Package 'Metab'

April 23, 2016

Version 1.4.1

Date 2016-02-01

Title Metab: An R Package for a High-Throughput Analysis of Metabolomics Data Generated by GC-MS.

Author Raphael Aggio <ragg005@aucklanduni.ac.nz>

Maintainer Raphael Aggio <ragg005@aucklanduni.ac.nz>

Depends xcms, R (>= 3.0.1), svDialogs

Imports pander

Suggests RUnit, BiocGenerics

Description Metab is an R package for high-throughput processing of metabolomics data analysed by the Automated Mass Spectral Deconvolution and Identification System (AMDIS) (http://chemdata.nist.gov/mass-spc/amdis/downloads/). In addition, it performs statistical hypothesis test (t-test) and analysis of variance (ANOVA). Doing so, Metab considerably speed up the data mining process in metabolomics and produces better quality results. Metab was developed using interactive features, allowing users with lack of R knowledge to appreciate its functionalities.

License GPL (>=2)

biocViews Metabolomics, MassSpectrometry, AMDIS, GCMS

NeedsCompilation no

R topics documented:

`Metab-package` *Metab processes metabolomics data previously analyzed by the Automated Mass Spectral and Deconvolution System (AMDIS).*

Description

Metab is an R package for high-throughput processing of metabolomics data analysed by the Automated Mass Spectral Deconvolution and Identification System (AMDIS) (http://chemdata.nist.gov/mass-spc/amdis/downloads/). In addition, it performs statistical hypothesis test (t-test) and analysis of variance (ANOVA). Doing so, Metab considerably speed up the data mining process in metabolomics and produces better quality results. Metab was developed using interactive features, allowing users with lack of R knowledge to appreciate its functionalities.

Details

Package:	Metab
Type:	Package
Version:	0.99.6
Date:	2013-10-11
License:	GNU General Public License as published by the Free\ Software Foundation, either version 3 of the License, or any

Author(s)

Raphael Aggio

Maintainer: Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

`htest`, `MetReport`, `MetReportNames`, `normalizeByInternalStandard`, `removeFalsePositives`, `buildLib`, `normalizeByBiomass`

Examples

```
##### Load example data #####
data(exampleMetReport)
##### print data #############
print(exampleMetReport)
```

`buildLib` *buildLib converts an AMDIS' library into a CSV file in the formated required by Metab.*

Description

buildLib is a function to convert a .MSL file of an AMDIS' library into a CSV file with the format required by Metab.

Usage

```
buildLib(AmdisLib, folder, save = TRUE, output = "ion_lib", verbose = TRUE)
```

Arguments

`AmdisLib`	when AmdisLib is missing, a dialog box will pop up allowing the user to click-and-point to the .MSL file from which the data is to be read. Alternatively, AmdisLib can take a character string naming the path to the .MSL file to be read or the name of a variable (data frame) containing the .MSL file.
`folder`	when save = TRUE and folder is missing, a pop up dialog box will be presented to the user. The user can then select the directory to which the results will be saved. Alternatively, folder can take a character string naming the path to the folder where the results must be saved.
`save`	a logical vector (TRUE or FALSE) defining if the results must be saved into a CSV file (default = TRUE).
`output`	A character string indicating the name of the file storing the results generated by buildLib (default = ion_lib.csv).
`verbose`	A 'logical' defining if the progress bar should be displayed.

Details

The Automated Mass Spectral Deconvolution and Identification System (AMDIS) is a software developed by NIST (http://chemdata.nist.gov/mass-spc/amdis/). It makes use of a mass spectral library composed of two files, a .CID and a .MSL file. buildLib allows a quick conversion of the AMDIS' library into a .CSV file with the format required by Metab. For this, buildLib requires only the .MSL file of the AMDIS' library.

Value

buildLib returns a data frame containing the following information:

Column 1: The name of each metabolite present in the .MSL file; Column 2: The expected retention time (RT) of each metabolite; Columns 3 to 6: The 4 ion mass fragments used to identify each metabolite. The ion mass fragment 1 is used by MetaBox as reference to detect and quantify each metabolite; Columns 7 to 9: The expected ratio of the ion mass fragments 2, 3 and 4 in relation to the ion mass fragment 1.

