Package 'GUniFrac'

January 20, 2025

Type Package

Title Generalized UniFrac Distances, Distance-Based Multivariate Methods and Feature-Based Univariate Methods for Microbiome Data Analysis

Version 1.8

Date 2023-09-13

Author Jun Chen, Xianyang Zhang, Lu Yang, Lujun Zhang

Maintainer Jun Chen <chen.jun2@mayo.edu>

Description A suite of methods for powerful and robust microbiome data analysis including data normalization, data simulation, community-level association testing and differential abundance analysis. It implements generalized UniFrac distances, Geometric Mean of Pairwise Ratios (GMPR) normalization, semiparametric data simulator, distance-based statistical methods, and feature-based statistical methods. The distance-based statistical methods include three extensions of PERMANOVA: (1) PERMANOVA using the Freedman-Lane permutation scheme, (2) PERMANOVA omnibus test using multiple matrices, and (3) analytical approach to approximating PERMANOVA p-value. Feature-based statistical methods include linear model-based methods for differential abundance analysis of zero-inflated high-dimensional compositional data.

Depends R (>= 3.5.0)

Suggests ade4, knitr, markdown, ggpubr

Imports Rcpp (>= 0.12.13), vegan, ggplot2, matrixStats, Matrix, ape, parallel, stats, utils, statmod, rmutil, dirmult, MASS, ggrepel, foreach, modeest, inline, methods

LinkingTo Rcpp

NeedsCompilation yes

VignetteBuilder knitr

License GPL-3

Encoding UTF-8

Repository CRAN

Date/Publication 2023-09-14 00:02:32 UTC

Contents

`adonis3` *Permutational Multivariate Analysis of Variance Using Distance Matrices (Freedman-Lane permutation)*

Description

Analysis of variance using distance matrices — for partitioning distance matrices among sources of variation and fitting linear models (e.g., factors, polynomial regression) to distance matrices; uses a permutation test (Freedman-Lane permutation) with pseudo-F ratios.

Usage

```
adonis3(formula, data, permutations = 999, method = "bray",
    strata = NULL, contr.unordered = "contr.sum",
    contr.ordered = "contr.poly", parallel = getOption("mc.cores"), ...)
```

Arguments

`formula`	model formula. The LHS must be either a community data matrix or a dissimilarity matrix, e.g., from `vegdist` or `dist`. If the LHS is a data matrix, function `vegdist` will be used to find the dissimilarities. The RHS defines the independent variables. These can be continuous variables or factors, they can be transformed within the formula, and they can have interactions as in a typical `formula`.

`data`	the data frame for the independent variables.
`permutations`	a list of control values for the permutations as returned by the function `how`, or the number of permutations required, or a permutation matrix where each row gives the permuted indices.
`method`	the name of any method used in `vegdist` to calculate pairwise distances if the left hand side of the `formula` was a data frame or a matrix.
`strata`	groups (strata) within which to constrain permutations.
`contr.unordered, contr.ordered`	contrasts used for the design matrix (default in R is dummy or treatment contrasts for unordered factors).
`parallel`	number of parallel processes or a predefined socket cluster. With `parallel = 1` uses ordinary, non-parallel processing. The parallel processing is done with **parallel** package.
`...`	Other arguments passed to `vegdist`.

Details

`adonis3` is the re-implementation of the `adonis` function in the `vegan` package based on the Freedman-Lane permutation scheme (Freedman & Lane (1983), Hu & Satten (2020)). The original implementation in the `vegan` package is directly based on the algorithm of Anderson (2001) and performs a sequential test of terms. Statistical significance is assessed based on permuting the distance matrix. We found that such permutation will lead to power loss in testing the effect of a covariate of interest while adjusting for other covariates (confounders). The power loss is more evident when the confounders' effects are strong, the correlation between the covariate of interest and the confounders is high, and the sample size is small. When the sample size is large than 100, the difference is usually small. The new implementation is revised on the `adonis` function with the same interface.

Value

Function `adonis3` returns an object of class `"adonis"` with following components:

`aov.tab`	typical AOV table showing sources of variation, degrees of freedom, sequential sums of squares, mean squares, F statistics, partial R^2 and P values, based on N permutations.
`coefficients`	matrix of coefficients of the linear model, with rows representing sources of variation and columns representing species; each column represents a fit of a species abundance to the linear model. These are what you get when you fit one species to your predictors. These are NOT available if you supply the distance matrix in the formula, rather than the site x species matrix
`coef.sites`	matrix of coefficients of the linear model, with rows representing sources of variation and columns representing sites; each column represents a fit of a sites distances (from all other sites) to the linear model. These are what you get when you fit distances of one site to your predictors.
`f.perms`	an N by m matrix of the null F statistics for each source of variation based on N permutations of the data. The permutations can be inspected with `permustats` and its support functions.

model.matrix	the model.matrix for the right hand side of the formula.
terms	the terms component of the model.

Author(s)

Martin Henry H. Stevens (adonis) and Jun Chen (adonis3).

References

Anderson, M.J. 2001. A new method for non-parametric multivariate analysis of variance. *Austral Ecology*, **26**: 32–46.

Freedman D. & Lane D. 1983. A nonstochastic interpretation of reported significance levels. *Journal of Business and Economic Statistics*, **1**292–298.

Hu, Y. J. & Satten, G. A. 2020. Testing hypotheses about the microbiome using the linear decomposition model (LDM). *JBioinformatics*, **36(14)** : 4106-4115.

Examples

```
## Not run:
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

groups <- throat.meta$SmokingStatus

# Rarefaction
otu.tab.rff <- Rarefy(throat.otu.tab)$otu.tab.rff

# Calculate the UniFrac distance
unifracs <- GUniFrac(otu.tab.rff, throat.tree, alpha=c(0, 0.5, 1))$unifracs

# Test the smoking effect based on unweighted UniFrac distance, adjusting sex
adonis3(as.dist(unifracs[, , 'd_UW']) ~ Sex + SmokingStatus, data = throat.meta)

## End(Not run)
```

---

dICC *Distance-based Intra-Class Correlation Coefficient*

Description

Distance-based Intra-Class Correlation Coefficient (ICC) is an extension of the traditional univariate ICC to multivariate case, where the relationship between the multivariate measurements is summarized in a distance matrix. It quantifies the ability of a measurement method in reproducing the inter-sample relationship.

Usage

```
dICC(dist.mat, strata)
```

Arguments

dist.mat	a symmetrical distance matrix between all the replicates (technical and biological replicates).
strata	a factor with each level corresponding to a biological replicate.

Value

Function dICC returns a list with the following component:

ICC	the distance-based ICC value.

Author(s)

Jun Chen and Xianyang Zhang

References

Chen, J. & Zhang, X. 2022. dICC: Distance-based Intraclass Correlation Coefficient for Metagenomic Reproducibility Studies. submitted.

See Also

dICC.SE.asympt, dICC.SE.bt

Examples

```
# Generate the error-free measurements of 20 biological replicates, each with four dimensions
y <- matrix(rnorm(80), nrow = 20, ncol = 4)

# Generate two technical replicates for each biological replicate and add measurement errors
y1 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
y2 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
y12 <- rbind(y1, y2)

# Create the design vector
strata <- factor(rep(1 : 20, 2))

# Calculate the distance and distance-based ICC
dist.mat <- as.matrix(dist(y12))
dICC(dist.mat, strata)
```

`dICC.SE.asympt` *Asymptotic Standard Error of Distance-based Intra-Class Correlation Coefficient*

Description

Calculate the standard error of the distance-based intra-class correlation coefficient based on the asymptotic distribution.

