Package 'MSdata'

September 14, 2015

Title Mass-spectrometry data analysis package

Description Package for mass-spectrometry data analysis working in connection with MApckg (MetaboAnalyst).

Version 0.1

Suggests knitr

VignetteBuilder knitr

URL http://github.com/flajole/MSdata

BugReports http://github.com/flajole/MSdata/issues

Imports abind,pcaMethods,impute,methods

License GPL-3

LazyData true

Collate 'Annotation.R' 'MSdata_class.R' 'EvalMissVal.R' 'MQI_to_MA.R' 'MSdata_to_MA.R' 'MSoutput.R' 'MSupload.R' 'MZindexing.R' 'Norm.R' 'Norm_Biomass.R' 'Norm_Scaling.R' 'Norm_Standards.R' 'Norm_Transform.R' 'PQI_to_MA.R' 'PeakFilter.R' 'RT_MZ_rename.R' 'RTindexing.R' 'SetRepGroup.R'

R topics documented:

`BasicFilter` *Basic peak filtering*

Description

The purpose of the data filtering is to identify and remove variables that are unlikely to be of use when modeling the data. No phenotype information are used in the filtering process, so the result can be used with any downstream analysis. This step is strongly recommended for untargeted metabolomics datasets (i.e. spectral binning data, peak lists) with large number of variables, many of them are from baseline noises. Filtering can usually improve the results.

Usage

```
## S4 method for signature 'MSdata'
BasicFilter(msdata, method = "none")
```

Arguments

`msdata`	`MSdata-class` object to be filtered
`method`	Method of filtering one of: `"none"` - no filtering applied `"iqr"` - interquantile range (IQR) `"sd"` - standard deviation (SD) `"mad"` - median absolute deviation (MAD) `"rsd"` - relative standard deviation (RSD = SD/mean) `"nprsd"` - non-parametric relative standard deviation (MAD/median) `"mean"` - mean intensity value `"median"` - median intensity value none (less than 2000 features) the final variable should be less than 5000 for effective computing

Details

Non-informative variables can be characterized in two groups:

- variables of very small values - can be detected using mean or median;
- variables that are near-constant throughout the experiment conditions - can be detected using different variance measures.

The following empirical rules are applied during data filtering:

- less than 250 variables: 5% will be filtered;
- between 250 - 500 variables: 10% will be filtered;
- between 500 - 1000 variables: 25% will be filtered;
- over 1000 variables: 40% will be filtered.

Please note, that "none" option is only for less than 2000 features. Over that, if you choose "none", the IQR filter will still be applied.

The maximum allowed number of variables is 5000. If over 5000 variables were left after filtering, only the top 5000 will be used in the subsequent analysis.

Value

MSdata-class object without filtered peaks

BiomassNorm *Normalisation by biomass*

Description

Normalisation of intensities by the list of sample masses/volumes/etc.

Usage

```
## S4 method for signature 'MSdata'
BiomassNorm(msdata, biomass.list)
```

Arguments

msdata	MSdata-class object
biomass.list	One of: 1. a path to the file containing the simple table with two columns: sample ID and biomass/volum/etc. 2. a name of the corresponding column in sample data table sampleData(msdata) (if biomass were uploaded there)

Value

MSdata-class object with normalised intensity matrix

DataScaling *Data scaling*

Description

This function realizes two steps: mean-centring and scaling.
Mean-centring means that for each feature (peak/compound) all samples intensities are considered as differences from the mean intensity of this feature.
Scaling means that these differences are standardized for all the features by dividing by standard deviation, or range, or another measure of variance.
Therefore, after scaling all the features have mean intensity 0 and corresponding variance measure 1.

Usage

```
## S4 method for signature 'MSdata'
DataScaling(msdata, method = "pareto")
```

Arguments

msdata	MSdata-class object
method	The method of scaling, one of: "auto" - Autoscaling, mean-centring and dividing by the standard deviation of each variable; "pareto" - Pareto scaling, mean-centring and dividing by the square root of standard deviation of each variable; "range" - Range scaling, mean-centring and dividing by the range of each variable

Value

MSdata-class object with normalised intensity matrix

DataTransform *Data transformation*

Description

Logarithmical or cube root data transformation.

