

Package ‘MSstatsTMT’

November 21, 2024

Title Protein Significance Analysis in shotgun mass spectrometry-based proteomic experiments with tandem mass tag (TMT) labeling

Version 2.15.1

Date 2024-04-23

Description The package provides statistical tools for detecting differentially abundant proteins in shotgun mass spectrometry-based proteomic experiments with tandem mass tag (TMT) labeling. It provides multiple functionalities, including aata visualization, protein quantification and normalization, and statistical modeling and inference. Furthermore, it is inter-operable with other data processing tools, such as Proteome Discoverer, MaxQuant, OpenMS and SpectroMine.

License Artistic-2.0

Depends R (>= 4.2)

Imports limma, lme4, lmerTest, methods, data.table, stats, utils, ggplot2, grDevices, graphics, MSstats, MSstatsConvert, checkmate, plotly, htmltools

Suggests BiocStyle, knitr, rmarkdown, testthat

VignetteBuilder knitr

biocViews ImmunoOncology, MassSpectrometry, Proteomics, Software

Encoding UTF-8

LazyData true

URL <http://msstats.org/msstatstmt/>

BugReports <https://groups.google.com/forum/#!forum/msstats>

RoxygenNote 7.3.1

git_url <https://git.bioconductor.org/packages/MSstatsTMT>

git_branch devel

git_last_commit 6bf3322

git_last_commit_date 2024-11-14

Repository Bioconductor 3.21

Date/Publication 2024-11-20

Author Devon Kohler [aut, cre],
 Ting Huang [aut],
 Meena Choi [aut],
 Mateusz Staniak [aut],
 Sicheng Hao [aut],
 Olga Vitek [aut]

Maintainer Devon Kohler <kohler.d@northeastern.edu>

Contents

.calculatePower	3
.checkContrastMatrix	4
.checkSummarizationParams	4
.countRunsWithNorm	6
.documentFunction	6
.getMedianSigmaRun	7
.getMedianSigmaSubject	8
.getNormalizationAbundance	8
.getNumSample	9
.getPhilosopherInput	9
.getRunsMedian	10
.getVarComponentTMT	10
.handleSingleContrastTMT	11
.logSum	11
.logSummarizationParams	12
.makeContrastSingleTMT	13
.makeFactorColumnsTMT	13
.medianPolish	14
.normalizePeptides	14
.normalizeProteins	15
.prepareForSummarization	15
.removeRedundantChannels	16
.summarizeMSstats	16
.summarizeSimpleStat	17
.summarizeTMP	18
.summarizeTMT	18
annotation.mine	19
annotation.mq	20
annotation.pd	21
dataProcessPlotsTMT	22
designSampleSizeTMT	24
evidence	26
getProcessedTMT	27
getSummarizedTMT	27
groupComparisonTMT	28
input.pd	29
MaxQtoMSstatsTMTFormat	30

MSstatsComparisonModelSingleTMT 32

MSstatsFitComparisonModelsTMT 32

MSstatsGroupComparisonOutputTMT 33

MSstatsGroupComparisonTMT 33

MSstatsModerateTTest 34

MSstatsNormalizeTMT 34

MSstatsPrepareForGroupComparisonTMT 35

MSstatsPrepareForSummarizationTMT 35

MSstatsSummarizationOutputTMT 36

MSstatsSummarizeTMT 37

MSstatsTestSingleProteinTMT 38

MSstatsTMT 38

OpenMStoMSstatsTMTFormat 39

PDtoMSstatsTMTFormat 41

PhilosophertoMSstatsTMTFormat 42

proteinGroups 44

proteinSummarization 45

quant.pd.msstats 46

raw.mine 47

raw.om 48

raw.pd 49

SpectroMinetoMSstatsTMTFormat 50

test.pairwise 51

Index **53**

<code>.calculatePower</code>	<i>Power calculation</i>
------------------------------	--------------------------

Description

Power calculation

Usage

```
.calculatePower(
  desiredFC,
  FDR,
  delta,
  median_sigma_error,
  median_sigma_subject,
  median_sigma_run,
  numSample
)
```

Arguments

desiredFC	the range of a desired fold change which includes the lower and upper values of the desired fold change.
FDR	a pre-specified false discovery ratio (FDR) to control the overall false positive rate. Default is 0.05
delta	difference between means (?)
median_sigma_error	median of error standard deviation
median_sigma_subject	median standard deviation per subject
numSample	minimal number of biological replicates per condition. TRUE represents you require to calculate the sample size for this category, else you should input the exact number of biological replicates.

.checkContrastMatrix *check whether pairwise comparison. If pairwise, generate a contrast matrix.*

Description

check whether pairwise comparison. If pairwise, generate a contrast matrix.

Usage

```
.checkContrastMatrix(contrast_matrix)
```

Value

a contrast matrix

.checkSummarizationParams
Check validity of parameters to proteinSummarization function

Description

Check validity of parameters to proteinSummarization function

Usage

```
.checkSummarizationParams(  
  data,  
  method,  
  global_norm,  
  reference_norm,  
  remove_norm_channel,  
  remove_empty_channel,  
  MBimpute,  
  maxQuantileforCensored  
)
```

Arguments

data	Name of the output of PDtoMSstatsTMTFormat function or peptide-level quantified data from other tools. It should have columns ProteinName, PeptideSequence, Charge, PSM, Mixture, TechRepMixture, Run, Channel, Condition, BioReplicate, Intensity
method	Four different summarization methods to protein-level can be performed : "msstats"(default), "MedianPolish", "Median", "LogSum".
global_norm	Global median normalization on peptide level data (equalizing the medians across all the channels and MS runs). Default is TRUE. It will be performed before protein-level summarization.
reference_norm	Reference channel based normalization between MS runs on protein level data. TRUE(default) needs at least one reference channel in each MS run, annotated by 'Norm' in Condition column. It will be performed after protein-level summarization. FALSE will not perform this normalization step. If data only has one run, then reference_norm=FALSE.
remove_norm_channel	TRUE(default) removes 'Norm' channels from protein level data.
remove_empty_channel	TRUE(default) removes 'Empty' channels from protein level data.
MBimpute	only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.
maxQuantileforCensored	We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.

Value

TRUE invisibly if all parameters are valid

`.countRunsWithNorm` *Utility function: count runs with "Norm" channel*

Description

Utility function: count runs with "Norm" channel

Usage

```
.countRunsWithNorm(run, condition)
```

Arguments

<code>run</code>	vector of run labels
<code>condition</code>	vector of condition labels

Value

integer

`.documentFunction` *A dummy function to store shared documentation items.*

Description

A dummy function to store shared documentation items.