Usage

```
dICC.SE.asympt(dist.mat, strata)
```

Arguments

`dist.mat`	a symmetrical distance matrix between all the replicates (technical and biological replicates).
`strata`	a factor with each level corresponding to a biological replicate. Currently only supports the same number of technical replicates for each biological replicate.

Value

Function `dICC` returns a list with the following component:

`ICC`	the distance-based ICC value.
`SE`	the standard error.

Author(s)

Jun Chen and Xianyang Zhang

References

Chen, J. & Zhang, X. 2022. dICC: Distance-based Intraclass Correlation Coefficient for Metagenomic Reproducibility Studies. submitted.

See Also

`dICC`, `dICC.SE.bt`

Examples

```
# Generate the error-free measurements of 20 biological replicates, each with four dimensions
y <- matrix(rnorm(80), nrow = 20, ncol = 4)

# Generate two technical replicates for each biological replicate and add measurement errors
y1 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
```

```
y2 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
y12 <- rbind(y1, y2)

# Create the design vector
strata <- factor(rep(1 : 20, 2))

# Calculate the distance and distance-based ICC
dist.mat <- as.matrix(dist(y12))
dICC.SE.asympt(dist.mat, strata)
```

`dICC.SE.bt` *Bootstrap Standard Error of Distance-based Intra-Class Correlation Coefficient*

Description

Calculate the standard error of the distance-based intra-class correlation coefficient based on bootstrapping. Biological replicates are sampled by replacement. More conservative than the asymptotic approach.

Usage

```
dICC.SE.bt(dist.mat, strata, B = 199)
```

Arguments

`dist.mat`	a symmetrical distance matrix between all the replicates (technical and biological replicates).
`strata`	a factor with each level corresponding to a biological replicate. Supports an unequal number of technical replicates for each biological replicate.
`B`	integer, the number of bootstrap iterations.

Value

Function `dICC` returns a list with the following component:

`ICC`	the distance-based ICC value.
`SE`	the standard error.

Author(s)

Jun Chen and Xianyang Zhang

References

Chen, J. & Zhang, X. 2022. dICC: Distance-based Intraclass Correlation Coefficient for Metagenomic Reproducibility Studies. submitted.

See Also

dICC, dICC.SE.asympt

Examples

```
# Generate the error-free measurements of 20 biological replicates, each with four dimensions
y <- matrix(rnorm(80), nrow = 20, ncol = 4)

# Generate two technical replicates for each biological replicate and add measurement errors
y1 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
y2 <- y + matrix(rnorm(80, sd = 0.5), nrow = 20, ncol = 4)
y12 <- rbind(y1, y2)

# Create the design vector
strata <- factor(rep(1 : 20, 2))

# Calculate the distance and distance-based ICC
dist.mat <- as.matrix(dist(y12))
dICC.SE.bt(dist.mat, strata)
```

---

dmanova *Distance-based Multivariate Analysis of Variance (Analytical P-value Calculation)*

Description

Analysis of variance using distance matrices — for partitioning distance matrices among sources of variation and fitting linear models (e.g., factors, polynomial regression) to distance matrices; calculate the analytical p-value based on pseudo-F statistic without permutation.

Usage

```
dmanova(formula, data = NULL, positify = FALSE,
contr.unordered = "contr.sum", contr.ordered = "contr.poly",
returnG = FALSE)
```

Arguments

formula	model formula. The LHS must be a dissimilarity matrix (either class matrix or class dist, e.g., from vegdist or dist. The RHS defines the independent variables. These can be continuous variables or factors, they can be transformed within the formula, and they can have interactions as in a typical formula.
data	the data frame for the independent variables.

`positify` a logical value indicating whether to make the Gower's matrix positive definite using the `nearPD` function in `Matrix` package. This is equivalent to modifying the distance matrix so that it has an Euclidean embedding.

`contr.unordered, contr.ordered`
contrasts used for the design matrix (default in R is dummy or treatment contrasts for unordered factors).

`returnG` a logical value indicating whether the Gower's matrix should be returned.

Details

`dmanova` is a permutation-free method for approximating the p-value from distance-based permutational multivariate analysis of variance (PERMANOVA). PERMANOVA is slow when the sample size is large. In contrast, `dmanova` provides an analytical solution, which is several orders of magnitude faster for large sample sizes. The covariate of interest should be put as the last term in `formula` while the variables to be adjusted are put before the covariate of interest.

Value

Function `dmanova` returns a list with the following components:

`aov.tab` typical AOV table showing sources of variation, degrees of freedom, sums of squares, mean squares, F statistics, partial R^2 and P values.

`df` degree of freedom for the Chisquared distribution.

`G` The Gower's matrix if `returnG` is true.

`call` the call made

Author(s)

Jun Chen and Xianyang Zhang

References

Chen, J. & Zhang, X. 2021. D-MANOVA: fast distance-based multivariate analysis of variance for large-scale microbiome association studies. Bioinformatics. https://doi.org/10.1093/bioinformatics/btab498

See Also

`adonis3`

Examples

```
## Not run:
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

groups <- throat.meta$SmokingStatus

# Rarefaction
```

```
otu.tab.rff <- Rarefy(throat.otu.tab)$otu.tab.rff

# Calculate the UniFrac distance
unifracs <- GUniFrac(otu.tab.rff, throat.tree, alpha=c(0, 0.5, 1))$unifracs

# Test the smoking effect based on unweighted UniFrac distance, adjusting sex
# 'Sex' should be put before 'SmokingStatus'
dmanova(as.dist(unifracs[, , 'd_UW']) ~ Sex + SmokingStatus, data = throat.meta)

## End(Not run)
```

GMPR *Geometric Mean of Pairwise Ratios (GMPR) Normalization for Zero-inflated Count Data*

Description

A robust normalization method for zero-inflated count data such as microbiome sequencing data.

Usage

```
GMPR(OTUmatrix, min_ct = 2, intersect_no = 4)
```

Arguments

`OTUmatrix`	An OTU count table, where OTUs are arranged in rows and samples in columns.
`min_ct`	The minimal number of OTU counts. Only those OTU pairs with at least `min_ct` counts are considered in the ratio calculation. The default is 2.
`intersect_no`	The minimal number of shared OTUs between samples. Only those sample pairs sharing at least `intersect_no` OTUs are considered in geometric mean calculation. The default is 4.

Details

Normalization is a critical step in microbiome sequencing data analysis to account for variable library sizes. Microbiome data contains a vast number of zeros, which makes the traditional RNA-Seq normalization methods unstable. The proposed GMPR normalization remedies this problem by switching the two steps in DESeq2 normalization:

First, to calculate rij, the median count ratio of nonzero counts between samples: rij=median(cki/ckj) (k in 1:OTU_number and cki, ckj is the non-zero count of the kth OTU)

Second, to calculate the size factor si for a given sample i: si=geometric_mean(rij)

Value

A vector of GMPR size factor for each sample.

