Package 'miMeta'

November 26, 2023

Type Package

Title Meta-analysis of microbiome association studies

Version 0.1.0

Author Zhoujingpeng Wei, Guanhua Chen, Zheng-Zheng Tang

Maintainer Zhoujingpeng <zwei74@wisc.edu>

Description This R package implements the new methods for meta-analysis of microbiome association studies that respect the unique features of microbiome data such as compositionality.

Data 2023-09-23

Reference Wei Z, Chen G, Tang ZZ. Melody identifies generalizable microbial signatures in microbiome association meta-analysis. Submitted.

License GPL (>= 3) | file LICENSE

Imports MASS,
brglm2,
abess,
R6,
UpSetR,
dplyr,
ggplot2,
doParallel,
parallel,
foreach

RoxygenNote 7.2.3

Suggests knitr,
rmarkdown

BugReports https://github.com/ZjpWei/miMeta/issues

URL https://github.com/ZjpWei/miMeta

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

R topics documented:

`CRC_abd` *Metagenomics studies of colorectal cancer (CRC)*

Description

The "CRC_abd" is a feature-by-sample matrix of species-level relative abundance counts from the five studies. For the easy of demonstration, we include 267 species under order ‘Clostridiales‘. The "meta" is a data frame including the sample-level variables from the five studies.

Usage

```
data(CRC_abd, meta)
```

Format

An object of class `matrix` (inherits from `array`) with 267 rows and 572 columns.

Source

<https://github.com/zellerlab/crc_meta>

References

Wirbel, Jakob et al. Nat Med. 2019 Apr;25(4):679-689.

`melody` *meta-analysis for selecting microbial signatures*

Description

Melody is a meta-analysis method designed to account for the unique features of compositional microbiome data in selecting microbial signatrues.

Usage

```
melody(
  rel.abd,
  sample.data,
  sample.id,
  study,
  disease,
  covariates = NULL,
  cluster = NULL,
```

```
  depth.filter = 0,
  prev.filter = 0,
  ref = NULL,
  cov.type = c("diag", "ridge"),
  tune.path = c("gsection", "sequence"),
  tune.size.sequence = NULL,
  tune.size.range = NULL,
  tune.type = c("BIC", "HBIC", "KBIC", "EBIC"),
  tol = 0.001,
  parallel.core = NULL,
  ouput.best.one = TRUE,
  verbose = FALSE
)
```

Arguments

`rel.abd`	Microbial feature-by-sample matrix of relative abundance counts. Must have microbial feature IDs as row names and sample IDs as column names.
`sample.data`	A dataframe. data frame of sample-level variables. Columns must include variables for study, disease, covariates (if specified), and cluster (if specified). The row names must agree with column names in `rel.abd` matrix.
`sample.id`	Name of the variable that defines the sample's ID in `sample.data`.
`study`	Name of the variable that defines different studies in `sample.data`.
`disease`	Name of the variable that defines the disease to identify microbial signature.
`covariates`	A vector of names of the variables to adjust for in the microbiome-disease association model. Default is NULL.
`cluster`	Name of the variable that define the sample clusters. For example, the values of this variable are subject IDs if each subject has multiple correlated samples (e.g., measured in a longitudinal study). For a study only contains independent samples, the value of this variable is unique for each sample (can set to sample ID) or set to NA for that study. Default is NULL (all samples in all studies are independent).
`depth.filter`	A cutoff value (>= 0) to remove samples with sequencing depth less than or equal to the cutoff. Default is 0 (not removing any samples).
`prev.filter`	A cutoff value to remove microbial features with prevalence (proportion of nonzero observations in rel.abd matrix) less than or equal to the cutoff. The cutoff value must be in the range of 0-1. Default is 0 (only removing microbial features that have zero in all samples).
`ref`	A vector with length L (Study number L>=1) or a character or NULL. The name of reference taxa/features in each study when generating summary statistics by multinomial logistic regression. If ref is a character, all studies will use ref as reference taxon; if ref is a vector with length L, the study l will use `ref_l` as reference taxon; if ref is NULL, melody will pick reference taxa automatically (Check argument ref.iden to see how melody picks reference taxa).Default is NULL.
`cov.type`	The type of covariate matrix of summary statistics in each study. Options include "ridge", "diag". If cov.type is "ridge", melody will do ridge regularization to decide full covriate matrix of summary statistics, if cov.type is "diag", melody will use the diagonal covariate matrix in summary statistics. Default is "diag".

`tune.path` Method to choose the optimal subset size in the best subset selection. If tune.path="sequence", we solve the best subset selection problem for each size in tune.size.sequence. If tune.path="gsection", we solve the best subset selection problem with the size ranged in tune.size.range, where the considered size sequence within the range is determined by golden section search. Default is "gsection".

