Package 'lfproQC'

September 30, 2024

Type Package

Title Quality Control for Label-Free Proteomics Expression Data

Version 1.3.0

Maintainer Kabilan S <kabilan151414@gmail.com>

Description Label-free bottom-up proteomics expression data is often affected by data heterogeneity and missing values. Normalization and missing value imputation are commonly used techniques to address these issues and make the dataset suitable for further downstream analysis. This package provides an optimal combination of normalization and imputation methods for the dataset. The package utilizes three normalization methods and three imputation methods.The statistical evaluation measures named pooled co-efficient of variance, pooled estimate of variance and pooled median absolute deviation are used for selecting the best combination of normalization and imputation method for the given dataset. The user can also visualize the results by using various plots available in this package. The user can also perform the differential expression analysis between two sample groups with the function included in this package. The chosen three normalization methods, three imputation methods and three evaluation measures were chosen for this study based on the research papers published by Välikangas et al. (2016) <doi:10.1093/bib/bbw095>, Jin et al. (2021) <doi:10.1038/s41598-021-81279-4> and Srivastava et al. (2023) <doi:10.2174/1574893618666230223150253>.

Imports VIM, dplyr, limma, matrixStats, pcaMethods, vsn, reshape2, laeken, ggplot2, Hmisc, reshape, stats, tidyr, magrittr, plotly, MASS, tidyselect

Suggests knitr, plyr, rmarkdown, testthat (>= 3.0.0), tibble

License GPL-3

URL https://github.com/kabilansbio/lfproQC

BugReports https://github.com/kabilansbio/lfproQC/issues

Encoding UTF-8

Depends R (>= 4.2)

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Kabilan S [aut, cre],
Dr Shashi Bhushan Lal [aut, ths],
Dr Sudhir Srivastava [aut, ths],
Dr Krishna Kumar Chaturvedi [ths],
Dr Yasin Jeshima K [ths],
Dr Ramasubramanian V [ths],
Dr Girish Kumar Jha [ctb]

Config/testthat/edition 3

VignetteBuilder knitr

Repository CRAN

Date/Publication 2024-09-30 07:20:02 UTC

Contents

`best_combination` *Best combination of normalization and imputation method*

Description

This function will provide the best combinations of normalization and imputation methods for the user given dataset based on the intragroup variation evaluation parameters called PCV, PEV, PMAD and NRMSE.

Usage

```
best_combination(data_input, groups, data_type, aggr_method)
```

Arguments

data_input	Label-free proteomics expression data as a dataframe
groups	Group information about the input data
data_type	A character string specifying the type of data being used. Use "Peptide" if your dataset contains peptide information, where the first column represents peptide IDs and the second column represents protein IDs. Use "Protein" if your dataset consists of protein data, where the first column represents protein IDs. The function will handle the data accordingly based on this parameter.
aggr_method	A character string specifying the method for aggregating peptide data to corresponding protein values. Use "sum" to aggregate by the sum of peptide intensities, "mean" to aggregate by the mean of peptide intensities, or "median" to aggregate by the median of peptide intensities. This parameter is only applicable when "data_type" is set to "Peptide".

Details

Label-free LC-MS proteomics expression data is often affected by heterogeneity and missing values. Normalization and missing value imputation are the commonly used techniques to solve these issues and make the dataset suitable for further downstream analysis. This function provides the best combination of normalization and imputation methods for the dataset, choosing from the three normalization methods (vsn, loess, and rlr) and three imputation methods (knn, lls, svd). The intra-group variation evaluation measures named pooled co-efficient of variance (PCV), pooled estimate of variance (PEV) and pooled median absolute deviation (PMAD) are used for selecting the best combination of normalization and imputation method for the given dataset. It will return the best combinations based on each evaluation parameters of PCV, PEV, and PMAD.

Along with this, the user can get all three normalized datasets, nine combinations of normalized and missing values imputed datasets, and the PCV, PEV, and PMAD result values. The user can also obtain the Normalized Root Mean Square Error (NRMSE) values, calculated by comparing the normalized and imputed dataset with the original dataset across all nine combinations. These NRMSE values provide insight into the accuracy of the imputation and normalization processes.

Value

This function gives the list which consist of following results.

'Best Combinations' The best combinations based on each PCV, PEV and PMAD for the given dataset.

'PCV Result' Values of groupwise PCV, overall PCV, PCV mean, PCV median and PCV standard deviation for all combinations.

'PEV Result' Values of groupwise PEV, overall PEV, PEV mean, PEV median and PEV standard deviation for all combinations.

'PMAD Result' Values of groupwise PMAD, overall PMAD, PMAD mean, PMAD median and PMAD standard deviation for all combinations.

'NRMSE Result' NRMSE values calculated for the normalized and imputed dataset to the original dataset.

'rollup_protein' The aggregated protein values for the peptide dataset are based on either the sum, mean, or median.

‘vsn_data‘ The ‘vsn‘ normalized dataset

‘loess_data‘ The ‘loess‘ normalized dataset

‘rlr_data‘ The ‘rlr‘ normalized dataset

‘vsn_knn_data‘ The dataset normalized by ‘vsn‘ method and missing values imputed by ‘knn‘ method.