Author(s)

Jun Chen and Lujun Zhang

References

Li Chen, James Reeve, Lujun Zhang, Shenbing Huang, and Jun Chen. 2018. GMPR: A robust normalization method for zero-inflated count data with application to microbiome sequencing data. PeerJ, 6, e4600.

Examples

```
data(throat.otu.tab)
size.factor <- GMPR(t(throat.otu.tab))
```

GUniFrac *Generalized UniFrac distances for comparing microbial communities.*

Description

A generalized version of commonly used UniFrac distances. It is defined as:

$$d^{(\alpha)} = \frac{\sum_{i=1}^{m} b_i (p_i^A + p_i^B)^\alpha \left| \frac{p_i^A - p_i^B}{p_i^A + p_i^B} \right|}{\sum_{i=1}^{m} b_i (p_i^A + p_i^B)^\alpha},$$

where m is the number of branches, b_i is the length of ith branch, p_i^A, p_i^B are the branch proportion for community A and B.

Generalized UniFrac distance contains an extra parameter α controlling the weight on abundant lineages so the distance is not dominated by highly abundant lineages. $\alpha = 0.5$ ("d_0.5") is overall very robust.

The unweighted ("d_1") and weighted UniFrac ("d_UW") are also implemented.

Usage

```
GUniFrac(otu.tab, tree, size.factor = NULL, alpha = c(0, 0.5, 1), verbose = TRUE)
```

Arguments

otu.tab	a matrix, the OTU count table, row - n sample, column - q OTU
tree	a rooted phylogenetic tree of R class "phylo"
size.factor	a numeric vector of the normalizing factors to divide the counts. The length is the number of samples. This provides the flexibility to normalize data using the preferred normalization method (e.g. GMPR normalizing factor). If not supplied, the total sum will be used.
alpha	a numeric vector, parameters controlling the weight on abundant lineages
verbose	logical value, whether to print out the messages

Value

Return a list containing

`unifracs` a three dimensional array containing all the UniFrac distance matrices

Note

The function only accepts rooted tree. To root a tree, you may consider using `midpoint` from the package `phangorn`.

Author(s)

Jun Chen <chen.jun2@mayo.edu>

References

Chen, J., Bittinger, K., Charlson, E.S., Hoffmann, C., Lewis, J., Wu, G.D., Collman, R.G., Bushman, F.D. and Li, H. (2012). Associating microbiome composition with environmental covariates using generalized UniFrac distances. 28(16): 2106–2113.

See Also

`Rarefy`, `PermanovaG`

Examples

```
## Not run:
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

groups <- throat.meta$SmokingStatus

# Rarefaction
otu.tab.rff <- Rarefy(throat.otu.tab)$otu.tab.rff

# Calculate the UniFracs
unifracs <- GUniFrac(otu.tab.rff, throat.tree, alpha=c(0, 0.5, 1))$unifracs

dw <- unifracs[, , "d_1"] # Weighted UniFrac
du <- unifracs[, , "d_UW"] # Unweighted UniFrac
d0 <- unifracs[, , "d_0"]      # GUniFrac with alpha 0
d5 <- unifracs[, , "d_0.5"]    # GUniFrac with alpha 0.5

# Permanova - Distance based multivariate analysis of variance
adonis3(as.dist(d5) ~ groups)

## End(Not run)
```

PermanovaG *Permutational Multivariate Analysis of Variance Using Multiple Distance Matrices*

Description

In practice, we do not know a priori which type of change happens in the microbiome. Each distance measure is most powerful in detecting only a certain scenario. When multiple distance matrices are available, separate tests using each distance matrix will lead to loss of power due to multiple testing correction. Combing the distance matrices in a single test will improve power. PermanovaG combines multiple distance matrices by taking the minimum of the P values for individual distance matrices. Significance is assessed by permutation.

Usage

```
PermanovaG(formula, data = NULL, ...)
```

Arguments

formula	a formula, left side of the formula (Y ~ X) is a three dimensional ARRAY containing the supplied distance matrices as produced by GUniFrac function. Or it could be a list of distance matrices.
data	a data frame containing the covariates
...	parameter passing to adonis function

Value

Return a list containing:

p.tab	a data frame, columns: p-values for individual distance matrices and the omnibus test, rows: covariates. (Note: they are sequential p-values, put the variable of interest in the end)
aov.tab.list	a list of adonis AOV tables for individual distance matrices

Author(s)

Jun Chen <chen.jun2@mayo.edu>

References

Chen, J., Bittinger, K., Charlson, E.S., Hoffmann, C., Lewis, J., Wu, G.D., Collman, R.G., Bushman, F.D. and Li, H.(2012). Associating microbiome composition with environmental covariates using generalized UniFrac distances. 28(16): 2106–2113.

See Also

Rarefy, GUniFrac

Examples

```
## Not run:
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

groups <- throat.meta$SmokingStatus

# Rarefaction
otu.tab.rff <- Rarefy(throat.otu.tab)$otu.tab.rff

# Calculate the UniFracs
unifracs <- GUniFrac(otu.tab.rff, throat.tree, alpha=c(0, 0.5, 1))$unifracs

# Combine unweighted and weighted UniFrac for testing
PermanovaG(unifracs[, , c("d_1", "d_UW")] ~ groups)
# Combine d(0), d(0.5), d(1) for testing

## End(Not run)
```

`PermanovaG2` *Permutational Multivariate Analysis of Variance Using Multiple Distance Matrices(Freedman-Lane Permutation)*

Description

In practice, we do not know a priori which type of change happens in the microbiome. Each distance measure is most powerful in detecting only a certain scenario. When multiple distance matrices are available, separate tests using each distance matrix will lead to loss of power due to multiple testing correction. Combing the distance matrices in a single test will improve power. PermanovaG combines multiple distance matrices by taking the minimum of the P values for individual distance matrices. Significance is assessed by permutation.

Usage

```
PermanovaG2(formula, data = NULL, ...)
```

Arguments

`formula`	a formula, left side of the formula (`Y ~ X`) is a three dimensional ARRAY containing the supplied distance matrices as produced by `GUniFrac` function. Or it could be a list of distance matrices.
`data`	a data frame containing the covariates
`...`	parameters passing to `adonis` function

Value

Return a list containing:

p.tab	a data frame, columns: p-values for individual distance matrices and the omnibus test, rows: covariates. (Note: they are sequential p-values, put the variable of interest in the end)
aov.tab.list	a list of adonis AOV tables for individual distance matrices

Author(s)

Jun Chen <chen.jun2@mayo.edu>

References

Chen, J., Bittinger, K., Charlson, E.S., Hoffmann, C., Lewis, J., Wu, G.D., Collman, R.G., Bushman, F.D. and Li, H. (2012). Associating microbiome composition with environmental covariates using generalized UniFrac distances. 28(16): 2106–2113.

See Also

Rarefy, GUniFrac, adonis3

Examples

```
## Not run:
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

groups <- throat.meta$SmokingStatus

# Rarefaction
otu.tab.rff <- Rarefy(throat.otu.tab)$otu.tab.rff

# Calculate the UniFracs
unifracs <- GUniFrac(otu.tab.rff, throat.tree, alpha=c(0, 0.5, 1))$unifracs

# Combine unweighted and weighted UniFrac for testing
PermanovaG2(unifracs[, , c("d_1", "d_UW")] ~ groups)

## End(Not run)
```

Rarefy *Rarefy a Count Table to Equal Sequencing Depth*

Description

GUniFrac is also sensitive to different sequencing depth. To compare microbiomes on an equal basis, rarefaction might be used.

