

Package ‘NormalyzerDE’

April 21, 2025

Title Evaluation of normalization methods and calculation of differential expression analysis statistics

Version 1.27.0

Author Jakob Willforss

Description NormalyzerDE provides screening of normalization methods for LC-MS based expression data. It calculates a range of normalized matrices using both existing approaches and a novel time-segmented approach, calculates performance measures and generates an evaluation report. Furthermore, it provides an easy utility for Limma- or ANOVA- based differential expression analysis.

Imports vsn, preprocessCore, limma, MASS, ape, car, ggplot2, methods, utils, stats, SummarizedExperiment, matrixStats, ggforce

Suggests knitr, testthat, rmarkdown, roxygen2, hexbin, BiocStyle

VignetteBuilder knitr

biocViews Normalization, MultipleComparison, Visualization, Bayesian, Proteomics, Metabolomics, DifferentialExpression

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.2.3

URL <https://github.com/ComputationalProteomics/NormalyzerDE>

Depends R (>= 4.1.0)

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

git_branch devel

git_last_commit 01c5da0

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-04-20

Maintainer Jakob Willforss <jakob.willforss@hotmail.com>

Contents

analyzeNormalizations	3
calculateANOVAPValues	4

calculateAvgMadMem	5
calculateAvgReplicateVariation	5
calculateContrasts	6
calculateCorrSum	7
calculateFeatureCV	7
calculatePercentageAvgDiffInMat	8
calculateReplicateCV	8
calculateSummarizedCorrelationVector	9
createDirectory	9
detectSingleReplicate	10
detectSingletonSample	10
elapsedSecondsBetweenSystimes	11
example_data	11
example_data_only_values	12
example_design	12
example_stat_data	12
example_stat_summarized_experiment	13
example_summarized_experiment	13
example_wide_data	13
example_wide_design	14
filterLowRep	14
findLowlyVariableFeaturesCVs	15
generateAnnotatedMatrix	15
generatePlots	16
generateStatsReport	17
getCombinedMatrix	18
getIndexList	19
getLowCountSampleFiltered	19
getReplicateSortedData	20
getRowNAFilterContrast	20
getRTNormalizedMatrix	21
getSmoothedRTNormalizedMatrix	22
getVerifiedNormalizerObject	23
getWidenedRTRange	24
globalIntensityNormalization	25
loadData	25
loadDesign	26
loadRawDataFromFile	27
meanNormalization	27
medianNormalization	28
normalizer	28
NormalizerDataset	31
normalizerDE	32
NormalizerEvaluationResults	34
NormalizerResults	35
NormalizerStatistics	36
normMethods	37
performCyclicLoessNormalization	38
performGlobalRLRNormalization	38
performNoNormalization	39
performNormalizations	39
performQuantileNormalization	40

performSMADNormalization	41
performVSNNormalization	41
plotBoxPlot	42
plotComparisonVenns	42
plotContrastPCA	43
plotContrastPHists	44
plotCorrelation	44
plotCVvsIntensity	45
plotDendrograms	45
plotDensity	46
plotFrontPage	46
plotMA	47
plotMDS	47
plotMeanSD	48
plotPHist	48
plotQQ	49
plotReplicateVarAndStableVariables	49
plotReplicateVariance	50
plotRLE	50
plotSampleMappingPage	51
plotSampleOutlierSummary	51
plotScatter	52
plotSigScatter	52
preprocessData	53
printMeta	53
printPlots	54
reduceTechnicalReplicates	54
setupJobDir	55
setupPlotting	56
setupRawContrastObject	56
setupRawDataObject	57
setupTestData	58
validateSampleReplication	58
verifyContrasts	59
verifyDesignMatrix	59
verifyMultipleSamplesPresent	60
verifySummarizedExperiment	60
verifyValidNumbers	61
writeNormalizedDatasets	61
Index	63

analyzeNormalizations *Calculate measures for normalization results*

Description

This function prepares an `NormalizerEvaluationResults` object containing the evaluation measures CV (coefficient of variance), MAD (median absolute deviation), average variance, significance measures (ANOVA between condition groups) and correlation between replicates.

Usage

```
analyzeNormalizations(nr, categoricalAnova = FALSE)
```

Arguments

`nr` Normalyzer results object with calculated results.
`categoricalAnova` Whether categorical or numerical (ordered) ANOVA should be calculated.

Value

Normalyzer results with attached evaluation results object.

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalizerObject("job_name", example_summarized_experiment)
normResults <- normMethods(normObj)
normResultsWithEval <- analyzeNormalizations(normResults)
```

`calculateANOVAPValues` *Calculates ANOVA p-values comparing the different condition groups returning a vector of resulting p-values with NA-values where too few values were present in at least one of the groups.*

Description

Calculates ANOVA p-values comparing the different condition groups returning a vector of resulting p-values with NA-values where too few values were present in at least one of the groups.

Usage

```
calculateANOVAPValues(methodList, sampleReplicateGroups, categoricalANOVA)
```

Arguments

`methodList` List containing normalized matrices
`sampleReplicateGroups` Condition header
`categoricalANOVA` Whether the ANOVA should be calculated using ordered or categorical groups

Value

`avgVarianceMat` Matrix with average variance for each biological condition

calculateAvgMadMem	<i>Calculate average MAD (Median Absolute Deviation) for each feature in each condition and then calculates the average for each replicate group</i>
--------------------	--

Description

Calculate average MAD (Median Absolute Deviation) for each feature in each condition and then calculates the average for each replicate group

Usage

```
calculateAvgMadMem(methodList, sampleReplicateGroups)
```

Arguments

methodList	List containing normalized matrices.
sampleReplicateGroups	Condition header.

Value

condAvgMadMat Matrix with average MAD for each biological condition.

calculateAvgReplicateVariation	<i>Calculate average variance for each feature in each condition and then calculate the average for each replicate group</i>
--------------------------------	--

Description

Calculate average variance for each feature in each condition and then calculate the average for each replicate group

Usage

```
calculateAvgReplicateVariation(methodList, sampleReplicateGroups)
```

Arguments

methodList	List containing normalized matrices.
sampleReplicateGroups	Condition header.

Value

avgVarianceMat Matrix with average variance for each biological condition

calculateContrasts	<i>Performs statistical comparisons between the supplied conditions. It uses the design matrix and data matrix in the supplied Normalyzer-Statistics object. A column is supplied specifying which of the columns in the design matrix that is used for deciding the sample groups. The comparisons vector specifies which pairwise comparisons between condition levels that are to be calculated.</i>
--------------------	---

Description

Optionally, a batch column can be specified allowing compensation for covariate variation in the statistical model. This is only compatible with a Limma-based statistical analysis.

Usage

```
calculateContrasts(
  nst,
  comparisons,
  condCol,
  batchCol = NULL,
  splitter = "-",
  type = "limma",
  leastRepCount = 1
)

## S4 method for signature 'NormalyzerStatistics'
calculateContrasts(
  nst,
  comparisons,
  condCol,
  batchCol = NULL,
  splitter = "-",
  type = "limma",
  leastRepCount = 1
)
```

Arguments

nst	Results evaluation object.
comparisons	String with comparisons for contrasts.
condCol	Column name in design matrix containing condition information.
batchCol	Column name in design matrix containing batch information.
splitter	Character dividing contrast conditions.
type	Type of statistical test (Limma or welch).
leastRepCount	Least replicates in each group to be retained for contrast calculations

Value

nst Statistics object with statistical measures calculated

Examples

```
data(example_stat_summarized_experiment)
nst <- NormalyzerStatistics(example_stat_summarized_experiment)
results <- calculateContrasts(nst, c("1-2", "2-3"), "group")
resultsBatch <- calculateContrasts(nst, c("1-2", "2-3"), "group", batchCol="batch")
```

calculateCorrSum	<i>Calculates internal correlations for each condition having at least two samples and returns a vector with correlation values corresponding to each condition</i>
------------------	---

Description

Calculates internal correlations for each condition having at least two samples and returns a vector with correlation values corresponding to each condition

Usage

```
calculateCorrSum(
  methodData,
  allReplicateGroups,
  sampleGroupsWithReplicates,
  corrType
)
```

Arguments

methodData	Expression data matrix
allReplicateGroups	Full condition header corresponding to data tables columns
sampleGroupsWithReplicates	Unique conditions where number of replicates exceeds one
corrType	Type of correlation (Pearson or Spearman)

Value

corSums

calculateFeatureCV	<i>Calculate CV values for each feature. Iterates through each normalization method and calculates a matrix of CV values where each column correspond to a method and each row corresponds to a feature.</i>
--------------------	--

Description

Calculate CV values for each feature. Iterates through each normalization method and calculates a matrix of CV values where each column correspond to a method and each row corresponds to a feature.

Usage

```
calculateFeatureCV(methodList)
```

Arguments

```
methodList      List containing normalized matrices.
sampleReplicateGroups
                  Condition header.
```

Value

```
methodFeatureCVMatrix Matrix with feature as rows and normalization method as columns
```