Usage

```
.documentFunction(  
  fewMeasurements,  
  useUniquePeptide,  
  summaryforMultipleRows,  
  removeProtein_with1Feature,  
  removeProtein_with1Protein,  
  removeOxidationMpeptides,  
  removeMpeptides  
)
```

Arguments

- fewMeasurements
'remove'(default) will remove the features that have 1 or 2 measurements across runs.
- useUniquePeptide
TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
- summaryforMultipleRows
max(default) or sum - when there are multiple measurements for certain feature and certain run, use highest or sum of multiple intensities.
- removeProtein_with1Feature
TRUE will remove the proteins which have only 1 feature, which is the combination of peptide, precursor charge, fragment and charge. FALSE is default.
- removeOxidationMpeptides
TRUE will remove the peptides including 'oxidation (M)' in modification. FALSE is default.
- removeMpeptides
TRUE will remove the peptides including 'M' sequence. FALSE is default.
- removeProtein_with1Peptide
TRUE will remove the proteins which have only 1 peptide and charge. FALSE is default.
- use_log_file
logical. If TRUE, information about data processing will be saved to a file.
- append
logical. If TRUE, information about data processing will be added to an existing log file.
- verbose
logical. If TRUE, information about data processing will be printed to the console.
- log_file_path
character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.

Value

NULL.

.getMedianSigmaRun *Get median per subject or group by subject*

Description

Get median per subject or group by subject

Usage

.getMedianSigmaRun(var_component)

Arguments

`var_component` data.frame, output of `.getVarComponent`

`.getMedianSigmaSubject`

Get median per run or run by mix

Description

Get median per run or run by mix

Usage

```
.getMedianSigmaSubject(var_component)
```

Arguments

`var_component` data.frame, output of `.getVarComponent`

`.getNormalizationAbundance`

Utility function: get mean abundance for "Norm" channels

Description

Utility function: get mean abundance for "Norm" channels

Usage

```
.getNormalizationAbundance(abundance, condition)
```

Arguments

`abundance` vector of abundances

`condition` vector of condition labels

Value

numeric

.getNumSample *Get sample size*

Description

Get sample size

Usage

```
.getNumSample(  
  desiredFC,  
  power,  
  alpha,  
  delta,  
  median_sigma_error,  
  median_sigma_subject,  
  median_sigma_run  
)
```

Arguments

desiredFC	the range of a desired fold change which includes the lower and upper values of the desired fold change.
power	a pre-specified statistical power which defined as the probability of detecting a true fold change. TRUE represent you require to calculate the power for this category, else you should input the average of power you expect. Default is 0.9
alpha	significance level
delta	difference between means (?)
median_sigma_error	median of error standard deviation
median_sigma_subject	median standard deviation per subject

.getPhilosopherInput *Convert Philosopher parameters to consistent format*

Description

Convert Philosopher parameters to consistent format

Usage

```
.getPhilosopherInput(input, path, folder)
```

Arguments

input	data.frame of 'msstats.csv' file produced by Philosopher
-------	--

<code>.getRunsMedian</code>	<i>Utility function: get median from unique values per run</i>
-----------------------------	--

Description

Utility function: get median from unique values per run

Usage

```
.getRunsMedian(input)
```

Arguments

<code>input</code>	<code>data.table / list</code>
--------------------	--------------------------------

Value

numeric

<code>.getVarComponentTMT</code>	<i>Get variances from models fitted by the groupComparison function</i>
----------------------------------	---

Description

Get variances from models fitted by the groupComparison function

Usage

```
.getVarComponentTMT(fitted_models)
```

Arguments

<code>fitted_models</code>	FittedModels element of groupComparison output
----------------------------	--

.handleSingleContrastTMT
perform statistical inference for single protein and single contrast

Description

perform statistical inference for single protein and single contrast

Usage

```
.handleSingleContrastTMT(  
  contrast,  
  fit,  
  single_protein,  
  coefs,  
  protein,  
  groups,  
  s2_posterior,  
  rho,  
  vss,  
  df_prior,  
  s2_df  
)
```

.logSum *Utility function: compute log of sum of 2^x*

Description

Utility function: compute log of sum of 2^x

Usage

```
.logSum(x)
```

Arguments

x numeric

Value

numeric

`.logSummarizationParams`

Log parameters for proteinSummarization function

Description

Log parameters for proteinSummarization function

Usage

```
.logSummarizationParams(  
  method,  
  global_norm,  
  reference_norm,  
  remove_norm_channel,  
  remove_empty_channel  
)
```

Arguments

<code>method</code>	Four different summarization methods to protein-level can be performed : "msstats"(default), "MedianPolish", "Median", "LogSum".
<code>global_norm</code>	Global median normalization on peptide level data (equalizing the medians across all the channels and MS runs). Default is TRUE. It will be performed before protein-level summarization.
<code>reference_norm</code>	Reference channel based normalization between MS runs on protein level data. TRUE(default) needs at least one reference channel in each MS run, annotated by 'Norm' in Condition column. It will be performed after protein-level summarization. FALSE will not perform this normalization step. If data only has one run, then <code>reference_norm=FALSE</code> .
<code>remove_norm_channel</code>	TRUE(default) removes 'Norm' channels from protein level data.
<code>remove_empty_channel</code>	TRUE(default) removes 'Empty' channels from protein level data.

Value

TRUE invisibly after logging successfully

`.makeContrastSingleTMT`
Make a contrast

Description

Make a contrast

Usage

`.makeContrastSingleTMT(fit, contrast, single_protein, coefs)`

Value

a contrast vector

`.makeFactorColumnsTMT` *Converts required columns to factor in summarization output*

Description

Converts required columns to factor in summarization output

Usage

`.makeFactorColumnsTMT(input)`

Arguments

input data.table

Value

a data table with factored columns

`.medianPolish` *Tukey median polish*

Description

Tukey median polish

Usage

```
.medianPolish(intensities, num_channels)
```

Arguments

`intensities` vector of log-intensities per protein and run
`num_channels` number of channels

Value

numeric vector with length 'num_channels'

`.normalizePeptides` *Normalization between channels (before summarization)*

Description

Normalization between channels (before summarization)

Usage

```
.normalizePeptides(input, normalize)
```

Arguments

`input` data.table
`normalize` logical, if TRUE, 'input' data will be normalized

Value

data.table

.normalizeProteins *Normalization between MS runs (after protein summarization)*

Description

Normalization between MS runs (after protein summarization)

Usage

```
.normalizeProteins(input, normalize)
```

Arguments

input	data.table
normalize	logical, if TRUE, data will be normalized

Value

data.table

.prepareForSummarization
Prepare TMT data for protein-level summarization

Description

Prepare TMT data for protein-level summarization

Usage

```
.prepareForSummarization(input)
```

Arguments

input	data.table
-------	------------

Value

data.table with required column types

```
.removeRedundantChannels
```

Remove empty and normalization channels

Description

Remove empty and normalization channels

Usage

```
.removeRedundantChannels(input, remove_empty_channel, remove_norm_channel)
```

Arguments

`input` data.table processed by the protein summarization function
`remove_empty_channel` TRUE(default) removes 'Empty' channels from protein level data.
`remove_norm_channel` TRUE(default) removes 'Norm' channels from protein level data.

