Package ‘metabolomicsR’

April 29, 2022

Type Package

Title Tools for Metabolomics Data

Version 1.0.0

Date 2022-04-27

Maintainer Xikun Han <hanxikun2017@gmail.com>

URL https://github.com/XikunHan/metabolomicsR

Description Tools to preprocess, analyse, and visualize metabolomics data. We included a set of functions for sample and metabolite quality control, outlier detection, missing value imputation, dimensional reduction, normalization, data integration, regression, metabolite annotation, enrichment analysis, and visualization of data and results. The package is designed to be a comprehensive R package that can be easily used by researchers with basic R programming skills. The framework designed here is versatile and is extensible to other various methods.

License GPL-2

Encoding UTF-8

Depends methods, R (>= 4.1)

Imports ggplot2, data.table, plotROC, utils, stats

Suggests ggthemes, knitr, rmarkdown, testthat (>= 3.0.0), lme4, nlme, broom, reshape2, impute, M3C, FNN, RColorBrewer, readxl, survival, future, pbapply, future.apply, progressr, ggrepel, here, genuMet, ggstatsplot, cowplot, pROC, BiocStyle, MASS, xgboost

RoxygenNote 7.1.2

biocViews Software, Metabolomics, MassSpectrometry, Regression, Normalization, QualityControl

LazyData FALSE

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Xikun Han [cre, aut]
## R topics documented:

- assayData
- assayData<-
- batch_norm
- bridge
- column_missing_rate
- correlation
- create_Metabolite
- df_plasma
- featureData
- featureData<-
- filter_column_constant
- filter_column_missing_rate
- filter_row_missing_rate
- fit_lm
- impute
- inverse_rank_transform
- is_outlier
- load_data
- load_excel
- merge_data
- Metabolite-class
- modelling_norm
- nearestQC_norm
- outlier_rate
- pareto_scale
- plot_injection_order
- plot_Metabolite
- plot_PCA
- plot_ROC
- plot_tsne
- plot_UMAP
- plot_volcano
- QCmatrix_norm
- QC_pipeline
- regression
- replace_outlier
- row_missing_rate
- RSD
- run_PCA
- sampleData
- sampleData<-
- save_data
Description
Accessors for Metabolite object. Get the assayData in the Metabolite object.

Usage
assayData(object)

## S4 method for signature 'Metabolite'
assayData(object)

Arguments
object A Metabolite object.

Value
A data.table of assayData.

Description
Accessors for Metabolite object. ‘assayData<-’ will update the assayData in the Metabolite object.

Usage
assayData(object) <- value

## S4 replacement method for signature 'Metabolite'
assayData(object) <- value

Arguments
object A Metabolite object.
value The new assayData.
Value

A data.table of assayData.

Description

Normalization data by the median value of each batch

Usage

batch_norm(
  object,
  feature_platform = "PLATFORM",
  QC_ID_pattern = "MTRX",
  test = FALSE,
  verbose = TRUE
)

Arguments

object A Metabolite object. In the feature annotation slot ‘feature’, a platform column should be provided for metabolite measurement platform (eg. ‘PLATFORM’). The values in the ‘PLATFORM’ column (eg. ‘Neg’, ‘Polar’, ‘Pos Early’, and ‘Pos Late’) are column names in the sample annotation ‘sample’ to determine the batches of samples.

feature_platform The column name of feature platform for metabolite measurements (eg. ‘PLATFORM’).

QC_ID_pattern A character pattern to determine QC samples. Default value: "MTRX".

test test the function for the first 20 columns.

verbose print log information.

Value

A Metabolite object after normalization.

See Also

QCmatrix_norm
**bridge**

*bridge different data sets based on conversion factors*

**Description**

Bridge metabolite data based on a conversion factor file

**Usage**

```r
bridge(
  object,
  conversion_factor_data = NULL,
  QC_ID_pattern = "MTRX",
  verbose = TRUE
)
```

**Arguments**

- **object**: A Metabolite object. In the `featureData`, `conversion_factor_ID` column should be created to match with `conversion_factor_data`.
- **conversion_factor_data**: A data set with columns `conversion_factor_ID` and `conversion_factor_value`.
- **QC_ID_pattern**: A character pattern to determine QC samples. Default value: "MTRX". Skip QC samples when rescale (median value is already 1).
- **verbose**: print log information.

**Value**

A Metabolite object after multiplying by conversion factor.

---

**column_missing_rate**

*column missing rate*

**Description**

Calculate column missing rate – metabolite missingness.

