Package 'POMA'

December 24, 2024

Title Tools for Omics Data Analysis

Version 1.17.6

Description The POMA package offers a comprehensive toolkit designed for omics data analysis, streamlining the process from initial visualization to final statistical analysis.
Its primary goal is to simplify and unify the various steps involved in omics data processing, making it more accessible and manageable within a single, intuitive R package.
Emphasizing on reproducibility and user-friendliness, POMA leverages the standardized SummarizedExperiment class from Bioconductor, ensuring seamless integration and compatibility
with a wide array of Bioconductor tools. This approach guarantees maximum flexibility and replicability, making POMA an essential asset for researchers handling omics datasets.
See https://github.com/pcastellanoescuder/POMAShiny. Paper: Castellano-Escuder et al. (2021) <doi:10.1371/journal.pcbi.1009148> for more details.

License GPL-3

Encoding UTF-8

LazyData true

biocViews BatchEffect, Classification, Clustering, DecisionTree, DimensionReduction, MultidimensionalScaling, Normalization, Preprocessing, PrincipalComponent, Regression, RNASeq, Software, StatisticalMethod, Visualization

Imports broom, caret, ComplexHeatmap, dbscan, dplyr, DESeq2, fgsea, FSA, ggcorrplot, ggplot2, ggrepel, glmnet, grid, impute, janitor, limma, lme4, magrittr, MASS, mixOmics, multcomp, msigdbr, purrr, randomForest, RankProd (>= 3.14), rlang, SummarizedExperiment, sva, tibble, tidyr, utils, uwot, vegan

Suggests BiocStyle, covr, ggraph, ggtext, knitr, patchwork, plotly, tidyverse, testthat (>= 2.3.2)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 4.0)

VignetteBuilder knitr

URL https://github.com/pcastellanoescuder/POMA

BugReports https://github.com/pcastellanoescuder/POMA/issues

git_url https://git.bioconductor.org/packages/POMA

git_branch devel

git_last_commit 09bb015

git_last_commit_date 2024-11-26

Repository Bioconductor 3.21

Date/Publication 2024-12-23

Author Pol Castellano-Escuder [aut, cre] (ORCID: <https://orcid.org/0000-0001-6466-877X>)

Maintainer Pol Castellano-Escuder <polcaes@gmail.com>

Contents

`box_cox_transformation` *Box-Cox Transformation*

Description

Compute Box-Cox normalization.

Usage

```
box_cox_transformation(data)
```

Arguments

`data`	A single variable.

`cor_pmat` *Correlation P-Values*

Description

Compute correlation p-values.

Usage

```
cor_pmat(x, method)
```

Arguments

`x`	A data matrix.
`method`	Character indicating which correlation coefficient has to be computed. Options are "pearson" (default), "kendall" and "spearman".

detect_decimals *Detect decimals*

Description

Detect decimal variables.

Usage

```
detect_decimals(data)
```

Arguments

data	A data matrix (samples in rows).

flattenCorrMatrix *Flatten Correlation Matrix*

Description

Flatten Correlation Matrix

Usage

```
flattenCorrMatrix(cormat, pmat = NULL)
```

Arguments

cormat	Output from cor.
pmat	Output from cor_pmat.

PomaBatch *Batch Correction*

Description

PomaBatch performs batch correction on a SummarizedExperiment object given a batch factor variable.

Usage

```
PomaBatch(data, batch, mod = NULL)
```

Arguments

data	A SummarizedExperiment object.
batch	Character. The name of the column in colData that contains the batch information.
mod	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates).

Value

A SummarizedExperiment object with batch-corrected data.

Author(s)

Pol Castellano-Escuder

References

Leek JT, Johnson WE, Parker HS, Fertig EJ, Jaffe AE, Zhang Y, Storey JD, Torres LC (2023). sva: Surrogate Variable Analysis. doi:10.18129/B9.bioc.sva https://doi.org/10.18129/B9.bioc.sva

Examples

```
# Output is a batch corrected SummarizedExperiment object
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

data %>%
  PomaBatch(batch = "gender")
```

PomaBoxplots *Boxplots and Violin Plots*

Description

PomaBoxplots generates boxplots and violin plots for samples and features. This function can be used for data exploration (e.g., comparison between pre and post normalized datasets).

Usage

```
PomaBoxplots(
  data,
  x = "samples",
  violin = FALSE,
  outcome = NULL,
  feature_name = NULL,
  theme_params = list(legend_title = FALSE, axis_x_rotate = TRUE)
)
```

Arguments

data	A SummarizedExperiment object.
x	Character. Options are "samples" (to visualize sample boxplots) and "features" (to visualize feature boxplots). Default is "samples".
violin	Logical. Indicates if violin plots should be displayed instead of boxplots. Default is FALSE.
outcome	Character. Indicates the name of the colData column to be used as the outcome factor. Default is NULL (first factor variable in colData).
feature_name	Character vector. Indicates the feature/s to display. Default is NULL (all features will be displayed).
theme_params	List. Indicates theme_poma parameters.