In addition, buildLib verifies the existence of metabolites expected at similar RT (less than 1 minute difference) and that use the same ion mass fragments as reference. These metabolites are probably strongly coeluted, which may dificult their correct identification. Metabolites showing these characteristics are presented to the user at the end of the run. We strongly suggest selecting different ion mass fragments for identifying such compounds.

See data(exampleIonLib).

Note

The .MSL file of the AMDIS' library must contain the expected RT of each compound.

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReport, MetReportNames, normalizeByBiomass, normalizeByInternalStandard, removeFalsePositives,

Examples

```
### Load example MSL file and show it to user #######
data(exampleMSLfile)
print(exampleMSLfile)
######## Convert library ########
testLib <- buildLib(exampleMSLfile, save = FALSE)
############ Print new library ########
print(testLib)
```

exampleAMDISReport *AMDIS report.*

Description

An example of the AMDIS report in batch mode.

Usage

```
data(exampleAMDISReport)
```

Examples

```
data(exampleAMDISReport)
print(exampleAMDISReport)
```

exampleBiomass *An example of the biomass data frame to be used in conjunction with the function normalizeByBiomass.*

Description

The function normalizeByBiomass requires a data frame containing the name of the samples under analysis and their respective biomasses.

Usage

```
data(exampleBiomass)
```

Format

A data frame containing the name of the samples in the first column and their respective amount of biomass (e.g. grams, O.D., cells, etc...) in the second column. The name of each column will be Sample and Biomass, respectively.

`Sample` column containing the name of samples under analysis

`Biomass` column containing the amount of biomass (e.g. grams, O.D., cells, etc...) relative to each sample

Examples

```
data(exampleBiomass)
```

exampleHtest *An example of the results obtained using the function htest.*

Description

The result from htest is a data frame containing the metabolites present in each biological sample, their respective intensities across samples and their associated p-values resultant from the statistical analysis performed.

Usage

```
data(exampleHtest)
```

Format

A data frame consisting of metabolites and their intensities/abundances in each biological sample.

Examples

```
data(exampleHtest)
```

exampleIonLib *An example of the ionLib required by Metab.*

Description

The function MetReport requires a spectral library for identifying metabolites in GC-MS samples.

Usage

```
data(exampleIonLib)
```

Format

A data frame containing a spectral library.

`Name` Example of spectral library

`RT` Example of spectral library

`ion2to1` Example of spectral library

`ion3to1` Example of spectral library

`ion4to1` Example of spectral library

`ref_ion1` Example of spectral library

`ref_ion2` Example of spectral library

`ref_ion3` Example of spectral library

`ref_ion4` Example of spectral library

Examples

```
data(exampleIonLib)
print(exampleIonLib)
```

`exampleMetReport` *An example of the results obtained using the function exampleMetReport.*

Description

The result from exampleMetReport is a data frame containing the metabolites present in each biological sample with their respective intensities across samples.

Usage

```
data(exampleMetReport)
```

Format

A data frame consisting of metabolites and their intensities/abundances in each biological sample.

Examples

```
data(exampleMetReport)
```

`exampleMSLfile` *An example of MSL file of an AMDIS library.*

Description

The MSL file of an AMDIS library is a text file containing the spectrum and RT of each metabolite in the spectral library.

Usage

```
data(exampleMSLfile)
```

Format

A data frame.

`V1` Example of the MSL file of an AMDIS library

Examples

```
data(exampleMSLfile)
print(exampleMSLfile)
```

htest *Function developed to apply t-test or ANOVA on a data frame.*

Description

htest applies t-test or ANOVA to a data frame. For that, the first row of the input data must contain a label defining the experimental condition associated to each sample or replicate. Thus, the first column of the first row must receive the word Replicates and the remaining columns must receive the name of the condition associated to each sample. See data(exampleMetReport) for more details.