Usage

```
Rarefy(otu.tab, depth = min(rowSums(otu.tab)))
```

Arguments

otu.tab	OTU count table, row - n sample, column - q OTU
depth	required sequencing depth; If not specified, the lowest sequencing depth is used.

Value

Return a list containing:

otu.tab.rff	rarefied OTU table
discard	IDs of samples that does not reach the specified sequencing depth

Author(s)

Jun Chen <chen.jun2@mayo.edu>

References

Chen, J., Bittinger, K., Charlson, E.S., Hoffmann, C., Lewis, J., Wu, G.D., Collman, R.G., Bushman, F.D. and Li, H. (2012). Associating microbiome composition with environmental covariates using generalized UniFrac distances. 28(16): 2106–2113.

Examples

```
data(throat.otu.tab)
# Rarefaction
otu.tab.rff <- Rarefy(throat.otu.tab, 1024)$otu.tab.rff
```

SimulateMSeq *A Semiparametric Model-based Microbiome Sequencing Data Simulator for Cross-sectional and Case-control Studies*

Description

The function generates synthetic microbiome sequencing data for studying the performance of differential abundance analysis methods. It uses a user-supplied (large) reference OTU table as a template to generate a synthetic OTU table of specified size. A subset of OTUs are affected by a simulated covariate of interest, either binary or continuous. Confounder effects can also be simulated. The function allows simulating different signal structures, i.e., the percentage of differential OTUs, their effect sizes, their direction of change, and whether these OTUs are relatively abundant or rare.

Usage

```
SimulateMSeq(
  ref.otu.tab,
  nSam = 100,
  nOTU = 500,
  diff.otu.pct = 0.1,
  diff.otu.direct = c("balanced", "unbalanced"),
  diff.otu.mode = c("abundant", "rare", "mix"),
  covariate.type = c("binary", "continuous"),
  grp.ratio = 1,
  covariate.eff.mean = 1,
  covariate.eff.sd = 0,
  confounder.type = c("none", "binary", "continuous", "both"),
  conf.cov.cor = 0.6,
  conf.diff.otu.pct = 0,
  conf.nondiff.otu.pct = 0.1,
  confounder.eff.mean = 0,
  confounder.eff.sd = 0,
  error.sd = 0,
  depth.mu = 10000,
  depth.theta = 5,
  depth.conf.factor = 0
)
```

Arguments

`ref.otu.tab`	a matrix, the reference OTU count table (row - OTUs, column - samples), serving as the template for synthetic sample generation.
`nSam`	the number of samples to be simulated.
`nOTU`	the number of OTUs to be simulated.
`diff.otu.pct`	a numeric value between 0 and 1, the percentage of differential OTUs to be simulated. If 0, global null setting is simulated. The default is 0.1.
`diff.otu.direct`	a character string of "balanced" or "unbalanced". "balanced" - the direction of change for these differential OTUs is random, "unbalanced" - direction of change is the same. The default is "balanced".
`diff.otu.mode`	a character string of "rare", "mix" or "abundant". "abundant" - differential OTU come from the top quartile of the abundance distribution, "rare" - differential OTU come from the bottom quartile of the abundance distribution, and "mix" - random set. The default is "abundant".
`covariate.type`	a character string of "binary" or "continuous", indicating the type of the covariate to be simulated. The default is "binary" (e.g., case v.s. control).
`grp.ratio`	a numeric value between 0 and 1. Group size ratio. The default is 1, i.e., equal group size. Only relevant when `covariate.type` is "binary".
`covariate.eff.mean`	a numeric value, the mean log fold change (effect size) in response to one unit change of the covariate. The default is 1.

`covariate.eff.sd`
: a positive numeric value, the standard deviation of the log fold change. The default is 0, i.e., the log fold change is the same across differential OTUs.

`confounder.type`
: a character string of "none", "binary", "continuous" or "both". The default is "none", no confounder will be simulated. If "both", both a binary and continuous confounder will be simulated. The default is "none".

`conf.cov.cor`
: a numeric value between 0 and 1. The correlation between the covariate of interest and the confounder. The default is 0.6.

`conf.diff.otu.pct`
: a numeric value between 0 and 1. The percentage of OTUs affected by the confounder and the covariate of interest. The default is 0.

`conf.nondiff.otu.pct`
: a numeric value between 0 and 1. The percentage of OTUs affected by the confounder but not the covariate of interest. The default is 0.1.

`confounder.eff.mean`
: a numeric value, the mean log fold change (effect size) in response to one unit change of the confounder. The default is 1.

`confounder.eff.sd`
: a positive numeric value, the standard deviation of the log fold change for the confounder. The default is 0, i.e., the log fold change is the same across OTUs affected by the confounder.

`error.sd`
: the sd of the log fold change unexplained by the covariate and the confounder (i.e., the error term under the log linear model). The default is 0.

`depth.mu`
: the mean sequencing depth to be simulated. The default is 10,000.

`depth.theta`
: the theta value of the negative binomial distribution controlling the variance (`mu + mu^2/theta`). The default is 5.

`depth.conf.factor`
: a numeric value controlling the dependence of the sequencing depth on the covariate of interest (`depth.mu * exp(scale(X) * depth.conf.factor)`). The default is 0, i.e., the depth is not associated with the covariate of interest. This parameter can be used to simulate depth confounding.

Details

This function implements a semiparametric approach for realistic independent microbiome sequencing data generation. The method draws random samples from a large reference dataset (nonparametric part) and uses these reference samples as templates to generate new samples (parametric part). Specifically, for each drawn reference sample, it infers the underlying composition based on a Bayesian model and then adds covariate/confounder effects to the composition vector, based on which a new sequencing sample is generated. The method circumvents the difficulty in modeling the inter-subject variation of the microbiome composition.

Value

Return a list with the elements:

`otu.tab.sim`
: simulated OTU table

covariate	simulated covariate of interest
confounder	simulated confounder(s)
diff.otu.ind	indices of the differential OTUs, i.e., affected by the covariate of interest
otu.names	the names of the simulated OTUs
conf.otu.ind	indices of OTUs affected by the confounder(s)

Author(s)

Jun Chen and Lu Yang

References

Yang, L. & Chen, J. 2022. A comprehensive evaluation of differential abundance analysis methods: current status and potential solutions. Microbiome. In Press.

