`	ps a `phyloseq::phyloseq` object.
`group`	character, the variable to set the group, must be one of the var of the sample metadata.
`confounders`	character vector, the confounding variables to be adjusted. default `character(0)`, indicating no confounding variable.
`contrast`	this parameter only used for two groups comparison while there are multiple groups. For more please see the following details.

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(ps)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(ps)`, e.g., OTU or ASV).

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods.

`method` character, which model used for differential analysis, "ZILN" (Zero-inflated Log-Normal mixture model)" or "ZIG" (Zero-inflated Gaussian mixture model). And the zero-inflated log-normal model is preferred due to the high sensitivity and low FDR.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05

`...` extra arguments passed to the model. more details see `metagenomeSeq::fitFeatureModel()` and `metagenomeSeq::fitZig()`, e.g. `control` (can be setted using `metagenomeSeq::zigControl()` for `metagenomeSeq::fitZig()`.

Details

metagnomeSeq provides two differential analysis methods, zero-inflated log-normal mixture model (implemented in `metagenomeSeq::fitFeatureModel()`) and zero-inflated Gaussian mixture model (implemented in `metagenomeSeq::fitZig()`). We recommend fitFeatureModel over fitZig due to high sensitivity and low FDR. Both `metagenomeSeq::fitFeatureModel()` and `metagenomeSeq::fitZig()` require the abundance profiles before normalization.

For `metagenomeSeq::fitZig()`, the output column is the coefficient of interest, and logFC column in the output of `metagenomeSeq::fitFeatureModel()` is analogous to coefficient. Thus, logFC is really just the estimate the coefficient of interest in `metagenomeSeq::fitFeatureModel()`. For more details see these question Difference between fitFeatureModel and fitZIG in metagenomeSeq.

`contrast` must be a two length character or `NULL` (default). It is only required to set manually for two groups comparison when there are multiple groups. The order determines the direction of comparison, the first element is used to specify the reference group (control). This means that, the first element is the denominator for the fold change, and the second element is used as baseline (numerator for fold change). Otherwise, users do required to concern this paramerter (set as default `NULL`), and if there are two groups, the first level of groups will set as the reference group; if there are multiple groups, it will perform an ANOVA-like testing to find markers which difference in any of the groups.

Of note, `metagenomeSeq::fitFeatureModel()` is not allows for multiple groups comparison.

Value

a `microbiomeMarker` object.

Author(s)

Yang Cao

References

Paulson, Joseph N., et al. "Differential abundance analysis for microbial marker-gene surveys." Nature methods 10.12 (2013): 1200-1202.

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
run_metagenomeseq(ps, group = "Enterotype")
```

---

run_posthoc_test *Post hoc pairwise comparisons for multiple groups test.*

Description

Multiple group test, such as anova and Kruskal-Wallis rank sum test, can be used to uncover the significant feature among all groups. Post hoc tests are used to uncover specific mean differences between pair of groups.

Usage

```
run_posthoc_test(
  ps,
  group,
  transform = c("identity", "log10", "log10p"),
  norm = "TSS",
```

```
  norm_para = list(),
  conf_level = 0.95,
  method = c("tukey", "games_howell", "scheffe", "welch_uncorrected")
)
```

Arguments

ps	a phyloseq::phyloseq object
group	character, the variable to set the group
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation (default). • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include: • a integer, e.g. 1e6 (default), indicating pre-sample normalization of the sum of the values to 1e6. • "none": do not normalize. • "rarefy": random subsampling counts to the smallest library size in the data set. • "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size. • "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference. • "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference. • "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold. • "CLR": centered log-ratio normalization.
norm_para	arguments passed to specific normalization methods
conf_level	confidence level, default 0.95
method	one of "tukey", "games_howell", "scheffe", "welch_uncorrected", defining the method for the pairwise comparisons. See details for more information.

Value

a postHocTest object

See Also

postHocTest, run_test_multiple_groups()

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2", "Enterotype 1")
) %>%
    phyloseq::subset_taxa(Phylum == "Bacteroidetes")
pht <- run_posthoc_test(ps, group = "Enterotype")
pht
```

---

run_simple_stat *Simple statistical analysis of metagenomic profiles*

Description

Perform simple statistical analysis of metagenomic profiles. This function is a wrapper of `run_test_two_groups` and `run_test_multiple_groups`.