```
calculatePercentageAvgDiffInMat
```

General function for calculating percentage difference of average column means in matrix

Description

General function for calculating percentage difference of average column means in matrix

Usage

```
calculatePercentageAvgDiffInMat(targetMat)
```

Arguments

```
targetMat      Matrix for which column means should be compared
```

Value

```
percDiffVector Vector with percentage difference, where first element always will be 100
```

```
calculateReplicateCV    Calculate CV per replicate group and normalization technique
```

Description

Iterates through each normalization method and calculate average CV values per replicate group.

Usage

```
calculateReplicateCV(methodList, sampleReplicateGroups)
```

Arguments

```
methodList      List containing normalized matrices.
sampleReplicateGroups
                  Condition header.
```


Value

avgCVPerNormAndReplicates Matrix with group CVs as rows and normalization technique as columns

calculateSummarizedCorrelationVector

Calculates correlation values between replicates for each condition matrix. Finally returns a matrix containing the results for all dataset

Description

Calculates correlation values between replicates for each condition matrix. Finally returns a matrix containing the results for all dataset

Usage

```
calculateSummarizedCorrelationVector(
  methodlist,
  allReplicateGroups,
  sampleGroupsWithReplicates,
  corrType
)
```

Arguments

methodlist	List containing normalized matrices for each normalization method
allReplicateGroups	Vector with condition groups matching the columns found in the normalization methods
sampleGroupsWithReplicates	Unique vector with condition groups present in two or more samples
corrType	Type of correlation (Pearson or Spearman)

Value

avgCorSum Matrix with column corresponding to normalization approaches and rows corresponding to replicate group

createDirectory	<i>Create directory, or return error if already present</i>
-----------------	---

Description

Create directory, or return error if already present

Usage

```
createDirectory(targetPath)
```

Arguments

targetPath Path where to attempt to create directory

Value

None

detectSingleReplicate *Detect single replicate, and assign related logical*

Description

Detect single replicate, and assign related logical

Usage

```
detectSingleReplicate(nds, quiet)

## S4 method for signature 'NormalizerDataset'
detectSingleReplicate(nds, quiet = FALSE)
```

Arguments

nds Normalizer dataset
quiet Don't give non-error output

Value

bool on whether sample contains only one sample group

detectSingletonSample *Detect single sample group*

Description

Detect single sample group

Usage

```
detectSingletonSample(nds, quiet)

## S4 method for signature 'NormalizerDataset'
detectSingletonSample(nds, quiet = FALSE)
```

Arguments

nds Normalizer dataset.
quiet Only print error messages

Value

None

`elapsedSecondsBetweenSystimes`*Get number of seconds between two Sys.time() objects*

Description

Get number of seconds between two Sys.time() objects

Usage

```
elapsedSecondsBetweenSystimes(start, end)
```

Arguments

<code>start</code>	Start-time object
--------------------	-------------------

<code>end</code>	End-time object
------------------	-----------------

Value

None

`example_data`*Small example dataset used to demonstrate code consistency in testing and as dummy data in the vignette.*

Description

Small example dataset used to demonstrate code consistency in testing and as dummy data in the vignette.

Usage

```
example_data
```

Format

A data frame containing annotation and expression data

example_data_only_values

Same data as in "example_data", but omitting the annotation meaning that it only contains the expression data.

Description

Same data as in "example_data", but omitting the annotation meaning that it only contains the expression data.

Usage

example_data_only_values

Format

A data frame containing expression data

example_design

Design matrix corresponding to the small example datasets.

Description

Design matrix corresponding to the small example datasets.

Usage

example_design

Format

A design matrix corresponding to the dataset "example_data"

example_stat_data

Same data as in "example_data", but normalized and ready for statistical processing.

Description

Same data as in "example_data", but normalized and ready for statistical processing.

Usage

example_stat_data

Format

A normalized data frame ready for statistical processing

```
example_stat_summarized_experiment
```

SummarizedExperiment object prepared with design-matrix, data-matrix and annotation columns for normalized data

Description

SummarizedExperiment object prepared with design-matrix, data-matrix and annotation columns for normalized data

Usage

```
example_stat_summarized_experiment
```

Format

An instance of the class SummarizedExperiment with stats data

```
example_summarized_experiment
```

SummarizedExperiment object prepared with design-matrix, data-matrix and annotation columns loaded for raw data

Description

SummarizedExperiment object prepared with design-matrix, data-matrix and annotation columns loaded for raw data

Usage

```
example_summarized_experiment
```

Format

An instance of the class SummarizedExperiment

```
example_wide_data
```

Full raw NormalyzerDE matrix used for internal testing

Description

Full raw NormalyzerDE matrix used for internal testing

Usage

```
example_wide_data
```

Format

A data table ready for analysis in NormalyzerDE

example_wide_design	<i>Design matrix belonging together with example_wide_data. Used for internal testing.</i>
---------------------	--

Description

Design matrix belonging together with example_wide_data. Used for internal testing.

Usage

```
example_wide_design
```

Format

A design table ready for analysis in NormalyzerDE

filterLowRep	<i>Filter rows with lower than given number of replicates for any condition</i>
--------------	---

Description

Filter rows with lower than given number of replicates for any condition

Usage

```
filterLowRep(df, groups, leastRep = 2)
```

Arguments

df	Dataframe with expression data to filter
groups	Condition groups header
leastRep	Minimum number of replicates in each group to retain

Value

collDesignDf Reduced design matrix

findLowlyVariableFeaturesCVs

Uses a list of FDR-values to extract features with low variance in the log2-transformed dataset. This is then used to calculate the average CV for these 'lowly variable' features in each normalization approach

Description

Uses a list of FDR-values to extract features with low variance in the log2-transformed dataset. This is then used to calculate the average CV for these 'lowly variable' features in each normalization approach

Usage

```
findLowlyVariableFeaturesCVs(referenceFDR, methodList)
```

Arguments

referenceFDR	List of FDR values used as non-normalized reference
methodList	List containing normalized matrices

Value

lowVarFeaturesAverageCVs Average CV values for lowly variable features in each normalization approach

generateAnnotatedMatrix

Generate an annotated data frame from statistics object

Description

Extracts key values (p-value, adjusted p-value, log2-fold change and average expression values) from an NormalyzerStatistics instance and appends these to the annotation- and data-matrices

Usage

```
generateAnnotatedMatrix(nst, prefixSep = "_", compLabels = NULL)
```

Arguments

nst	NormalyzerDE statistics object.
prefixSep	Character string for separating the prefix names from the statistics suffix
compLabels	Vector containing strings to use as prefix for statistical comparisons

Value

outDf Annotated statistics matrix

Examples

```
data(example_stat_summarized_experiment)
statObj <- NormalyzerStatistics(example_stat_summarized_experiment)
statObj <- calculateContrasts(statObj, comparisons=c("1-2", "2-3"), condCol="group", type="limma")
annotDf <- generateAnnotatedMatrix(statObj)
```

generatePlots	<i>Generates a number of visualizations for the performance measures calculated for the normalized matrices. These contain both general measures and direct comparisons for different normalization approaches.</i>
---------------	---

Description

They include:

Usage

```
generatePlots(nr, jobdir, plotRows = 3, plotCols = 4, writeAsPngs = FALSE)
```

Arguments

nr	Normalyzer results object.
jobdir	Path to output directory for run.
plotRows	Number of plot rows.
plotCols	Number of plot columns.
writeAsPngs	Output the report as PNG-plots instead of a single PDF

Details

"Total intensity" Barplot showing the summed intensity in each sample for thelog2-transformed data

"Total missing" Barplot showing the number of missing values found in each sample for the log2-transformed data

Log2-MDS plot: MDS plot where data is reduced to two dimensions allowing inspection of the main global changes in the data

PCV - Intragroup: Mean of intragroup CV of all replicate groups

PMAD - Intragroup: Mean of intragroup median absolute deviation across replicate groups

PEV - Intragroup: Mean of intragroup pooled estimate of variance across the replicate groups

Relative PCV, PMAD and PEV compared to log2: The results from PCV, PMAD and PEV from all normalized data compared to the log2 data

Stable variables plot: 5 analysis of log2 transformed data. Thereafter, global CV of these variables is estimated from different normalized datasets. A plot of global CV of the stable variables from all datasets on the y-axis and PCV-compared to log2 on the x-axis is generated.

CV vs Raw Intensity plots: For the first replicate group in each of the normalized dataset, a plot of PCV of each variable compared to the average intensity of the variable in the replicate group is plotted.

MA plots: Plotted using the plotMA function of the limma package. The first sample in each dataset is plotted against the average of the replicate group that sample belong to.

Scatterplots: The first two samples from each dataset are plotted.

Q-Q plots: QQ-plots are plotted for the first sample in each normalized dataset.

Boxplots: Boxplots for all samples are plotted and colored according to the replicate grouping.

Relative Log Expression (RLE) plots: Relative log expression value plots. Ratio between the expression of the variable and the median expression of this variable across all samples. The samples should be aligned around zero. Any deviation would indicate discrepancies in the data.

Density plots: Density distributions for each sample using the density function. Can capture outliers (if single densities lies far from the others) and see if there is batch effects in the dataset (if for instance there is two clear collections of lines in the data).

MDS plots Multidimensional scaling plot using the cmdscale() function from the stats package. Is often able to show whether replicates group together, and whether there are any clear outliers in the data.

MeanSDplots Displays the standard deviation values against values ordered according to mean. If no dependency on mean is present (as is desired) a flat red line is shown.

Pearson and Spearman correlation Mean of intragroup Pearson and Spearman correlation values for each method.

Dendograms Generated using the hclust function. Data is centered and scaled prior to analysis. Coloring of replicates is done using as.phylo from the ape package.

P-value histograms Histogram plots of p-values after calculating an ANOVA between different condition groups. If no effect is present in the data a flat distribution is expected. If an effect is present a flat distribution is still expected, but with a sharp peak close to zero. If other effects are present it might indicate that the data doesn't support the assumptions of ANOVA, for instance if there are batch effects present in the data.

Value

None

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalizerObject("job_name", example_summarized_experiment)
normResults <- normMethods(normObj)
normResultsWithEval <- analyzeNormalizations(normResults)
outputDir <- tempdir()
generatePlots(normResultsWithEval, outputDir)
```