Value

data.table

```
.summarizeMSstats
```

Summarization based on MSstats

Description

Summarization based on MSstats

Usage

```
.summarizeMSstats(  
  input,  
  annotation,  
  impute,  
  max_quantile_censored = NULL,  
  log_file_path = NULL  
)
```


Arguments

- input data.table
- annotation data.table with run and channel annotation
- impute only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.
- max_quantile_censored
 We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.
- log_file_path path to a MSstats log file

Value

data.table

.summarizeSimpleStat *Summarize TMT data with a simple aggregate of log-intensities*

Description

Summarize TMT data with a simple aggregate of log-intensities

Usage

```
.summarizeSimpleStat(input, annotation, stat_aggregate)
```

Arguments

- input data.table
- annotation data.table with run and channel annotation
- stat_aggregate function that will be used to compute protein-level summary

Value

data.table

<code>.summarizeTMP</code>	<i>Summarize TMT data with median polish</i>
----------------------------	--

Description

Summarize TMT data with median polish

Usage

```
.summarizeTMP(input, annotation)
```

Arguments

input	data.table
annotation	data.table with run and channel annotation

Value

data.table with summarized protein intensities

<code>.summarizeTMT</code>	<i>Performs summarization for TMT data</i>
----------------------------	--

Description

Performs summarization for TMT data

Usage

```
.summarizeTMT(
  input,
  method,
  annotation,
  impute,
  max_quantile_censored,
  log_file_path
)
```

Arguments

input	data.table
method	"mstats"/"MedianPolish"/"LogSum"/"Median"
annotation	data.table with run and channel annotation

impute	only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.
max_quantile_censored	We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.
log_file_path	path to a MSstats log file

Value

data.table

annotation.mine	<i>Example of annotation file for raw.mine, which is the output of SpectroMine.</i>
-----------------	---

Description

Annotation of example data, raw.mine, in this package. It should be prepared by users. The variables are as follows:

Usage

annotation.mine

Format

A data frame with 72 rows and 7 variables.

Details

- Run : MS run ID. It should be the same as R.FileName info in raw.mine
- Channel : Labeling information (TMT6_126, ..., TMT6_131). The channels should be consistent with the channel columns in raw.mine.
- Condition : Condition (ex. Healthy, Cancer, Time0). If the channel doesn't have sample, please add 'Empty' under Condition.
- Mixture : Mixture of samples labeled with different TMT reagents, which can be analyzed in a single mass spectrometry experiment.
- TechRepMixture : Technical replicate of one mixture. One mixture may have multiple technical replicates. For example, if 'TechRepMixture' = 1, 2 are the two technical replicates of one mixture, then they should match with same 'Mixture' value.
- Fraction : Fraction ID. One technical replicate of one mixture may be fractionated into multiple fractions to increase the analytical depth. Then one technical replicate of one mixture should correspond to multiple fractions. For example, if 'Fraction' = 1, 2, 3 are three fractions of the first technical replicate of one TMT mixture of biological subjects, then they should have same 'TechRepMixture' and 'Mixture' value.

- BioReplicate : Unique ID for biological subject. If the channel doesn't have sample, please add 'Empty' under BioReplicate

Examples

```
head(annotation.mine)
```

annotation.mq	<i>Example of annotation file for evidence, which is the output of MaxQuant.</i>
---------------	--

Description

Annotation of example data, evidence, in this package. It should be prepared by users. The variables are as follows:

Usage

```
annotation.mq
```

Format

A data frame with 150 rows and 7 variables.

Details

- Run : MS run ID. It should be the same as Raw.file info in raw.mq
- Channel : Labeling information (channel.0, ..., channel.9). The channel index should be consistent with the channel columns in raw.mq.
- Condition : Condition (ex. Healthy, Cancer, Time0)
- Mixture : Mixture of samples labeled with different TMT reagents, which can be analyzed in a single mass spectrometry experiment. If the channel doesn't have sample, please add 'Empty' under Condition.
- TechRepMixture : Technical replicate of one mixture. One mixture may have multiple technical replicates. For example, if 'TechRepMixture' = 1, 2 are the two technical replicates of one mixture, then they should match with same 'Mixture' value.
- Fraction : Fraction ID. One technical replicate of one mixture may be fractionated into multiple fractions to increase the analytical depth. Then one technical replicate of one mixture should correspond to multiple fractions. For example, if 'Fraction' = 1, 2, 3 are three fractions of the first technical replicate of one TMT mixture of biological subjects, then they should have same 'TechRepMixture' and 'Mixture' value.
- BioReplicate : Unique ID for biological subject. If the channel doesn't have sample, please add 'Empty' under BioReplicate.

Examples

```
head(annotation.mq)
```

annotation.pd	<i>Example of annotation file for raw.pd, which is the PSM output of Proteome Discoverer</i>
---------------	--

Description

Annotation of example data, raw.pd, in this package. It should be prepared by users. The variables are as follows:

Usage

```
annotation.pd
```

Format

A data frame with 150 rows and 7 variables.

Details

- Run : MS run ID. It should be the same as Spectrum.File info in raw.pd.
- Channel : Labeling information (126, ... 131). It should be consistent with the channel columns in raw.pd.
- Condition : Condition (ex. Healthy, Cancer, Time0)
- Mixture : Mixture of samples labeled with different TMT reagents, which can be analyzed in a single mass spectrometry experiment. If the channel doesn't have sample, please add 'Empty' under Condition.
- TechRepMixture : Technical replicate of one mixture. One mixture may have multiple technical replicates. For example, if 'TechRepMixture' = 1, 2 are the two technical replicates of one mixture, then they should match with same 'Mixture' value.
- Fraction : Fraction ID. One technical replicate of one mixture may be fractionated into multiple fractions to increase the analytical depth. Then one technical replicate of one mixture should correspond to multiple fractions. For example, if 'Fraction' = 1, 2, 3 are three fractions of the first technical replicate of one TMT mixture of biological subjects, then they should have same 'TechRepMixture' and 'Mixture' value.
- BioReplicate : Unique ID for biological subject. If the channel doesn't have sample, please add 'Empty' under BioReplicate.

Examples

```
head(annotation.pd)
```

dataProcessPlotsTMT *Visualization for explanatory data analysis - TMT experiment*

Description

To illustrate the quantitative data and quality control of MS runs, dataProcessPlotsTMT takes the quantitative data and summarized data from function 'proteinSummarization' as input and generate two types of figures in pdf files as output : (1) profile plot (specify "ProfilePlot" in option type), to identify the potential sources of variation for each protein; (2) quality control plot (specify "QCPlot" in option type), to evaluate the systematic bias between MS runs and channels.