Usage

```
htest(
inputData,
signif.level = 0.05,
log.transform = TRUE,
save = TRUE,
folder,
StatTest,
output,
adjust.pValue = TRUE,
method = "bonferroni"
)
```

Arguments

inputData	When inputData is missing, a dialog box will pop up allowing the user to click-and-point to the .csv file from which the data is to be read. It may also receive a character string pointing to a .csv file containing a data frame such as data(exampleMetReport), generated by MetReport. Alternatively, inputData takes an R vector containing the desired data frame.
signif.level	A numerical string indicating the p-value cut-off. Compounds presenting a p-value higher than specified through signif.level will not be reported.
log.transform	A logical parameter (TRUE or FALSE) indicating if the data should be log transformed before t-test or ANOVA. Log transformation is skipped if the input data contains any cell filled with negative value.
save	A logical vector (TRUE or FALSE) indicating if the resultant data frame should be saved in a .csv file. If save = TRUE, the .csv file will be saved in the path defined in the argument folder.
folder	A character string pointing to the folder where the results will be saved.
StatTest	A character string defining the statistical test to be performed (See details).
output	A character string indicating the name of the .csv file to be generated.
adjust.pValue	A logical vector indicating if p-values should be adjusted by the p.adjust.
method	A character string indicating the method used when applying p.adjust. See p.adjust.methods for possible methods. Default is 'bonferroni'.

Details

The argument StatTest may receive any of the following options for performing t-Test: "T-TEST", "T-test", "t-test", "t-TEST", "t", "T". The argument StatTest may receive any of the following options for performing ANOVA: "ANOVA", "Anova", "anova", "A", "a". If StatTest is missing, ANOVA is performed when more than 2 experimental conditions are under analysis and t-test is applied otherwise. A column containing the p-values resulting from the analysis is added to the data frame defined in inputData. As a result, htest produces a data frame consisting of only compounds statistically significantly different at the specified signif.level. If log.transform = TRUE (default), the t-test or ANOVA will be calculated using the log transformed data.

Value

htest produces a data frame containing only metabolites showing a p-value lower than the one specified through signif.level.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

MetReport, MetReportNames, normalizeByBiomass, normalizeByInternalStandard, removeFalsePositives, buildLib t.test p.adjust

Examples

```
### Load Metab ###
library(Metab)
### Load the inputData ###
data(exampleMetReport)
### Perform t-test ####
tTestResults <- htest(
exampleMetReport,
signif.level = 0.05,
StatTest = "T",
save = FALSE
)
### Show results ###
print(tTestResults)
### Perform ANOVA ####
AnovaResults <- htest(
exampleMetReport,
```

```
signif.level = 0.05,
StatTest = "Anova",
save = FALSE
)
### Show results ###
print(AnovaResults)
```

`MetReport` *MetReport cleans and corrects results generated by the Automated Mass Spectral Deconvolution and Identification System (AMDIS).*

Description

MetReport automatically process ADMIS results by selecting only one compound for each retention time, correcting peak intensities by assigning a fixed mass fragment for each compound across samples or simply extracting their respective areas or base peaks.

Usage

```
MetReport(
inputData,
singleFile = FALSE,
AmdisReport,
ionLib,
save = TRUE,
output = "metab_data",
TimeWindow = 2.5,
Remove,
abundance = "recalculate",
folder)
```

Arguments

`inputData`	the value of input data depend on the value of the argument singleFile. If singleFile = FALSE, inputData must receive a character vector indicating the path to the mainFolder (See details). If singleFile = TRUE, inputData must receive a character vector indicating the path to the CDF file to be analysed. If inputData is missing, a dialog box pops up allowing the user to click and point to the desired mainFolder or CDF file.
`singleFile`	singleFile = FALSE when analysing a batch of CDF files organized in a single mainFolder (See details). singleFile = TRUE when analysing a single CDF file.
`AmdisReport`	a character vector indicating the path to the TXT file containing the AMDIS report in batch mode (See details).
`ionLib`	The default behaviour is to allow the user to point to a .csv file interactively from a popup dialog box. Alternatively, ionLib can take a character string indicating the path to the reference ion library (See details).

