Package 'mbOmic'

March 20, 2023

Type Package

Title Integrative analysis of the microbiome and metabolome

Version 1.2.0

Description The mbOmic package contains a set of analysis functions for microbiomics and metabolomics data, designed to analyze the inter-omic correlation between microbiology and metabolites. Integrative analysis of the microbiome and metabolome is the aim of mbOmic. Additionally, the identification of enterotype using the gut microbiota abundance is preliminaryimplemented.

License Artistic-2.0

Maintainer@R CongcongGong <congconggong33@gmail.com>

URL https://github.com/gongcongcong/mbOmic

BugReports https://github.com/gongcongcong/mbOmic/issues

Encoding UTF-8

RoxygenNote 7.1.2

Imports parallel, doParallel, psych, WGCNA, data.table, igraph, visNetwork, cluster, clusterSim, methods, graphics, stats

Suggests testthat (>= 3.0.0), knitr, rmarkdown, devtools, impute

VignetteBuilder knitr

biocViews Metabolomics, Microbiome, Network

Config/testthat/edition 3

Depends R (>= 4.1.0)

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

git_branch RELEASE_3_16

git_last_commit be27122

git_last_commit_date 2022-11-01

Date/Publication 2023-03-20

NeedsCompilation no

Author Congcong Gong [aut, cre] (<https://orcid.org/0000-0002-9483-1424>)

Maintainer Congcong Gong <congconggong33@gmail.com>

R topics documented:

`bSet` *bSet*

Description

Function to return 'bSet' class.

Usage

```
bSet(b, ...)
```

Arguments

b	data.table, metabolites aundance matrix. if ‘rn‘ column is not contained in this data.table, the ‘Features‘ parameter should be given by character vector.
...	‘Samples‘ or ‘Features‘. if the ‘Samples‘ not given, the colnames of ‘b‘ data.table will be taken as the Samples names.

Value

a ‘bSet‘ class

Examples

```
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(genera)[1] <- 'rn'
bSet(b = genera)
```

---

bSet-class *bSet*

Description

‘bSet‘ class is similar to the ‘mSet‘ class but it store the OTU abundance matrix rather than the metabolite abundance.

Value

S4 class

Slots

`Samples` character a character vector contains the samples

`Features` character a character vector contains the features

`dt` data.table OTU abundance matrix

Examples

```
b.path <- system.file("extdata", "metabolites_and_genera.rda", package = "mbOmic")
load(b.path)
names(genera)[1] <- 'rn'
bSet(b = genera)
```

clean_analytes *clean_analytes*

Description

Remove unqualified metabolites and OTU which only measured in few samples.

Usage

```
clean_analytes(object, fea_num = 2)
```

Arguments

object	a ‘bSet‘ or ‘mSet‘ class
fea_num,	integer, Removal of features only measured in ‘fea_num‘ of samples

Value

return a quantified ‘bSet‘ or ‘mSet‘ object

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(genera)[1] <- 'rn'
names(metabolites)[1] <- 'rn'
b <- bSet(b = genera)
m <- mSet(m = metabolites)
clean_analytes(b)
```

cluster_jsd *cluster_jsd*

Description

‘cluster_jsd‘ cluster the sample using the pam based the JSD.

Usage

```
cluster_jsd(dist, b, k)
```

Arguments

dist	distance matrix, if miss distance matrix the bacterial abundance matrix ‘b‘ was used to calculate the Jensen-Shannon divergence
b	bacterial abundance matrix, if the distance matrix ‘dist‘ was given, it is useless
k	number of cluster

Value

cluster vector

Examples

```
data(iris)
## using the matrix to cluster samples.
b <- bSet(b = data.table::as.data.table(t(iris[1:6, 1:4])),
          Features = letters[1:4],
          Samples = LETTERS[1:6])
res_cluster <- cluster_jsd(b=b, k = 3)
## or using the resoult of dist_jsd
## dist <- dist_jsd(iris[, 1:4])
## res_cluster <- cluster_jsd(dist = iris[,1:4], k = 3)
table(res_cluster, iris[1:6, 5])
```

`coExpress` *coExpress*

Description

'coExpress' identify the co-expression metabolites by performing the metabolites adjant matrix basing their relative abundace. This process was mainly implemented using the WGCNA package.

Usage