Usage

```
dataProcessPlotsTMT(
  data,
  type,
  featureName = "Transition",
  ylimUp = FALSE,
  ylimDown = FALSE,
  x.axis.size = 10,
  y.axis.size = 10,
  text.size = 2,
  text.angle = 90,
  legend.size = 7,
  dot.size.profile = 2,
  ncol.guide = 5,
  width = 10,
  height = 10,
  which.Protein = "all",
  originalPlot = TRUE,
  summaryPlot = TRUE,
  address = "",
  isPlotly = FALSE
)
```

Arguments

data	the output of proteinSummarization function. It is a list with data frames 'FeatureLevelData' and 'ProteinLevelData'
type	choice of visualization. "ProfilePlot" represents profile plot of log intensities across MS runs. "QCPlot" represents box plots of log intensities across channels and MS runs.
featureName	for "ProfilePlot" only, "Transition" (default) means printing feature legend in transition-level; "Peptide" means printing feature legend in peptide-level; "NA" means no feature legend printing. FALSE(Default) for Profile Plot and QC Plot uses the upper limit as rounded off maximum of $\log_2(\text{intensities})$ after normalization + 3..


```

dataProcessPlotsTMT(data=quant.msstats,
                    type='ProfilePlot',
                    width = 21,
                    height = 7)

## NottoRun: QC plot
# dataProcessPlotsTMT(data=quant.msstats,
#                     # type='QCPlot',
#                     # width = 21,
#                     # height = 7)

```

designSampleSizeTMT *Planning future experimental designs of Tandem Mass Tag (TMT) experiments acquired with Data-Dependent Acquisition (DDA or shotgun)*

Description

Calculate sample size for future experiments of a TMT experiment based on intensity-based linear model. Two options of the calculation: (1) number of biological replicates per condition, (2) power.

Usage

```

designSampleSizeTMT(
  data,
  desiredFC,
  FDR = 0.05,
  numSample = TRUE,
  power = 0.9,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL
)

```

Arguments

data	'FittedModel' in testing output from function groupComparisonTMT.
desiredFC	the range of a desired fold change which includes the lower and upper values of the desired fold change.
FDR	a pre-specified false discovery ratio (FDR) to control the overall false positive rate. Default is 0.05
numSample	minimal number of biological replicates per condition. TRUE represents you require to calculate the sample size for this category, else you should input the exact number of biological replicates.

power	a pre-specified statistical power which defined as the probability of detecting a true fold change. TRUE represent you require to calculate the power for this category, else you should input the average of power you expect. Default is 0.9
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.

Details

The function fits the model and uses variance components to calculate sample size. The underlying model fitting with intensity-based linear model with technical MS run replication. Estimated sample size is rounded to 0 decimal. The function can only obtain either one of the categories of the sample size calculation (numSample, numPep, numTran, power) at the same time.

Value

data.frame - sample size calculation results including variables: desiredFC, numSample, FDR, and power.

Examples

```
data(input.pd)
# use protein.summarization() to get protein abundance data
quant.pd.msstats = proteinSummarization(input.pd,
                                       method="msstats",
                                       global_norm=TRUE,
                                       reference_norm=TRUE)

test.pairwise = groupComparisonTMT(quant.pd.msstats, save_fitted_models = TRUE)
head(test.pairwise$ComparisonResult)

## Calculate sample size for future experiments:
#(1) Minimal number of biological replicates per condition
designSampleSizeTMT(data=test.pairwise$FittedModel, numSample=TRUE,
                   desiredFC=c(1.25,1.75), FDR=0.05, power=0.8)
#(2) Power calculation
designSampleSizeTMT(data=test.pairwise$FittedModel, numSample=2,
                   desiredFC=c(1.25,1.75), FDR=0.05, power=TRUE)
```

evidence

Example of output from MaxQuant for TMT-10plex experiments.

Description

Example of evidence.txt from MaxQuant. It is the input for MaxQtoMSstatsTMTFormat function, with proteinGroups.txt and annotation file. Annotation file should be made by users. It includes peak intensities for 10 proteins among 15 MS runs with TMT10. The important variables are as follows:

Usage

evidence

Format

A data frame with 1075 rows and 105 variables.

Details

- Proteins
- Protein.group.IDs
- Modified.sequence
- Charge
- Raw.file
- Score
- Potential.contaminant
- Reverse
- Channels : Reporter.intensity.corrected.0, ..., Reporter.intensity.corrected.9

Examples

```
head(evidence)
```

`getProcessedTMT` *Get processed feature-level data*

Description

Get processed feature-level data

Usage

```
getProcessedTMT(summarized, input)
```

Arguments

`summarized` output of the `MSstatsSummarizeTMT` function
`input` output of `MSstatsNormalizeTMT` function

Value

`data.table`

`getSummarizedTMT` *Get protein-level data from MSstatsSummarizeTMT output*

Description

Get protein-level data from `MSstatsSummarizeTMT` output

Usage

```
getSummarizedTMT(summarized)
```

Arguments

`summarized` output of the `MSstatsSummarizeTMT` function

Value

`data.table`

groupComparisonTMT	<i>Finding differentially abundant proteins across conditions in TMT experiment</i>
--------------------	---

Description

Tests for significant changes in protein abundance across conditions based on a family of linear mixed-effects models in TMT experiment. Experimental design of case-control study (patients are not repeatedly measured) is automatically determined based on proper statistical model.

Usage

```
groupComparisonTMT(
  data,
  contrast.matrix = "pairwise",
  moderated = FALSE,
  adj.method = "BH",
  remove_norm_channel = TRUE,
  remove_empty_channel = TRUE,
  save_fitted_models = FALSE,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL
)
```

Arguments

data	the output of proteinSummarization function. It is a list with data frames 'FeatureLevelData' and 'ProteinLevelData'
contrast.matrix	Comparison between conditions of interests. 1) default is "pairwise", which compare all possible pairs between two conditions. 2) Otherwise, users can specify the comparisons of interest. Based on the levels of conditions, specify 1 or -1 to the conditions of interests and 0 otherwise. The levels of conditions are sorted alphabetically.
moderated	TRUE will moderate t statistic; FALSE (default) uses ordinary t statistic.
adj.method	adjusted method for multiple comparison. "BH" is default.
remove_norm_channel	TRUE(default) removes "Norm" channels from protein level data.
remove_empty_channel	TRUE(default) removes "Empty" channels from protein level data.
save_fitted_models	logical, if TRUE, fitted models will be added to
use_log_file	logical. If TRUE, information about data processing will be saved to a file.

append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.

Value

a list that consists of the following elements: (1) ComparisonResult: statistical testing results; (2) FittedModel: the fitted linear models

Examples

```
data(input.pd)
# use protein.summarization() to get protein abundance data
quant.pd.msstats = proteinSummarization(input.pd,
                                       method="msstats",
                                       global_norm=TRUE,
                                       reference_norm=TRUE)

test.pairwise = groupComparisonTMT(quant.pd.msstats, moderated = TRUE)
head(test.pairwise$ComparisonResult)

# Only compare condition 0.125 and 1
levels(quant.pd.msstats$ProteinLevelData$Condition)

# Compare condition 1 and 0.125
comparison=matrix(c(-1,0,0,1),nrow=1)

# Set the names of each row
row.names(comparison)="1-0.125"

# Set the column names
colnames(comparison)= c("0.125", "0.5", "0.667", "1")
test.contrast = groupComparisonTMT(data = quant.pd.msstats,
contrast.matrix = comparison,
moderated = TRUE)
head(test.contrast$ComparisonResult)
```

input.pd

Example of output from PDtoMSstatsTMTFormat function

Description

It is made from [raw.pd](#) and [annotation.pd](#), which is the output of PDtoMSstatsTMTFormat function. It should include the required columns as below.