save	If TRUE (default), a data frame is saved to a .csv file in folder.
output	A character string with the name of the .csv file produced.
TimeWindow	A numeric vector defining the maximum allowed different between expected and observed retention. Any compound showing a difference between expected and observed retention higher than the value defined through TimeWindow will be removed from results.
Remove	A character string with the names of compounds to be skipped during analysis. Compounds defined through remove (e.g. remove = c("Zylene1", "Pyridine")) will not be considered during analysis.
abundance	it may receive one of the values: "recalculated", if the abundances of metabolites will be corrected by fixing a single mass fragment as reference (See details); "Area", "AREA", "a" or "A", if the area associated with each compound should be extracted from the AMDIS report; or "Base.Peak", "BasePeak", "base.peak", "basepeak", "B" or "b", if the Base.Peak associated with each compound should be extracted from the AMDIS report.
folder	a character vector indicating the path to the folder where the results must be saved, if save = TRUE.

Details

Metab is an R package for processing metabolomics data previously analysed by the Automated Mass Spectral Deconvolution and Identification System (AMDIS) (http://chemdata.nist.gov/mass-spc/amdis/downloads/). AMDIS is one of the most used software for deconvoluting and identifying metabolites analysed by Gas Chromatography - Mass Spectrometry (GC-MS). It is execellent in deconvoluting chromatograms and identifying metabolites based on a spectral library, which is a list of metabolites with their respective mass spectrum and their associated retention times. Although AMDIS is widely and successfully applied to chemistry and many other fields, it shows some limitations when applied to biological studies. First, it generates results in a single spreadsheet per sample, which means that one must manually merge the results provided by AMDIS in a unique spreadsheet for performing further comparisons and statistical analysis, for example, comparing the abundances of metabolites across experimental conditions. AMDIS also allows users to generate a single report containing the results for a batch of samples. However, this report contains the results of samples placed on top of each other, which also requires extensive manual process before statistical analysis. In addition, AMDIS shows some limitations when quantifying metabolites. It quantifies metabolites by calculating the area (Area) under their respective peaks or by calculating the abundance of the ion mass fragment (Base.Peak) used as model to deconvolute the peak associated with each specific metabolite. As the area of a peak may be influenced by coelution of different metabolites, the abundance of the most abundant ion mass fragment is commonly used for quantifying metabolites in biological samples. However, AMDIS may use different ion mass fragments for quantifying the same metabolite across samples, which indicates that using AMDIS results one is not comparing the same variable across experimental conditions. Finally, according to the configurations used when applying AMDIS, it may report more than one metabolite identified for the same retention time. Therefore, AMDIS data requires manual inspection to define the correct metabolite to be assigned to each retention time. Metab solves AMDIS limitations by selecting the most probable metabolite associated to each retention time, by correcting the Base.Peak values calculated by AMDIS and by combining results in a single spreadsheet and in a format that suits further data processing. In order to select the most probable metabolite associated to each retention

time, Metab considers the number of question marks reported by AMDIS, which indicates its certainty in identification, and the difference between expected and observed retention times associated with each metabolite. For correcting abundances calculated by AMDIS, Metab makes use of an ion library containing the ion mass fragment to be used as reference when quantifying each metabolite present in the mass spectral library applied. For this, Metab collects from the AMDIS report the scan used to identify each metabolite and collects from the raw data (CDF files) the intensities of their reference ion mass fragments defined in the ion library. In addition, MetReport can be used to simply reformat AMDIS reports into a single spreadsheet containing identified metabolites and their Areas or Base.Peaks calculated by AMDIS in each analysed sample. Therefore, MetReport can be used to quickly process AMDIS reports correcting or not metabolite abundances previously calculated by AMDIS.

When singleFile = FALSE, MetReport requires CDF files organised in a mainFolder with subfolders for each experimental condition. Metab's functions were developed to automatically identify the experimental condition associated with each sample. For this, the CDF files to be analysed by MetReport must be organised in subfolders according to their experimental condition, as follows:

```
————————————
mainFolder
——Condition1
————Sample1_1.cdf
————Sample1_2.cdf
————Sample1_3.cdf
——Condition2
————Sample2_1.cdf
————Sample2_2.cdf
————Sample2_3.cdf
——Condition3
————Sample3_1.cdf
————Sample3_2.cdf
————Sample3_3.cdf
————————————
```

One mainFolder containing one subfolder for each experimental condition. Each subfolder contains the CDF files associated with this specific experimental condition. Alternatively, all the CDF files can be placed in a single folder and MetReport will analyse every sample as belonging to the same experimental condition.