```
coExpress(
  object,
  power = NULL,
  powerVec = seq_len(30),
  threshold = 0.8,
  message = TRUE,
  plot = FALSE,
  ...
)
```

Arguments

`object`	mbSet class
`power`	integer if the pickSoftThreshold function (WGCNA) can find appropriate power, this param is invalid
`powerVec`	vector was passed to PickST function to get the power value
`threshold`	numeric as the threshold to filter power value
`message`	logical whether to show verbose info
`plot`	logical whether plot in PickST function
`...`	additional arguments passed to WGCNA

Value

network

Author(s)

Congcong Gong

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(genera)[1] <- 'rn'
names(metabolites)[1] <- 'rn'
b <- bSet(b = genera)
m <- mSet(m = metabolites)
res <- corr(m, b, method = 'spearman')
net <- coExpress(m, minN = 2, power = 9, message = FALSE)
```

---

coExpress,Set-method *coExpress*

Description

coExpress

Usage

```
## S4 method for signature 'Set'
coExpress(
  object,
  power = NULL,
  powerVec = seq_len(30),
  threshold = 0.8,
  message = TRUE,
  plot = FALSE,
  ...
)
```

Arguments

object	mbSet class
power	integer, if the pickSoftThreshold function (WGCNA) can find appropriate power, this param is invalid
powerVec	vector was passed to PickST function to get the power value
threshold	numeric as the threshold to filter power value
message	logical, whether to show verbose info

plot	logical, whether plot in PickST function
...	args passed to WGCNA

Value

network

---

cor2df *cor2df*

Description

Coverting the correlation result to data table class.

Usage

```
cor2df(res)
```

Arguments

res	output of corr.test

Value

data table

---

corr *corr*

Description

Calculating spearman correlation between each OTU and each metabolites.

Usage

```
corr(m, b, method = "spearman", parallel = FALSE, ncore = 4)
```

Arguments

m	mSet class
b	bSet class
method	character default is spearman
parallel	logical
ncore	integer number of core,default is 4

Value

the correlation data table

Examples

```
library(data.table)
path <- system.file('extdata','metabolites_and_genera.rda',package = 'mbOmic')
load(path)
names(genera)[1] <- 'rn'
names(metabolites)[1] <- 'rn'
b <- bSet(b = genera)
m <- mSet(m = metabolites)
res <- corr(m, b, method = 'spearman')
```

---

corr,mSet,bSet-method *corr*

Description

genetic methods to perform the correlation test.

Usage

```
## S4 method for signature 'mSet,bSet'
corr(m, b, method = "spearman", parallel = FALSE, ncore = 4)
```

Arguments

m	mSet class
b	bSet class
method	character default is spearman
parallel	logical
ncore	integer number of core,default is 4

Value

correlation matrix

`dist_jsd` *dist_jsd Calculating Jensen-Shannon divergence basing the formulate* $JSD(P||Q) = \sqrt{0.5 \times KLD(P||\frac{P+Q}{2}) + 0.5 \times KLD(Q||\frac{P+Q}{2})}$.

Description

dist_jsd

Calculating Jensen-Shannon divergence basing the formulate $JSD(P||Q) = \sqrt{0.5 \times KLD(P||\frac{P+Q}{2}) + 0.5 \times KLD(Q||\frac{P}$

Usage

```
dist_jsd(b)
```

Arguments

b	a ‘bSet‘ class

Value

Jensen-Shannon divergence data frame

Examples

```
data(iris)
b <- data.table::as.data.table(iris[1:6, 1:4])
b <- bSet(b = b,
          Features = letters[1:6],
          Samples = LETTERS[1:4])
dist <- dist_jsd(b)
```

`enterotyping` *enterotyping*

Description

identifing enterotype basing the OTU abundance using the cluster anlysis refer to the XXX works.