Usage

```
input.pd
```

Format

A data frame with 20110 rows and 11 variables.

Details

- ProteinName : Protein ID
- PeptideSequence : peptide sequence
- Charge : peptide charge
- PSM : peptide ion and spectra match
- Channel : Labeling information (126, ... 131)
- Condition : Condition (ex. Healthy, Cancer, Time0)
- BioReplicate : Unique ID for biological subject.
- Run : MS run ID
- Mixture : Unique ID for TMT mixture.
- TechRepMixture : Unique ID for technical replicate of one TMT mixture.
- Intensity: Protein Abundance

Examples

```
head(input.pd)
```

MaxQtoMSstatsTMTFormat

Generate MSstatsTMT required input format from MaxQuant output

Description

Generate MSstatsTMT required input format from MaxQuant output

Usage

```
MaxQtoMSstatsTMTFormat(  
  evidence,  
  proteinGroups,  
  annotation,  
  which.proteinid = "Proteins",  
  rmProt_Only.identified.by.site = FALSE,  
  useUniquePeptide = TRUE,  
  rmPSM_withfewMea_withinRun = TRUE,
```

```

    rmProtein_with1Feature = FALSE,
    summaryforMultipleRows = sum,
    use_log_file = TRUE,
    append = FALSE,
    verbose = TRUE,
    log_file_path = NULL,
    ...
)

```

Arguments

evidence	name of 'evidence.txt' data, which includes feature-level data.
proteinGroups	name of 'proteinGroups.txt' data.
annotation	data frame which contains column Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition. Refer to the example 'annotation.mq' for the meaning of each column.
which.proteinid	Use 'Proteins' (default) column for protein name. 'Leading.proteins' or 'Leading.razor.proteins' or 'Gene.names' can be used instead to get the protein ID with single protein. However, those can potentially have the shared peptides.
rmProt_Only_identified.by.site	TRUE will remove proteins with '+' in 'Only.identified.by.site' column from proteinGroups.txt, which was identified only by a modification site. FALSE is the default.
useUniquePeptide	TRUE(default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
summaryforMultipleRows	sum(default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
...	additional parameters to 'data.table::fread'.

Value

data.frame of class "MSstatsTMT"

Examples

```
head(evidence)
head(proteinGroups)
head(annotation.mq)
input.mq <- MaxQtoMSstatsTMTFormat(evidence, proteinGroups, annotation.mq)
head(input.mq)
```

MSstatsComparisonModelSingleTMT

Fit a linear model for group comparison for a single protein

Description

Fit a linear model for group comparison for a single protein

Usage

```
MSstatsComparisonModelSingleTMT(single_protein, protein_name)
```

Arguments

`single_protein` protein-level data for a single protein (single element of list created by the `MSstatsPrepareForGroupComparisonTMT` function)
`protein_name` name of a protein from the `single_protein` data.table

Value

list

MSstatsFitComparisonModelsTMT

Fit linear models for group comparison

Description

Fit linear models for group comparison

Usage

```
MSstatsFitComparisonModelsTMT(input)
```


Arguments

input output of the MSstatsPrepareForGroupComparisonTMT function

Value

list

MSstatsGroupComparisonOutputTMT

Combine testing results for individual proteins

Description

Combine testing results for individual proteins

Usage

```
MSstatsGroupComparisonOutputTMT(testing_results, adj_method)
```

Arguments

testing_results

output of the MSstatsGroupComparisonTMT function

adj_method

method that will be used to adjust p-values for multiple comparisons

Value

data.table

MSstatsGroupComparisonTMT

Group comparison for TMT data

Description

Group comparison for TMT data

Usage

```
MSstatsGroupComparisonTMT(fitted_models, contrast_matrix)
```

Arguments

fitted_models output of the MSstatsModerateTTest function

contrast_matrix

contrast matrix

Value

data.table

MSstatsModerateTTest *Moderate T statistic for group comparison*

Description

Moderate T statistic for group comparison

Usage

```
MSstatsModerateTTest(summarized, fitted_models, moderated)
```

Arguments

summarized	protein-level data produced by the proteinSummarization function
fitted_models	output of the MSstatsFitComparisonModelsTMT function
moderated	if TRUE, moderation will be performed

Value

list

MSstatsNormalizeTMT *Normalization for TMT data*

Description

Normalization for TMT data

Usage

```
MSstatsNormalizeTMT(input, type, normalize)
```

Arguments

input	data.table
type	"peptides" for peptide normalization between channel and run, "proteins" for protein normalization
normalize	logical, if TRUE, data will be normalized

Value

data.table

MSstatsPrepareForGroupComparisonTMT
Prepare output of proteinSummarization for group comparison

Description

Prepare output of proteinSummarization for group comparison

Usage

```
MSstatsPrepareForGroupComparisonTMT(  
  input,  
  remove_norm_channel,  
  remove_empty_channel  
)
```

Arguments

input output of proteinSummarization
remove_norm_channel if TRUE, "Norm" channel will be removed
remove_empty_channel if TRUE, empty channel will be removed

Value

data.table

MSstatsPrepareForSummarizationTMT
Prepare output of MSstatsTMT converters for protein-level summarization

Description

Prepare output of MSstatsTMT converters for protein-level summarization

Usage

```
MSstatsPrepareForSummarizationTMT(  
  data,  
  method,  
  global_norm,  
  reference_norm,  
  remove_norm_channel,
```

```

    remove_empty_channel,
    MBimpute,
    maxQuantileforCensored
  )

```

Arguments

<code>data</code>	Name of the output of PDtoMSstatsTMTFormat function or peptide-level quantified data from other tools. It should have columns ProteinName, PeptideSequence, Charge, PSM, Mixture, TechRepMixture, Run, Channel, Condition, BioReplicate, Intensity
<code>method</code>	Four different summarization methods to protein-level can be performed : "msstats"(default), "MedianPolish", "Median", "LogSum".
<code>global_norm</code>	Global median normalization on peptide level data (equalizing the medians across all the channels and MS runs). Default is TRUE. It will be performed before protein-level summarization.
<code>reference_norm</code>	Reference channel based normalization between MS runs on protein level data. TRUE(default) needs at least one reference channel in each MS run, annotated by 'Norm' in Condition column. It will be performed after protein-level summarization. FALSE will not perform this normalization step. If data only has one run, then reference_norm=FALSE.
<code>remove_norm_channel</code>	TRUE(default) removes 'Norm' channels from protein level data.
<code>remove_empty_channel</code>	TRUE(default) removes 'Empty' channels from protein level data.
<code>MBimpute</code>	only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.
<code>maxQuantileforCensored</code>	We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.