Value

A ggplot object.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaNorm()

# Sample boxplots
data %>%
  PomaBoxplots(x = "samples",
               violin = FALSE,
               outcome = NULL,
               feature_name = NULL,
               theme_params = list(axistext = "y")) # If too many samples

# Sample boxplots with covariate as outcome
data %>%
  PomaBoxplots(x = "samples",
               violin = FALSE,
               outcome = "gender", # change outcome
               feature_name = NULL,
               theme_params = list(axistext = "y")) # If too many samples

# Sample violin plots
data %>%
  PomaBoxplots(x = "samples",
               violin = TRUE,
               outcome = NULL,
               feature_name = NULL,
               theme_params = list(axistext = "y")) # If too many samples
```

```
# All feature boxplots
data %>%
  PomaBoxplots(x = "features",
               theme_params = list(axis_x_rotate = TRUE))

# Specific feature boxplots
data %>%
  PomaBoxplots(x = "features",
               feature_name = c("ornithine", "orotate"))

# Specific feature violin plots
data %>%
  PomaBoxplots(x = "features",
               violin = TRUE,
               feature_name = c("ornithine", "orotate"))
```

PomaClust *Cluster Analysis*

Description

PomaClust performs a k-means clustering and plots the results in a classical multidimensional scaling (MDS) plot.

Usage

```
PomaClust(
  data,
  method = "euclidean",
  k = NA,
  k_max = floor(min(dim(data))/2),
  show_clusters = TRUE,
  labels = FALSE
)
```

Arguments

data	A SummarizedExperiment object.
method	Character. Indicates the distance method to perform MDS. Options are "euclidean", "maximum", "manhattan", "canberra" and "minkowski". See ?dist().
k	Numeric. Indicates the number of clusters (default is NA). The optimal number of clusters will be used by default.
k_max	Numeric. Indicates the number of clusters among which the optimal k will be selected.
show_clusters	Logical. Indicates if clusters should be plotted or not.
labels	Logical. Indicates if sample names should be plotted or not.

Value

A `list` with results including plots and tables.

Author(s)

Pol Castellano-Escuder

Examples

```
## Output is a list with objects `mds_coordinates` (tibble), `mds_plot` (ggplot2 object), `optimal_clusters_number
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

data %>%
  PomaClust(method = "euclidean",
            k = NA,
            k_max = floor(min(dim(data))/2),
            show_clusters = TRUE,
            labels = FALSE)
```

`PomaCorr` *Correlation Analysis*

Description

`PomaCorr` computes all pairwise correlations in the data and generates a correlation plot.

Usage

```
PomaCorr(
  data,
  method = "pearson",
  cluster = TRUE,
  corrplot_shape = "square",
  sig_level = 1
)
```

Arguments

`data`	A `SummarizedExperiment` object.
`method`	Character. Indicates which correlation coefficient has to be computed. Options are "pearson" (default), "kendall", and "spearman".
`cluster`	Logical. Indicates whether the correlation plot will be ordered using the `hclust` function. Default is TRUE.
`corrplot_shape`	Character. Indicates the visualization method of the correlation plot to be used. Allowed values are "square" (default) and "circle".

sig_level	Numeric. Indicates the significance level. If the correlation p-value exceeds this threshold, the corresponding correlation coefficient is considered insignificant, and that pair will be hidden in the correlation plot. The default is 1, meaning all correlations are included in the plot. For datasets with more than 500 features, this threshold is ignored, and all pairwise correlations are displayed in the plot.

Value

A list with the results.

Author(s)

Pol Castellano-Escuder

Examples

```
## Output is a list with objects `correlations` (tibble) and `corrplot` (ggplot2 object)
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

data %>%
  PomaCorr(method = "pearson")
```

PomaCreateObject *Create a* SummarizedExperiment *Object*

Description

PomaCreateObject creates a SummarizedExperiment object from data frames.

Usage

```
PomaCreateObject(metadata = NULL, features = NULL, factor_levels = 10)
```

Arguments

metadata	Data frame. Metadata variables structured in columns. Sample ID must be the first column.
features	Matrix of features. Each feature is a column.
factor_levels	Numeric. Integer variables with less levels than indicated by this parameter will be treated as factors.

Value

A SummarizedExperiment object.

Author(s)

Pol Castellano-Escuder

References

Morgan M, Obenchain V, Hester J, Pagès H (2021). SummarizedExperiment: SummarizedExperiment container. R package version 1.24.0, https://bioconductor.org/packages/SummarizedExperiment.

Examples

```
data(iris)

# Create metadata: Data frame with sample names and a group factor
metadata <- data.frame(sample_id = paste0("sample_", 1:150), group = iris$Species)

# Create features: `p` column data frame with features
features <- iris[, 1:4]

# Create a `SummarizedExperiment` object with `POMA`
object <- PomaCreateObject(metadata = metadata,
                           features = features)
```

`PomaDensity` *Density Plots*

Description

`PomaDensity` generates a density plot for samples and features. This function can be used for data exploration (e.g., comparison between pre and post normalized datasets).

Usage