generateStatsReport	<i>Generate full output report plot document. Plots p-value histograms for each contrast in the NormalizerStatistics instance and writes these to a PDF report.</i>
---------------------	---

Description

Generate full output report plot document. Plots p-value histograms for each contrast in the NormalizerStatistics instance and writes these to a PDF report.

Usage

```
generateStatsReport(
  nst,
  jobName,
  jobDir,
  sigThres = 0.1,
  sigThresType = "fdr",
  log2FoldThres = 0,
  plotRows = 3,
  plotCols = 4,
  writeAsPngs = FALSE
)
```

Arguments

nst	NormalyzerDE statistics object.
jobName	Name of processing run.
jobDir	Path to output directory.
sigThres	Significance threshold for indicating as significant
sigThresType	Type of significance threshold (FDR or p)
log2FoldThres	log2 fold-change required for being counted as significant
plotRows	Number of plot rows.
plotCols	Number of plot columns.
writeAsPngs	Output the report as separate PNG files instead of a single PDF file

Value

None

Examples

```
data(example_stat_summarized_experiment)
statObj <- NormalyzerStatistics(example_stat_summarized_experiment)
statObj <- calculateContrasts(statObj, comparisons=c("1-2", "2-3"),
  condCol="group", type="limma")
outputDir <- tempdir()
generateStatsReport(statObj, "jobName", outputDir)
```

getCombinedMatrix	<i>Merge multiple dataframes using provided function</i>
-------------------	--

Description

Merge multiple dataframes using provided function

Usage

```
getCombinedMatrix(mList, combFunc)
```

Arguments

mList	List containing dataframes of same shape
combFunc	Function performing elementwise merge of matrices

Value

combinedMatrix A single dataframe with combined data

getIndexList	<i>Return list containing vector positions of values in string</i>
--------------	--

Description

Return list containing vector positions of values in string

Usage

```
getIndexList(targetVector)
```

Arguments

targetVector

Value

indexList List where key is condition level and values are indices for the condition

getLowCountSampleFiltered	<i>Verify that samples contain at least a lowest number of values</i>
---------------------------	---

Description

Verify that samples contain at least a lowest number of values

Usage

```
getLowCountSampleFiltered(
  dataMatrix,
  groups,
  threshold = 15,
  stopIfTooFew = TRUE
)
```

Arguments

dataMatrix	Dataframe with processed input data.
groups	Vector containing condition levels.
threshold	Lowest number of allowed values in a column.
stopIfTooFew	Abort run if lower than threshold number of values in column

Value

None

 getReplicateSortedData

Get dataframe with raw data column sorted on replicates

Description

Get dataframe with raw data column sorted on replicates

Usage

```
getReplicateSortedData(rawDataOnly, groups)
```

Arguments

rawDataOnly	Dataframe with unparsed input data matrix.
groups	Vector containing condition levels.

Value

rawData sorted on replicate

 getRowNAFilterContrast

Get contrast vector (TRUE/FALSE-values) indicating whether both at least half values are present, and each sample has at least one non-NA value

Description

Get contrast vector (TRUE/FALSE-values) indicating whether both at least half values are present, and each sample has at least one non-NA value

Usage

```
getRowNAFilterContrast(dataMatrix, replicateHeader, minCount = 1)
```

Arguments

dataMatrix	Matrix with expression values for entities in replicate samples.
replicateHeader	Header showing how samples in matrix are replicated.
minCount	Minimum number of required values present in samples.

Value

Contrast vector

`getRTNormalizedMatrix` *Perform RT-segmented normalization by performing the supplied normalization over retention-time sliced data*

Description

The function orders the retention times and steps through them using the supplied step size (in minutes). If smaller than a fixed lower boundary the window is expanded to ensure a minimum amount of data in each normalization step. An offset can be specified which can be used to perform multiple RT-segmentations with partial overlapping windows.

Usage

```
getRTNormalizedMatrix(
  rawMatrix,
  retentionTimes,
  normMethod,
  stepSizeMinutes = 1,
  windowMinCount = 100,
  offset = 0,
  noLogTransform = FALSE
)
```

Arguments

<code>rawMatrix</code>	Target matrix to be normalized
<code>retentionTimes</code>	Vector of retention times corresponding to <code>rawMatrix</code>
<code>normMethod</code>	The normalization method to apply to the time windows
<code>stepSizeMinutes</code>	Size of windows to be normalized
<code>windowMinCount</code>	Minimum number of values for window to not be expanded.
<code>offset</code>	Whether time window should shifted half step size
<code>noLogTransform</code>	Don't log-transform the data

Value

Normalized matrix

Examples

```
data(example_data_small)
data(example_design_small)
data(example_data_only_values)
dataMat <- example_data_only_values
retentionTimes <- as.numeric(example_data[, "Average.RT"])
performCyclicLoessNormalization <- function(rawMatrix) {
  log2Matrix <- log2(rawMatrix)
  normMatrix <- limma::normalizeCyclicLoess(log2Matrix, method="fast")
  colnames(normMatrix) <- colnames(rawMatrix)
  normMatrix
}
```

```
rtNormMat <- getRTNormalizedMatrix(dataMat, retentionTimes,
performCyclicLoessNormalization, stepSizeMinutes=1, windowMinCount=100)
```

```
getSmoothedRTNormalizedMatrix
```

Generate multiple RT time-window normalized matrices where one is shifted. Merge them using a specified method (mean or median) and return the result.

Description

Uses the function `getRTNormalizedMatrix` to generate multiple normalized matrices which are shifted respective to each other and finally merged into a single matrix. This could potentially reduce effect of fluctuations within individual windows.

Usage

```
getSmoothedRTNormalizedMatrix(
  rawMatrix,
  retentionTimes,
  normMethod,
  stepSizeMinutes,
  windowShifts = 2,
  windowMinCount = 100,
  mergeMethod = "mean",
  noLogTransform = FALSE
)
```

Arguments

<code>rawMatrix</code>	Target matrix to be normalized
<code>retentionTimes</code>	Vector of retention times corresponding to <code>rawMatrix</code>
<code>normMethod</code>	The normalization method to apply to the time windows
<code>stepSizeMinutes</code>	Size of windows to be normalized
<code>windowShifts</code>	Number of frame shifts.
<code>windowMinCount</code>	Minimum number of features within window.
<code>mergeMethod</code>	Layer merging approach. Mean or median.
<code>noLogTransform</code>	Don't log transform the input

Value

Normalized matrix

Examples

```

data(example_data_small)
data(example_data_only_values)
data(example_design_small)
retentionTimes <- as.numeric(example_data[, "Average.RT"])
dataMat <- example_data_only_values
performCyclicLoessNormalization <- function(rawMatrix) {
  log2Matrix <- log2(rawMatrix)
  normMatrix <- limma::normalizeCyclicLoess(log2Matrix, method="fast")
  colnames(normMatrix) <- colnames(rawMatrix)
  normMatrix
}
rtNormMat <- getSmoothedRTNormalizedMatrix(dataMat, retentionTimes,
  performCyclicLoessNormalization, stepSizeMinutes=1, windowMinCount=100,
  windowShifts=2, mergeMethod="median")

```

```
getVerifiedNormalizerObject
```

Verify that input data is in correct format, and if so, return a generated NormalizerDE data object from that input data

Description

This function performs a number of checks on the input data and provides informative error messages if the data isn't fulfilling the required format. Checks include verifying that the design matrix matches to the data matrix, that the data matrix contains valid numbers and that samples have enough values for analysis

Usage

```

getVerifiedNormalizerObject(
  jobName,
  summarizedExp,
  threshold = 15,
  omitSamples = FALSE,
  requireReplicates = TRUE,
  quiet = FALSE,
  noLogTransform = FALSE,
  tinyRunThres = 50
)

```

Arguments

jobName	Name of ongoing run.
summarizedExp	Summarized experiment input object
threshold	Minimum number of features.
omitSamples	Automatically omit invalid samples from analysis.
requireReplicates	Require there to be at least to samples per condition
quiet	Don't print output messages during processing
noLogTransform	Don't log-transform the provided data
tinyRunThres	If less features in run, a limited run is performed

Value

Normalyzer data object representing verified input data.