**Usage**

```r
column_missing_rate(object)
```

```
## Default S3 method:
column_missing_rate(object)
```

```
## S3 method for class 'Metabolite'
column_missing_rate(object)
```
correlation

correlation of features between two Metabolite objects

Arguments

object object X The first Metabolite object.
object_Y The second Metabolite object.
method a character string to calculate correlation coefficient. One of "pearson" (default), "kendall", or "spearman".
verbose print log information.

Value

A data.table with correlation coefficients.

See Also

cor

correlation

Arguments

object An object, data.frame, data.table or Metabolite.

Value

Returns a vector of the missing rate for each column
A data.table of column missing rate.

Examples

# for a Metabolite object
data(df_plasma)
v <- column_missing_rate(df_plasma)

correlation

correlation of features between two Metabolite objects

Description

Calculate the correlation of features between two Metabolite objects

Usage

correlation(
  object_X = NULL,
  object_Y = NULL,
  method = "pearson",
  verbose = TRUE
)

Arguments

object_X The first Metabolite object.
object_Y The second Metabolite object.
method a character string to calculate correlation coefficient. One of "pearson" (default), "kendall", or "spearman".
verbose print log information.

Value

A data.table with correlation coefficients.
create_Metabolite

Create a Metabolite object

Description

Create a Metabolite object from three input data sets: 1) metabolite measurements (e.g., peak area data or normalized data), and 2) metabolite annotation (e.g., chemical annotation) 3) sample annotation (e.g., sample meta data)

Usage

create_Metabolite(
  assayData,
  featureData,
  sampleData,
  featureID,
  sampleID,
  logs
)

Arguments

assayData a data.frame or data.table of metabolite measurements (peak area data or normalized data, sample [row] * feature [column]).

featureData a data.frame or data.table of metabolite annotation (chemical annotation)

sampleData a data.frame or data.table of sample annotation (sample meta data).

featureID a character of the metabolite ID column (in feature file and the column names of data), default: CHEM_ID (provided from Metabolon file).

sampleID a character of the sample ID column (in sample and the first column of data), default: PARENT_SAMPLE_NAME (provided from Metabolon file).

logs Log information.

Value

A Metabolite object with slots: assayData, featureData, and sampleData.

A Metabolite object.

See Also

Metabolite, load_excel, load_data

Examples

# df <- create_Metabolite(assayData = df_data, featureData = df_feature, sampleData = df_sample)
**Example data.**

A dataset containing 356 samples and 758 features.

**Usage**

```r
data(df_plasma)
```

**Format**

An object of class `Metabolite` of length 1.

**featureData**

**Description**

Accessors for Metabolite object. Get the featureData in the Metabolite object.

**Usage**

```r
featureData(object)
```

**Arguments**

- `object` A Metabolite object.

**Value**

A data.table of featureData.
Description

Accessors for Metabolite object. ‘featureData<-’ will update the featureData in the Metabolite object.

Usage

featureData(object) <- value

## S4 replacement method for signature 'Metabolite'
featureData(object) <- value

Arguments

object A Metabolite object.
value The new featureData.

Value

A data.table of featureData.

filter_column_constant

filter columns if values are constant

Description

Remove columns if values are constant

Usage

filter_column_constant(object, verbose)

## Default S3 method:
filter_column_constant(object, verbose = TRUE)

## S3 method for class 'Metabolite'
filter_column_constant(object, verbose = TRUE)

Arguments

object An object, data.frame, data.table or Metabolite.
verbose print log information.
Examples

data(df_plasma)
v <- filter_column_constant(df_plasma)

filter_column_missing_rate

filter columns using missing rate

Description

Remove columns below a specific missing rate threshold.

Usage

filter_column_missing_rate(object, threshold, verbose)

## Default S3 method:
filter_column_missing_rate(object, threshold = 0.5, verbose = TRUE)

## S3 method for class 'Metabolite'
filter_column_missing_rate(object, threshold = 0.5, verbose = TRUE)

Arguments

object An object, data.frame, data.table or Metabolite.
threshold missing rate threshold, default is 0.5. Other values: 0.2, 0.8.
verbose print log information.

Value

An object after filtering column missing rate.

Examples

data(df_plasma)
d <- filter_column_missing_rate(df_plasma)
**filter_row_missing_rate**

Filter rows using missing rate

**Description**

Remove samples below a specific missing rate threshold.