```
PomaDensity(
  data,
  x = "samples",
  outcome = NULL,
  feature_name = NULL,
  theme_params = list(legend_title = FALSE)
)
```

Arguments

`data`	A `SummarizedExperiment` object.
`x`	Character. Options are "samples" (to visualize sample density plots) and "features" (to visualize feature density plots). Default is "samples".
`outcome`	Character. Indicates the name of the `colData` column to be used as the outcome factor. Default is NULL (first factor variable in `colData`).
`feature_name`	Character vector. Indicates the feature/s to display. Default is NULL (all features will be displayed).
`theme_params`	List. Indicates `theme_poma` parameters.

Value

A ggplot object.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaNorm()

# Sample density plots
data %>%
  PomaDensity(x = "samples",
              outcome = NULL)

# Sample density plots with covariate as outcome
data %>%
  PomaDensity(x = "samples",
              outcome = "gender") # change outcome

# All feature density plots
data %>%
  PomaDensity(x = "features",
              theme_params = list(legend_position = "none"))

# Specific feature density plots
data %>%
  PomaDensity(x = "features",
              feature_name = c("ornithine", "orotate"))
```

PomaDESeq *Differential Expression Analysis Based on the Negative Binomial Distribution*

Description

PomaDESeq estimates variance-mean dependence in count data from high-throughput sequencing assays and test for differential expression based on a model using the negative binomial distribution.

Usage

```
PomaDESeq(data, contrast = NULL, outcome = NULL, covs = NULL, adjust = "fdr")
```

Arguments

data	A SummarizedExperiment object.
contrast	Character. Indicates the comparison. For example, "Group1-Group2" or "control-intervention".
outcome	Character. Indicates the name of the colData column to be used as the outcome factor. Default is NULL (first factor variable in colData).
covs	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates). If not NULL, a limma model will be fitted using the specified covariates. Note: The order of the covariates is important and should be listed in increasing order of importance in the experimental design.
adjust	Character. Indicates the multiple comparisons correction method. Options are: "fdr", "holm", "hochberg", "hommel", "bonferroni", "BH" and "BY".

Value

A tibble with the results.

Author(s)

Pol Castellano-Escuder

References

Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550 (2014)

Examples

```
#library("airway")
#data("airway")
#se <- airway
#
## Classic DESeq2
#DESeq_results <- se %>%
#  PomaDESeq(contrast = NULL,
#            outcome = "dex",
#            covs = NULL,
#            adjust = "fdr")
#
#DESeq_results %>%
#  dplyr::slice(1:10)
#
### Volcano plot
#DESeq_results %>%
#  dplyr::select(feature, log2FC, pvalue) %>%
#  PomaVolcano(labels = TRUE)
#
### Boxplot of top features
```

```
#se %>%
#  PomaBoxplots(x = "features",
#            outcome = "cell", # factorial variable to group by (e.g., treatment, sex, etc)
#               feature_name = DESeq_results$feature[1:10])
#
### Heatmap of top features
#se[rownames(se) %in% DESeq_results$feature[1:10]] %>%
#  PomaHeatmap(covs = c("cell", "dex"), # covariates to plot (e.g., treatment, sex, etc)
#              feature_names = TRUE)
#
## DESeq2 with covariates
#DESeq_results <- se %>%
#  PomaDESeq(contrast = NULL,
#            outcome = "dex",
#            covs = "cell",
#            adjust = "fdr")
#
#DESeq_results %>%
#  dplyr::slice(1:10)
#
### Volcano plot
#DESeq_results %>%
#  dplyr::select(feature, log2FC, adj_pvalue) %>%
#  PomaVolcano(labels = TRUE, y_label = "-log10 (Adjusted P-value)")
#
### Boxplot of top features
#se %>%
#  PomaBoxplots(x = "features",
#            outcome = "dex", # factorial variable to group by (e.g., treatment, sex, etc)
#               feature_name = DESeq_results$feature[1:10])
#
### Heatmap of top features
#se[rownames(se) %in% DESeq_results$feature[1:10]] %>%
#  PomaHeatmap(covs = c("cell", "dex"), # covariates to plot (e.g., treatment, sex, etc)
#              feature_names = TRUE)
#
## DESeq2 with covariates and batch
#DESeq_results <- se %>%
#  PomaDESeq(contrast = NULL,
#            outcome = "dex",
#            covs = c("batch", "cell"),
#            adjust = "fdr")
#
#DESeq_results %>%
#  dplyr::slice(1:10)
#
### Volcano plot
#DESeq_results %>%
#  dplyr::select(feature, log2FC, adj_pvalue) %>%
#  PomaVolcano(labels = TRUE, y_label = "-log10 (Adjusted P-value)")
#
### Boxplot of top features
#se %>%
```

```
#  PomaBoxplots(x = "features",
#            outcome = "cell", # factorial variable to group by (e.g., treatment, sex, etc)
#               feature_name = DESeq_results$feature[1:10])
#
### Heatmap of top features
#se[rownames(se) %in% DESeq_results$feature[1:10]] %>%
#  PomaHeatmap(covs = c("cell", "dex"), # covariates to plot (e.g., treatment, sex, etc)
#              feature_names = TRUE)
```

PomaEnrichment *Enrichment Analysis*

Description

PomaEnrichment performs enrichment analysis on a set of query gene symbols using specified methods and gene set collections. It allows for the analysis of over-representation (ORA) or gene set enrichment (GSEA) in various model organisms.