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalyzerObject("job_name", example_summarized_experiment)
```

getWidenedRTRange	<i>Pick datapoints before and after window until a minimum number is reached Expects the start and end retention times to match actual retention times present in the data</i>
-------------------	--

Description

Pick datapoints before and after window until a minimum number is reached Expects the start and end retention times to match actual retention times present in the data

Usage

```
getWidenedRTRange(
  rtStart,
  rtEnd,
  minimumDatapoints,
  retentionTimes,
  allowTooWideData = FALSE
)
```

Arguments

rtStart	Original retention time start point
rtEnd	Original retention time end point
minimumDatapoints	Required number of datapoints to fulfill
retentionTimes	Vector with all retention times

Value

Vector with start and end of new RT range

globalIntensityNormalization

The normalization divides the intensity of each variable in a sample with the sum of intensities of all variables in the sample and multiplies with the median of sum of intensities of all variables in all samples. The normalized data is then log2-transformed.

Description

The normalization divides the intensity of each variable in a sample with the sum of intensities of all variables in the sample and multiplies with the median of sum of intensities of all variables in all samples. The normalized data is then log2-transformed.

Usage

```
globalIntensityNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix Target matrix to be normalized
noLogTransform Assumes no need for log transformation

Value

Normalized and log-transformed matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- globalIntensityNormalization(example_data_only_values)
```

loadData	<i>Load raw data into dataframe</i>
----------	-------------------------------------

Description

General function which allows specifying different types of input data including "proteios", "maxquant-pep" (peptide output from MaxQuant) and "maxquantprot" (protein output from MaxQuant) formats.

Usage

```
loadData(dataPath, inputFormat = "default")
```

Arguments

dataPath File path to design matrix.
inputFormat If input is given in standard NormalyzerDE format, Proteios format or in MaxQuant protein or peptide format

Value

rawData Raw data loaded into data frame

Examples

```
## Not run:  
df <- loadData("data.tsv")  
  
## End(Not run)
```

loadDesign	<i>Load raw design into dataframe</i>
------------	---------------------------------------

Description

Takes a design path, loads the matrix and ensures that the sample column is in character format and that the group column is in factor format.

Usage

```
loadDesign(designPath, sampleCol = "sample", groupCol = "group")
```

Arguments

designPath	File path to design matrix.
sampleCol	Column name for column containing sample names.
groupCol	Column name for column containing condition levels.

Value

designMatrix Design data loaded into data frame

Examples

```
## Not run:  
df <- loadDesign("design.tsv")  
  
## End(Not run)
```

loadRawDataFromFile	<i>Try reading raw Normalyzer matrix from provided filepath</i>
---------------------	---

Description

Try reading raw Normalyzer matrix from provided filepath

Usage

```
loadRawDataFromFile(inputPath)
```

Arguments

inputPath Path to Normalyzer data.

Value

Table containing raw data from input file.

meanNormalization	<i>Intensity of each variable in a given sample is divided by the mean of sum of intensities of all variables in the sample and then multiplied by the mean of sum of intensities of all variables in all samples. The normalized data is then transformed to log2.</i>
-------------------	---

Description

Intensity of each variable in a given sample is divided by the mean of sum of intensities of all variables in the sample and then multiplied by the mean of sum of intensities of all variables in all samples. The normalized data is then transformed to log2.

Usage

```
meanNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix Target matrix to be normalized
noLogTransform Assumes no need for log transformation

Value

Normalized and log-transformed matrix

Examples

```
data(example_data_only_values_small)  
normMatrix <- meanNormalization(example_data_only_values)
```

medianNormalization	<i>Intensity of each variable in a given sample is divided by the median of intensities of all variables in the sample and then multiplied by the mean of median of sum of intensities of all variables in all samples. The normalized data is then log2-transformed.</i>
---------------------	---

Description

Intensity of each variable in a given sample is divided by the median of intensities of all variables in the sample and then multiplied by the mean of median of sum of intensities of all variables in all samples. The normalized data is then log2-transformed.

Usage

```
medianNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix Target matrix to be normalized
noLogTransform Assumes no need for log transformation

Value

Normalized and log-transformed matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- medianNormalization(example_data_only_values)
```

normalyzer	<i>NormalyzerDE pipeline entry point</i>
------------	--

Description

This function is the main execution point for the normalization part of the NormalyzerDE analysis pipeline. When executed it performs the following steps:

Usage

```
normalyzer(
  jobName,
  designPath = NULL,
  dataPath = NULL,
  experimentObj = NULL,
  outputDir = ".",
  forceAllMethods = FALSE,
  omitLowAbundSamples = FALSE,
  sampleAbundThres = 5,
  tinyRunThres = 50,
```

```

    requireReplicates = TRUE,
    normalizeRetentionTime = TRUE,
    plotRows = 3,
    plotCols = 4,
    zeroToNA = FALSE,
    sampleColName = "sample",
    groupColName = "group",
    inputFormat = "default",
    skipAnalysis = FALSE,
    quiet = FALSE,
    noLogTransform = FALSE,
    writeReportAsPngs = FALSE,
    rtStepSizeMinutes = 1,
    rtWindowMinCount = 100,
    rtWindowShifts = 1,
    rtWindowMergeMethod = "mean"
)

```

Arguments

jobName	Give the current run a name.
designPath	Path to file containing design matrix.
dataPath	Specify an output directory for generated files. Defaults to current working directory.
experimentObj	SummarizedExperiment object, can be provided as input as alternative to 'designPath' and 'dataPath'
outputDir	Directory where results folder is created.
forceAllMethods	Debugging function. Run all normalizations even if they aren't in the recommended range of number of values
omitLowAbundSamples	Automatically remove samples with fewer non-NA values compared to threshold given by sampleAbundThres. Will otherwise stop with error message if such sample is encountered.
sampleAbundThres	Threshold for omitting low-abundant samples. Is by default set to 15.
tinyRunThres	If total number of features is less than this, a limited run is performed.
requireReplicates	Require multiple samples per condition to pass input validation.
normalizeRetentionTime	Perform normalizations over retention time.
plotRows	Number of plot-rows in output documentation.
plotCols	Number of plot-columns in output documentation.
zeroToNA	Convert zero values to NA.
sampleColName	Column name in design matrix containing sample IDs.
groupColName	Column name in design matrix containing condition IDs.
inputFormat	Type of input format.
skipAnalysis	Only perform normalization steps.

quiet Omit status messages printed during run.
 noLogTransform Don't log-transform the input.
 writeReportAsPngs Output the evaluation report as PNG files instead of a single PDF
 rtStepSizeMinutes Retention time normalization window size.
 rtWindowMinCount Minimum number of datapoints in each retention-time segment.
 rtWindowShifts Number of layered retention time normalized windows.
 rtWindowMergeMethod Merge approach for layered retention time windows.

Details

1: Loads the data matrix containing expression values and optional annotations, as well as the design matrix containing the experimental setup 2: Performs input data verification to validate that the data is in correct format. This step captures many common formatting errors. It returns an instance of the NormalyzerDataset class representing the unprocessed data. 3: Calculate a range of normalizations for the dataset. The result is provided as a NormalyzerResults object containing the resulting data matrices from each normalization. 4: Analyze the normalizations and generate performance measures for each of the normalized datasets. This result is provided as a NormalyzerEvaluationResults object. 5: Output the matrices containing the normalized datasets to files. 6: Generate visualizations overviewing the performance measures and write them to a PDF report.

Value

None

Examples

```
## Not run:
data_path <- system.file(package="NormalyzerDE", "extdata", "tiny_data.tsv")
design_path <- system.file(package="NormalyzerDE", "extdata", "tiny_design.tsv")
out_dir <- tempdir()
normalyzer(
  jobName="my_jobname",
  designPath=design_path,
  dataPath=data_path,
  outputDir=out_dir)
normalyzer(
  "my_jobname",
  designMatrix="design.tsv",
  "data.tsv",
  outputDir="path/to/output",
  normalizeRetentionTime=TRUE,
  retentionTimeWindow=2)
normalyzer(
  "my_jobname",
  designMatrix="design.tsv",
  "data.tsv",
  outputDir="path/to/output",
  inputFormat="maxquantprot")

## End(Not run)
```

NormalyzerDataset	<i>Represents raw input data together with basic annotation information</i>
-------------------	---

Description

Takes a job name, a data matrix, a design matrix as well as specification of the group and sample columns in the design matrix. Provides the basic representation of a dataset in the NormalyzerDE normalization part.