`tune.size.sequence` An integer vector containing the subset sizes to be considered. Only used when tune.path="sequence". Default is 1:round(K/2), where K is number of microbial features.

`tune.size.range` An integer vector with two elements that define the range of the size. Default is c(1, K/2)

`tune.type` Type of information criterion for choosing the optimal tunning parameters. Available options are "BIC", "HBIC", "KBIC" and "EBIC". Default is "BIC".

`tol` Converge tolerance for detecting the best model. Default is 1e-3.

`parallel.core` The number of cores to be concurrently used for generating summary statistics. It's an integer between 1 and cl - 1 (cl is the total core number). parallel.core = 1: no parallel. Default is cl - 1.

`ouput.best.one` Whether only output the best model. Default is TRUE.

`verbose:` whether to print verbose information. Default is FALSE. (see details in Value)

Details

Melody first generates summary statistics (microbiome-disease association coefficient estimates and their variances) for individual studies. Melody then combines the summary statistics across studies to select disease-associated microbial signatures based on the average absolute-abundance association coefficients inferred from the summary statistics. In particular, the selection of signature is operated through a best-subset selection. There are two sets of tunning parameters in the best subset selection including the subset size (i.e. the number of microbial features selected) and delta's which are introduced to recover the absolute-abundance coefficients from the relative-abundance coefficients.

Value

If output.best.one=TRUE (default), output a list with the following components about the single best model over the subset sizes considered.

`coef` A coefficient vector that contains the absolute-abundance coefficient estimates for the microbial features under the best subset size. The vector names are microbial features IDs.

`delta` A vector that contains the best values of the delta tunning parameters under the best subset size. The vector names are in the format "<study ID>_<reference microbial feature ID>"

`dev` The value of the deviance for the best subset size.

`ic` The value of the information criterion (specified in tune.type) for the best subset size.

`best.size` The best subset size.

If ouput.best.one=FALSE, output a list object with the following components for multiple best subset models, each of which is under a specific subset size considered (depends on tune.path, see details in Arguments).

coef	A coefficient matrix with each column contains the absolute-abundance coefficient estimates for the microbial features under a specific subset size. The row names are microbial features IDs and the column names are the subset sizes considered.
delta	A matrix with each column contains the best values of the delta tunning parameters under a specific subset size. The row names are in the format "<study ID>_<reference microbial feature ID>" and the column names are the subset sizes considered.
dev	A vector contains the values of the deviance for the subset sizes considered (shown as vector names).
ic	A vector contains the values of the information criterion (specified in tune.type) for the subset sizes considered (shown as vector names).
best.size	The best subset size.

If verbose=TRUE, Generate two plots and print information about the progress of meta-analysis. plot for microbial feature overlap among studies: this plot shows the number of features shared among studies. a plot showing the absolute-abundance coefficient estimates of the selected microbial features in the best model under the best subset size.

Author(s)

Zhoujingpeng Wei, Guanhua Chen, Zheng-Zheng Tang

References

Wei Z, Chen G, Tang ZZ. Melody identifies generalizable microbial signatures in microbiome association meta-analysis. Submitted.

See Also

melody.get.summary, melody.meta.summary, melody.merge.summary,

Examples

```
library("miMeta")
data("CRC_abd")
data("meta")

meta_FR <- meta[meta$Study == "FR-CRC",]
CRC_abd_FR <- CRC_abd[,meta_FR$Sample_ID]

Melody.model <- melody(rel.abd = CRC_abd_FR,
                       sample.data = meta_FR,
                       sample.id = "Sample_ID",
                       study = "Study",
                       disease = "Group")
```

`melody.get.summary` *Generate Melody summary statistics for one or multiple studies.*

Description

The melody.get.summary, melody.merge.summary, and melody.meta.summary are three functions that represent individual components in the melody pipeline. The function melody.get.summary takes individual-level data from one or multiple studies and outputs summary statistics for the studies in a Melody object. The output can be directly used by the functions melody.merge.summary and melody.meta.summary.

Usage

```
melody.get.summary(
  rel.abd,
  sample.data,
  sample.id,
  study,
  disease,
  covariates = NULL,
  cluster = NULL,
  depth.filter = 0,
  prev.filter = 0,
  ref = NULL,
  cov.type = c("diag", "ridge"),
  parallel.core = NULL,
  verbose = FALSE
)
```

Arguments

...	Same arguments as the function melody

Value

A Melody class.

`summary.stat.study`
: A list includes summary statistics for each study.