Examples

```
# Use throat microbiome for illustration
data(throat.otu.tab)
comm <- t(throat.otu.tab)
comm <- comm[rowMeans(comm != 0) > 0.2, ]

# Simulate binary covariate, 10% signal density, abundant differential OTUs, unbalanced change
# This setting simulates strong compositional effects
sim.obj <- SimulateMSeq(
ref.otu.tab = comm, nSam = 50, nOTU = 50,
# True signal setting
diff.otu.pct = 0.1, diff.otu.direct = c("unbalanced"),
diff.otu.mode = c("abundant"),
covariate.type = c("binary"), grp.ratio = 1,
covariate.eff.mean = 1.0, covariate.eff.sd = 0,
# Confounder signal setting
confounder.type = c("both"), conf.cov.cor = 0.6,
conf.diff.otu.pct = 0.1, conf.nondiff.otu.pct = 0.1,
confounder.eff.mean = 1.0, confounder.eff.sd = 0,
# Depth setting
depth.mu = 10000, depth.theta = 5, depth.conf.factor = 0
)

meta.dat <- data.frame(X = sim.obj$covariate, Z1 = sim.obj$confounder[, 1],
                       Z2 = sim.obj$confounder[, 2])
otu.tab.sim <- sim.obj$otu.tab.sim

# Run ZicoSeq for differential abundance analysis
zico.obj <- ZicoSeq(meta.dat = meta.dat, feature.dat = otu.tab.sim,
grp.name = 'X', adj.name = c('Z1', 'Z2'), feature.dat.type = "count",
# Filter to remove rare taxa
prev.filter = 0.2, mean.abund.filter = 0,  max.abund.filter = 0.002, min.prop = 0,
# Winsorization to replace outliers
is.winsor = TRUE, outlier.pct = 0.03, winsor.end = 'top',
# Posterior sampling to impute zeros
```

```
is.post.sample = TRUE, post.sample.no = 25,
# Multiple link functions to capture diverse taxon-covariate relation
link.func = list(function (x) x^0.25, function (x) x^0.5, function (x) x^0.75),
stats.combine.func = max,
# Permutation-based multiple testing correction
perm.no = 99,  strata = NULL,
# Reference-based multiple stage normalization
ref.pct = 0.5, stage.no = 6, excl.pct = 0.2,
# Family-wise error rate control
is.fwer = FALSE,
verbose = TRUE, return.feature.dat = FALSE)

# Detected differential OTUs
which(zico.obj$p.adj.fdr <= 0.05)

# True differential OTUs
sim.obj$otu.names[sim.obj$diff.otu.ind]
```

`SimulateMSeqC` *A Semiparametric Model-based Microbiome Sequencing Data Simulator for Longitudinal, Matched-pair, and Replicate Sampling Designs*

Description

The function generates synthetic microbiome sequencing data for studying the performance of differential abundance analysis methods for correlated microbiome data generated in longitudinal, matched-pair and replicate sampling study designs. It uses a user-supplied (large) reference OTU table as a template to generate a synthetic OTU table of specified size. A subset of OTUs can be affected by by a binary variable (group effect) and/or a time variable (temporal effect). Time X group interaction and confounder effects can also be simulated. The function allows simulating different signal structures, i.e., the percentage of differential OTUs, their effect sizes, and their direction of change.

Usage

```
SimulateMSeqC(
  ref.otu.tab,
  nSubject = 40,
  nOTU = 50,
  nTime = 2,
  error.sd = 1,
  MgX = 0.5,
  SgX = 0,
  X.diff.otu.pct = 0.1,
  grp.ratio = 1,
  balanced.X = TRUE,
  MgT = 0,
  SgT = 0,
```

```
  SbT = 0,
  T.diff.otu.pct = 0,
  balanced.T = TRUE,
  MgXT = 0,
  SgXT = 0,
  XT.diff.otu.pct = 0,
  balanced.XT = TRUE,
  conf.cov.cor = 0.6,
  confounder = c('X', 'T'),
  MgZ = 0.5,
  SgZ = 0,
  Z.diff.otu.pct = 0.05,
  Z.nondiff.otu.pct = 0.1,
  depth.mu = 10000,
  depth.theta = 5,
  depth.conf.factor = 0
)
```

Arguments

ref.otu.tab	a matrix, the reference OTU count table (row - OTUs, column - samples), serving as the template for synthetic sample generation.
nSubject	the number of subjects to be simulated.
nOTU	the number of OTUs to be simulated.
nTime	the number of time points to be simulated.
error.sd	the standar deviation of the random error controlling the within-subject correlation strength. Large k = 1, small k = 4.
MgX	a numeric value indicating the mean group (X) effect (log fold change) across the associated OTUs. The default is 0.
SgX	a numeric value indicating the variance of group (X) effect (log fold change) across the associated OTUs. The default is 0.
X.diff.otu.pct	a numeric value between 0 and 1, the percentage of differential OTUs regarding the group (X) to be simulated. The default is 0.1.
grp.ratio	a numeric value between 0 and 1. Group size ratio. The default is 1, i.e., equal group size.
balanced.X	a logical value. TRUE - the direction of change for these group differential OTUs is random, FALSE - the direction of change is the same. The default is "balanced".
MgT	a numeric value indicating the population mean of the time (T) effect (log fold change) across the associated OTUs. The default is 0.
SgT	a numeric value indicating the population variance of the time (T) effect (log fold change) across the associated OTUs. The default is 0.
SbT	a numeric value indicating the variance of time (T) effect (log fold change) across the subjects. This parameter is to generate a subject-level random slope (temporal trends differ by subjects). The default is 0.

`T.diff.otu.pct` a numeric value between 0 and 1, the percentage of time differential OTUs to be simulated. The default is 0.

`balanced.T` a logical value. TRUE - the direction of change for these time differential OTUs is random, FALSE - the direction of change is the same. The default is "balanced".

`MgXT` a numeric value indicating the mean X:T interaction effect (log fold change) across the associated OTUs. The default is 0.

`SgXT` a numeric value indicating the variance of X:T interaction effect (log fold change) across the associated OTUs. The default is 0.

`XT.diff.otu.pct`
a numeric value between 0 and 1, the percentage of X:T interaction OTUs to be simulated. The default is 0.

`balanced.XT` a logical value. TRUE - the direction of change for the interaction effects is random, FALSE - the direction of change is the same. The default is "balanced".

`conf.cov.cor` a numeric value between 0 and 1. The correlation strength between the group and the confounder. The default is 0.6.

`confounder` one of 'X' or 'T', whether the confounder is correlated with 'X' or 'T'.

`MgZ` a numeric value indicating the mean confounder (Z) effect (log fold change) across the associated OTUs. The default is 0.

`SgZ` a numeric value indicating the variance of confounder (Z) effect (log fold change) across the associated OTUs. The default is 0.

`Z.diff.otu.pct` a numeric value between 0 and 1, less than the percentage of group differential OTUs, the percentage of confounder-associated OTUs, which are also group differential. The default is 0.05.

`Z.nondiff.otu.pct`
a numeric value between 0 and 1, less than the percentage of group non-differential OTUs, the percentage of confounder-associated OTUs, which are not group differential. The default is 0.1.

`depth.mu` the mean sequencing depth to be simulated. The default is 10,000.

`depth.theta` the theta value of the negative binomial distribution controlling the variance (`mu + mu^2/theta`). The default is 5.

`depth.conf.factor`
a numeric value controlling the dependence of the sequencing depth on the group variable (`depth.mu * exp(scale(X) * depth.conf.factor)`). The default is 0, i.e., the depth is not different between groups. This parameter can be used to simulate depth confounding.

Details

This function implements a semiparametric approach for realistic correlated microbiome data generation. The method draws random samples from a large reference dataset (non-parametric part) and uses these reference samples as templates to generate new samples (parametric part). Specifically, for each drawn reference sample, it infers the underlying composition based on a Bayesian model and then adds group/time/group:time/confounder effects to the composition vector, based on which a new sequencing sample is generated. The method circumvents the difficulty in modeling the inter-subject variation of the microbiome composition.