Usage

```
run_simple_stat(
  ps,
  group,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "TSS",
  norm_para = list(),
  method = c("welch.test", "t.test", "white.test", "anova", "kruskal"),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  diff_mean_cutoff = NULL,
  ratio_cutoff = NULL,
  eta_squared_cutoff = NULL,
  conf_level = 0.95,
  nperm = 1000,
  ...
)
```

Arguments

ps	a `phyloseq::phyloseq` object
group	character, the variable to set the group
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).
transform	character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include: • "identity", return the original data without any transformation (default).

- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods

`method` test method, options include: "welch.test", "t.test" and "white.test" for two groups comparison, "anova"and "kruskal" for multiple groups comparison.

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05

`diff_mean_cutoff`, `ratio_cutoff`
only used for two groups comparison, cutoff of different means and ratios, default `NULL` which means no effect size filter.

`eta_squared_cutoff`
only used for multiple groups comparison, numeric, cutoff of effect size (eta squared) default `NULL` which means no effect size filter.

`conf_level` only used for two groups comparison, numeric, confidence level of interval.

`nperm` integer, only used for two groups comparison, number of permutations for white non parametric t test estimation

`...` only used for two groups comparison, extra arguments passed to `t.test()` or `fisher.test()`.

Value

a `microbiomeMarker` object.

See Also

`run_test_two_groups()`,`run_test_multiple_groups()`

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2")
)
run_simple_stat(ps, group = "Enterotype")
```

---

run_sl *Identify biomarkers using supervised leaning (SL) methods*

Description

Identify biomarkers using logistic regression, random forest, or support vector machine.

Usage

```
run_sl(
  ps,
  group,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "none",
  norm_para = list(),
  nfolds = 3,
  nrepeats = 3,
  sampling = NULL,
  tune_length = 5,
  top_n = 10,
  method = c("LR", "RF", "SVM"),
  ...
)
```

Arguments

ps	a phyloseq-class object.
group	character, the variable to set the group.
taxa_rank	character to specify taxonomic rank to perform differential analysis on. Should be one of phyloseq::rank_names(phyloseq), or "all" means to summarize the taxa by the top taxa ranks (summarize_taxa(ps, level = rank_names(ps)[1])), or "none" means perform differential analysis on the original taxa (taxa_names(phyloseq), e.g., OTU or ASV).
transform	character, the methods used to transform the microbial abundance. See transform_abundances() for more details. The options include: • "identity", return the original data without any transformation (default). • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).
norm	the methods used to normalize the microbial abundance data. See normalize() for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` named `list`. other arguments passed to specific normalization methods. Most users will not need to pass any additional arguments here.

`nfolds` the number of splits in CV.

`nrepeats` the number of complete sets of folds to compute.

`sampling` a single character value describing the type of additional sampling that is conducted after resampling (usually to resolve class imbalances). Values are "none", "down", "up", "smote", or "rose". For more details see `caret::trainControl()`.

`tune_length` an integer denoting the amount of granularity in the tuning parameter grid. For more details see `caret::train()`.

`top_n` an integer denoting the top n features as the biomarker according the importance score.

`method` supervised learning method, options are "LR" (logistic regression), "RF" (rando forest), or "SVM" (support vector machine).

`...` extra arguments passed to the classification. e.g., `importance` for `randomForest::randomForest`.

Details

Only support two groups comparison in the current version. And the marker was selected based on its importance score. Moreover, The hyper-parameters are selected automatically by a grid-search based method in the N-time K-fold cross-validation. Thus, the identified biomarker based can be biased due to model overfitting for small datasets (e.g., with less than 100 samples).

The argument `top_n` is used to denote the number of markers based on the importance score. There is no rule or principle on how to select `top_n`, however, usually it is very useful to try a different `top_n` and compare the performance of the marker predictions for the testing data.

Value

a microbiomeMarker object.

Author(s)

Yang Cao

See Also

caret::train(),caret::trainControl()

Examples

```
data(enterotypes_arumugam)
# small example phyloseq object for test
ps_small <- phyloseq::subset_taxa(
    enterotypes_arumugam,
    Phylum %in% c("Firmicutes", "Bacteroidetes")
)

set.seed(2021)
mm <- run_sl(
    ps_small,
    group = "Gender",
    taxa_rank = "Genus",
    nfolds = 2,
    nrepeats = 1,
    top_n = 15,
    norm = "TSS",
    method = "LR",
)
mm
```

---

run_test_multiple_groups

Statistical test for multiple groups

Description

Statistical test for multiple groups

Usage