Usage

```
enterotyping(b, cluster, threshold = 0.02)
```

Arguments

b	bacterial abundance matrix
cluster	cluster vector
threshold	abundance threshold

Value

list

Examples

```
dat <- read.delim('http://enterotypes.org/ref_samples_abundance_MetaHIT.txt')
```

`estimate_k` *estimate_k*

Description

To estimate the optimal cluster number , 'estimate_k' takes advantage of two measures, Calinski-Harabasz (CH) Index and silhouette coefficients.

Usage

```
estimate_k(b, verK = 2:10)
```

Arguments

b	bacterial abundance matrix
verK	number vector of cluster

Value

list

Examples

```
data(iris)
b <- bSet(b = data.table::as.data.table(t(iris[1:6, 1:4])),
          Features = letters[1:4],
          Samples = LETTERS[1:6])
ret <- estimate_k(b = b, 2:3)
```

features *features*

Description

'features' get or set features names vector.

'features' get or set features names vector.

Usage

```
features(object)
features(object) <- value

features(object)
features(object) <- value
```

Arguments

object	a 'Set' object
value	character vector

Value

'features' return a character vector or update the features names

'features' return a character vector or update the features names

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(metabolites)[1] <- 'rn'
m <- mSet(m = metabolites)
features(m)
```

mb_initialize *mb_initialize initialized function for 'mSet' and 'bSet' classs.*

Description

mb_initialize

initialized function for 'mSet' and 'bSet' classs.

Usage

```
mb_initialize(type, dt, Samples, Features)
```

Arguments

type	character ‘mSet‘ or ‘bSet‘
dt	data.table abundance matrix
Samples	character samples names
Features	character Features names

Value

object

---

mSet *mSet*

Description

Function to return ‘mSet‘ class.

Usage

```
mSet(m, ...)
```

Arguments

m	data.table, metabolites aundance matrix. if ‘rn‘ column is not contained in this data.table, the ‘Features‘ parameter should be given by character vector.
...	‘Samples‘ or ‘Features‘. if the ‘Samples‘ not given, the colnames of ‘m‘ data.table will be taken as the Samples names.

Value

a ‘mSet‘ class

Examples

```
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(metabolites)[1] <- 'rn'
mSet(m = metabolites)
```

mSet-class *mSet*

Description

'mSet' is a S4 class extended from the virtual 'Set' used as object to store metabolites abundance matrix.

Value

S4 class

Slots

`Samples` character a character vector contains the samples

`Features` character a character vector contains the features

`dt` data.table metabolties abundance matrix

Examples

```
m.path <- system.file("extdata", "metabolites_and_genera.rda", package = "mbOmic")
load(m.path)
names(metabolites)[1] <- 'rn'
bSet(b = metabolites)
```

ncol,Set-method *ncol*

Description

ncol method for 'Set' to obtain the number of samples. nrow and ncol return the number of features or samples of a 'Set' class.

Usage

```
## S4 method for signature 'Set'
ncol(x)
```

Arguments

x	a 'Set' Object

Value

an 'integer' of length 1

nrow,Set-method *nrow*

Description

nrow method for 'Set' to obtain the number of features. nrow and ncol return the number of features or samples of a 'Set' class.

Usage

```
## S4 method for signature 'Set'
nrow(x)
```

Arguments

x	a 'Set' Object

Value

an 'integer' of length 1

pickST *pickST*

Description

Picking up the Soft threshold of metabolites abundance matrix.

Usage

```
pickST(m, threshold.d = 0.05, threshold = 0.8, plot = TRUE, powers = NULL)
```

Arguments

m	data.table metabolites abundance
threshold.d	numeric rol
threshold	numeric threshold
plot	logical whether to plot
powers	vector

Value

integer

plot_coExpress *plot_coExpress*

Description

Plotting the dendro of metabolites modules.

Usage

```
plot_coExpress(net)
```

Arguments

net	output of coExpress function

Value

ploting

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(metabolites)[1] <- 'rn'
m <- mSet(m = metabolites)
net <- coExpress(m, minN = 2, power = 9, message = FALSE)
```

plot_network *plot_network*

Description

plotting the network of metabolites and OTU. Orange nodes represent the OTU, while the other color represent the metabolite. Same color metabolites nodes are constructed in the same modules.