Value

Return a list with the elements:

`otu.tab.sim`	simulated OTU table
`meta`	meta data containing the simulated covariates (group, time, confounder)
`otu.names`	the names of the simulated OTUs
`X.diff.otu.ind`	indices of the group differential OTUs
`T.diff.otu.ind`	indices of the time differential OTUs
`XT.diff.otu.ind`	indices of the OTUs with group:time interaction
`Z.diff.otu.ind`	indices of OTUs affected by the confounder

Author(s)

Lu Yang and Jun Chen

References

Yang, L. & Chen, J. 2022. Benchmarking Differential Abundance Analysis Methods for Correlated Microbiome Sequencing Data. Submitted.

Examples

```
# Use throat microbiome for illustration
data(throat.otu.tab)
comm <- t(throat.otu.tab)
comm <- comm[rowMeans(comm != 0) > 0.2, ]

# Example1: Simulate replicate sampling data, 40 subjects each with two replicates (nTime =2),
# two group comparison, 10% group differential OTUs
## Not run:
sim.obj <- SimulateMSeqC(ref.otu.tab= comm,
                         nSubject = 40, nOTU = 50, nTime = 2,
                         # Within-subject correlation setting
                         error.sd = 1,
                         # Group effect setting, unbalanced
                         MgX = 0.5, SgX = 0, X.diff.otu.pct = 0.1, grp.ratio = 1,
                         balanced.X = FALSE,
                         # Time effect setting (No time effect)
                         MgT = 0, SgT = 0, SbT = 0, T.diff.otu.pct = 0,
                         # Interaction effect setting (No interaction effect)
                         MgXT = 0, SgXT = 0, XT.diff.otu.pct = 0,
                         # Confounder effect setting
                         conf.cov.cor = 0.6, confounder = 'X',
                       MgZ = 0.5, SgZ = 0,  Z.diff.otu.pct = 0.05, Z.nondiff.otu.pct = 0.1,
                         # Sequencing Depth setting
                         depth.mu = 10000, depth.theta = 5,  depth.conf.factor = 0)

## End(Not run)
# Example2: Simulate matched-pair data, 100 subjects each with pre- and post-treatment (nTime = 2),
```

```
# 10% differential OTUs
## Not run:
sim.obj <- SimulateMSeqC(ref.otu.tab= comm,
                         nSubject = 100, nOTU = 50, nTime = 2,
                         # Within-subject correlation setting
                         error.sd = 1,
                         # Group effect setting (No group effect)
                         MgX = 0, SgX = 0, X.diff.otu.pct = 0, grp.ratio = 1,
                         # Time effect setting (No random slope, SbT=0)
                         MgT = 0.5, SgT = 0, SbT = 0, T.diff.otu.pct = 0.1,
                         # Interaction effect setting (No interaction effect)
                         MgXT = 0, SgXT = 0, XT.diff.otu.pct = 0,
                         # Confounder effect setting (T!)
                         conf.cov.cor = 0.6, confounder = 'T',
                        MgZ = 0, SgZ = 0,  Z.diff.otu.pct = 0.05, Z.nondiff.otu.pct = 0.1,
                         # Sequencing Depth setting
                         depth.mu = 10000, depth.theta = 5,  depth.conf.factor = 0)

## End(Not run)

# Example3: Simulate the general longitudinal data, 40 Subjects each with five time points,
# two groups, 10% group differential OTUs, 10 % time differential OTUs and 10 % interaction OTUs.
## Not run:
sim.obj <- SimulateMSeqC(ref.otu.tab= comm,
                         nSubject = 40, nOTU = 50, nTime = 5,
                         # Within-subject correlation setting
                         error.sd = 1,
                         # Group effect setting, balanced
                         MgX = 0.5, SgX = 0, X.diff.otu.pct = 0.1, grp.ratio = 1,
                         balanced.X = TRUE,
                         # Time effect setting (random slope)
                         MgT = 0.5, SgT = 0, SbT = 0.5, T.diff.otu.pct = 0.1,
                         # Interaction effect setting
                         MgXT = 0.5, SgXT = 0, XT.diff.otu.pct = 0.1,
                         # Confounder effect setting
                         conf.cov.cor = 0.6, confounder = 'X',
                       MgZ = 0.5, SgZ = 0,  Z.diff.otu.pct = 0.05, Z.nondiff.otu.pct = 0.1,
                         # Depth setting
                         depth.mu = 10000, depth.theta = 5,  depth.conf.factor = 0)

## End(Not run)
```

`stool.otu.tab` *Stool Microbiome OTU Count Table*

Description

OTU count table from 16S V3-V5 targeted sequencing of the stool microbiome samples from the HMP project. A total of 2,094 OTUs from 295 samples.

Usage

```
data(stool.otu.tab)
```

Format

The format is: chr "stool.otu.tab"

Details

The OTU table was taken from R bioconductor "HMP16SData" package. OTUs with prevalence less than 10% and maximum proportion less than 0.2% were removed. This OTU table can be used for simulating stool microbiome sequencing data.

Source

Schiffer L, Azhar R, Shepherd L, Ramos M, Geistlinger L, Huttenhower C, Dowd JB, Segata N, Waldron L (2019). "HMP16SData: Efficient Access to the Human Microbiome Project through Bioconductor." American Journal of Epidemiology. doi: 10.1093/aje/kwz006.

Examples

```
data(stool.otu.tab)
```

---

throat.meta *Throat Microbiome Meta Data*

Description

It is part of a microbiome data set for studying the effect of smoking on the upper respiratory tract microbiome. The original data set contains samples from both throat and nose microbiomes, and from both body sides. This data set comes from the throat microbiome of left body side. It contains 60 subjects consisting of 32 nonsmokers and 28 smokers.

Usage

```
data(throat.meta)
```

Source

Charlson ES, Chen J, Custers-Allen R, Bittinger K, Li H, et al. (2010) Disordered Microbial Communities in the Upper Respiratory Tract of Cigarette Smokers. PLoS ONE 5(12): e15216.

Examples

```
data(throat.meta)
```

throat.otu.tab *Throat Microbiome OTU Count Table*

Description

It is part of a microbiome data set (16S V12-targeted 454 pyrosequencing) for studying the effect of smoking on the upper respiratory tract microbiome. The original data set contains samples from both throat and nose microbiomes, and from both body sides. This data set comes from the throat microbiome of left body side. It contains 60 subjects consisting of 32 nonsmokers and 28 smokers.

Usage

```
data(throat.otu.tab)
```

Details

The OTU table is produced by the QIIME software. Singleton OTUs have been discarded.

Source

Charlson ES, Chen J, Custers-Allen R, Bittinger K, Li H, et al. (2010) Disordered Microbial Communities in the Upper Respiratory Tract of Cigarette Smokers. PLoS ONE 5(12): e15216.

Examples

```
data(throat.otu.tab)
```

throat.tree *UPGMA Tree of Throat Microbiome OTUs*

Description

The OTU tree is constructed using UPGMA on the K80 distance matrice of the OTUs. It is a rooted tree of class "phylo".

Usage

```
data(throat.tree)
```

Details

The OTUs are produced by the QIIME software. Singleton OTUs have been discarded.