Usage

```
NormalyzerDataset(
  jobName,
  designMatrix,
  rawData,
  annotationData,
  sampleNameCol,
  groupNameCol,
  tinyRunThres = 50,
  quiet = FALSE
)
```

```
NormalyzerDataset(
  jobName,
  designMatrix,
  rawData,
  annotationData,
  sampleNameCol,
  groupNameCol,
  tinyRunThres = 50,
  quiet = FALSE
)
```

Arguments

jobName	Name of the NormalyzerDE processing run
designMatrix	Matrix containing sample conditions
rawData	Matrix containing raw input data
annotationData	Matrix containing annotation information for each input feature. Is expected to contain the same number of rows as the data but can contain any number of features
sampleNameCol	Name of column in design matrix containing sample information
groupNameCol	Name of column in design matrix containing condition information
tinyRunThres	If fewer features than this is present in the input a limited run will be performed to avoid some steps requiring a more extensive number of features.
quiet	If set to TRUE no information messages will be printed

Value

nds Generated NormalyzerDataset instance

Slots

jobName Name of the job represented by the dataset.
 rawData Matrix with raw values.
 sampleNameCol Name column for sample.
 groupNameCol Name column for groups.
 designMatrix Data frame containing design.
 sampleNames Vector containing sample names.
 filterrawdata Reduced raw data matrix where low abundance rows are removed
 sampleReplicateGroups Vector with sample replicate information
 samplesGroupsWithReplicates Vector with replicated sample replicate information
 annotationValues Annotation part of original dataframe.
 retentionTimes Vector of retention time values.
 singleReplicateRun Conditional whether run is single replicate.

 normalyzerDE

NormalyzerDE differential expression

Description

Performs differential expression analysis on a normalization matrix. This command executes a pipeline processing the data and generates an annotated normalization matrix and a report containing p-value histograms for each of the performed comparisons.

Usage

```

normalyzerDE(
  jobName,
  comparisons,
  designPath = NULL,
  dataPath = NULL,
  experimentObj = NULL,
  outputDir = ".",
  logTrans = FALSE,
  type = "limma",
  sampleCol = "sample",
  condCol = "group",
  batchCol = NULL,
  techRepCol = NULL,
  leastRepCount = 1,
  quiet = FALSE,
  sigThres = 0.1,
  sigThresType = "fdr",
  log2FoldThres = 0,
  writeReportAsPngs = FALSE
)

```


Arguments

jobName	Name of job
comparisons	Character vector containing target contrasts. If comparing condA with condB, then the vector would be c("condA-condB")
designPath	File path to design matrix
dataPath	File path to normalized matrix
experimentObj	SummarizedExperiment object, can be provided as input as alternative to 'designPath' and 'dataPath'
outputDir	Path to output directory
logTrans	Log transform the input (needed if providing non-logged input)
type	Type of statistical comparison, "limma", "limma_intensity" or "welch", where "limma_intensity" allows the prior to be fit according to intensity rather than using a flat prior
sampleCol	Design matrix column header for column containing sample IDs
condCol	Design matrix column header for column containing sample conditions
batchCol	Provide an optional column for inclusion of possible batch variance in the model
techRepCol	Design matrix column header for column containing technical replicates
leastRepCount	Minimum required replicate count
quiet	Omit status messages printed during run
sigThres	Significance threshold use for illustrating significant hits in diagnostic plots
sigThresType	Type of significance threshold, "fdr" or "p". "fdr" is strongly recommended (Benjamini-Hochberg corrected p-values)
log2FoldThres	Fold-size cutoff for being considered significant in diagnostic plots
writeReportAsPngs	Output report as separate PNG files instead of a single PDF

Details

When executed, it performs the following steps:

1: Read the data and the design matrices into dataframes. 2: Generate an instance of the NormalyzerStatistics class representing the data and their statistical comparisons. 3: Optionally reduce technical replicates in both the data matrix and the design matrix 4: Calculate statistical contrasts between supplied groups 5: Generate an annotated version of the original dataframe where columns containing statistical key measures have been added 6: Write the table to file 7: Generate a PDF report displaying p-value histograms for each calculated contrast

Value

None

Examples

```
data_path <- system.file(package="NormalyzerDE", "extdata", "tiny_data.tsv")
design_path <- system.file(package="NormalyzerDE", "extdata", "tiny_design.tsv")
out_dir <- tempdir()
normalyzerDE(
  jobName="my_jobname",
  comparisons=c("4-5"),
```

```

designPath=design_path,
dataPath=data_path,
outputDir=out_dir,
condCol="group")

```

NormalyzerEvaluationResults

Representation of evaluation results by calculating performance measures for an an NormalyzerResults instance

Description

Contains the resulting information from the processing which subsequently can be used to generate the quality assessment report.

Usage

```
NormalyzerEvaluationResults(nr)
```

```
NormalyzerEvaluationResults(nr)
```

Arguments

nr NormalyzerResults object

Value

nds Generated NormalyzerEvaluationResults instance

Slots

avgcvmem Average coefficient of variance per method

avgcvmempdiff Percentage difference of mean coefficient of variance compared to log2-transformed data

featureCVPerMethod CV calculated per feature and normalization method.

avgmadmem Average median absolute deviation

avgmadmempdiff Percentage difference of median absolute deviation compared to log2-transformed data

avgvarmem Average variance per method

avgvarmempdiff Percentage difference of mean variance compared to log2-transformed data

lowVarFeaturesCVs List of 5 for log2-transformed data

lowVarFeaturesCVsPercDiff Coefficient of variance for least variable entries

anovaP ANOVA calculated p-values

repCorPear Within group Pearson correlations

repCorSpear Within group Spearman correlations

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalizerObject("job_name", example_summarized_experiment)
normResults <- normMethods(normObj)
normEval <- NormalizerEvaluationResults(normResults)
```

NormalizerResults	<i>Representation of the results from performing normalization over a dataset</i>
-------------------	---

Description

It is linked to a NormalizerDataset instance representing the raw data which has been processed. After performing evaluation it also links to an instance of NormalizerEvaluationResults representing the results from the evaluation.

Usage

```
NormalizerResults(nds)
```

```
NormalizerResults(nds)
```

Arguments

nds NormalizerDataset object

Value

nr Prepared NormalizerResults object

Slots

normalizations SummarizedExperiment object containing calculated normalization results

nds Normalizer dataset representing run data

ner Normalizer evaluation results for running extended normalizations

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalizerObject("job_name", example_summarized_experiment)
emptyNormResults <- NormalizerResults(normObj)
```

NormalyzerStatistics	<i>Class representing a dataset for statistical processing in NormalyzerDE</i>
----------------------	--

Description

Is initialized with an annotation matrix, a data matrix and a design data frame. This object can subsequently be processed to generate statistical values and in turn used to write a full matrix with additional statistical information as well as a graphical report of the comparisons.

Usage

```
NormalyzerStatistics(experimentObj, logTrans = FALSE)
```

```
NormalyzerStatistics(experimentObj, logTrans = FALSE)
```

Arguments

experimentObj	Instance of SummarizedExperiment containing matrix and design information as column data
logTrans	Whether the input data should be log transformed

Value

nds Generated NormalyzerStatistics instance

Slots

annotMat	Matrix containing annotation information
dataMat	Matrix containing (normalized) expression data
filteredDataMat	Filtered matrix with low-count rows removed
designDf	Data frame containing design conditions
filteringContrast	Vector showing which entries are filtered (due to low count)
pairwiseCompsP	List with P-values for pairwise comparisons
pairwiseCompsFdr	List with FDR-values for pairwise comparisons
pairwiseCompsAve	List with average expression values
pairwiseCompsFold	List with log2 fold-change values for pairwise comparisons
contrasts	Spot for saving vector of last used contrasts
condCol	Column containing last used conditions
batchCol	Column containing last used batch conditions

Examples

```
data(example_stat_summarized_experiment)
nst <- NormalyzerStatistics(example_stat_summarized_experiment)
```

normMethods

*Perform normalizations on Normalyzer dataset***Description**

Perform normalizations on Normalyzer dataset

Usage

```
normMethods(
  nds,
  forceAll = FALSE,
  normalizeRetentionTime = TRUE,
  quiet = FALSE,
  rtStepSizeMinutes = 1,
  rtWindowMinCount = 100,
  rtWindowShifts = 1,
  rtWindowMergeMethod = "mean",
  noLogTransform = FALSE
)
```

Arguments

nds	Normalyzer dataset object.
forceAll	Force all methods to run despite not qualifying for thresholds.
normalizeRetentionTime	Perform retention time based normalization methods.
quiet	Prevent diagnostic output
rtStepSizeMinutes	Retention time normalization window size.
rtWindowMinCount	Minimum number of datapoints in each retention-time segment.
rtWindowShifts	Number of layered retention time normalized windows.
rtWindowMergeMethod	Merge approach for layered retention time windows.
noLogTransform	Per default NormalyzerDE performs a log-transformation on the input data. If not needed, specify this option

Value

Returns Normalyzer results object with performed analyzes assigned as attributes

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalyzerObject("job_name", example_summarized_experiment)
normResults <- normMethods(normObj)
```

```
performCyclicLoessNormalization
```

Cyclic Loess normalization

Description

Log2 transformed data is normalized by Loess method using the function "normalizeCyclicLoess". Further information is available for the function "normalizeCyclicLoess" in the Limma package.

Usage

```
performCyclicLoessNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix Target matrix to be normalized
noLogTransform Assumes no need for log transformation

Value

Normalized matrix

Examples

```
data(example_data_only_values_small)  
normMatrix <- performCyclicLoessNormalization(example_data_only_values)
```

```
performGlobalRLRNormalization
```

Global linear regression normalization

Description

Log2 transformed data is normalized by robust linear regression using the function "rlm" from the MASS package.