**Usage**

```r
filter_row_missing_rate(object, threshold, verbose)
```

## Default S3 method:

```r
filter_row_missing_rate(object, threshold = 0.5, verbose = TRUE)
```

## S3 method for class 'Metabolite'

```r
filter_row_missing_rate(object, threshold = 0.5, verbose = TRUE)
```

**Arguments**

- **object**: An object, data.frame, data.table or Metabolite.
- **threshold**: missing rate threshold, default is 0.5. Other values: 0.2, 0.8.
- **verbose**: print log information.

**Examples**

```r
data(df_plasma)
v <- filter_row_missing_rate(df_plasma)
```

---

**fit_lm**

Available regression methods

**Description**

- 'fit_lm': linear regression model lm.
- 'fit_logistic': logistic regression model glm.
- 'fit_poisson': poisson regression model glm.
- 'fit_cox': proportional hazards regression model coxph.
- 'fit_lme': linear mixed-effects model lme.
- 'fit_glmer': logistic linear mixed-effects model glmer.
- 'fit_lmer': linear mixed-effects model lmer.
Usage

fit_lm(data = NULL, formula = NULL, keep = NULL)
fit_logistic(data = NULL, formula = NULL, keep = NULL)
fit_poisson(data = NULL, formula = NULL, keep = NULL)
fit_cox(data = NULL, formula = NULL, keep = NULL)
fit_lme(data = NULL, formula = NULL, keep = NULL, ...)
fit_glmer(data = NULL, formula = NULL, keep = NULL, ...)
fit_lmer(data = NULL, formula = NULL, keep = NULL, ...)

Arguments

data A data.table with all variables to be fitted.
formula A "formula" object to be fitted.
keep Variables to keep regression results.
... Further arguments passed to regression model.

Value
term estimate std.error statistic p.value n.

See Also

regression

impute

impute missing values

Description

impute missing values

Usage

impute(object, method)

## S3 method for class 'Metabolite'
impute(object, method = c("half-min", "median", "mean", "zero", "kNN"))

## Default S3 method:
impute(object, method = "half-min")
impute_kNN(object)
### inverse_rank_transform

**Description**

Rank-based inverse normal transformation for a metabolite.

**Usage**

```r
inverse_rank_transform(x)
```

**Arguments**

- `x` A vector

**Value**

A vector after transformation.

---

### Arguments

- **object** An object, a vector, data.frame, data.table or Metabolite.
- **method** Imputation method, the default method is half the minimum value ('half-min') of the metabolite. Currently support 'half-min', "median", "mean", "zero", "kNN".

**Value**

An object after imputing missing values.

**Note**


'impute_kNN': Imputation using nearest neighbor averaging (kNN) method, the input is a Metabolite object, assayData was first transposed to row as metabolites and column as samples.

**Examples**

```r
data(df_plasma)
d <- impute(df_plasma)
```
is_outlier

Description

is outlier

Usage

is_outlier(object, nSD = 5)

Arguments

object An object, a vector.
nSD N times of the SD as outliers.

Value

TRUE or FALSE for a vector.

load_data

Load metabolite data from three separate files

Description

Load metabolite data from three separate files (import files using 'fread' from data.table).

Usage

load_data(
  data_path = NULL,
  feature_path = NULL,
  sample_path = NULL,
  featureID = "CHEM_ID",
  sampleID = "PARENTSAMPLE_NAME"
)

Arguments

data_path Path to the metabolite measurements (peak area data or normalized data, sample [row] * feature [column])
feature_path Path to the metabolite annotation (chemical annotation)
sample_path Path to the sample annotation (sample meta data)
featureID a character of the metabolite ID column (in feature file and the column names of data file), default: CHEM_ID (provided from Metabolon file)
sampleID a character of the sample ID column (in sample file and the first column of data file), default: PARENTSAMPLE_NAME (provided from Metabolon file).
load_excel

Value

A Metabolite object with slots: assayData, featureData, and sampleData.

load_excel Load metabolite data from an excel file

Description

Load metabolite data from an excel file

Usage

load_excel(
  path,
  data_sheet = NULL,
  feature_sheet = NULL,
  sample_sheet = NULL,
  featureID = "CHEM_ID",
  sampleID = "PARENT_SAMPLE_NAME"
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Path to the xls/xlsx file.</td>
</tr>
<tr>
<td>data_sheet</td>
<td>A integer of xlsx sheet number for metabolite measurements (peak area data or normalized data, sample [row] * feature [column])</td>
</tr>
<tr>
<td>feature_sheet</td>
<td>A integer of xlsx sheet number for metabolite annotation (chemical annotation)</td>
</tr>
<tr>
<td>sample_sheet</td>
<td>A integer of xlsx sheet number for sample annotation (sample meta data)</td>
</tr>
<tr>
<td>featureID</td>
<td>a character of the metabolite ID column (in feature file and the column names of data file), default: CHEM_ID (provided from Metabolon file)</td>
</tr>
<tr>
<td>sampleID</td>
<td>a character of the sample ID column (in sample file and the first column of data file), default: PARENT_SAMPLE_NAME (provided from Metabolon file)</td>
</tr>
</tbody>
</table>