Usage

```
PomaEnrichment(
  genes,
  method = "ora",
  organism = "Homo sapiens",
  collection = "C5",
  universe = NULL,
  rank = NULL,
  pval_cutoff = 0.05,
  fdr_cutoff = 0.1,
  min_size = 2,
  max_size = if (method == "gsea") {
     length(genes) - 1
 } else {
     NULL
 },
  max_genes = 10
)
```

Arguments

genes	Character vector. Set of query gene symbols.
method	Character. Enrichment method. Options are: 'ora' (simple over-representation analysis based on hypergeometric test) and 'gsea' (gene set enrichment analysis on a ranked list of genes).
organism	Character. Indicates the model organism name. Default is 'Homo sapiens'. Other options are: 'Anolis carolinensis', 'Bos taurus', 'Caenorhabditis elegans',

	'Canis lupus familiaris', 'Danio rerio', 'Drosophila melanogaster', 'Equus caballus', 'Felis catus', 'Gallus gallus', 'Macaca mulatta', 'Monodelphis domestica', 'Mus musculus', 'Ornithorhynchus anatinus', 'Pan troglodytes', 'Rattus norvegicus', 'Saccharomyces cerevisiae', 'Schizosaccharomyces pombe 972h-', 'Sus scrofa', 'Xenopus tropicalis'. See `msigdbr::msigdbr_show_species()`.
`collection`	Character. Indicates the gene set collection. Default is 'C5' (Gene Ontology gene sets). Other options are: 'C1' (positional gene sets), 'C2' (curated gene sets), 'C3' (regulatory target gene sets), 'C4' (computational gene sets), 'C6' (oncogenic signature gene sets), 'C7' (immunologic signature gene sets), 'C8' (cell type signature gene sets), 'H' (Hallmark gene sets). See `msigdbr::msigdbr_collections()`.
`universe`	Character vector. A universe from which 'genes' were selected.
`rank`	Numeric vector. Ranking factor to sort genes for GSEA (e.g., logFC, -log10(p-value), etc).
`pval_cutoff`	Numeric. Raw p-value cutoff on enrichment tests to report.
`fdr_cutoff`	Numeric. Adjusted p-value cutoff on enrichment tests to report.
`min_size`	Numeric. Minimal size of a gene set to test. All pathways below the threshold are excluded.
`max_size`	Numeric. Maximal size of a gene set to test. All pathways above the threshold are excluded.
`max_genes`	Numeric. The number of genes to retain from the `overlap_genes` or `leading_edge` columns. If `max_genes` is greater than the number of genes available, all genes are retained.

Value

A `tibble` with the enriched gene sets.

Author(s)

Pol Castellano-Escuder

Examples

```
# Example genes
genes <- c("BRCA1", "TP53", "EGFR", "MYC", "PTEN")

# Perform ORA on Gene Ontology (C5) gene sets for Homo sapiens
PomaEnrichment(
  genes = genes,
  method = "ora",
  organism = "Homo sapiens",
  collection = "C5",
  pval_cutoff = 0.05,
  fdr_cutoff = 0.1,
  min_size = 10,
  max_size = 500)

# Example genes with ranking factors (e.g., logFC values)
```

```
genes <- c("Actb", "Gapdh", "Cdkn1a", "Cd44", "Pten")
rank <- c(2.5, -1.8, 3.1, -2.2, 1.7)

# Perform GSEA on Hallmark (H) gene sets for Mus musculus
PomaEnrichment(
  genes = genes,
  method = "gsea",
  organism = "Mus musculus",
  collection = "H",
  rank = rank,
  pval_cutoff = 0.05,
  fdr_cutoff = 0.25,
  min_size = 15,
  max_size = 500)
```

PomaHeatmap *Heatmap Plot*

Description

PomaHeatmap generates a heatmap.

Usage

```
PomaHeatmap(
  data,
  covs = NULL,
  sample_names = TRUE,
  feature_names = FALSE,
  show_legend = TRUE
)
```

Arguments

data	A SummarizedExperiment object.
covs	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates).
sample_names	Logical. Indicates if sample names should be displayed or not. Default is TRUE.
feature_names	Logical. Indicates if feature names should be displayed or not. Default is FALSE.
show_legend	Logical. Indicates if legend should be displayed or not. Default is TRUE.