Usage

```
performGlobalRLRNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix Target matrix to be normalized
noLogTransform Assumes no need for log transformation

Value

Normalized matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- performGlobalRLRNormalization(example_data_only_values)
```

```
performNoNormalization
```

Do no normalization (For debugging purposes)

Description

Do no normalization (For debugging purposes)

Usage

```
performNoNormalization(rawMatrix)
```

Arguments

rawMatrix Target matrix to be normalized

Value

Normalized matrix

```
performNormalizations    Main function for executing normalizations
```

Description

Main function for executing normalizations

Usage

```
performNormalizations(
  nr,
  forceAll = FALSE,
  rtNorm = FALSE,
  rtStepSizeMinutes = 1,
  rtWindowMinCount = 100,
  rtWindowShifts = 1,
  rtWindowMergeMethod = "median",
  noLogTransform = FALSE,
  quiet = FALSE
)

## S4 method for signature 'NormalyzerResults'
performNormalizations(
  nr,
  forceAll = FALSE,
```

```

rtNorm = FALSE,
rtStepSizeMinutes = 1,
rtWindowMinCount = 100,
rtWindowShifts = 1,
rtWindowMergeMethod = "median",
noLogTransform = FALSE,
quiet = FALSE
)

```

Arguments

nr	Normalyzer results object.
forceAll	Ignore dataset size limits and run all normalizations (only meant for testing purposes)
rtNorm	Perform retention time based normalizations
rtStepSizeMinutes	Retention time normalization window size.
rtWindowMinCount	Minimum number of datapoints in each retention-time segment.
rtWindowShifts	Number of layered retention time normalized windows.
rtWindowMergeMethod	Merge approach for layered retention time windows.
noLogTransform	Prevent log-transforming input
quiet	Don't show regular output messages

Value

nr NormalyzerDE results object

performQuantileNormalization

Quantile normalization is performed by the function "normalize.quantiles" from the package preprocessCore.

Description

It makes the assumption that the data in different samples should originate from an identical distribution. It does this by generating a reference distribution and then scaling the other samples accordingly.

Usage

```
performQuantileNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

rawMatrix	Target matrix to be normalized
noLogTransform	Assumes no need for log transformation

Value

Normalized matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- performQuantileNormalization(example_data_only_values)
```

```
performSMADNormalization
```

Median absolute deviation normalization Normalization subtracts the median and divides the data by the median absolute deviation (MAD).

Description

Median absolute deviation normalization Normalization subtracts the median and divides the data by the median absolute deviation (MAD).

Usage

```
performSMADNormalization(rawMatrix, noLogTransform = FALSE)
```

Arguments

`rawMatrix` Target matrix to be normalized
`noLogTransform` Assumes no need for log transformation

Value

Normalized matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- performSMADNormalization(example_data_only_values)
```

```
performVSNNormalization
```

Log2 transformed data is normalized using the function "justvsnn" from the VSN package.

Description

The VSN (Variance Stabilizing Normalization) attempts to transform the data in such a way that the variance remains nearly constant over the intensity spectrum

Usage

```
performVSNNormalization(rawMatrix)
```

Arguments

rawMatrix Target matrix to be normalized

Value

Normalized matrix

Examples

```
data(example_data_only_values_small)
normMatrix <- performVSNNormalization(example_data_only_values)
```

plotBoxPlot	<i>Boxplots showing distribution of values after different normalizations</i>
-------------	---

Description

Boxplots showing distribution of values after different normalizations

Usage

```
plotBoxPlot(nr, currentLayout, pageno)
```

Arguments

nr Normalyzer results object.
currentLayout Layout used for document.
pageno Current page number.

Value

None

plotComparisonVenns	<i>If multiple comparisons - Show overlap in Venn diagrams</i>
---------------------	--

Description

If multiple comparisons - Show overlap in Venn diagrams

Usage

```
plotComparisonVenns(
  nst,
  jobName,
  currentLayout,
  pageno,
  sigThres = 0.1,
  sigThresType = "fdr",
  log2FoldThres = 0,
  maxContrasts = 4
)
```

Arguments

nst	NormalyzerDE statistics object.
jobName	Name of processing run.
currentLayout	Layout used for document.
pageno	Current page number.
sigThres	Cutoff value for significance threshold
sigThresType	Type of significance cutoff
log2FoldThres	Log2-fold based cutoff threshold
maxContrasts	Maximum contrasts to show pairwise comparisons for

Value

None

plotContrastPCA	<i>Show in a PCA plot what samples are compared in statistical contrast This is useful to understand what conditions are compared and for checking for outliers in the contrast</i>
-----------------	---

Description

Show in a PCA plot what samples are compared in statistical contrast This is useful to understand what conditions are compared and for checking for outliers in the contrast

Usage

```
plotContrastPCA(nst, jobName, currentLayout, pageno, pcs = c(1, 2))
```

Arguments

nst	NormalyzerDE statistics object.
jobName	Name of processing run.
currentLayout	Layout used for document.
pageno	Current page number.
pcs	Principal components to show.

Value

None

plotContrastPHists	<i>Takes an NormalyzerStatistics instance and generates and prints a p-value histogram for each onto the viewport</i>
--------------------	---

Description

Takes an NormalyzerStatistics instance and generates and prints a p-value histogram for each onto the viewport

Usage

```
plotContrastPHists(nst, jobName, currentLayout, pageno)
```

Arguments

nst	NormalyzerDE statistics object.
jobName	Name of processing run.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotCorrelation	<i>Visualize within-replicates correlations</i>
-----------------	---

Description

Visualize within-replicates correlations

Usage

```
plotCorrelation(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotCVvsIntensity	<i>Plots page displaying coefficient of variance (CV) against raw intensity for features across the performed normalizations</i>
-------------------	--

Description

Plots page displaying coefficient of variance (CV) against raw intensity for features across the performed normalizations

Usage

```
plotCVvsIntensity(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotDendrograms	<i>Visualize dendrogram grouping of samples</i>
-----------------	---

Description

Visualize dendrogram grouping of samples

Usage

```
plotDendrograms(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotDensity	<i>Density plots showing value distributions after normalizations</i>
-------------	---

Description

Density plots showing value distributions after normalizations

Usage

```
plotDensity(nr, currentLayout, pageno)
```

Arguments

nr	Normalizer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotFrontPage	<i>Generate first page in output report and write to viewport</i>
---------------	---

Description

Generate first page in output report and write to viewport

Usage

```
plotFrontPage(currentjob, currentFont)
```

Arguments

currentjob	Name of current run.
currentFont	Font used for output document.

Value

None

plotMA	<i>Produces a page containing expression vs. fold-change figures (MA plots) The visualized fold is between the first sample in each group and the average of the replicate to which that sample belongs</i>
--------	---

Description

Produces a page containing expression vs. fold-change figures (MA plots) The visualized fold is between the first sample in each group and the average of the replicate to which that sample belongs

Usage

```
plotMA(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotMDS	<i>MDS plots showing grouping of samples after normalizations</i>
---------	---

Description

MDS plots showing grouping of samples after normalizations

Usage

```
plotMDS(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotMeanSD	<i>Visualize standard deviation over (expression?) for different values</i>
------------	---

Description

Visualize standard deviation over (expression?) for different values

Usage

```
plotMeanSD(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotPHist	<i>Generate P-histograms for ANOVA calculated after each normalization</i>
-----------	--

Description

Generate P-histograms for ANOVA calculated after each normalization

Usage

```
plotPHist(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotQQ	<i>Produces page showing QQ-plots for the first sample for each normalization method. This plot can be used to assess whether the data follows a normal distribution.</i>
--------	---

Description

Produces page showing QQ-plots for the first sample for each normalization method. This plot can be used to assess whether the data follows a normal distribution.