Value

A Metabolite object with slots: assayData, featureData, and sampleData.
merge_data

merge two Metabolite objects

Description

Merge two Metabolite objects.

Usage

merge_data(object_X = NULL, object_Y = NULL, all = TRUE, verbose = TRUE)

Arguments

object_X The first Metabolite object.
object_Y The second Metabolite object.
all logical; all = TRUE: keep all metabolites; all = FALSE, keep common metabolites that were present in both datasets.
verbose print log information.

Value

A Metabolite object after merging with slots: assayData, featureData, and sampleData.

Metabolite-class

The Metabolite class

Description

The Metabolite object is a representation of metabolomic data, metabolomic annotation, and sample annotation.

Value

A Metabolite class.

Slots

assayData a data.frame or data.table of metabolite measurements (peak area data or normalized data, sample [row] * feature [column]).
featureData a data.frame or data.table of metabolite annotation (chemical annotation)
sampleData a data.frame or data.table of sample annotation (sample meta data).
featureID a character of the metabolite ID column (in feature file and the column names of data),
default: CHEM_ID (provided from Metabolon file).
sampleID a character of the sample ID column (in sample and the first column of data), default:
PARENT_SAMPLE_NAME (provided from Metabolon file).
logs Log information of data analysis process.
miscData Ancillary data.
modelling_norm

See Also

Metabolite, load_excel, load_data

modelling_norm  LOESS normalization

Description

Normalization data by machine learning modelling, eg. locally estimated scatterplot smoothing (LOESS) on QC samples in each batch. For each metabolite, the values (eg. raw peak area data) were divided by the median value of QC samples in that batch. QC samples and metabolite batches should be specified (see parameters below).

Usage

modelling.norm(
  object,
  method = c("LOESS", "KNN", "XGBoost"),
  feature.platform = "PLATFORM",
  QC.ID_pattern = "MTRX",
  span = 0.75,
  degree = 2,
  k = 3,
  test = FALSE,
  verbose = TRUE
)

Arguments

object  A Metabolite object. In the feature annotation slot ‘feature’, a platform column should be provided for metabolite measurement platform (eg. ‘PLATFORM’). The values in the ‘PLATFORM’ column (eg. ‘Neg’, ‘Polar’, ‘Pos Early’, and ‘Pos Late’) are column names in the sample annotation ‘sample’ to determine the batches of samples.

method  Modelling method for the normalization, currently support LOESS and KNN.

feature.platform  The column name of feature platform for metabolite measurements (eg. ‘PLATFORM’).

QC.ID_pattern  A character pattern to determine QC samples. Default value: "MTRX".

span  default value 0.4

degree  default value 2

k  Number of neighbors in KNN modelling (default value 3)

test  test the function for the first 20 columns.

verbose  print log information.
Value
A Metabolite object after normalization.

See Also
batch_norm

nearestQC_norm nearest QC sample normalization

Description
Normalization data by the median value of the nearest QC samples. For each metabolite, the values (eg. raw peak area data) were divided by the median value of nearest QC samples (eg. the nearest three QC samples). To identify the nearest QC samples, '@assayData' should be ordered by the injection order.

Usage
nearestQC_norm(
    object,
    n_nearest_QCsample = 3,
    feature_platform = "PLATFORM",
    QC_ID_pattern = "MTRX",
    test = FALSE,
    verbose = TRUE
)

Arguments

object A Metabolite object. In the feature annotation slot ‘feature’, a platform column should be provided for metabolite measurement platform (eg. ‘PLATFORM’). The values in the ‘PLATFORM’ column (eg. ‘Neg’, ‘Polar’, ‘Pos Early’, and ‘Pos Late’) are column names in the sample annotation ‘sample’ to determine the batches of samples.

n_nearest_QCsample Number of nearest QC samples to calculate the median value. The default value is 3 (an outlier QC sample might be used if only n_nearest_QCsample = 1).

feature_platform The column name of feature platform for metabolite measurements (eg. ‘PLATFORM’).