Value

data.table

MSstatsSummarizationOutputTMT

Combine feature-level and protein-level data into single output

Description

Combine feature-level and protein-level data into single output

Usage

```
MSstatsSummarizationOutputTMT(
  summarized,
  processed,
  remove_empty_channel,
  remove_norm_channel
)
```

Arguments

summarized output of the getSummarizedTMT function
 processed output of the getProcessedTMT function
 remove_empty_channel
 TRUE(default) removes 'Empty' channels from protein level data.
 remove_norm_channel
 TRUE(default) removes 'Norm' channels from protein level data.

Value

list that consists of two data.frames with feature-level and protein-level data

MSstatsSummarizeTMT *Protein summarization for TMT data*

Description

Protein summarization for TMT data

Usage

```
MSstatsSummarizeTMT(
  input,
  method,
  impute,
  max_quantile_censored = NULL,
  log_file_path = NULL
)
```

Arguments

input data.table with TM quant data
 method Four different summarization methods to protein-level can be performed : "msstats"(default), "MedianPolish", "Median", "LogSum".
 impute only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.

max_quantile_censored
 We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.

log_file_path path to a MSstats log file

Value

data.table

MSstatsTestSingleProteinTMT
Hypothesis tests for a single protein in TMT data

Description

Hypothesis tests for a single protein in TMT data

Usage

MSstatsTestSingleProteinTMT(fitted_model, contrast_matrix)

Arguments

fitted_model single element of the MSstatsModerateTTest output
 contrast_matrix contrast matrix

Value

list

MSstatsTMT
MSstatsTMT: A package for protein significance analysis in shotgun mass spectrometry-based proteomic experiments with tandem mass tag (TMT) labeling

Description

A set of tools for detecting differentially abundant peptides and proteins in shotgun mass spectrometry-based proteomic experiments with tandem mass tag (TMT) labeling.

functions

- [PDtoMSstatsTMTFormat](#) : generates MSstatsTMT required input format for Proteome discoverer output.
- [MaxQtoMSstatsTMTFormat](#) : generates MSstatsTMT required input format for MaxQuant output.
- [SpectroMinetoMSstatsTMTFormat](#) : generates MSstatsTMT required input format for SpectroMine output.
- [OpenMStoMSstatsTMTFormat](#) : generates MSstatsTMT required input format for OpenMS output.
- [proteinSummarization](#) : summarizes PSM level quantification to protein level quantification.
- [dataProcessPlotsTMT](#) : visualizes for explanatory data analysis.
- [groupComparisonTMT](#) : tests for significant changes in protein abundance across conditions.

Author(s)

Maintainer: Devon Kohler <kohler.d@northeastern.edu>

Authors:

- Ting Huang <thuang0703@gmail.com>
- Meena Choi <mnchoi67@gmail.com>
- Mateusz Staniak <mtst@mstaniak.pl>
- Sicheng Hao <hao.sic@husky.neu.edu>
- Olga Vitek <o.vitek@northeastern.edu>

See Also

Useful links:

- <http://msstats.org/msstatstmt/>
- Report bugs at <https://groups.google.com/forum/#!forum/msstats>

OpenMStoMSstatsTMTFormat

Generate MSstatsTMT required input format for OpenMS output

Description

Generate MSstatsTMT required input format for OpenMS output

Usage

```
OpenMStoMSstatsTMTFormat(
  input,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultiplePSMs = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

Arguments

<code>input</code>	MSstatsTMT report from OpenMS
<code>useUniquePeptide</code>	TRUE(default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
<code>rmPSM_withfewMea_withinRun</code>	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
<code>rmProtein_with1Feature</code>	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
<code>summaryforMultiplePSMs</code>	sum(default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
<code>use_log_file</code>	logical. If TRUE, information about data processing will be saved to a file.
<code>append</code>	logical. If TRUE, information about data processing will be added to an existing log file.
<code>verbose</code>	logical. If TRUE, information about data processing wil be printed to the console.
<code>log_file_path</code>	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
<code>...</code>	additional parameters to 'data.table::fread'.

Value

'data.frame' of class 'MSstatsTMT'.

Examples

```
head(raw.om)
input.om <- OpenMStoMSstatsTMTFormat(raw.om)
```



```
head(input.om)
```

PDtoMSstatsTMTFormat *Convert Proteome Discoverer output to MSstatsTMT format.*

Description

Convert Proteome Discoverer output to MSstatsTMT format.

Usage

```
PDtoMSstatsTMTFormat(
  input,
  annotation,
  which.proteinid = "Protein.Accessions",
  useNumProteinsColumn = TRUE,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

Arguments

input	PD report or a path to it.
annotation	annotation with Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition columns or a path to file. Refer to the example 'annotation' for the meaning of each column.
which.proteinid	Use 'Protein.Accessions' (default) column for protein name. 'Master.Protein.Accessions' can be used instead to get the protein name with single protein.
useNumProteinsColumn	logical, TRUE (default) remove shared peptides by information of # Proteins column in PSM sheet.
useUniquePeptide	logical, if TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.

rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
summaryforMultipleRows	sum (default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
...	additional parameters to 'data.table::fread'.

Value

'data.frame' of class 'MSstatsTMT'

Examples

```
head(raw.pd)
head(annotation.pd)
input.pd <- PDtoMSstatsTMTFormat(raw.pd, annotation.pd)
head(input.pd)
```

PhilosophertoMSstatsTMTFormat

Convert Philosopher (Fragpipe) output to MSstatsTMT format.

Description

Convert Philosopher (Fragpipe) output to MSstatsTMT format.

Usage

```
PhilosophertoMSstatsTMTFormat(
  input,
  annotation,
  protein_id_col = "Protein",
  peptide_id_col = "Peptide.Sequence",
  Purity_cutoff = 0.6,
  PeptideProphet_prob_cutoff = 0.7,
  useUniquePeptide = TRUE,
```

```

    rmPSM_withfewMea_withinRun = TRUE,
    rmPeptide_OxidationM = TRUE,
    rmProtein_with1Feature = FALSE,
    summaryforMultipleRows = sum,
    use_log_file = TRUE,
    append = FALSE,
    verbose = TRUE,
    log_file_path = NULL,
    ...
)

```

Arguments

input	data.frame of 'msstats.csv' file produced by Philosopher
annotation	annotation with Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition columns or a path to file. Refer to the example 'annotation' for the meaning of each column. Channel column should be consistent with the channel columns (Ignore the prefix "Channel ") in msstats.csv file. Run column should be consistent with the Spectrum.File columns in msstats.csv file.
protein_id_col	Use 'Protein'(default) column for protein name. 'Master.Protein.Accessions' can be used instead to get the protein ID with single protein.
peptide_id_col	Use 'Peptide.Sequence'(default) column for peptide sequence. 'Modified.Peptide.Sequence' can be used instead to get the modified peptide sequence.
Purity_cutoff	Cutoff for purity. Default is 0.6
PeptideProphet_prob_cutoff	Cutoff for the peptide identification probability. Default is 0.7. The probability is confidence score determined by PeptideProphet and higher values indicate greater confidence.
useUniquePeptide	logical, if TRUE (default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
rmPeptide_OxidationM	TRUE (default) will remove the peptides including oxidation (M) sequence.
rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.
summaryforMultipleRows	sum (default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.

verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
...	additional parameters to 'data.table::fread'.