Usage

```
plotQQ(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotReplicateVarAndStableVariables	<i>Write figures displaying pooled coefficient of variance, median absolute deviation and pooled estimate of variance percentage compared to log2-transformed and stable variables plot displaying CV of stable variables against pooled CV measure. The stable variables are calculated by an ANOVA comparison across sample conditions and selecting features with the least clear difference.</i>
------------------------------------	--

Description

Write figures displaying pooled coefficient of variance, median absolute deviation and pooled estimate of variance percentage compared to log2-transformed and stable variables plot displaying CV of stable variables against pooled CV measure. The stable variables are calculated by an ANOVA comparison across sample conditions and selecting features with the least clear difference.

Usage

```
plotReplicateVarAndStableVariables(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotReplicateVariance	<i>Generate normalization replicate variance summary by displaying CV (coefficient of variance), MAD (mean of intragroup median absolute deviation) and PEV (Pooled Estimate of Variance) as mean of intragroups</i>
-----------------------	--

Description

Generate normalization replicate variance summary by displaying CV (coefficient of variance), MAD (mean of intragroup median absolute deviation) and PEV (Pooled Estimate of Variance) as mean of intragroups

Usage

```
plotReplicateVariance(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotRLE	<i>Boxplots showing relative log expression after normalizations</i>
---------	--

Description

Boxplots showing relative log expression after normalizations

Usage

```
plotRLE(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotSampleMappingPage *Write page with sample mapping*

Description

Write page with sample mapping

Usage

```
plotSampleMappingPage(nr, currentFont, currentLayout, currentjob, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotSampleOutlierSummary
*Write page containing sample summary of intensities, missing values
and MDS plot to the viewport*

Description

Write page containing sample summary of intensities, missing values and MDS plot to the viewport

Usage

```
plotSampleOutlierSummary(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotScatter	<i>Produces page containing scatter plot plotting the first two samples from each dataset against each other for each normalization method</i>
-------------	--

Description

Produces page containing scatter plot plotting the first two samples from each dataset against each other for each normalization method

Usage

```
plotScatter(nr, currentLayout, pageno)
```

Arguments

nr	Normalyzer results object.
currentLayout	Layout used for document.
pageno	Current page number.

Value

None

plotSigScatter	<i>Takes an NormalyzerStatistics instance and generates and prints a volcano plot</i>
----------------	---

Description

Takes an NormalyzerStatistics instance and generates and prints a volcano plot

Usage

```
plotSigScatter(
  nst,
  jobName,
  currentLayout,
  pageno,
  type = "Volcano",
  sigThres = 0.1,
  sigThresType = "fdr",
  log2FoldThres = 0
)
```

Arguments

nst	NormalyzerDE statistics object.
jobName	Name of processing run.
currentLayout	Layout used for document.
pageno	Current page number.
type	Specify whether to plot 'Volcano' or 'MA'.
sigThres	FDR threshold for DE coloring.

Value

None

preprocessData	<i>Replace empty values (0 or empty field) with NA in input data</i>
----------------	--

Description

Replace empty values (0 or empty field) with NA in input data

Usage

```
preprocessData(dataMatrix, quiet = FALSE)
```

Arguments

dataMatrix	Matrix with raw data.
quiet	Don't show diagnostic messages

Value

Parsed rawdata where 0 values are replaced with NA

printMeta	<i>Print meta information for Normalyzer plot page ! Needs refactoring to reduce redundancy in code ! Needs double check of functionality</i>
-----------	---

Description

Print meta information for Normalyzer plot page ! Needs refactoring to reduce redundancy in code ! Needs double check of functionality

Usage

```
printMeta(plotname, pageno, jobname, currentLayout)
```

Arguments

plotname	Name of current plot.
pageno	Current page number.
jobname	Name of ongoing job.
currentLayout	Custom viewport layout.

Value

None

printPlots	<i>Generate PDF grid page filling it with provided list of plots</i>
------------	--

Description

Generate PDF grid page filling it with provided list of plots

Usage

```
printPlots(plotlist, plotname, pageno, jobname, currentLayout)
```

Arguments

plotlist	List of target plots to display.
plotname	List of names corresponding to the provided plot list.
pageno	Current page number.
jobname	Name of ongoing job.
currentLayout	Custom viewport layout.

Value

None

reduceTechnicalReplicates	<i>Remove technical replicates from data and design</i>
---------------------------	---

Description

Collapses sample values into their average. If only one value is present due to NA-values in other technical replicates, then that value is used.

Usage

```
reduceTechnicalReplicates(se, techRepColName, sampleColName)
```

Arguments

se	Summarized experiment where the assay contains the data to be reduced, and the colData the data frame
techRepColName	Technical replicates column name in colData
sampleColName	Sample names column name in colData

Details

Takes a SummarizedExperiment where the data is present as the assay and the colData contains the design conditions. In the design conditions there should be one column with the technical replicate groups and one column containing the sample names

Value

reducedSe Summarized experiment with reduced data

Examples

```
testData <- as.matrix(data.frame(
  c(1,1,1),
  c(1,2,1),
  c(7,7,7),
  c(7,9,7)))
colnames(testData) <- c("a1", "a2", "b1", "b2")
designDf <- data.frame(
  sample=c("a1", "a2", "b1", "b2"),
  techrep=c("a", "a", "b", "b"))
se <- SummarizedExperiment::SummarizedExperiment(
  assay=testData,
  colData=designDf
)
statObj <- reduceTechnicalReplicates(se, "techrep", "sample")
```

setupJobDir	<i>Create empty directory for run</i>
-------------	---------------------------------------

Description

Creates a directory at provided path named to the jobname.

Usage

```
setupJobDir(jobName, outputDir)
```

Arguments

jobName	Name of the run.
outputDir	Path to directory where to create the output directory.

Value

Path to newly created directory.

Examples

```
setupJobDir("job_name", "path/to/outdir")
```

setupPlotting	<i>Setup PDF report settings by initializing the color palette, format for the PDF report and the graphical device</i>
---------------	--

Description

Setup PDF report settings by initializing the color palette, format for the PDF report and the graphical device

Usage

```
setupPlotting(currentJob, jobDir, suffix)
```

Arguments

currentJob	Name of current run.
jobDir	Path to output directory for run.
suffix	Text to add to output filename.

Value

None

setupRawContrastObject	<i>Prepare SummarizedExperiment object for statistics data</i>
------------------------	--

Description

Prepare SummarizedExperiment object for statistics data

Usage

```
setupRawContrastObject(dataPath, designPath, sampleColName)
```

Arguments

dataPath	Path to raw data matrix
designPath	Path to design matrix
sampleColName	Name for column in design matrix containing sample names

Value

experimentObj Prepared instance of SummarizedExperiment

Examples

```
data_path <- system.file(package="NormalyzerDE", "extdata", "tiny_data.tsv")
design_path <- system.file(package="NormalyzerDE", "extdata", "tiny_design.tsv")
sumExpObj <- setupRawContrastObject(data_path, design_path, "sample")
```

setupRawDataObject	<i>Prepare SummarizedExperiment object for raw data to be normalized containing data, design and annotation information</i>
--------------------	---

Description

Prepare SummarizedExperiment object for raw data to be normalized containing data, design and annotation information

Usage

```
setupRawDataObject(
  dataPath,
  designPath,
  inputFormat = "default",
  zeroToNA = FALSE,
  sampleColName = "sample",
  groupColName = "group"
)
```

Arguments

dataPath	File path to data matrix.
designPath	File path to design matrix.
inputFormat	Type of matrix for data, can be either 'default', 'proteios', 'maxquantprot' or 'maxquantpep'
zeroToNA	If TRUE zeroes in the data is automatically converted to NA values
sampleColName	Column name for column containing sample names
groupColName	Column name for column containing condition levels

Value

experimentObj SummarizedExperiment object loaded with the data

Examples

```
data_path <- system.file(package="NormalyzerDE", "extdata", "tiny_data.tsv")
design_path <- system.file(package="NormalyzerDE", "extdata", "tiny_design.tsv")
df <- setupRawDataObject(data_path, design_path)
```

setupTestData	<i>Generate a random test dataset with features, sample values and retention times</i>
---------------	--

Description

Generate a random test dataset with features, sample values and retention times

Usage

```
setupTestData(nSamples, nFeatures, rtMin = 40, rtMax = 80, mean = 20, sd = 4)
```

Arguments

nSamples	Number of samples
nFeatures	Number of features
rtMin	Minimum retention time
rtMax	Maximum retention time
mean	Mean value for sample intensities
sd	Standard deviation for sample intensities

Value

Test dataset

Examples

```
df <- setupTestData(6, 20)
df <- setupTestData(6, 20, mean=15, sd=1)
```

validateSampleReplication	<i>Check whether all samples have replicates</i>
---------------------------	--

Description

Check whether all samples have replicates

Usage

```
validateSampleReplication(  
  dataMatrix,  
  groups,  
  requireReplicates = TRUE,  
  quiet = FALSE  
)
```

Arguments

dataMatrix	Prepared matrix containing expression data.
groups	Vector containing condition levels
requireReplicates	By default stops processing if not all samples have replicates

Value

None

verifyContrasts	<i>Check that a given contrast string is valid given a particular design matrix. Each level tested for in the contrast should be present in the condition column for the design matrix.</i>
-----------------	---

Description

Mainly meant to verify strings received during server usage.