Usage

```
## S4 method for signature 'MSdata'
DataTransform(msdata, method = "glog10")
```

Arguments

msdata	MSdata-class object
method	The method of transformation, one of: "log10" - log10-transformation; "log2" - log2-transformation; "glog10" - generalised log10, tolerant to zeros (zeros are replaced with 1/10 of the minimal value); "glog2" - generalised log2, tolerant to zeros (zeros are replaced with 1/10 of the minimal value); "cuberoot" - cube root transformation

Value

MSdata-class object with transformed intensity matrix

EvalMissVal *Evaluate Missing Values*

Description

Fill NA positions in data set. See more: pca

Usage

```
## S4 method for signature 'MSdata'
EvalMissVal(msdata, method = "min", percent = 0.5)
```

Arguments

msdata	MSdata-class
method	Method of evaluation, one of: • "min", - certain percent of minimal intensity throughout all dataset; • "featureMin", "featureMean", "featureMedian" - NAs in feature are replaced by corresponding statistics of this feature intensity values; • "GausSim" - random generation of normally distributed values; • "knn" - nearest neighbour averaging. impute.knn function is used; • "bpca", "ppca", "svdImpute" - imputation based on PCA analysis. pca function is used.
percent	For method = "min". Quotient of minimal value used to replace missing values.

MSdata-class *MSdata class*

Description

MSdata-S4 class description, with it's accessors and replacement methods.

Usage

```
## S4 method for signature 'MSdata'
peakData(msdata)

## S4 replacement method for signature 'MSdata'
peakData(msdata) <- value

## S4 method for signature 'MSdata'
sampleData(msdata)

## S4 replacement method for signature 'MSdata'
```

```
sampleData(msdata) <- value

## S4 method for signature 'MSdata'
intMatrix(msdata)

## S4 replacement method for signature 'MSdata'
intMatrix(msdata) <- value

## S4 method for signature 'MSdata'
processLog(msdata)

## S4 replacement method for signature 'MSdata'
processLog(msdata) <- value

## S4 method for signature 'MSdata'
peakNames(msdata)

## S4 method for signature 'MSdata'
sampleNames(msdata)
```

Arguments

msdata	MSdata-class object
value	The data to replace data in corresponding slot of msdata

Slots

intMatrix The matrix of peak intensities / compound concentrations.

peakData Peak metadata.

sampleData Sample metadata.

processLog Processing log.

MSdata_to_MA *Convert MSdata object to MetaboAnalyst object*

Description

Create an object for storing data for processing in MetaboAnalysis (MApckg).

Usage

```
## S4 method for signature 'MSdata'
MSdata_to_MA(msdata, designType = "regular",
  facA = names(sampleData(msdata))[1], facB = NULL)
```

Arguments

msdata	An object of MSdata class.
designType	"time" if the data is time-series data; "regular" otherwise. In the first case one of the arguments facA or facB have to be equal "Time"
facA	Grouping factor, one of the names of sample data columns in msdata@sampleData
facB	Optional second factor.

MSoutput *Output MSdata in files*

Description

Writes MSdata to four files, containing intensity matrix, sample metadata, peak metadata and processing log file.

Usage

```
## S4 method for signature 'MSdata'
MSoutput(msdata, dir = "", file = "MSdata")
```

Arguments

msdata	An object of MSdata class.
dir	The output directory path.
file	File name prefix.

MSupload *Upload MSdata*

Description

Create a MSdata-class object from external data tables. Automatically adds a ReplicationGroup column in sampleData table according to group factors combinations.

Usage

```
MSupload(object, ...)

## S4 method for signature 'character'
MSupload(object, orientation = "SamplesInCol",
  zeros.as.NA = TRUE, sampleDataLines = 1, peakDataLines = 1,
  sampleNames = TRUE, peakNames = TRUE)

## S4 method for signature 'list'
MSupload(object = list(intFile = "", sampleFile = "",
  peakFile = ""), orientation = "SamplesInCol", zeros.as.NA = TRUE)
```

Arguments

object	One of: 1. a character vector of length 1 - just a file path name of one .csv or .txt data frame. 2. a list of paths to three files: matrix of intensities, sample metadata and peak metadata.
orientation	Orientation of table, one of "SamplesInCol" or "SamplesInRow"
zeros.as.NA	If TRUE then zero intensities are treated as NAs.
sampleDataLines	The number of lines containing data about samples.
peakDataLines	The number of lines containing data about peaks/compounds.
sampleNames	If TRUE, sample names are taken from the first column/row of table. If FALSE, standard names like "sample1" are created.
peakNames	If TRUE, sample names are taken from the first row/column of table. If FALSE, standard names like "peak1" are created.