- Amdis report in batch mode. It is a text file containing the results for a batch of samples and can be obtained in AMDIS through: File > Batch Job > Create and Run Job.... Select the Analysis Type to be used, generally Simple, click on Generate Report and Report all hits. Click on Add.., select the files to be analysed, click on Save As..., select the folder where the report will be generated and a name for this report (any name you desire). Finally, click on Run. A new .TXT file with the name specified will be generated in the folder specified.

- ion library in the specific format required by Metab. The ion library is a data frame containing the name and the reference ion mass fragment to quantify each metabolite present in the mass spectral library used by AMDIS when generating the batch report. To facilitate the process, MetReport accepts the .msl file used by AMDIS. An AMDIS library is stored in two files, a file with extension .CID and a file with extension .msl. Metab requires only the .msl file.

To see an example of an ion library from AMDIS converted to the format required by Metab, simply

enter the following code in the R console:

library(Metab)

data(exampleIonLib)

print(exampleIonLib)

When all the requirements described above are ready and available, MetReport can be applied. If an essential argument is missing, a dialog box will pop up allowing the user to point and click on the main folder to be analysed, the AMDIS report to be used or the ion library. Thus, MetReport can be applied by simply entering

MetReport()

at the R console. In this case, the user will be prompted to point to the required files while the arguments save, output, TimeWindow and Remove will receive their default values. The default value of save is TRUE, which indicates that the report generated by MetReport must be saved into a CSV file with the name specified in the argument output. The argument TimeWindow defines, in minutes, the accepted difference between expected and observed retention times to consider a true identification. Expected retention time is the retention time defined in the spectral library, while the observed retention time is the retention time where AMDIS actually identified each metabolite. For example, if TimeWindow = 0.5, every metabolite showing more than half minute difference between expected and observed retention times will be removed from the analysis. The argument Remove is used to remove specific compounds from the analysis. For example, if Remove = "Ethanol", every observation of Ethanol in the AMDIS report will not be considered in the analysis. It may receive as many names of metabolites as desired, for example, Remove = c("Ethanol", "Alanine", "Pyridine").

As a result, MetReport generates a data frame containing the metabolites identified in the first column and their abundances in the different samples analysed in the following columns. To see an example, enter the following code at the R console:

library(Metab)

data(exampleMetReport)

print(exampleMetReport)

Value

MetReport generates a data frame containing the metabolites identified in each biological sample and their respective abundancies/intensities. See data(exampleMetReport) to see an example of the data frame produced by MetReport.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReportNames, normalizeByBiomass, normalizeByInternalStandard, removeFalsePositives, buildLib

Examples

```
library(Metab)
###### Load exampleAMDISReport ######
data(exampleAMDISReport)
###### Analyse a single file ########
test <- MetReport(
inputData = unzip(system.file("extdata/130513_REF_SOL2_2_50_50_1.CDF.zip", package = "Metab")),
singleFile = TRUE,
AmdisReport = exampleAMDISReport,
abundance = "Area",
save = FALSE)
###### Show results #################
print(test)
```

`MetReportNames` *MetReportNames cleans results obtained with the Automated Mass Spectral Deconvolution and Identification System (AMDIS).*

Description

MetReportNames automatically process ADMIS results keeping only one compound for each retention time and by assigning the area or the base peak of each compound accross samples.