```
run_test_multiple_groups(
  ps,
  group,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "TSS",
  norm_para = list(),
  method = c("anova", "kruskal"),
  p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  effect_size_cutoff = NULL
)
```

Arguments

ps	a phyloseq::phyloseq object
group	character, the variable to set the group

`taxa_rank` character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).

`transform` character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include:

- "identity", return the original data without any transformation (default).
- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods

`method` test method, must be one of "anova" or "kruskal"

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05.

`effect_size_cutoff` numeric, cutoff of effect size default `NULL` which means no effect size filter. The eta squared is used to measure the effect size for anova/kruskal test.

Value

a `microbiomeMarker` object.

See Also

`run_posthoc_test()`,`run_test_two_groups()`,`run_simple_stat()`

Examples

```
data(enterotypes_arumugam)
ps <- phyloseq::subset_samples(
    enterotypes_arumugam,
    Enterotype %in% c("Enterotype 3", "Enterotype 2", "Enterotype 1")
)
mm_anova <- run_test_multiple_groups(
    ps,
    group = "Enterotype",
    method = "anova"
)
```

`run_test_two_groups` *Statistical test between two groups*

Description

Statistical test between two groups

Usage

```
run_test_two_groups(
  ps,
  group,
  taxa_rank = "all",
  transform = c("identity", "log10", "log10p"),
  norm = "TSS",
  norm_para = list(),
  method = c("welch.test", "t.test", "white.test"),
 p_adjust = c("none", "fdr", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY"),
  pvalue_cutoff = 0.05,
  diff_mean_cutoff = NULL,
  ratio_cutoff = NULL,
  conf_level = 0.95,
  nperm = 1000,
  ...
)
```

Arguments

`ps`	a `phyloseq::phyloseq` object
`group`	character, the variable to set the group
`taxa_rank`	character to specify taxonomic rank to perform differential analysis on. Should be one of `phyloseq::rank_names(phyloseq)`, or "all" means to summarize the taxa by the top taxa ranks (`summarize_taxa(ps, level = rank_names(ps)[1])`), or "none" means perform differential analysis on the original taxa (`taxa_names(phyloseq)`, e.g., OTU or ASV).
`transform`	character, the methods used to transform the microbial abundance. See `transform_abundances()` for more details. The options include: • "identity", return the original data without any transformation (default).

- "log10", the transformation is `log10(object)`, and if the data contains zeros the transformation is `log10(1 + object)`.
- "log10p", the transformation is `log10(1 + object)`.

`norm` the methods used to normalize the microbial abundance data. See `normalize()` for more details. Options include:

- "none": do not normalize.
- "rarefy": random subsampling counts to the smallest library size in the data set.
- "TSS": total sum scaling, also referred to as "relative abundance", the abundances were normalized by dividing the corresponding sample library size.
- "TMM": trimmed mean of m-values. First, a sample is chosen as reference. The scaling factor is then derived using a weighted trimmed mean over the differences of the log-transformed gene-count fold-change between the sample and the reference.
- "RLE", relative log expression, RLE uses a pseudo-reference calculated using the geometric mean of the gene-specific abundances over all samples. The scaling factors are then calculated as the median of the gene counts ratios between the samples and the reference.
- "CSS": cumulative sum scaling, calculates scaling factors as the cumulative sum of gene abundances up to a data-derived threshold.
- "CLR": centered log-ratio normalization.
- "CPM": pre-sample normalization of the sum of the values to 1e+06.

`norm_para` arguments passed to specific normalization methods

`method` test method, must be one of "welch.test", "t.test" or "white.test"

`p_adjust` method for multiple test correction, default `none`, for more details see stats::p.adjust.

`pvalue_cutoff` numeric, p value cutoff, default 0.05

`diff_mean_cutoff`, `ratio_cutoff` cutoff of different means and ratios, default `NULL` which means no effect size filter.

`conf_level` numeric, confidence level of interval.

`nperm` integer, number of permutations for white non parametric t test estimation

`...` extra arguments passed to `t.test()` or `fisher.test()`

Value

a `microbiomeMarker` object.

Author(s)

Yang Cao

See Also

`run_test_multiple_groups()`,`run_simple_stat`

Examples