‘vsn_lls_data‘ The dataset normalized by ‘vsn‘ method and missing values imputed by ‘lls‘ method.

‘vsn_svd_data‘ The dataset normalized by ‘vsn‘ method and missing values imputed by ‘svd‘ method.

‘loess_knn_data‘ The dataset normalized by ‘loess‘ method and missing values imputed by ‘knn‘ method.

‘loess_lls_data‘ The dataset normalized by ‘loess‘ method and missing values imputed by ‘lls‘ method.

‘loess_svd_data‘ The dataset normalized by ‘loess‘ method and missing values imputed by ‘svd‘ method.

‘rlr_knn_data‘ The dataset normalized by ‘rlr‘ method and missing values imputed by ‘knn‘ method.

‘rlr_lls_data‘ The dataset normalized by ‘rlr‘ method and missing values imputed by ‘lls‘ method.

‘rlr_svd_data‘ The dataset normalized by ‘rlr‘ method and missing values imputed by ‘svd‘ method.

Author(s)

Dr Sudhir Srivastava ("Sudhir.Srivastava@icar.gov.in")

Kabilan S ("kabilan151414@gmail.com")

Examples

```
result <- best_combination(yeast_data, yeast_groups, data_type = "Protein")
result$`Best combinations`
result$`PCV Result`
result$`PMAD Result`
result$`rlr_knn_data`
```

`Boxplot_data` *Creating Boxplot for a dataset*

Description

The box and whiskers plot displays the distribution of a continuous variable. It visualises five summary statistics (the median, two hinges and two whiskers), and all "outlying" points individually. The ‘ggplot2‘ package is used here for creating the boxplot.

Usage

```
Boxplot_data(data)
```

Arguments

data	Proteomics expression dataset (original or normalized dataset)

Details

This can also be used for comparing the original dataset with the normalized dataset.

Value

Interactive box and whiskers plot

See Also

'geom_boxplot()'

Examples

```
Boxplot_data(yeast_data)
Boxplot_data(rlr_knn_yeast_data)
```

Corrplot_data *Creating Correlation matrix plot for a dataset*

Description

A graphical display of a correlation matrix.

Usage

```
Corrplot_data(data)
```

Arguments

data	Proteomics expression dataset (original or normalized dataset) along with the protein information

Details

This can also be used for comparing the original dataset with the normalized dataset.

Value

Interactive corrleation matrix plot

Examples

```
Corrplot_data(yeast_data)
Corrplot_data(rlr_knn_yeast_data)
```

`Densityplot_data` *Creating Density plot for a dataset*

Description

Computes and draws kernel density estimate, which is a smoothed version of the histogram. This is a useful alternative to the histogram for continuous data that comes from an underlying smooth distribution. The 'ggplot2' package is used here for creating the boxplot.

Usage

```
Densityplot_data(data)
```

Arguments

`data`	Proteomics expression dataset (original or normalized dataset) along with the protein information

Details

This can also be used for comparing the original dataset with the normalized dataset.

Value

Interactive column-wise density plot

See Also

'geom_density()'

Examples

```
Densityplot_data(yeast_data)
Densityplot_data(rlr_knn_yeast_data)
```

`MAplot_DE_fn` *Find out the Up and Down regulated proteins from MA plot*

Description

MA plot is used for visualizing the differentially expressed proteins by plotting the log mean intensity data in x axis and log fold change values in y axis.

This function can be used for visualizing the up regulated, down regulated, and non-significant proteins along with their information.

Usage

```
MAplot_DE_fn(top_table, x1 = NULL, x2 = NULL, p = NULL)
```

Arguments

top_table	Top table information
x1	Cut-off limit for down-regulated proteins
x2	Cut-off limit for up-regulated proteins
p	Cut-off limit for p-values

Value

'Result' Top table along with up, down, significant and non-significant protein information.

'MA plot' Interacive MA plot with the details of up and down regulated proteins

'Up-regulated' Up-regulated protein information

'Down-regulated' Down-regulated protein information

'Non-significant' Non-significant protein information

Examples

```
result <- MAplot_DE_fn(yeast_top_table, -1, 1, 0.05)
result$`MA Plot`
result$`Result`
result$`Up-regulated`
result$`Down-regulated`
result$`Non-significant`
```

MDSplot_data *Creating MDS plot for a dataset*

Description

Multi-dimensional scaling (MDS) plots showing a 2-dimensional projection of distances between the dataset samples.

Usage

```
MDSplot_data(data)
```

Arguments

data	Normalized and imputed Proteomics expression dataset along with the protein information

Value

MDS plot

See Also

'mdsPlot'

Examples

```
MDSplot_data(rlr_knn_yeast_data)
```

`QQplot_data` *Creating QQ-Plot for a dataset*

Description

A Q–Q plot (quantile-quantile plot) is a plot of the quantiles of two distributions against each other, or a plot based on estimates of the quantiles. The normality of the data can be understand by this plot.

Usage

```
QQplot_data(data)
```

Arguments

`data`	Proteomics expression dataset (original or normalized dataset)

Details

This can be used for comparing the original dataset with the normalized dataset.