Usage

```
MetReportNames(
data,
AmdisReport,
base.peak = FALSE,
save = TRUE,
folder,
output = "metab_data",
TimeWindow = 2.5,
Remove)
```

Arguments

`data`	A character vector defining the names of the samples to be extracted from the AMDIS report under analysis. If missing, a dialog box will pop up allowing the user to select the samples to be extracted.
`AmdisReport`	The default behaviour is to allow the user to point to a .csv file interactively from a popup dialog box. Alternatively, AmdisReport can take a character string indicating the path to the AMDIS report generated in batch mode (See details).

base.peak	If TRUE, the base.peak of each compound is returned. If FALSE, the area of each compound is returned.
save	If TRUE (default), a data frame is saved to a .csv file in the dataFolder.
folder	A character string pointing to the folder where the results will be saved.
output	A character string with the name of the .csv file produced.
TimeWindow	A numeric vector defining the maximum allowed different between expected and observed retention. Any compound showing a difference between expected and observed retention higher than the value defined through TimeWindow will be removed from results.
Remove	A character string with the names of compounds to be skipped during analysis. Compounds defined through remove (e.g. remove = c("Zylene1", "Pyridine")) will not be considered during analysis.

Details

Metab is an R package for processing metabolomics data previously analysed by the Automated Mass Spectral Deconvolution and Identification System (AMDIS) (http://chemdata.nist.gov/mass-spc/amdis/downloads/). AMDIS is one of the most used software for deconvoluting and identifying metabolites analysed by Gas Chromatography - Mass Spectrometry (GC-MS). It is execellent in deconvoluting chromatograms and identifying metabolites based on a spectral library, which is a list of metabolites with their respective mass spectrum and their associated retention times. Although AMDIS is widely and successfully applied to chemistry and many other fields, it shows some limitations when applied to biological studies. First, it generates results in a single spreadsheet per sample, which means that one must manually merge the results provided by AMDIS in a unique spreadsheet for performing further comparisons and statistical analysis, for example, comparing the abundances of metabolites across experimental conditions. AMDIS also allows users to generate a single report containing the results for a batch of samples. However, this report contains the results of samples placed on top of each other, which also requires extensive manual process before statistical analysis. In addition, AMDIS shows some limitations when quantifying metabolites. It quantifies metabolites by calculating the area (Area) under their respective peaks or by calculating the abundance of the ion mass fragment (Base.Peak) used as model to deconvolute the peak associated with each specific metabolite. As the area of a peak may be influenced by coelution of different metabolites, the abundance of the most abundant ion mass fragment is commonly used for quantifying metabolites in biological samples. However, AMDIS may use different ion mass fragments for quantifying the same metabolite across samples, which indicates that using AMDIS results one is not comparing the same variable across experimental conditions. Finally, according to the configurations used when applying AMDIS, it may report more than one metabolite identified for the same retention time. Therefore, AMDIS data requires manual inspection to define the correct metabolite to be assigned to each retention time. MetReportNames processes an AMDIS report produced in batch mode by selecting the most probable metabolite associated to each retention time, extracting their base.peak or their area and by combining results in a single spreadsheet and in a format that suits further data processing. In order to select the most probable metabolite associated to each retention time, MetReportNames considers the number of question marks reported by AMDIS, which indicates its certainty in identification, and the difference between expected and observed retention times associated with each metabolite. See below examples of MetReportNames applications.

Value

MetReportNames generates a data frame containing the metabolites identified in the analyzed sample and their respective abundancies/intensities. See data(exampleMetReport) to see an example of the data frame produced by MetReport. If save = TRUE, MetReportNames also generates a log file with the parameters used in the analysis.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReport, normalizeByBiomass, normalizeByInternalStandard, removeFalsePositives, buildLib

Examples

```
#### Load exmaple of AMDIS report #####
data(exampleAMDISReport)

#### Example 1 ###

#### Filter files "130513_REF_SOL2_2_100_1" and "130513_REF_SOL2_2_100_2" from AMDIS report ##############
#### using a difference between expected and real RT of 0.5min and obtaining the base.peaks ##############
#### of each compound.
test <- MetReportNames(
data = c("130513_REF_SOL2_2_100_1", "130513_REF_SOL2_2_100_2"),
exampleAMDISReport,
save = FALSE,
TimeWindow = 0.5,
base.peak = TRUE)
print(test)

#### Example 2 ###

#### Filter files "130513_REF_SOL2_2_100_1" and "130513_REF_SOL2_2_100_2" from AMDIS report  ##############
#### using a difference between expected and real RT of 1 min and obtaining the AREA of each ##############
#### compound.
test <- MetReportNames(
data = c("130513_REF_SOL2_2_100_1", "130513_REF_SOL2_2_100_2"),
exampleAMDISReport,
save = FALSE,
TimeWindow = 0.5,
```

```
base.peak = FALSE)
print(test)
```