Value

A ggplot object.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaNorm()

# Basic heatmap
data %>%
  PomaHeatmap()

# Heatmap with one covariate
data %>%
  PomaHeatmap(covs = "factors")

# Heatmap with two covariates
data %>%
  PomaHeatmap(covs = c("factors", "smoking_condition"))
```

PomaImpute *Impute Missing Values*

Description

PomaImpute performs missing value imputation on a dataset using various imputation methods.

Usage

```
PomaImpute(
  data,
  zeros_as_na = FALSE,
  remove_na = TRUE,
  cutoff = 20,
  group_by = NULL,
  method = "knn"
)
```

Arguments

data	A SummarizedExperiment object.
zeros_as_na	Logical. Indicates if the zeros in the data are missing values. Default is FALSE.
remove_na	Logical. Indicates if features with a percentage of missing values over the cutoff parameter should be removed. Default is TRUE.
cutoff	Numeric. Percentage of missing values allowed in each feature.

group_by — Character. Indicates the name of the colData column to be used as the grouping factor. Default is NULL (no grouping). This variable specifies the grouping criteria for evaluating missing values. Missing values will be assessed within each group defined by this variable, and features will be removed only if all groups exceed the allowed threshold of missing values.

method — Character. The imputation method to use. Options include "none" (no imputation, replace missing values by zeros), "half_min" (replace missing values with half of the minimum value), "median" (replace missing values with the median), "mean" (replace missing values with the mean), "min" (replace missing values with the minimum value), "knn" (replace missing values using k-nearest neighbors imputation), and "random_forest" (replace missing values using random forest imputation).

Value

A SummarizedExperiment object without missing values.

Author(s)

Pol Castellano-Escuder

References

Armitage, E. G., Godzien, J., Alonso-Herranz, V., López-Gonzálvez, Á., & Barbas, C. (2015). Missing value imputation strategies for metabolomics data. Electrophoresis, 36(24), 3050-3060.

Examples

```
# Output is a imputed SummarizedExperiment object
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

# No sample normalization
data %>%
  PomaImpute(zeros_as_na = FALSE,
             remove_na = TRUE,
             cutoff = 20,
             group_by = NULL,
             method = "knn")
```

PomaLasso — *Lasso, Ridge, and Elasticnet Regularized Generalized Linear Models for Binary Outcomes*

Description

PomaLasso performs LASSO, Ridge, and Elasticnet regression for feature selection and prediction purposes for binary outcomes.

Usage

```
PomaLasso(
  data,
  alpha = 1,
  ntest = NULL,
  nfolds = 10,
  lambda = NULL,
  labels = FALSE
)
```

Arguments

data	A SummarizedExperiment object.
alpha	Numeric. Indicates the elasticnet mixing parameter. alpha = 1 is the LASSO penalty and alpha = 0 is the Ridge penalty.
ntest	Numeric. Indicates the percentage of observations that will be used as test set. Default is NULL (no test set).
nfolds	Numeric. Indicates number of folds for cross-validation (default is 10). Although nfolds can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowable is nfolds = 3.
lambda	Numeric. Indicates the user supplied lambda sequence. Typical usage is to have the program compute its own lambda sequence based on nlambda and lambda.min.ratio. See ?glmnet::glmnet().
labels	Logical. Indicates if feature names should be plotted in coefficient plot or not. Default is FALSE.

Value

A list with results.

Author(s)

Pol Castellano-Escuder

References

Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/.

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `coefficients` (tibble), `coefficients_plot` (ggplot2 object), `cv_plot` (ggplot2
# LASSO
```

```
data %>%
  PomaLasso(alpha = 1,
            ntest = NULL,
            nfolds = 10,
            lambda = NULL,
            labels = TRUE)

# Elasticnet
data %>%
  PomaLasso(alpha = 0.5,
            ntest = NULL,
            nfolds = 10,
            lambda = NULL,
            labels = TRUE)

# Ridge Regression
data %>%
  PomaLasso(alpha = 0,
            ntest = NULL,
            nfolds = 10,
            lambda = NULL,
            labels = FALSE)
```

PomaLimma *Differential Expression Analysis Using* limma

Description

PomaLimma uses the classical limma package to compute differential expression analysis.

Usage

```
PomaLimma(
  data,
  contrast = NULL,
  outcome = NULL,
  covs = NULL,
  adjust = "fdr",
  block = NULL,
  weights = FALSE
)
```

Arguments

data	A SummarizedExperiment object.
contrast	Character. Indicates the comparison. For example, "Group1-Group2" or "control-intervention".
outcome	Character. Indicates the name of the colData column to be used as the outcome factor. Default is NULL (first factor variable in colData).

covs	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates). If not NULL, a limma model will be fitted using the specified covariates. Note: The order of the covariates is important and should be listed in increasing order of importance in the experimental design.
adjust	Character. Indicates the multiple comparisons correction method. Options are: "fdr", "holm", "hochberg", "hommel", "bonferroni", "BH" and "BY".
block	Character. Specifies the name of the colData factor column that includes the random effect variable to be considered (e.g., replicate). The default is NULL, indicating no random effect.
weights	Logical. Indicates whether the limma model should estimate the relative quality weights for each group. See ?limma::arrayWeights().

Value

A tibble with the results.