Value

'data.frame' of class 'MSstatsTMT'

proteinGroups	<i>Example of proteinGroups file from MaxQuant for TMT-10plex experiments.</i>
---------------	--

Description

Example of proteinGroup.txt file from MaxQuant, which is identified protein group information file. It is the input for MaxQtoMSstatsTMTFormat function, with evidence.txt and annotation file. It includes identified protein groups for 10 proteins among 15 MS runs with TMT10. The important variables are as follows:

Usage

```
proteinGroups
```

Format

A data frame with 1075 rows and 105 variables.

Details

- id
- Protein.IDs
- Only.identified.by.site
- Potential.contaminant
- Reverse

Examples

```
head(proteinGroups)
```

proteinSummarization *Summarizing peptide level quantification to protein level quantification*

Description

We assume missing values are censored and then impute the missing values. Protein-level summarization from peptide level quantification are performed. After all, global median normalization on peptide level data and normalization between MS runs using reference channels will be implemented.

Usage

```
proteinSummarization(
  data,
  method = "msstats",
  global_norm = TRUE,
  reference_norm = TRUE,
  remove_norm_channel = TRUE,
  remove_empty_channel = TRUE,
  MBimpute = TRUE,
  maxQuantileforCensored = NULL,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  msstats_log_path = NULL
)
```

Arguments

data	Name of the output of PDtoMSstatsTMTFormat function or peptide-level quantified data from other tools. It should have columns ProteinName, PeptideSequence, Charge, PSM, Mixture, TechRepMixture, Run, Channel, Condition, BioReplicate, Intensity
method	Four different summarization methods to protein-level can be performed : "msstats"(default), "MedianPolish", "Median", "LogSum".
global_norm	Global median normalization on peptide level data (equalizing the medians across all the channels and MS runs). Default is TRUE. It will be performed before protein-level summarization.
reference_norm	Reference channel based normalization between MS runs on protein level data. TRUE(default) needs at least one reference channel in each MS run, annotated by 'Norm' in Condition column. It will be performed after protein-level summarization. FALSE will not perform this normalization step. If data only has one run, then reference_norm=FALSE.

remove_norm_channel	TRUE(default) removes 'Norm' channels from protein level data.
remove_empty_channel	TRUE(default) removes 'Empty' channels from protein level data.
MBimpute	only for method="msstats". TRUE (default) imputes missing values by Accelerated failure model. FALSE uses minimum value to impute the missing value for each peptide precursor ion.
maxQuantileforCensored	We assume missing values are censored. maxQuantileforCensored is Maximum quantile for deciding censored missing value, for instance, 0.999. Default is Null.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
msstats_log_path	path to a MSstats log file

Value

list that consists of two data.frames with feature-level (FeatureLevelData) and protein-level data (ProteinLevelData)

Examples

```
data(input.pd)
quant.pd.msstats <- proteinSummarization(input.pd,
                                         method = "msstats",
                                         global_norm = TRUE,
                                         reference_norm = TRUE)
head(quant.pd.msstats$ProteinLevelData)
```

quant.pd.msstats

Example of output from proteinSummarization function

Description

It is made from [input.pd](#). It is the output of proteinSummarization function. It is a list that consists of two data.frames with feature-level (FeatureLevelData) and protein-level data (ProteinLevelData). ProteinLevelData should include the required columns as below.

Usage

```
quant.pd.msstats
```

Format

A data frame with 100 rows and 8 variables.

Details

- Run : MS run ID
- Protein : Protein ID
- Abundance: Protein-level summarized abundance
- Channel : Labeling information (126, ... 131)
- Condition : Condition (ex. Healthy, Cancer, Time0)
- BioReplicate : Unique ID for biological subject.
- TechRepMixture : Unique ID for technical replicate of one TMT mixture.
- Mixture : Unique ID for TMT mixture.

Examples

```
head(quant.pd.msstats$ProteinLevelData)
```

raw.mine

Example of output from SpectroMine for TMT-6plex experiments.

Description

Example of SpectroMine PSM sheet. It is the output of SpectroMine and the input for SpectroMine-toMSstatsTMTFormat function, with annotation file. Annotation file should be made by users. It includes peak intensities for 10 proteins among 12 MS runs with TMT-6plex. The important variables are as follows:

Usage

```
raw.mine
```

Format

A data frame with 170 rows and 28 variables.

Details

- PG.ProteinAccessions
- P.MoleculeID
- PP.Charge
- R.FileName
- PG.QValue
- PSM.Qvalue
- Channels : PSM.TMT6_126..Raw., ..., PSM.TMT6_131..Raw.

Examples

```
head(raw.mine)
```

```
raw.om
```

Example of MSstatsTMT report from OpenMS for TMT-10plex experiments.

Description

Example of MSstatsTMT PSM sheet from MaxQuant. It is the input for OpenMStoMSstatsTMT-Format function. It includes peak intensities for 10 proteins among 27 MS runs from three TMT10 mixtures. The important variables are as follows:

Usage

```
raw.om
```

Format

A data frame with 860 rows and 13 variables.

Details

- RetentionTime
- ProteinName
- PeptideSequence
- Charge
- Channel
- Condition
- BioReplicate
- Run
- Mixture

- TechRepMixture
- Fraction
- Intensity
- Reference

Examples

```
head(raw.om)
```

raw.pd

Example of output from Proteome Discoverer 2.2 for TMT-10plex experiments.

Description

Example of Proteome discover PSM sheet. It is the input for PDtoMSstatsTMTFormat function, with annotation file. Annotation file should be made by users. It includes peak intensities for 10 proteins among 15 MS runs with TMT-10plex. The variables are as follows:

Usage

```
raw.pd
```

Format

A data frame with 2858 rows and 50 variables.

Details

- Master.Protein.Accessions
- Protein.Accessions
- Annotated.Sequence
- Charge
- Ions.Score
- Spectrum.File
- Quan.Info
- Channels : 126, ..., 131

Examples

```
head(raw.pd)
```

SpectroMineToMSstatsTMTFormat
Import data from SpectroMine

Description

Import data from SpectroMine

Usage

```
SpectroMineToMSstatsTMTFormat(
  input,
  annotation,
  filter_with_Qvalue = TRUE,
  qvalue_cutoff = 0.01,
  useUniquePeptide = TRUE,
  rmPSM_withfewMea_withinRun = TRUE,
  rmProtein_with1Feature = FALSE,
  summaryforMultipleRows = sum,
  use_log_file = TRUE,
  append = FALSE,
  verbose = TRUE,
  log_file_path = NULL,
  ...
)
```