Value

Interactive column-wise QQ-plot

Examples

```
qqplot <- QQplot_data(rlr_knn_yeast_data)
```

rlr_knn_yeast_data *Normalized and imputed complete yeast lysate - UPS1 benchmark dataset*

Description

This is the groupwise normalized and missing values imputed dataset of the complete yeast lysate - UPS1 benchmark dataset. Normalization has been done by RLR normalization method and missing values imputation has been done by KNN imputation method.

Usage

```
rlr_knn_yeast_data
```

Format

A data frame with 835 rows and 7 variables:

Majority protein IDs Protein ID information

A1 1st sample group, 1st technical replicate

A2 1st sample group, 2nd technical replicate

A3 1st sample group, 3rd technical replicate

B1 2nd sample group, 1st technical replicate

B2 2nd sample group, 2nd technical replicate

B3 2nd sample group, 3rd technical replicate

top_table_fn *Creating the top table*

Description

Top table can be used for identifying the pairwise differential abundance analysis of proteins in the dataset.

Usage

```
top_table_fn(data, groups, ch_gr1, ch_gr2)
```

Arguments

data	Normalized and missing values imputed expression dataset containing protein information
groups	Group information about the input data
ch_gr1	Group number of the dataset for pairwise comparison with the another group
ch_gr2	Group number of the dataset to be compared with the chosen group

Value

Top table consists of following values

'logFC' - Log fold change values,

'AveExpr' - Average intensity values,

't'- t-statistic values,

'P.Value' - P-values,

'adj.P.Val' - Adjusted P-values,

'B'- B-statistic values

See Also

'limma::topTable'

Examples

```
top_table <- top_table_fn(rlr_knn_yeast_data, yeast_groups, 2, 1)
top_table
```

`volcanoplot_DE_fn` *Find out the Up and Down regulated proteins from volcano plot*

Description

Volcano plot is used for visualizing the differentially expressed proteins by plotting the log fold change values in x axis and (-log10 p-values) in y axis.

This function can be used for visualizing the up regulated, down regulated, and non-significant proteins along with their information.

Usage

```
volcanoplot_DE_fn(top_table, x1 = NULL, x2 = NULL, p = NULL)
```

Arguments

`top_table`	Top table information
`x1`	Cut-off limit for down-regulated proteins
`x2`	Cut-off limit for up-regulated proteins
`p`	Cut-off limit for p-values

Value

'Result' Top table along with up, down, significant and non-significant protein information.

'Volcano plot' Interactive MA plot with the details of up and down regulated proteins

'Up-regulated' Up-regulated protein information

'Down-regulated' Down-regulated protein information

'Non-significant' Non-significant protein information

Examples

```
result <- volcanoplot_DE_fn(yeast_top_table, -1, 1, 0.05)
result$`Volcano Plot`
result$`Result`
result$`Up-regulated`
result$`Down-regulated`
result$`Non-significant`
```

---

yeast_data *Yeast lysate - UPS1 benchmark dataset*

Description

This dataset was given by Ramus et al., (2016). It is based on a highly complex sample (yeast lysate) spiked with different spiked amounts of the UPS1 standard mixture of 48 recombinant proteins. The original dataset contains 2644 rows of proteins and 2 groups of samples with three replicates each. From the original dataset, only the pairwise comparison of 50vs0.5 has taken here. This dataset contains 874 rows of proteins which includes 48 spiked-in ups proteins in the yeast.

Usage

```
yeast_data
```

Format

A data frame with 874 rows and 7 variables:

Majority protein IDs Protein ID information

A1 1st condition, 1st technical replicate

A2 1st condition, 2nd technical replicate

A3 1st condition, 3rd technical replicate

B1 2nd condition, 1st technical replicate

B2 2nd condition, 2nd technical replicate

B3 2nd condition, 3rd technical replicate

Details

This standard proteomic dataset is suitable for benchmarking and comparing software for label-free quantification. And can also be applied to the evaluation of post-processing steps such as normalization, imputation of missing values, and statistical methods.

Source

doi:10.1016/j.dib.2015.11.063

`yeast_groups` *Group information of Yeast lysate - UPS1 benchmark dataset*

Description

The standard benchmark yeast lysate - UPS1 dataset contains the variables of 6 columns, 2 conditions and 3 technical replicates.

Usage

```
yeast_groups
```

Format

A data frame with 6 rows and 2 variables:

Samples Technical replicate details - Two kinds of technical replicates (A1, A2, A3) and (B1, B2, B3)

Groups Condition details - Two conditions (GrA) and (GrB)

Details

The above information is required for further analysis of the dataset.

Source

doi:10.1016/j.dib.2015.11.063

`yeast_top_table` *Top table for Yeast Lysate - UPS1 dataset*

Description

Top table can be used for identifying the differential abundance analysis of proteins in the dataset.

Usage

```
yeast_top_table
```

Format

A data frame with 835 rows and 6 variables along with 'Fasta headers':

logFC Log fold change values

AveExpr Average intensity values

t t-statistic values

P.Value P-values

adj.P.Val Adjusted P-values

B B-statistic values

Index