Author(s)

Pol Castellano-Escuder

References

Matthew E. Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, Gordon K. Smyth, limma powers differential expression analyses for RNA-sequencing and microarray studies, Nucleic Acids Research, Volume 43, Issue 7, 20 April 2015, Page e47, https://doi.org/10.1093/nar/gkv007

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaNorm()

# Basic limma
limma_results <- data %>%
  PomaLimma(contrast = "Healthy-CRC",
            covs = NULL,
            adjust = "fdr",
            block = NULL)

limma_results %>%
  dplyr::slice(1:10)

## Volcano plot
limma_results %>%
  dplyr::select(feature, log2FC, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
```

```
            outcome = "gender", # factorial variable to group by (e.g., treatment, sex, etc)
                feature_name = limma_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% limma_results$feature[1:10]] %>%
 PomaHeatmap(covs = c("gender", "smoking_condition", "alcohol_consumption"), # covariates to plot (e.g., treatmen
              feature_names = TRUE)

# Basic limma on alternative outcome
SummarizedExperiment::colData(data)$gender <- factor(ifelse(SummarizedExperiment::colData(data)$gender == 0, "ma
data %>%
  PomaLimma(contrast = "male-female",
            outcome = "gender",
            covs = NULL,
            adjust = "fdr",
            block = NULL)

limma_results %>%
  dplyr::slice(1:10)

## Volcano plot
limma_results %>%
  dplyr::select(feature, log2FC, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
           outcome = "gender", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = limma_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% limma_results$feature[1:10]] %>%
 PomaHeatmap(covs = c("gender", "smoking_condition", "alcohol_consumption"), # covariates to plot (e.g., treatmen
              feature_names = TRUE)

# limma with one covariate
data %>%
  PomaLimma(contrast = "Healthy-CRC",
            covs = "gender",
            adjust = "fdr",
            block = NULL)

limma_results %>%
  dplyr::slice(1:10)

## Volcano plot
limma_results %>%
  dplyr::select(feature, log2FC, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
```

```
  PomaBoxplots(x = "features",
           outcome = "gender", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = limma_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% limma_results$feature[1:10]] %>%
 PomaHeatmap(covs = c("gender", "smoking_condition", "alcohol_consumption"), # covariates to plot (e.g., treatmen
              feature_names = TRUE)

# limma with two covariates
data %>%
  PomaLimma(contrast = "Healthy-CRC",
            covs = c("gender", "age_at_consent"),
            adjust = "fdr",
            block = NULL)

limma_results %>%
  dplyr::slice(1:10)

## Volcano plot
limma_results %>%
  dplyr::select(feature, log2FC, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
           outcome = "gender", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = limma_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% limma_results$feature[1:10]] %>%
 PomaHeatmap(covs = c("gender", "smoking_condition", "alcohol_consumption"), # covariates to plot (e.g., treatmen
              feature_names = TRUE)

# limma with replicates
# data %>%
#   PomaLimma(contrast = "Healthy-CRC",
#             covs = NULL,
#             adjust = "fdr",
#             block = "replicate")
```

PomaLM *Linear Models*

Description

PomaLM performs a linear model on a SummarizedExperiment object.

Usage

```
PomaLM(data, x = NULL, y = NULL, adjust = "fdr")
```

Arguments

data	A SummarizedExperiment object.
x	Character vector. Indicates the names of independent variables. If it's NULL (default), all features will be used.
y	Character. Indicates the name of colData numeric columns to be used as dependent variable. If it's set to NULL, the first numeric variable in colData will be used as the dependent variable.
adjust	Character. Multiple comparisons correction method to adjust p-values. Available options are: "fdr" (false discovery rate), "holm", "hochberg", "hommel", "bonferroni", "BH" (Benjamini-Hochberg), and "BY" (Benjamini-Yekutieli).

Value

A list with results including plots and tables.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `lm_table` (tibble) and `regression_plot` (ggplot2 object)
# Perform linear model with all features
data %>%
  PomaLM()

# Perform linear model with two features
data %>%
  PomaLM(x = c("x1_methyladenosine", "x2_deoxyuridine"))
```

PomaLMM *Linear Mixed Models*

Description

PomaLMM performs linear mixed models on a SummarizedExperiment object.

Usage

```
PomaLMM(data, x = NULL, y = NULL, adjust = "fdr", clean_plot = FALSE)
```

Arguments

data	A SummarizedExperiment object.
x	Character vector. Indicates the names of colData columns to be used as random and fixed effects (independent variables). If it's set to NULL (default), all variables in colData will be used.
y	Character vector. Indicates the names of dependent variables. If it's NULL (default), all features will be used.
adjust	Character. Multiple comparisons correction method to adjust p-values. Available options are: "fdr" (false discovery rate), "holm", "hochberg", "hommel", "bonferroni", "BH" (Benjamini-Hochberg), and "BY" (Benjamini-Yekutieli).
clean_plot	Logical. Indicates if remove intercept and linear mixed model residues boxplots from the plot. Defasult is FALSE.