Source

Charlson ES, Chen J, Custers-Allen R, Bittinger K, Li H, et al. (2010) Disordered Microbial Communities in the Upper Respiratory Tract of Cigarette Smokers. PLoS ONE 5(12): e15216.

Examples

```
data(throat.tree)
```

`vaginal.otu.tab` *Vaginal Microbiome OTU Count Table*

Description

OTU count table from 16S V3-V5 targeted sequencing of the vaginal microbiome samples from the HMP project. A total of 780 OTUs from 381 samples.

Usage

```
data(vaginal.otu.tab)
```

Details

The OTU table was taken from R bioconductor "HMP16SData" package. OTUs with prevalence less than 10% and maximum proportion less than 0.2% were removed. This OTU table can be used for simulating vaginal microbiome sequencing data.

Source

Schiffer L, Azhar R, Shepherd L, Ramos M, Geistlinger L, Huttenhower C, Dowd JB, Segata N, Waldron L (2019). "HMP16SData: Efficient Access to the Human Microbiome Project through Bioconductor." American Journal of Epidemiology. doi: 10.1093/aje/kwz006.

Examples

```
data(stool.otu.tab)
```

`ZicoSeq` *A linear Model-based Permutation Test for Differential Abundance Analysis of Microbiome Data and Other Omics Data*

Description

ZicoSeq is a permutation test (Smith permutation) for differential abundance analysis of microbiome sequencing data. The input can be a count or a proportion matrix. When a count matrix is provided, it provides an option to draw posterior samples of the underlying proportions to account for the sampling variability during the sequencing process. The test results are aggregated over these posterior samples. For both count and proportion data, a reference-based ratio approach is used to account for compositional effects. As a general methodology, ZicoSeq can also be applied to differential analysis of other omics data. In this case, they are not treated as compositional data.

Usage

```
ZicoSeq(
  meta.dat,
  feature.dat,
  grp.name,
  adj.name = NULL,
  feature.dat.type = c('count', 'proportion', 'other'),
  prev.filter = 0,
  mean.abund.filter = 0,
  max.abund.filter = 0,
  min.prop = 0,
  is.winsor = TRUE,
  outlier.pct = 0.03,
  winsor.end = c('top', 'bottom', 'both'),
  is.post.sample = TRUE,
  post.sample.no = 25,
  link.func = list(function(x) sign(x) * (abs(x))^0.5),
  stats.combine.func = max,
  perm.no = 99,
  strata = NULL,
  ref.pct = 0.5,
  stage.no = 6,
  excl.pct = 0.2,
  p.max = 500,
  is.fwer = FALSE,
  verbose = TRUE,
  return.feature.dat = TRUE
)
```

Arguments

`meta.dat`	a data frame containing the sample meta data.
`feature.dat`	a matrix of feature data, row - features (OTUs, genes, etc) , column - samples.
`grp.name`	the name for the variable of interest. It could be numeric or categorical; should be in `meta.dat`.
`adj.name`	the name(s) for the variable(s) to be adjusted. Multiple variables are allowed. They could be numeric or categorical; should be in `meta.dat`.
`feature.dat.type`	the type of the feature data. It could be "count", "proportion" or "other". For "proportion" data type, posterior sampling will not be performed, but the reference-based ratio approach will still be used to address compositional effects. For "other" data type, neither posterior sampling or reference-base ratio approach will be used.
`prev.filter`	the prevalence (percentage of nonzeros) cutoff, under which the features will be filtered. The default is 0.

`mean.abund.filter`
: the mean relative abundance cutoff, under which the features will be filtered. The default is 0.

`max.abund.filter`
: the max relative abundance cutoff, under which the features will be filtered. The default is 0.

`min.prop`
: proportions less than this value will be replaced with this value. Only relevant when log transformation is used. Default is 0.

`is.winsor`
: a logical value indicating whether winsorization should be performed to replace outliers. The default is TRUE.

`outlier.pct`
: the expected percentage of outliers. These outliers will be winsorized. The default is 0.03. For count/proportion data, `outlier.pct` should be less than `prev.filter`.

`winsor.end`
: a character indicating whether the outliers at the "top", "bottom" or "both" will be winsorized. The default is "top". If the `feature.dat.type` is "other", "both" may be considered.

`is.post.sample`
: a logical value indicating whether to perform posterior sampling of the underlying proportions. Only relevant when the feature data are counts.

`post.sample.no`
: the number of posterior samples if posterior sampling is used. The default is 25.

`link.func`
: a list of transformation functions for the feature data or the ratios. Based on our experience, square-root transformation is a robust choice for many datasets.

`perm.no`
: the number of permutations. If the raw p values are of the major interest, set `perm.no` to at least 999.

`strata`
: a factor such as subject IDs indicating the permutation strata or characters indicating the strata variable in `meta.dat`. Permutation will be confined to each stratum. This can be used for paired or some longitudinal designs.

`stats.combine.func`
: function to combine the F-statistic for the omnibus test. The default is `max`.

`ref.pct`
: percentage of reference taxa. The default is 0.5.

`p.max`
: the maximum number of (most abundant) taxa to be considered in reference taxa selection; only relevant when the nubmer of taxa is huge. The default is 500, i.e., when the nubmer of taxa is larger than 500, only the 500 most abundant taxa will be used for reference selection. This is to reduce the computational time.

`stage.no`
: the number of stages if multiple-stage normalization is used. The default is 6.

`excl.pct`
: the maximum percentage of significant features (nominal p-value < 0.05) in the reference set that should be removed. Only relevant when multiple-stage normalization is used.

`is.fwer`
: a logical value indicating whether the family-wise error rate control (West-Young) should be performed.

`verbose`
: a logical value indicating whether the trace information should be printed out.

`return.feature.dat`
: a logical value indicating whether the winsorized, filtered "feature.dat" matrix should be returned.

Details

ZicoSeq is a linear model-based permutation test developed for differential abundance analysis of zero-inflated compositional data. Although its development is motivated by zero-inflated microbiome sequence count data, it can be applied to proportion (composition) data and more generally to other types of omics data. Currently, it has the following components: 1. Winsorization to decrease the influence of outliers; 2. Posterior sampling based on a beta mixture prior to address sampling variability and zero inflation; 3. Reference-based multiple-stage normalization to address compositional effects; 4. An omnibus test to address diverse feature-covariate relationships; 5. Permutation-based false discovery rate control / family-wise error rate control for multiple testing correction, which takes into account the correlation structure in the feature data.

Value

A list with the elements

call	the call
feature.dat	the winsorized, filtered feature.dat matrix.
meta.dat	meta.dat used.
grp.name	the name of the variable of interest.
filter.features	a vector of the names of the features that are filtered.
ref.features	a vector of the names of the reference features. Only relevant when reference approach is used.
R2	a matrix of percent explained variance (number of features by number of transformation functions).
F0	a matrix of F-statistics (number of features by number of transformation functions).
RSS	a matrix of residual sum squares (number of features by number of transformation functions).
df.model, df.residual	degrees of freedom for the model and residual space.
coef.list	a list of the linear regression coefficients under the specified transformations.
p.raw	the raw p-values based on permutations (not accurate if perm.no is small).
p.adj.fdr	permutation-based FDR-adjusted p-values.
p.adj.fwer	permutation-based FWER-adjusted (West-Young) p-values.