`normalizeByBiomass` *A function to normalize metabolomics data by the biomass associated to each biological sample (e.g. cell number or O.D.)*

Description

normalizeByBiomass divides the intensity of each metabolite in a specific sample by the value of the biomass measured for this specific sample.

Usage

```
normalizeByBiomass(inputData, biomass, save = TRUE, folder, output = "norm_bio")
```

Arguments

`inputData`	When inputData is missing, a dialog box will pop up allowing the user to click-and-point to the .csv file from which the data is to be read. It may also receive a character string pointing to a .csv file containing a data frame such as data(exampleMetReport), generated by `MetReport`. Alternatively, inputData takes an R vector containing the desired data frame.
`biomass`	When biomass is missing, a dialog box will pop up allowing the user to click-and-point to the .csv file from which the biomasses are to be read (See data(exampleBiomass)). It may also receive a character string pointing to a .csv file containing a data frame such as data(exampleBiomass). Alternativelly, it may receive a character vector indicating the data frame such as data(exampleBiomass).
`save`	A logical vector (TRUE or FALSE) indicating if the resultant data frame should be saved in a .csv file. If save = TRUE, the .csv file will be saved in the path defined in the argument folder.
`folder`	A character vector indicating the path to the folder where the results will be saved.
`output`	A character vector indicating the name of the .csv file to be generated.

Details

normalizeByBiomass loads the inputData and searches the data frame Biomass for the values of the biomasses associated with each sample present in the inputData. When the value is found, the abundances of metabolites associated with each sample is divided by the value of its respective biomass.

Value

norm.biomass produces a data frame normalized by biomass.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReport, MetReportNames, normalizeByInternalStandard, removeFalsePositives, buildLib

Examples

```
### Load the inputData ###
data(exampleMetReport)
### Load the list of biomasses ###
data(exampleBiomass)
### Normalize ####
normalizedData <- normalizeByBiomass(exampleMetReport, biomass = exampleBiomass, save = FALSE)
### Show results ###
print(normalizedData)
```

---

normalizeByInternalStandard

Normalize metabolomics data sets by a specific metabolite defined as internal standard.

Description

In the specified inputData, every metabolite from each sample will be divided by the intensity/abundance of the metabolite defined as internal standard, which is specified through the argument internalStandard.

Usage

```
normalizeByInternalStandard(
inputData,
internalStandard,
save = TRUE,
folder,
output = "normalizedByInternalStandard"
)
```

Arguments

inputData	When inputData is missing, a dialog box will pop up allowing the user to click-and-point to the .csv file from which the data is to be read. It may also receive a character string pointing to a .csv file containing a data frame such as data(exampleMetReport), generated by MetReport. Alternatively, inputData takes an R vector containing the desired data frame.
internalStandard	A character string indicating the name of the compound to be used as internal standard. If internalStandard is missing, a list of metabolites is presented to the user to interactively choose the correct compound.
save	A logical vector (TRUE or FALSE) indicating if the resultant data frame should be saved in a .csv file. If save = TRUE, the .csv file will be saved in the path defined in the argument folder.
folder	A character string indicating the path to the folder where the results will be saved.
output	A character string indicating the name of the .csv file to be generated.

Details

normalizeByInternalStandard will divide the abundances of each metabolite in a specific sample by the abundance of the chosen internal standard in this specific sample.

Value

normalizeByInternalStandard generates a data frame containing metabolite abundances normalized by the nominated internal standard.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReport, MetReportNames, normalizeByBiomass, removeFalsePositives, buildLib

Examples

```
### Load the inputData ###
data(exampleMetReport)
### Normalize ####
normalizedData <- normalizeByInternalStandard(
exampleMetReport,
internalStandard = "Acetone",
save = FALSE
)
### Show results ####
print(normalizedData)
```

`removeFalsePositives` *removeFalsePositives is a function to exclude from a data frame metabolites considered false positives.*

Description

removeFalsePositives is used to exclude compounds considered false positives. We consider false positive those compounds detected in just few samples from a specific experimental condition.