QC_ID_pattern A character pattern to determine QC samples. Default value: "MTRX".

test test the function for the first 20 columns.

verbose print log information.
outlier_rate

Value

A Metabolite object after normalization.

See Also

batch_norm, QCmatrix_norm

Description

Calculate outlier rate.

Usage

outlier_rate(object, nSD)

## Default S3 method:
outlier_rate(object, nSD = 5)

## S3 method for class 'data.frame'
outlier_rate(object, nSD = 5)

## S3 method for class 'Metabolite'
outlier_rate(object, nSD = 5)

Arguments

object An object, vector, data.frame, data.table or Metabolite.
nSD N times of the SD as outliers.

Value

Returns a vector of the outlier rate.

A data.table of outlier rate.

Examples

# for a Metabolite object
data(df_plasma)
v <- outlier_rate(df_plasma)
### pareto_scale

**Description**

pareto scale transformation

**Usage**

```r
description(x)
```

**Arguments**

- `x` A vector

**Value**

A vector after transformation.

---

### plot_injection_order

**Description**

Injection order scatterplot. The `@sampleData` should be sorted by injection order, with a new column `ID` from 1 to N.

**Usage**

```r
plot_injection_order(
  object,
  color = "NEG",
  shape = "NEG",
  size = 0.6,
  ID_order = "ID_injection_order",
  feature_name = NULL,
  random_select = 16
)
```
plot_Metabolite

Arguments

object  A Metabolite object.
color   A column in '@sampleData' to show the color of points.
shape   A column in '@sampleData' to show the shape of points.
size    Point size.
ID_order Injection ID order in the '@sampleData'.
feature_name A vector of selected metabolites to plot. If NULL, will randomly select 16 (default) metabolites to plot.
random_select An integer, number of randomly selected metabolites to plot.

Value

A scatterplot.

Description

Plot a Metabolite object including boxplot (more to add.).

Usage

plot_Metabolite(
  object,
  plot = "boxplot",
  x = "NEG",
  feature_name = NULL,
  color = "NEG",
  shape = "NEG",
  fill = "NEG",
  random_select = 16,
  size = 0.6,
  n_row = 1,
  n_col = 1,
  ylab = "featureID",
  height = 10,
  width = 10,
  save_to_file = NULL
)
plot_PCA

Arguments

object  A Metabolite object.
plot  type of plot, current support ‘boxplot’ and ‘betweenstats’.
 x  The x-axis coordinate.
feature_name  A vector of selected metabolites to plot. If NULL, will randomly select 16 (default) metabolites to plot.
color  A column in '@sampleData' to show the color of points.
shape  A column in '@sampleData' to show the shape of points.
fill  A column in '@sampleData' to show the ‘fill’ for histogram.
random_select  An integer, number of randomly selected metabolites to plot.
size  Point size.
n_row  Number of rows of subfigures for ‘betweenstats’
n_col  Number of columns of subfigures for ‘betweenstats’
ylab  Column name to annotate the y-axis in ‘betweenstats’ (eg. "BIOCHEMICAL"), default column: "featureID”.
height  Height of the figure.
width  Width of the figure.
save_to_file  Path to save the figure.

Value

A boxplot of a Metabolite object

plot_PCA

Description

Plot first two principal components.

Usage

plot_PCA(object, color = "NEG", shape = "NEG", size = 1.5)

Arguments

object  A Metabolite object.
color  A column in '@sampleData' to show the color of points.
shape  A column in '@sampleData' to show the shape of points.
size  Point size.

Value

PCA plot.
Description

Plot Receiver Operating Characteristic (ROC) curve for metabolites with or without covariates

Usage

```r
plot_ROC(
  object = NULL,
  y = NULL,
  x = NULL,
  model_a = NULL,
  model_b = NULL,
  lab = NULL
)
```

Arguments

- **object**: A Metabolite object.
- **y**: A column name for the disease (0, 1).
- **x**: One variable name (if x is provided, model_a and model_b should be NULL or vice versa).
- **model_a**: Column names for model a (one or more covariates, as the first model).
- **model_b**: Column names for model b (one or more covariates, as the second model).
- **lab**: Title (eg. "BIOCHEMICAL"), default value is x.

Value

ROC.

Description

Plot t-distributed stochastic neighbor embedding. See more details in tsne.

Usage

```r
plot_tsne(object, color = "NEG", shape = "NEG", size = 1.5)
```
plot_UMAP

**Arguments**

- **object**: A Metabolite object.
- **color**: A column in '@sampleData' to show the color of points.
- **shape**: A column in '@sampleData' to show the shape of points.
- **size**: Point size.