Usage

```
verifyContrasts(designLevels, contrasts)
```

Arguments

designLevels	Vector containing condition levels present in design
contrasts	A string containing one or several (comma delimited) strings for which contrasts should be performed

Value

None

verifyDesignMatrix	<i>Verify that design matrix setup matches the data matrix</i>
--------------------	--

Description

Verify that design matrix setup matches the data matrix

Usage

```
verifyDesignMatrix(fullMatrix, designMatrix, sampleCol)
```

Arguments

fullMatrix	Dataframe with input data.
designMatrix	Dataframe with design setup.
sampleCol	Column in design matrix containing sample IDs.

Value

None

 verifyMultipleSamplesPresent

Check whether more than one sample is present

Description

Check whether more than one sample is present

Usage

```
verifyMultipleSamplesPresent(
  dataMatrix,
  groups,
  requireReplicates = TRUE,
  quiet = FALSE
)
```

Arguments

dataMatrix Prepared dataframe.

groups Vector containing condition levels

requireReplicates
 By default stops processing if not all samples have replicates

Value

None

 verifySummarizedExperiment

Verify that design matrix setup matches the data matrix

Description

Verify that design matrix setup matches the data matrix

Usage

```
verifySummarizedExperiment(summarizedExp, sampleCol)
```

Arguments

sampleCol Column in design matrix containing sample IDs.

fullMatrix Dataframe with input data.

designMatrix Dataframe with design setup.

Value

None

verifyValidNumbers	<i>Verify that input fields conform to the expected formats</i>
--------------------	---

Description

Verify that input fields conform to the expected formats

Usage

```
verifyValidNumbers(rawDataOnly, groups, noLogTransform = FALSE, quiet = FALSE)
```

Arguments

rawDataOnly	Dataframe with input data.
groups	Condition levels for comparisons.

Value

None

writeNormalizedDatasets	<i>Write normalization matrices to file</i>
-------------------------	---

Description

Outputs each of the normalized datasets to the specified directory.

Usage

```
writeNormalizedDatasets(  
  nr,  
  jobdir,  
  includePairwiseComparisons = FALSE,  
  includeCvCol = FALSE,  
  includeAnovaP = FALSE,  
  normSuffix = "-normalized.txt",  
  rawDataName = "submitted_rawdata.txt"  
)
```

Arguments

<code>nr</code>	Results object.
<code>jobdir</code>	Path to output directory.
<code>includePairwiseComparisons</code>	Include p-values for pairwise comparisons.
<code>includeCvCol</code>	Include CV column in output.
<code>includeAnovaP</code>	Include ANOVA p-value in output.
<code>normSuffix</code>	String used to name output together with normalization names.
<code>rawdataName</code>	Name of output raw data file.

Value

None

Examples

```
data(example_summarized_experiment)
normObj <- getVerifiedNormalizerObject("job_name", example_summarized_experiment)
normResults <- normMethods(normObj)
normResultsWithEval <- analyzeNormalizations(normResults)
outputDir <- tempdir()
writeNormalizedDatasets(normResultsWithEval, outputDir)
```

Index

* internal

calculateANOVAPValues, [4](#)
calculateAvgMadMem, [5](#)
calculateAvgReplicateVariation, [5](#)
calculateCorrSum, [7](#)
calculateFeatureCV, [7](#)
calculatePercentageAvgDiffInMat, [8](#)
calculateReplicateCV, [8](#)
calculateSummarizedCorrelationVector, [9](#)
createDirectory, [9](#)
detectSingleReplicate, [10](#)
detectSingletonSample, [10](#)
elapsedSecondsBetweenSystimes, [11](#)
example_data, [11](#)
example_data_only_values, [12](#)
example_design, [12](#)
example_stat_data, [12](#)
example_stat_summarized_experiment, [13](#)
example_summarized_experiment, [13](#)
example_wide_data, [13](#)
example_wide_design, [14](#)
filterLowRep, [14](#)
findLowlyVariableFeaturesCVs, [15](#)
getCombinedMatrix, [18](#)
getIndexList, [19](#)
getLowCountSampleFiltered, [19](#)
getReplicateSortedData, [20](#)
getRowNAFilterContrast, [20](#)
getWidenedRTRange, [24](#)
loadRawDataFromFile, [27](#)
NormalyzerDataset, [31](#)
performNoNormalization, [39](#)
performNormalizations, [39](#)
plotBoxPlot, [42](#)
plotComparisonVenns, [42](#)
plotContrastPCA, [43](#)
plotContrastPHists, [44](#)
plotCorrelation, [44](#)
plotCVvsIntensity, [45](#)
plotDendrograms, [45](#)
plotDensity, [46](#)
plotFrontPage, [46](#)
plotMA, [47](#)
plotMDS, [47](#)
plotMeanSD, [48](#)
plotPHist, [48](#)
plotQQ, [49](#)
plotReplicateVarAndStableVariables, [49](#)
plotReplicateVariance, [50](#)
plotRLE, [50](#)
plotSampleMappingPage, [51](#)
plotSampleOutlierSummary, [51](#)
plotScatter, [52](#)
plotSigScatter, [52](#)
preprocessData, [53](#)
printMeta, [53](#)
printPlots, [54](#)
setupPlotting, [56](#)
setupTestData, [58](#)
validateSampleReplication, [58](#)
verifyContrasts, [59](#)
verifyDesignMatrix, [59](#)
verifyMultipleSamplesPresent, [60](#)
verifySummarizedExperiment, [60](#)
verifyValidNumbers, [61](#)

analyzeNormalizations, [3](#)

calculateANOVAPValues, [4](#)
calculateAvgMadMem, [5](#)
calculateAvgReplicateVariation, [5](#)
calculateContrasts, [6](#)
calculateContrasts, NormalyzerStatistics-method (calculateContrasts), [6](#)
calculateCorrSum, [7](#)
calculateFeatureCV, [7](#)
calculatePercentageAvgDiffInMat, [8](#)
calculateReplicateCV, [8](#)
calculateSummarizedCorrelationVector, [9](#)
createDirectory, [9](#)

detectSingleReplicate, [10](#)

- detectSingleReplicate, NormalizerDataset-method
 - (detectSingleReplicate), 10
- detectSingletonSample, 10
- detectSingletonSample, NormalizerDataset-method
 - (detectSingletonSample), 10
- elapsedSecondsBetweenSystemtimes, 11
- example_data, 11
- example_data_only_values, 12
- example_design, 12
- example_stat_data, 12
- example_stat_summarized_experiment, 13
- example_summarized_experiment, 13
- example_wide_data, 13
- example_wide_design, 14
- filterLowRep, 14
- findLowlyVariableFeaturesCVs, 15
- generateAnnotatedMatrix, 15
- generatePlots, 16
- generateStatsReport, 17
- getCombinedMatrix, 18
- getIndexList, 19
- getLowCountSampleFiltered, 19
- getReplicateSortedData, 20
- getRowNAFilterContrast, 20
- getRTNormalizedMatrix, 21
- getSmoothedRTNormalizedMatrix, 22
- getVerifiedNormalizerObject, 23
- getWidenedRTRange, 24
- globalIntensityNormalization, 25
- loadData, 25
- loadDesign, 26
- loadRawDataFromFile, 27
- meanNormalization, 27
- medianNormalization, 28
- normalizer, 28
- NormalizerDataset, 31
- normalizerDE, 32
- NormalizerEvaluationResults, 34
- NormalizerResults, 35
- NormalizerStatistics, 36
- normMethods, 37
- performCyclicLoessNormalization, 38
- performGlobalRLRNormalization, 38
- performNoNormalization, 39
- performNormalizations, 39
- performNormalizations, NormalizerResults-method
 - (performNormalizations), 39
- performQuantileNormalization, 40
- performSMADNormalization, 41
- performVSNNormalization, 41
- plotBoxPlot, 42
- plotComparisonVenns, 42
- plotContrastPCA, 43
- plotContrastPHists, 44
- plotCorrelation, 44
- plotCVvsIntensity, 45
- plotDendrograms, 45
- plotDensity, 46
- plotFrontPage, 46
- plotMA, 47
- plotMDS, 47
- plotMeanSD, 48
- plotPHist, 48
- plotQQ, 49
- plotReplicateVarAndStableVariables, 49
- plotReplicateVariance, 50
- plotRLE, 50
- plotSampleMappingPage, 51
- plotSampleOutlierSummary, 51
- plotScatter, 52
- plotSigScatter, 52
- preprocessData, 53
- printMeta, 53
- printPlots, 54
- reduceTechnicalReplicates, 54
- setupJobDir, 55
- setupPlotting, 56
- setupRawContrastObject, 56
- setupRawDataObject, 57
- setupTestData, 58
- validateSampleReplication, 58
- verifyContrasts, 59
- verifyDesignMatrix, 59
- verifyMultipleSamplesPresent, 60
- verifySummarizedExperiment, 60
- verifyValidNumbers, 61
- writeNormalizedDatasets, 61