Usage

```
removeFalsePositives(
inputData,
truePercentage = 50,
Name_medium_condition = "none",
truePercentageMedium = 50,
save = TRUE,
folder,
output = "NoFalse"
)
```

Arguments

`inputData`	When inputData is missing, a dialog box will pop up allowing the user to click-and-point to the .csv file from which the data is to be read. It may also receive a character string pointing to a .csv file containing a data frame such as data(exampleMetReport), generated by `MetReport`. Alternatively, inputData takes an R vector containing the desired data frame.
`truePercentage`	A numerical string indicating in which proportion of samples, per experimental condition, each metabolite must be present to be considered a true compound (See details).
`Name_medium_condition`	A character string indicating the name of the experimental condition containining samples from the uncultured medium (See details).

`truePercentageMedium`
: truePercentageMedium works in the same way as truePercentage. However, it refers specifically to samples belonging to the experimental condition defined in Name_medium_condition (See details).

`save`
: A logical vector (TRUE or FALSE) indicating if the resultant data frame should be saved in a .csv file. If save = TRUE, the .csv file will be saved in the path defined in the argument folder.

`folder`
: A character string pointing to the folder where the results will be saved.

`output`
: A character string indicating the name of the .csv file to be generated.

Details

The data argument takes the path to the input file or an R vector containing the input data. The user should see data(exampleMetReport) for what the input file should look like. The first row of the input data is used to define the experimental conditions associated with each sample. This row contains the word Replicates in the first collumn and the names of experimental conditions in the following columns, according to samples. The argument truePercentage takes a numerical vector between 0 to 100. It works as a proportion cut off indicating the required proportion of samples from an experiemntal condition where a compound must be present in order to be considered a true compound. For example, considering an experiment performed in 6 replicates and true = 50, compounds detected in fewer than 3 replicates will have their intensity replaced by NA. However, samples from the uncultured medium may have a different number of replicates, generally less replicates. In this case, the user may want to have a different proportion cut off applied to samples from the uncultured medium. The argument Name_medium_condition is then used to identify in the input data those samples from the uncultured medium. For this, the argument Name_medium_condition takes the same character string used in the input data, in the row Replicates, to define the experimental condition associated with the uncultured medium. truePercentageMedium works in the same way as truePercentage; however, it refers specifically to samples containing the name defined in Name_medium_condition. This feature is used when analyzing extracellular metabolites or footprinting. As a result, removeFalsePositives produces a data frame containing only metabolites present in a higher proportion of replicates than defined by the user. When the argument save = TRUE this data frame is saved in a folder defined in folder. The CSV file generated is named according to the character vector defined in the argument output (e.g. NoFalse). The extension .csv will be added automatically. There is no limit for the number of experimental conditions under analysis.

Value

removeFalsePositives processes the input data and produces a data frame containing only compounds present in a defined proportion of samples from each experimental condition.

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

Author(s)

Raphael Aggio <ragg005@aucklanduni.ac.nz>

References

Aggio, R., Villas-Boas, S. G., & Ruggiero, K. (2011). Metab: an R package for high-throughput analysis of metabolomics data generated by GC-MS. Bioinformatics, 27(16), 2316-2318. doi: 10.1093/bioinformatics/btr379

See Also

htest, MetReport, MetReportNames, normalizeByBiomass, normalizeByInternalStandard, buildLib

Examples

```
### Load the inputData ###
data(exampleMetReport)
### Normalize ####
normalizedData <- removeFalsePositives(exampleMetReport, truePercentage = 40, save = FALSE)
###################
# The abundances of compound Zylene3 will be replaced by NA in samples from experimental
#condition 50ul, as it is present in less than 40 per cent of the samples from this
#experimental condition.
### Show results ####
print(normalizedData)
```

Index