`dat.inf`
: A list includes study number; reference feature for each study; feature number and feature names.

`taxa.set`
: A list includes the which features are included or not in each study.

See Also

`melody`, `melody.meta.summary`, `melody.merge.summary`

Examples

```
library("miMeta")
data("CRC_abd")
data("meta")

########## Generate summary statistics for study FR ##########
meta_FR <- meta[meta$Study == "FR-CRC",]
CRC_abd_FR <- CRC_abd[,meta_FR$Sample_ID]

sumstats <- melody.get.summary(rel.abd = CRC_abd_FR,
                               sample.data = meta_FR,
                               sample.id = "Sample_ID",
                               study = "Study",
                               disease = "Group",
                               verbose = TRUE)
```

`melody.merge.summary` *Merge summary statistics across studies.*

Description

This function takes a list of Melody objects generated by the function melody.get.summary, align summary statistics by feature IDs, and output a Melody object containing summary statistics of all particpating studies. The output can be directly used by the function melody.meta.summary.

Usage

```
melody.merge.summary(melody.obj.lst, verbose = FALSE)
```

Arguments

`melody.obj.lst`	A list of summary statistics. Each element is a Melody object from `melody.get.summary`.
`verbose`	whether to generate a plot for microbial feature overlap among studies. Default is FALSE.

Value

A Melody object that contains the summary statistics over multiple studies participating the meta-analysis.

`summary.stat.study`
: A list includes summary statistics for each study.

If verbose=TRUE, generate a plot for microbial feature overlap among studies.

See Also

`melody`, `melody.get.summary`, `melody.meta.summary`

Examples

```
data("CRC_abd")
data("meta")

########## Generate summary statistics for study FR ##########
meta_FR <- meta[meta$Study == "FR-CRC",]
CRC_abd_FR <- CRC_abd[,meta_FR$Sample_ID]

sumstats_FR <- melody.get.summary(rel.abd = CRC_abd_FR,
                                  sample.data = meta_FR,
                                  sample.id = "Sample_ID",
                                  study = "Study",
                                  disease = "Group",
                                  verbose = TRUE)

########## Generate summary statistics for study DE ##########
meta_DE <- meta[meta$Study == "DE-CRC",]
CRC_abd_DE <- CRC_abd[,meta_DE$Sample_ID]

sumstats_DE <- melody.get.summary(rel.abd = CRC_abd_DE,
                                  sample.data = meta_DE,
                                  sample.id = "Sample_ID",
                                  study = "Study",
                                  disease = "Group",
                                  verbose = TRUE)

########## Merge summary statistics ##########
sumstats_merge <- melody.merge.summary(list(sumstats_FR, sumstats_DE))
```

`melody.meta.summary` *Meta-analyze summary statistics across studies*

Description

This function takes a Melody object that contains summary statistics of all studies participating the meta-analysis, then combines these summary statistics to select microbial signatures (see the Description of the function melody)

Usage

```
melody.meta.summary(
  Melody,
  tune.path = c("gsection", "sequence"),
  tune.size.sequence = NULL,
  tune.size.range = NULL,
  tune.type = c("BIC", "HBIC", "KBIC", "EBIC"),
  ouput.best.one = TRUE,
  tol = 0.001,
  verbose = FALSE
)
```

Arguments

Melody	Melody object.
...	Same arguments as the function melody

Value

Same output as the function melody

See Also

melody, melody.get.summary, melody.merge.summary

Examples

```
data("CRC_abd")
data("meta")

########## Generate summary statistics for study FR ##########
meta_FR <- meta[meta$Study == "FR-CRC",]
CRC_abd_FR <- CRC_abd[,meta_FR$Sample_ID]

sumstats_FR <- melody.get.summary(rel.abd = CRC_abd_FR,
                                  sample.data = meta_FR,
                                  sample.id = "Sample_ID",
                                  study = "Study",
                                  disease = "Group",
                                  verbose = TRUE)

########## Generate summary statistics for study DE ##########
meta_DE <- meta[meta$Study == "DE-CRC",]
CRC_abd_DE <- CRC_abd[,meta_DE$Sample_ID]

sumstats_DE <- melody.get.summary(rel.abd = CRC_abd_DE,
                                   sample.data = meta_DE,
                                   sample.id = "Sample_ID",
                                   study = "Study",
                                   disease = "Group",
                                   verbose = TRUE)

########## Merge summary statistics ##########
sumstats_merge <- melody.merge.summary(list(sumstats_FR, sumstats_DE))

########## Meta-analysis ##########
Melody.model <- melody.meta.summary(Melody = sumstats_merge)
```