Author(s)

Jun Chen

References

Yang, L. & Chen, J. 2022. A comprehensive evaluation of differential abundance analysis methods: current status and potential solutions. Microbiome, 10(1), 1-23.

See Also

ZicoSeq.plot

Examples

```
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

comm <- t(throat.otu.tab)
meta.dat <- throat.meta

set.seed(123)
# For count data
zico.obj <- ZicoSeq(meta.dat = meta.dat, feature.dat = comm,
grp.name = 'SmokingStatus', adj.name = 'Sex', feature.dat.type = "count",
# Filter to remove rare taxa
prev.filter = 0.2, mean.abund.filter = 0,  max.abund.filter = 0.002, min.prop = 0,
# Winsorization to replace outliers
is.winsor = TRUE, outlier.pct = 0.03, winsor.end = 'top',
# Posterior sampling to impute zeros
is.post.sample = TRUE, post.sample.no = 25,
# Multiple link functions to capture diverse taxon-covariate relation
link.func = list(function (x) x^0.25, function (x) x^0.5, function (x) x^0.75),
stats.combine.func = max,
# Permutation-based multiple testing correction
perm.no = 99,  strata = NULL,
# Reference-based multiple stage normalization
ref.pct = 0.5, stage.no = 6, excl.pct = 0.2,
# Family-wise error rate control
is.fwer = FALSE,
verbose = TRUE, return.feature.dat = FALSE)

which(zico.obj$p.adj.fdr <= 0.05)

# For proportion data
comm.p <- t(t(comm) / colSums(comm))
zico.obj <- ZicoSeq(meta.dat = meta.dat, feature.dat = comm.p,
grp.name = 'SmokingStatus', adj.name = 'Sex', feature.dat.type = "proportion",
# Filter to remove rare taxa
prev.filter = 0.2, mean.abund.filter = 0,  max.abund.filter = 0.002, min.prop = 0,
# Winsorization to replace outliers
is.winsor = TRUE, outlier.pct = 0.03, winsor.end = 'top',
# Posterior sampling will be automatically disabled
is.post.sample = FALSE, post.sample.no = 25,
# Use the square-root transformation
link.func = list(function (x) x^0.5), stats.combine.func = max,
# Permutation-based multiple testing correction
perm.no = 99,  strata = NULL,
# Reference-based multiple stage normalization
ref.pct = 0.5, stage.no = 6, excl.pct = 0.2,
# Family-wise error rate control
```

```
is.fwer = FALSE,
verbose = TRUE, return.feature.dat = FALSE)

which(zico.obj$p.adj.fdr <= 0.05)


# For other type of data. The user should be responsible for the filtering.
comm.o <- comm[rowMeans(comm != 0) >= 0.2, ] + 1
comm.o <- log(t(t(comm.o) / colSums(comm.o)))
zico.obj <- ZicoSeq(meta.dat = meta.dat, feature.dat = comm.o,
grp.name = 'SmokingStatus', adj.name = 'Sex', feature.dat.type = "other",
# Filter will not be applied
prev.filter = 0, mean.abund.filter = 0,  max.abund.filter = 0, min.prop = 0,
# Winsorization to both ends of the distribution
is.winsor = TRUE, outlier.pct = 0.03, winsor.end = 'both',
# Posterior sampling will be automatically disabled
is.post.sample = FALSE, post.sample.no = 25,
# Identity function is used
link.func = list(function (x) x), stats.combine.func = max,
# Permutation-based multiple testing correction
perm.no = 99,  strata = NULL,
# Reference-based multiple-stage normalization will not be performed
ref.pct = 0.5, stage.no = 6, excl.pct = 0.2,
# Family-wise error rate control
is.fwer = TRUE,
verbose = TRUE, return.feature.dat = FALSE)

which(zico.obj$p.adj.fdr <= 0.05)
```

`ZicoSeq.plot` *A Plot Function for Visualizing the ZicoSeq Results*

Description

`ZicoSeq.plot` produces volcano plots with the y-axis being the log10 (adjusted) p-value and the x-axis being the signed R^2^ with the sign indicating the association direction determined based on the sign of the regression coefficients (for multi-categorical variables, sign is not applicable). The names of differential taxa passing a specific cutoff will be printed on the figure. When data types are counts and proportions, the mean abundance and prevalence will be visualized; when the data type is 'other', mean and standard deviation of the features will be visualized. Users need to set `return.feature.dat = T` when using the plot function.

Usage

```
ZicoSeq.plot(
  ZicoSeq.obj,
  pvalue.type = c('p.adj.fdr','p.raw','p.adj.fwer'),
  cutoff = 0.1,
```

```
  text.size = 10,
  out.dir = NULL,
  file.name = 'ZicoSeq.plot.pdf',
  width = 10,
  height = 6)
```

Arguments

ZicoSeq.obj	object from calling the function ZicoSeq.
pvalue.type	character string, one of 'p.adj.fdr','p.raw' and 'p.adj.fwer'.
cutoff	a cutoff between 0 and 1 for pvalue.type, below which the names of the features will be printed.
text.size	text size for the plots.
out.dir	character string; the directory to save the figure, e.g., getwd(). Default is NULL. If NULL, figure will not be saved.
file.name	character string; name of the file to be saved.
width	the width of the graphics region in inches. See R function ggsave.
height	the height of the graphics region in inches. See R function ggsave.

Value

gtable of aligned plots from ggarrange.

Author(s)

Lu Yang, Jun Chen

References

Yang, L. & Chen, J. 2022. A comprehensive evaluation of differential abundance analysis methods: current status and potential solutions. Microbiome. Microbiome, 10(1), 1-23.

See Also

ZicoSeq

Examples

```
data(throat.otu.tab)
data(throat.tree)
data(throat.meta)

comm <- t(throat.otu.tab)
meta.dat <- throat.meta

set.seed(123)
# For count data
zico.obj <- ZicoSeq(meta.dat = meta.dat, feature.dat = comm,
grp.name = 'SmokingStatus', adj.name = 'Sex', feature.dat.type = "count",
```

```
# Filter to remove rare taxa
prev.filter = 0.2, mean.abund.filter = 0,  max.abund.filter = 0.002, min.prop = 0,
# Winsorization to replace outliers
is.winsor = TRUE, outlier.pct = 0.03, winsor.end = 'top',
# Posterior sampling to impute zeros
is.post.sample = TRUE, post.sample.no = 25,
# Multiple link functions to capture diverse taxon-covariate relation
link.func = list(function (x) x^0.25, function (x) x^0.5, function (x) x^0.75),
stats.combine.func = max,
# Permutation-based multiple testing correction
perm.no = 99,  strata = NULL,
# Reference-based multiple stage normalization
ref.pct = 0.5, stage.no = 6, excl.pct = 0.2,
# Family-wise error rate control
is.fwer = FALSE,
verbose = TRUE, return.feature.dat = TRUE)

which(zico.obj$p.adj.fdr <= 0.1)

ZicoSeq.plot(ZicoSeq.obj = zico.obj, pvalue.type = 'p.adj.fdr',
             cutoff = 0.1, text.size = 10, out.dir = NULL, width = 15, height = 10)
```

Index