Usage

```
plot_network(net, corr, seed = 123, return = FALSE, interaction = TRUE)
```

Arguments

net	result of function ‘coExpress‘
corr	result of function ‘corr.test‘
seed	set seed for layout in interaction ploting
return	whether return the igraph
interaction	plot method

Value

igraph or graph

Author(s)

Congcong Gong

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(genera)[1] <- 'rn'
names(metabolites)[1] <- 'rn'
b <- bSet(b = genera)
m <- mSet(m = metabolites)
res <- corr(m, b, method = 'spearman')
net <- coExpress(m, minN = 2, power = 9, message = FALSE)
plot_network(net, res[abs(rho)>=0.85])
```

`print.verCHI` *print.verCHI*

Description

print the optialmal numer of cluster, max CHI, and Silhouette and plot the Calinski-Harabasz (CH) Index and silhouette coefficients simultaneously.

Usage

```
## S3 method for class 'verCHI'
print(x, ..., verbose = TRUE, plotting = TRUE, cluster)
```

Arguments

`x`	estimate_k output
`...`	print
`verbose`	logical
`plotting`	logical
`cluster`	cluster result

Value

no return

Examples

```
data(iris)
b <- bSet(b = data.table::as.data.table(t(iris[1:6, 1:4])),
          Features = letters[1:4],
          Samples = LETTERS[1:6])
ret <- estimate_k(b =b, 2:3)
ret
```

---

quiteRun *quiteRun Run expression in quite mode.*

Description

quiteRun

Run expression in quite mode.

Usage

```
quiteRun(x)
```

Arguments

x	expression

Value

no return

---

samples *samples*

Description

'samples' get or set samples names vector.

'samples' get or set samples names vector.

Usage

```
samples(object)
samples(object) <- value

samples(object)
samples(object) <- value
```

Arguments

object	a ‘Set‘ object
value	character vector

Value

‘samples‘ return a character vector or update the samples names

‘samples‘ return a character vector or update the samples names

Examples

```
library(data.table)
path <- system.file('extdata', 'metabolites_and_genera.rda', package = 'mbOmic')
load(path)
names(metabolites)[1] <- 'rn'
m <- mSet(m = metabolites)
features(m)
```

Set-class *Set*

Description

‘Set‘ is a virtual class as the base class to extend the ‘mSet‘ and ‘bSet‘.

Value

no return

Slots

`Samples` character a character vector contains the samples

`Features` character a character vector contains the features

`dt` data.table

Examples

```
##`Set` is virtual class.
```

show,Set-method *show*

Description

show method for the ‘Set‘ class.

Usage

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

Arguments

object	a ‘Set‘ class

Value

none return but print the main info of ‘Set‘ class

vmh-class *vmh*

Description

‘vmh‘ is a API for vmh

Value

vmh class

Slots

type a character, "microbe", "metabolites", "gene", or "food items"

net a data.table class contain the network of microbe, metabolites, gene, and food items

Examples

```
b.path <- system.file("extdata", "metabolites_and_genera.rda", package = "mbOmic")
load(b.path)
names(genera)[1] <- 'rn'
bSet(b = genera)
```

`[.Set` *subset_Set*

Description

subset ‘Set‘ class.

Usage

```
## S3 method for class 'Set'
x[i, j]
```

Arguments

`x`	a ‘mSet‘ or ‘bSet‘ class
`i`	the data.table i
`j`	the data.table j

Value

return a subset object

Index