Value

A list with results including plots and tables. Table values indicate the percentage variance explained per variable.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `lm_table` (tibble) and `regression_plot` (ggplot2 object)
## Perform linear mixed model with all features
#data %>%
#  PomaLMM()
#
## Perform linear mixed model with two features
#data %>%
#  PomaLMM(y = c("x1_methyladenosine", "x1_methylhistamine"))
#
## Perform linear mixed model with one random effect
#data %>%
#  PomaLMM(x = "smoking_condition")
#
## Perform linear mixed model with two random effects and two features
#data %>%
#  PomaLMM(x = c("smoking_condition", "gender"),
#          y = c("x1_methyladenosine", "x1_methylhistamine"))
#
## Perform linear mixed model with no random effects and two features, therefore, a linear model will be fitted
#data %>%
#  PomaLMM(x = "age_at_consent", # Numerical, i.e., fixed effect
#          y = c("x1_methyladenosine", "x1_methylhistamine"))
```

```
#
## Perform linear mixed model with no random effects and all features, therefore, a linear model will be fitted
#data %>%
#  PomaLMM(x = "age_at_consent") # Numerical i.e., fixed effect
```

PomaNorm *Normalize Data*

Description

PomaNorm performs data normalization using various normalization methods.

Usage

```
PomaNorm(data, sample_norm = "none", method = "log_pareto")
```

Arguments

data	A SummarizedExperiment object.
sample_norm	Character. Sample normalization method. Options include "none" (default), "sum", or "quantile". Quantile is often used when >100 samples.
method	Character. The normalization method to use. Options include "none" (no normalization), "auto_scaling" (autoscaling, i.e., Z-score normalization), "level_scaling" (level scaling), "log_scaling" (log scaling), "log" (log transformation), "vast_scaling" (vast scaling), "log_pareto" (log Pareto scaling), "min_max" (min-max), and "box_cox" (Box-Cox transformation).

Value

A SummarizedExperiment object with normalized data.

Author(s)

Pol Castellano-Escuder

References

Van den Berg, R. A., Hoefsloot, H. C., Westerhuis, J. A., Smilde, A. K., & van der Werf, M. J. (2006). Centering, scaling, and transformations: improving the biological information content of metabolomics data. BMC genomics, 7(1), 142.

Examples

```
# Output is a normalized SummarizedExperiment object
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

# No sample normalization
data %>%
  PomaNorm(sample_norm = "none",
           method = "log_pareto")

# Sum sample normalization
data %>%
  PomaNorm(sample_norm = "sum",
           method = "log_pareto")
```

PomaOddsRatio *Logistic Regression Model Odds Ratios*

Description

PomaOddsRatio calculates the Odds Ratios for each feature from a logistic regression model using the binary outcome (group/type must be a binary factor) as a dependent variable.

Usage

```
PomaOddsRatio(data, feature_name = NULL, covs = NULL, show_ci = TRUE)
```

Arguments

data	A SummarizedExperiment object.
feature_name	Character vector. Indicates the name/s of feature/s that will be used to fit the model. If it's NULL (default), all variables will be included in the model.
covs	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates).
show_ci	Logical. Indicates if the 95% confidence intervals will be plotted. Default is TRUE.

Value

A list with results including plots and tables.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `odds_ratio_table` (tibble) and `odds_ratio_plot` (ggplot2 object)
data %>%
  PomaOddsRatio(feature_name = c("glutamic_acid", "glutamine", "glycine", "histidine"),
                covs = NULL,
                show_ci = TRUE)

# With covariates
data %>%
  PomaOddsRatio(feature_name = c("glutamic_acid", "glutamine", "glycine", "histidine"),
                covs = "steroids",
                show_ci = TRUE)
```

PomaOutliers *Analyse and Remove Statistical Outliers*

Description

PomaOutliers analyses and removes statistical outliers from the data.

Usage

```
PomaOutliers(
  data,
  method = "euclidean",
  type = "median",
  outcome = NULL,
  coef = 2,
  labels = FALSE
)
```

Arguments

data	A SummarizedExperiment object.
method	Character. Indicates the distance measure method to perform MDS.
type	Character. Indicates the type of outlier analysis to perform. Options are "median" (default) and "centroid". See vegan::betadisper.
outcome	Character. Indicates the name of the colData column to be used as the outcome factor. Default is NULL (first factor variable in colData).
coef	Numeric. Indicates the outlier coefficient. Lower values are more sensitive to outliers while higher values are less restrictive about outliers.
labels	Logical. Indicates if sample names should to be plotted.

Value

A `list` with the results.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `polygon_plot` (ggplot2 object), `distance_boxplot` (ggplot2 object), `outliers` (
outlier_results <- data %>%
  PomaOutliers(method = "euclidean",
               type = "median",
               outcome = NULL,
               coef = 2,
               labels = FALSE)

outlier_results$data # cleaned SummarizedExperiment object

## Change oulier group factor
outlier_results2 <- data %>%
  PomaOutliers(method = "euclidean",
               type = "median",
               outcome = "steroids",
               coef = 2,
               labels = FALSE)

outlier_results2$data # cleaned SummarizedExperiment object
```

`PomaPCA` *Principal Components Analysis*

Description

`PomaPCA` performs a principal components analysis on the given `SummarizedExperiment` object.

Usage

```
PomaPCA(
  data,
  outcome = NULL,
  center = TRUE,
  scale = TRUE,
  ncomp = 4,
  labels = FALSE,
```