**Value**

tSNE plot.

---

**Description**

Plot manifold approximation and projection (UMAP). See more details in `umap`.

**Usage**

```
plot_UMAP(object, color = "NEG", shape = "NEG", size = 1.5)
```

**Arguments**

- **object**: A Metabolite object.
- **color**: A column in '@sampleData' to show the color of points.
- **shape**: A column in '@sampleData' to show the shape of points.
- **size**: Point size.

**Value**

UMAP plot.
Description

volcano plot for regression results

Usage

plot_volcano(
  fit,
  x = "estimate",
  y = "p.value",
  p.value_log10 = TRUE,
  color = "outcome",
  label = "term",
  highlight = "significant",
  x_lab = "Effect size",
  y_lab = "-log10(P value)"
)

Arguments

fit regression summary results.

x The x-axis column, eg. effect size.

y The y-axis column, eg. p value.

p.value_log10 whether to transforme p.value by -log10.

color A column in fit to show different point colors. Set as NULL to turn off the color argument.

label A column in fit to label points.

highlight A column in fit to show the points to highlight. Values as 1 are highlighted.

x_lab labels for x-axis.

y_lab labels for y-axis.

Value

A volcano plot.
**QCmatrix_norm**

**QCmatrix normalization**

**Description**
Normalization data by the median value of QC samples in each batch. For each metabolite, the values (e.g., raw peak area data) were divided by the median value of QC samples in that batch. QC samples and metabolite batches should be specified (see parameters below).

**Usage**

```r
QCmatrix_norm(
  object,
  feature_platform = "PLATFORM",
  QC_ID_pattern = "MTRX",
  test = FALSE,
  verbose = TRUE
)
```

**Arguments**

- `object`: A Metabolite object. In the feature annotation slot ‘feature’, a platform column should be provided for metabolite measurement platform (e.g., ‘PLATFORM’). The values in the ‘PLATFORM’ column (e.g., ‘Neg’, ‘Polar’, ‘Pos Early’, and ‘Pos Late’) are column names in the sample annotation ‘sample’ to determine the batches of samples.

- `feature_platform`: The column name of feature platform for metabolite measurements (e.g., ‘PLATFORM’).

- `QC_ID_pattern`: A character pattern to determine QC samples. Default value: "MTRX".

- `test`: test the function for the first 20 columns.

- `verbose`: print log information.

**Value**
A Metabolite object after normalization.

**See Also**

- batch_norm
QC_pipeline

Quality control pipeline

Description

This function will run QC steps on a Metabolite object

Usage

QC_pipeline(
  object,
  filter_column_constant = TRUE,
  filter_column_missing_rate_threshold = 0.5,
  filter_row_missing_rate_threshold = NULL,
  replace_outlier_method = NULL,
  nSD = 5,
  impute_method = "half-min",
  verbose = TRUE
)

Arguments

object
  An object, data.frame, data.table or Metabolite.
filter_column_constant
  A logical value, whether to filter columns (features) with a constant value.
filter_column_missing_rate_threshold
  A numeric threshold to filter columns (features) below a missing rate, default: 0.5. Other values: 0.2, 0.8. If NULL, then skip this step.
filter_row_missing_rate_threshold
  A numeric threshold to filter rows (samples) below a missing rate. Default: NULL, to skip this step. Other values: 0.5, 0.2, 0.8.
replace_outlier_method
  Method to replace outlier value, see replace_outlier.
nSD
  Define the N times of the SD as outliers.
impute_method
  Imputation method, the default method is half the minimum value ('half-min') of the metabolite. Currently support 'half-min', "median", "mean", "zero".
verbose
  print log information.

Value

A Metabolite object after QC.
### Description

Run regression models with adjusting for covariates. ‘regression_each’ is used for one outcome. In ‘regression’, several outcomes can be specified to run together.

### Usage

```r
regression(
  object,
  phenoData = NULL,
  model = NULL,
  outcome = NULL,
  covars = NULL,
  factors = NULL,
  feature_name = NULL,
  time = NULL,
  verbose = TRUE,
  ncpus = 1,
  p.adjust.method = "bonferroni",
  ...
)
```

```r
regression_each(
  object,
  phenoData = NULL,
  model = NULL,
  formula = NULL,
  outcome = NULL,
  covars = NULL,
  factors = NULL,
  feature_name = NULL,
  time = NULL,
  verbose = TRUE,
  ncpus = 1,
  p.adjust.method = "bonferroni",
  ...
)
```

### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>A Metabolite object.</td>
</tr>
<tr>
<td><code>phenoData</code></td>
<td>A data.table with outcome and covariates. If <code>phenoData</code> is NULL, ‘@sample-Data’ will be used.</td>
</tr>
</tbody>
</table>
**replace_outlier**

Specify a regression model. See `fit_lm` for more details. 'auto' can be used to infer 'lm' or 'logistic' (with only 0, 1, and NA).