```
data(enterotypes_arumugam)
mm_welch <- run_test_two_groups(
    enterotypes_arumugam,
    group = "Gender",
    method = "welch.test"
)
mm_welch
```

`subset_marker` *Subset microbiome markers*

Description

Subset markers based on an expression related to the columns and values within the `marker_table` slot of `mm`.

Usage

```
subset_marker(mm, ...)
```

Arguments

`mm`	a `microbiomeMarker` or `marker_table` object.
`...`	the subsetting expression passed to `base::subset()`.

Value

a subset object in the same class with `mm`.

Examples

```
data(enterotypes_arumugam)
mm <- run_limma_voom(
    enterotypes_arumugam,
    "Enterotype",
    contrast = c("Enterotype 3", "Enterotype 2"),
    pvalue_cutoff = 0.01,
    p_adjust = "none"
)
subset_marker(mm, pvalue < 0.005)
```

summarize_taxa *Summarize taxa into a taxonomic level within each sample*

Description

Provides summary information of the representation of a taxonomic levels within each sample.

Usage

```
summarize_taxa(ps, level = rank_names(ps)[1], absolute = TRUE, sep = "|")
```

Arguments

ps	a phyloseq-class object.
level	taxonomic level to summarize, default the top level rank of the ps.
absolute	logical, whether return the absolute abundance or relative abundance, default FALSE.
sep	a character string to separate the taxonomic levels.

Value

a phyloseq::phyloseq object, where each row represents a taxa, and each col represents the taxa abundance of each sample.

Examples

```
data(enterotypes_arumugam)
summarize_taxa(enterotypes_arumugam)
```

summary.compareDA *Summary differential analysis methods comparison results*

Description

Summary differential analysis methods comparison results

Usage

```
## S3 method for class 'compareDA'
summary(
  object,
  sort = c("score", "auc", "fpr", "power"),
  boot = TRUE,
  boot_n = 1000L,
  prob = c(0.05, 0.95),
  ...
)
```

Arguments

object	an compareDA object, output from compare_DA().
sort	character string specifying sort method. Possibilities are "score" which is calculated as $(auc - 0.5) * power - fdr$, "auc" for area under the ROC curve, "fpr" for false positive rate, "power" for empirical power.
boot	logical, whether use bootstrap for confidence limites of the score, default TRUE. Recommended to be TRUE unless n_rep is larger then 100 in compare_DA().
boot_n	integer, number of bootstraps, default 1000L.
prob	two length numeric vector, confidence limits for score, default c(0.05, 0.95).
...	extra arguments affecting the summary produced.

Value

a data.frame containing measurements for differential analysis methods:

- call: differential analysis commands.
- auc: area under curve of ROC.
- fpr: false positive rate
- power: empirical power.
- fdr: false discover7y rate.
- score: score whch is calculated as $(auc - 0.5) * power - fdr$.
- score_*: confidence limits of score.

---

transform_abundances *Transform the taxa abundances in* otu_table *sample by sample*

Description

Transform the taxa abundances in otu_table sample by sample, which means the counts of each sample will be transformed individually.

Usage

```
transform_abundances(object, transform = c("identity", "log10", "log10p"))
```

Arguments

object	otu_table, phyloseq, or microbiomeMarker.
transform	transformation to apply, the options includle: • "identity", return the original data without any transformation. • "log10", the transformation is log10(object), and if the data contains zeros the transformation is log10(1 + object). • "log10p", the transformation is log10(1 + object).

Value

A object matches the class of argument object with the transformed otu_table.

See Also

abundances()

Examples

```
data(oxygen)
x1 <- transform_abundances(oxygen)
head(otu_table(x1), 10)
x2 <- transform_abundances(oxygen, "log10")
head(otu_table(x2), 10)
x3 <- transform_abundances(oxygen, "log10p")
head(otu_table(x3), 10)
```

[*Extract* marker_table *object*

Description

Operators acting on marker_table to extract parts.

Usage

```
## S4 method for signature 'marker_table,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

x	a marker_table object.
i, j	elements to extract.
...	see base::Extract().
drop	ignored now.

Value

a marker_table object.

See Also

base::Extract()

Index