Value

MSdata-class object

Methods (by class)

- character:
- list:

MZindexing *Mass-charge indexing*

Description

Perform mass-charge correction in MSdata-class object. Function takes a given list of internal standards, corresponding peaks are found in data set and basing on their M/Z shift the shifts for the rest of peaks are calculated. The resulting data set is realigned.

It's better to perform MZ indexing after RTindexing.

Usage

```
## S4 method for signature 'MSdata'
MZindexing(object, targets.list, mz.window = 0.02,
  rt.window = 3)
```

Arguments

object	MSdata-class object. For correct processing there should be "RT" and "MZ" columns in peak data (peakData(object))
targets.list	File path to simple table of the internal standards. Table should have three columns: compoundID, mass-charge and retention time (with or without header).
mz.window	Peak corresponding to standard compound is searched in this range of MZs around standard MZ value.
rt.window	Peak corresponding to standard compound is searched in this range of RTs around standard RT value.

Value

MSdata-class object with recalculated MZ values in peak data

See Also

RTindexing

Norm *MSdata normalisation*

Description

Sample-wise normalisation methods.

Usage

```
## S4 method for signature 'MSdata'
DataNorm(msdata, method, ref.cmpd = NULL)
```

Arguments

msdata	MSdata-class object
method	Method of normalisation. In each sample data are normalised by one of: "sum" - sum concentration of all compounds "median" - median concentration of all compounds "refCompound" - concentration of reference compound
ref.cmpd	For "refCompound" method, the name or the order number of reference compound.

Value

MSdata-class object with normalised intensity matrix

PeakFilter *Peak filtering*

Description

Remove peaks fitting some criteria from data set. Criteria are stated by function arguments. If an argument is equal NULL, corresponding filtering criterion is not applied.
Usually each peak in each replicate group considered separately in terms of each criterion. After that this peak/replicate group pair can be marked as "filterable" according to this criterion. If for at least one selecting criterion all replicate groups are marked as "filterable", this peak is removed from data set.

Usage

```
## S4 method for signature 'MSdata'
PeakFilter(msdata, blanks = NULL, above.blank = NULL,
  min.int = NULL, min.nonNAnum.repgroup = NULL, min.nonNApercent = 0.4)
```

Arguments

msdata	MSdata-class object to be filtered
blanks	The vector of blank samples. Either vector of sample numbers or sample names. (check names by command: sampleData(msdata)). These sample will be removed from dataset after filtering.
above.blank	Filter peaks with intensities close to blanks. mean - above.blank*SE for certain peak's intensities in replicate group have to be more than mean + above.blank*SE for this peak's intensities in blank group. (where SE - standard error of the mean.) Note: if above.blank = 0 then just mean values are compared.
min.int	Filter peaks by total intensity. Mean peak intensity in replicate group have to be higher than min.int value.
min.nonNAnum.repgroup	Filter peaks with too many missing values. nonNAnum.repgroup is minimal number of non-NA values in replicate group.
min.nonNApercent	Filter peaks with too many missing values. min.nonNApercent is minimal allowed quotient of non-NA values for each peak.

Value

MSdata-class object without filtered peaks and blank samples

QI_to_MA *Convert QI to MetaboAnalyst*

Description

Function for converting QI output .csv files into format convenient for further analysis with MetaboAnalyst. For proper work, please, make complete QI output files with all the options marked.
This function:

1. Cuts out all the extra data, leaving only columns with abundance values (either raw or normalised) and a column with compound's IDs.
2. Moves chosen compound ID column to the first position.
3. Expands sample group labels.
4. Separate group labels into grouping factors and creates rows with factors.
5. (for `MQI_to_MA`) Merges neg and pos files with matching names.
6. Writes the resulting table into new .csv file.