- **outcome**: Column name of the outcome variable.
- **covars**: Column names of covariates.
- **factors**: Variables to be treated as factor.
- **feature_name**: A vector of selected metabolites to run. If both feature_name and random_select are NULL, will run regression for all features.
- **time**: Column name of survival time, used in cox regression, see `coxph` for more details.
- **verbose**: Print log information.
- **ncpus**: Number of CPUS for parallele job.
- **p.adjust.method**: Adjust for P value method, see `p.adjust`.
- **formula**: A character or formula object to fit model (only used in 'regression_each')

**Value**

term estimate std.error statistic p.value n outcome p.value.adj.

**Examples**

data(df_plasma)
fit_lm <- regression(object = df_plasma, phenoData = NULL, model = "lm", outcome = "BMI", covars = c("AGE", "GENDER", "ETHNICITY"), factors = "ETHNICITY")

---

**replace_outlier**

change outlier values as NA or winsorize

**Description**

change outlier values as NA or winsorize

**Usage**

replace_outlier(object, method, nSD)

## Default S3 method:
replace_outlier(object, method = "winsorize", nSD = 5)

## S3 method for class 'data.frame'
replace_outlier(object, method = "winsorize", nSD = 5)

## S3 method for class 'Metabolite'
replace_outlier(object, method = "winsorize", nSD = 5)
Arguments

- **object**: An object, a vector, data.frame, data.table or Metabolite.
- **method**: Replace outlier value method, the default method is 'winsorize': replace the outlier values by the maximum and/or minimum values of the remaining values. 'as_NA': set as NA (do not use this method if using half-min imputation).
- **nSD**: Define the N times of the SD as outliers.

Value

An object after replacing outlier values.

Examples

```r
data(df_plasma)
d <- replace_outlier(df_plasma, method = "winsorize", nSD = 5)
```

row_missing_rate  row missing rate

Description

Calculate row missing rate – sample missingness.

Usage

```r
row_missing_rate(object)
```

## Default S3 method:
```r
row_missing_rate(object)
```

## S3 method for class 'Metabolite'
```r
row_missing_rate(object)
```

Arguments

- **object**: An object, data.frame, data.table or Metabolite.

Value

Returns a vector of the missing rate for each row

A data.table of row missing rate.

Examples

```r
# for a Metabolite object
data(df_plasma)
v <- row_missing_rate(df_plasma)
```
**RSD**  

**Description**

calculate RDS (  

**Usage**

RSD(x)

**Arguments**

x  

A vector

**Value**

A vector of RDS values.

---

**run_PCA**  

**Principal Components Analysis**

**Description**

Performs a principal components analysis on the Metabolite object.

**Usage**

run_PCA(  
  object,  
  nPCs = 10,  
  impute_method = "half-min",  
  log = TRUE,  
  scale = TRUE,  
  addPC = TRUE

)

**Arguments**

**object**  

A Metabolite object.

**nPCs**  

Number of principal components to be calculated. Default value 10.

**impute_method**  

Imputation method, the default method is half the minimum value (‘half-min’) of the metabolite. Currently support ‘half-min’, “median”, “mean”, “zero”. ‘NULL’ without imputation.

**log**

Performs natural logarithm transformation before PCA analysis.
sampleData

scale             scale feature in the PCA calculation.
addPC             If TRUE, merge PCs with '@sampleData' and return the 'object', else return 'PC'.

Value

A list of PCs and variances explained.

Examples

data(df_plasma)
d <- run_PCA(df_plasma)

Description

Accessors for Metabolite object. Get the sampleData in the Metabolite object.

Usage

sampleData(object)

## S4 method for signature 'Metabolite'
sampleData(object)

Arguments

object        A Metabolite object.

Value

A data.table of sampleData.
Description

Accessors for Metabolite object. ‘sampleData<-’ will update the sampleData in the Metabolite object.

Usage

sampleData(object) <- value

## S4 replacement method for signature 'Metabolite'
sampleData(object) <- value

Arguments

object A Metabolite object.
value The new sampleData.