Arguments

input	data name of SpectroMine PSM output. Read PSM sheet.
annotation	data frame which contains column Run, Fraction, TechRepMixture, Mixture, Channel, BioReplicate, Condition. Refer to the example 'annotation.mine' for the meaning of each column.
filter_with_Qvalue	TRUE(default) will filter out the intensities that have greater than qvalue_cutoff in EG.Qvalue column. Those intensities will be replaced with NA and will be considered as censored missing values for imputation purpose.
qvalue_cutoff	Cutoff for EG.Qvalue. default is 0.01.
useUniquePeptide	TRUE(default) removes peptides that are assigned for more than one proteins. We assume to use unique peptide for each protein.
rmPSM_withfewMea_withinRun	TRUE (default) will remove the features that have 1 or 2 measurements within each Run.
rmProtein_with1Feature	TRUE will remove the proteins which have only 1 peptide and charge. Default is FALSE.

summaryforMultipleRows	sum(default) or max - when there are multiple measurements for certain feature in certain run, select the feature with the largest summation or maximal value.
use_log_file	logical. If TRUE, information about data processing will be saved to a file.
append	logical. If TRUE, information about data processing will be added to an existing log file.
verbose	logical. If TRUE, information about data processing will be printed to the console.
log_file_path	character. Path to a file to which information about data processing will be saved. If not provided, such a file will be created automatically. If 'append = TRUE', has to be a valid path to a file.
...	additional parameters to 'data.table::fread'.

Value

'data.frame' of class 'MSstatsTMT'

Examples

```
head(raw.mine)
head(annotation.mine)
input.mine <- SpectroMinetoMSstatsTMTFormat(raw.mine, annotation.mine)
head(input.mine)
```

test.pairwise

Example of output from groupComparisonTMT function

Description

It is the output of groupComparisonTMT function, which is made from [quant.pd.msstats](#). It is a list that consists of the following elements: (1) ComparisonResult: statistical testing results; (2) FittedModel: the fitted linear models ComparisonResult should include the columns as below.

Usage

```
test.pairwise
```

Format

A data frame with 60 rows and 7 variables.

Details

- Protein : Protein ID
- Label: Label of the pairwise comparison or contrast
- log2FC: Log2 fold change
- SE: Standard error of the comparison of contrast results
- DF: Degree of freedom
- pvalue: Value of p statistic of the test
- adj.pvalue: adjusted p value
- issue: used for indicating the reason why a comparison is not testable. NA means the comparison is testable. 'oneConditionMissing' means the protein has no measurements in one condition of the comparison. Furthermore, when 'issue = oneConditionMissing', 'log2FC = Inf' means the negative condition (with coefficient -1 in the Label column) is missing and 'log2FC = -Inf' means the positive condition (with coefficient 1 in the Label column) is missing. 'completeMissing' means the protein has no measurements in all the conditions of the comparison. 'unfittableModel' means there is not enough measurements to fit the linear model. In other words, each condition has only one measurement.

Examples

```
head(test.pairwise$ComparisonResult)
```

Index

* datasets

- annotation.mine, 19
- annotation.mq, 20
- annotation.pd, 21
- evidence, 26
- input.pd, 29
- proteinGroups, 44
- raw.mine, 47
- raw.om, 48
- raw.pd, 49

* internal

- .calculatePower, 3
- .checkContrastMatrix, 4
- .checkSummarizationParams, 4
- .countRunsWithNorm, 6
- .documentFunction, 6
- .getMedianSigmaRun, 7
- .getMedianSigmaSubject, 8
- .getNormalizationAbundance, 8
- .getNumSample, 9
- .getPhilosopherInput, 9
- .getRunsMedian, 10
- .getVarComponentTMT, 10
- .handleSingleContrastTMT, 11
- .logSum, 11
- .logSummarizationParams, 12
- .makeContrastSingleTMT, 13
- .makeFactorColumnsTMT, 13
- .medianPolish, 14
- .normalizePeptides, 14
- .normalizeProteins, 15
- .prepareForSummarization, 15
- .removeRedundantChannels, 16
- .summarizeMSstats, 16
- .summarizeSimpleStat, 17
- .summarizeTMP, 18
- .summarizeTMT, 18
- getProcessedTMT, 27
- getSummarizedTMT, 27

- MSstatsComparisonModelSingleTMT, 32
- MSstatsFitComparisonModelsTMT, 32
- MSstatsGroupComparisonOutputTMT, 33
- MSstatsGroupComparisonTMT, 33
- MSstatsModerateTTest, 34
- MSstatsNormalizeTMT, 34
- MSstatsPrepareForGroupComparisonTMT, 35
- MSstatsPrepareForSummarizationTMT, 35
- MSstatsSummarizationOutputTMT, 36
- MSstatsSummarizeTMT, 37
- MSstatsTestSingleProteinTMT, 38
- quant.pd.msstats, 46
- test.pairwise, 51

- .calculatePower, 3
- .checkContrastMatrix, 4
- .checkSummarizationParams, 4
- .countRunsWithNorm, 6
- .documentFunction, 6
- .getMedianSigmaRun, 7
- .getMedianSigmaSubject, 8
- .getNormalizationAbundance, 8
- .getNumSample, 9
- .getPhilosopherInput, 9
- .getRunsMedian, 10
- .getVarComponentTMT, 10
- .handleSingleContrastTMT, 11
- .logSum, 11
- .logSummarizationParams, 12
- .makeContrastSingleTMT, 13
- .makeFactorColumnsTMT, 13
- .medianPolish, 14
- .normalizePeptides, 14
- .normalizeProteins, 15
- .prepareForSummarization, 15
- .removeRedundantChannels, 16

- .summarizeMSstats, 16
- .summarizeSimpleStat, 17
- .summarizeTMP, 18
- .summarizeTMT, 18

- annotation.mine, 19
- annotation.mq, 20
- annotation.pd, 21, 29

- dataProcessPlotsTMT, 22, 39
- designSampleSizeTMT, 24

- evidence, 26

- getProcessedTMT, 27
- getSummarizedTMT, 27
- groupComparisonTMT, 28, 39

- input.pd, 29, 46

- MaxQtoMSstatsTMTFormat, 30, 39
- MSstatsComparisonModelSingleTMT, 32
- MSstatsFitComparisonModelsTMT, 32
- MSstatsGroupComparisonOutputTMT, 33
- MSstatsGroupComparisonTMT, 33
- MSstatsModerateTTest, 34
- MSstatsNormalizeTMT, 34
- MSstatsPrepareForGroupComparisonTMT, 35
- MSstatsPrepareForSummarizationTMT, 35
- MSstatsSummarizationOutputTMT, 36
- MSstatsSummarizeTMT, 37
- MSstatsTestSingleProteinTMT, 38
- MSstatsTMT, 38
- MSstatsTMT-package (MSstatsTMT), 38

- OpenMStoMSstatsTMTFormat, 39, 39

- PDtoMSstatsTMTFormat, 39, 41
- PhilosophertoMSstatsTMTFormat, 42
- proteinGroups, 44
- proteinSummarization, 22, 28, 39, 45

- quant.pd.msstats, 46, 51

- raw.mine, 47
- raw.om, 48
- raw.pd, 29, 49

- SpectroMinetoMSstatsTMTFormat, 39, 50

- test.pairwise, 51