`MQI_to_MA` converts lipid and metabolite QI data output

`PQI_to_MA` converts protein QI data output

Usage

```
MQI_to_MA(path = getwd(), abundance = "Raw",
  compoundID = "Accepted Compound ID", facNames = NULL,
  unite_neg_pos = TRUE)

PQI_to_MA(path = getwd(), abundance = "Raw", facNames = NULL,
  compoundID = "shortDescription")
```

Arguments

`path`	Character vector. There could be two types of elements: 1) file path(s) to the proceeded .csv QI output file(s). 2) path(s) to the directory(ies) containing .csv files. In the second case all the files in all subdirectories are proceeded. Remember that in file paths you should use either "/" or double "\", not just "\".
`abundance`	If `"Normalised"`, then normalised data are used; if `"Raw"`, then raw data are used
`compoundID`	The name of the column in QI output table used as the set of compound IDs. For `MQI_to_MA` one of: `"Compound"`, `"Accepted Compound ID"`, `"Formula"`. For `PQI_to_MA` one of: `"Accession"`, `"Description"`, `"shortDescription"`. `"shortDescription"` means that the data from "Description" column are used, but they are cut starting with "OS=".
`facNames`	Character vector of grouping factor names.
`unite_neg_pos`	If `TRUE`, creates combined file from "pos" and "neg" file pairs with matching names if `FALSE`, just skips this step

Examples

```
# PQI_to_MA("//mpimp-golm/user/Homes/Zubkov/QI_data")
# PQI_to_MA("\\\\mpimp-golm\\user\\Homes\\Zubkov\\QI_data")
# PQI_to_MA("C:/QI_data", abundance = "Raw", compoundID = "Accession")
# MQI_to_MA("C:/QI_data", facNames = c("Phenotype", "Treatment", "Time"), unite_neg_pos = FALSE)
```

RTindexing *Retention time indexing*

Description

Perform retention time correction in MSdata-class object. Function takes a given list of internal standards, corresponding peaks are found in data set and basing on their RT shift the shifts for the rest of peaks are calculated. The resulting data set is realigned.

Usage

```
## S4 method for signature 'MSdata'
RTindexing(object, targets.list, mz.window = 0.02,
  rt.window = 30)
```

Arguments

object	MSdata-class object. For correct processing there should be "RT" and "MZ" columns in peak data (peakData(object))
targets.list	File path to simple table of the internal standards. Table should have three columns: compoundID, mass-charge and retention time (with or without header).
mz.window	Peak corresponding to standard compound is searched in this range of MZs around standard MZ value.
rt.window	Peak corresponding to standard compound is searched in this range of RTs around standard RT value.

Value

MSdata-class object with recalculated RT values in peak data

See Also

MZindexing

SetRepGroup *Set replication groups numbers*

Description

If there are data about replication groups, corresponding column is renamed in a standard way to "ReplicationGroup".
Otherwise function automatically adds a ReplicationGroup column into sampleData table according to group factors combinations.

Usage

```
## S4 method for signature 'MSdata'
SetRepGroup(msdata, repFac = NULL, impFac = NULL)
```

Arguments

msdata	MSdata-class object
repFac	If you already have replication group numbers in sample metadata, set repFac - the name of this factoring variable in table.
impFac	The vector of the names of factors which are taken into account during automatic replication group labeling. If NULL, all grouping factors are used.

Value

MSdata-class object with $ReplicationGroup in sample data table

StandNorm *Normalisation by standards*

Description

Normalisation by the list of external or internal standards. The list of the standards is provided as a file with three columns: compound, m/z, retention time. By these MZ and RT values corresponding peaks in dataset are determined and their sum intensity is used for normalisation.
Afterwards, standards are excluded from feature list.

Usage

```
## S4 method for signature 'MSdata'
StandNorm(msdata, standards.list, mzwindow = 0.01,
  rtwindow = 10, meanInt = 2000, recalculateMean = FALSE)
```

Arguments

msdata	MSdata-class object
standards.list	The link to the file with the table of standards looking like: compound, m/z, retention time
mzwindow	Range (in ppm) for searching the peak corresponding to a standard from the list.
rtwindow	Range (in seconds) for searching the peak corresponding to a standard from the list.
meanInt	Mean sum intensity of all the standards (usually, got from previous experiments).
recalculateMean	If TRUE then meanInt is not used, but is recalculated from these particular data.

Value

MSdata-class object with normalised intensity matrix

Index