Value

A data.table of sampleData.

---

save_data [Save metabolite data]

Description

Save metabolite data in separate txt files

Usage

save_data(object, file = "")

Arguments

object A Metabolite object
file Output file to save the metabolite measurements (suffixes: "_assay.txt", "_feature_annotation.txt", "_sample_annotation.txt", "_logs.txt").

Value

No return value.
## S4 method for signature 'Metabolite'

`show(object)`

### Arguments

- **object**: A Metabolite object.

### Value

print a Metabolite object.

---

### subset

`subset(object, subset, select)`

#### Usage

```
## S3 method for class 'Metabolite'
subset(object, subset, select)
```

#### Arguments

- **object**: An object, data.frame, data.table or Metabolite.
- **subset**: logical expression indicating rows to keep (samples). Expression will be evaluate in the `@sampleData`.
- **select**: expression indicating columns to select (features). See `subset`. Expression will be evaluate in the `@assayData`.

#### Value

An object after subsetting rows or columns.
transformation

**apply transformation to a Metabolite object**

**Description**
Apply transformation to Metabolite object

**Usage**

```r
transformation(object, method = "log")
```

**Arguments**

- `object`: A Metabolite object.
- `method`: Transform method, eg. "log", "pareto_scale", "scale", "inverse_rank_transform". A User defined method is also supported.

**Value**
A Metabolite object after transformation.

**Examples**

```r
data(df_plasma)
d <- transformation(df_plasma)
```

---

update_Metabolite

**Update a Metabolite object**

**Description**
Update a Metabolite object.

**Usage**

```r
update_Metabolite(object, dataset = NULL, action = NULL)
```
Arguments

- **object**: A Metabolite object
- **dataset**: A vector or data.table used for a specific action mode.
- **action**: Currently support:
  - "injection_order": `@sampleData` will be updated by the order of sampleID that provided in the injection order data
  - "keep_feature": feature ID list to keep
  - "remove_feature": feature ID list to remove
  - "keep_sample": sample ID list to keep
  - "remove_sample": sample ID list to remove
  - "add_sample_annotation": merge data with `@sampleData`
  - "change_featureID": change the name of featureID (provide the new column name in `@featureData` for dataset)

Value

A Metabolite object after updating.
Index

* datasets
  df_plasma, 8

  assayData, 3
  assayData,Metabolite-method
    (assayData), 3
  assayData<-, 3
  assayData<-,Metabolite-method
    (assayData<->), 3

  batch_norm, 4, 18, 19, 26
  bridge, 5

  column_missing_rate, 5
  cor, 6
  correlation, 6
  coxph, 11, 29
  create_Metabolite, 7

  df_plasma, 8

  featureData, 8
  featureData,Metabolite-method
    (featureData), 8
  featureData<-, 9
  featureData<-,Metabolite-method
    (featureData<->), 9

  fit_cox (fit_lm), 11
  fit_glmer (fit_lm), 11
  fit_lm, 11, 29
  fit_lme (fit_lm), 11
  fit_lmer (fit_lm), 11
  fit_logistic (fit_lm), 11
  fit_poisson (fit_lm), 11

  glm, 11
  glmer, 11

  impute, 12
  impute_kNN (impute), 12
  inverse_rank_transform, 13
  is_outlier, 14

  lm, 11
  lme, 11
  lmer, 11
  load_data, 7, 14, 17
  load_excel, 7, 15, 17

  merge_data, 16
  Metabolite, 7, 17
  Metabolite (Metabolite-class), 16
  Metabolite-class, 16
  modelling_norm, 17

  nearestQC_norm, 18

  outlier_rate, 19

  p.adjust, 29
  pareto_scale, 20
  plot_injection_order, 20
  plot_Metabolite, 21
  plot_PCA, 22
  plot_ROC, 23
  plot_tsne, 23
  plot_UMAP, 24
  plot_volcano, 25

  QC_pipeline, 27
  QCmatrix_norm, 4, 19, 26

  regression, 12, 28
  regression_each (regression), 28
  replace_outlier, 27, 29
  row_missing_rate, 30
  RSD, 31
  run_PCA, 31
sampleData, 32
sampleData, Metabolite-method
   (sampleData), 32
sampleData<-, 33
sampleData<-, Metabolite-method
   (sampleData<->), 33
save_data, 33
show, Metabolite-method, 34
subset, 34, 34

transformation, 35
tsne, 23

umap, 24
update_Metabolite, 35