```
  ellipse = FALSE,
  load_length = 1
)
```

Arguments

data	A SummarizedExperiment object.
outcome	Character. Indicates the name of the colData column to be used as the outcome factor. Default is NULL (first factor variable in colData).
center	Logical. Indicates whether the variables should be shifted to be zero centered. Default is TRUE.
scale	Logical. Indicates whether the variables should be scaled to have unit variance before the analysis takes place. Default is TRUE.
ncomp	Numeric. Number of components to be included in the results. Default is 4.
labels	Logical. Indicates if sample names should be displayed.
ellipse	Logical. Indicates whether a 95 percent confidence interval ellipse should be displayed in score plot and biplot. Default is FALSE.
load_length	Numeric. Indicates the length of biplot loading arrows. Value between 1 and 2. Default is 1.

Value

A list with results including plots and tables.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `factors` (tibble wth principal components), `eigenvalues` (tibble), `loadings` (t
# Default outcome (first factor variable in `colData`)
data %>%
  PomaPCA(outcome = NULL,
          center = TRUE,
          scale = TRUE,
          labels = FALSE,
          ellipse = FALSE)

# Alternative outcome
data %>%
  PomaPCA(outcome = "steroids",
          center = TRUE,
          scale = TRUE,
          labels = FALSE,
          ellipse = FALSE)
```

PomaPCR *Principal Components Regression*

Description

PomaPCR performs Principal Components Regression.

Usage

```
PomaPCR(data, center = TRUE, scale = TRUE, ncomp = 2, y = NULL, adjust = "fdr")
```

Arguments

data	A SummarizedExperiment object.
center	Logical. Indicates whether the variables should be shifted to be zero centered. Default is TRUE.
scale	Logical. Indicates whether the variables should be scaled to have unit variance before the analysis takes place. Default is TRUE.
ncomp	Numeric. Indicates the number of principal components used as predictors in the model. Default is 2.
y	Character. Indicates the name of colData columns to be used as dependent variable. If it's set to NULL, the first numeric variable in colData will be used as the dependent variable.
adjust	Character. Multiple comparisons correction method to adjust p-values. Available options are: "fdr" (false discovery rate), "holm", "hochberg", "hommel", "bonferroni", "BH" (Benjamini-Hochberg), and "BY" (Benjamini-Yekutieli).

Value

A tibble with the results.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaNorm()

# PCR with 2 components and the default outcome (1st column of `colData`)
data %>%
  PomaPCR(center = TRUE,
          scale = TRUE,
          ncomp = 2,
          y = NULL,
          adjust = "fdr")
```

```
# PCR with 2 components and alternative outcome
data %>%
  PomaPCR(center = TRUE,
          scale = TRUE,
          ncomp = 2,
          y = "age_at_consent",
          adjust = "fdr")

# PCR with 20 components and alternative outcome
data %>%
  PomaPCR(center = TRUE,
          scale = TRUE,
          ncomp = 20,
          y = "age_at_consent",
          adjust = "fdr")
```

PomaPLS *Partial Least Squares Methods*

Description

PomaPLS performs Partial Least Squares (PLS) regression, Partial Least Squares Discriminant Analysis (PLS-DA) to classify samples, and Sparse Partial Least Squares Discriminant Analysis (sPLS-DA) to classify samples (supervised analysis) and select variables.

Usage

```
PomaPLS(
  data,
  method = "pls",
  y = NULL,
  ncomp = 5,
  labels = FALSE,
  ellipse = TRUE,
  cross_validation = FALSE,
  validation = "Mfold",
  folds = 5,
  nrepeat = 10,
  vip = 1,
  num_features = 10,
  theme_params = list()
)
```

Arguments

data	A SummarizedExperiment object.
method	Character. PLS method. Options include "pls", "plsda", and "splsda".

y	Character. Indicates the name of colData columns to be used as dependent variable. If it's set to NULL, the first variable in colData will be used as the dependent variable.
ncomp	Numeric. Number of components in the model. Default is 5.
labels	Logical. Indicates if sample names should be displayed.
ellipse	Logical. Indicates whether a 95 percent confidence interval ellipse should be displayed. Default is TRUE.
cross_validation	Logical. Indicates if cross-validation should be performed for PLS-DA ("plsda") and sPLS-DA ("splsda") methods. Default is FALSE.
validation	Character. (Only for "plsda" and "splsda" methods). Indicates the cross-validation method. Options are "Mfold" and "loo" (Leave-One-Out).
folds	Numeric. (Only for "plsda" and "splsda" methods). Number of folds for "Mfold" cross-validation method (default is 5). If the validation method is "loo", this value is set to 1.
nrepeat	Numeric. (Only for "plsda" and "splsda" methods). Number of times the cross-validation process is repeated.
vip	Numeric. (Only for "plsda" method). Indicates the variable importance in the projection (VIP) cutoff.
num_features	Numeric. (Only for "splsda" method). Number of features to discriminate groups.
theme_params	List. Indicates theme_poma parameters.

Value

A list with results including plots and tables.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000284 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `factors` (tibble), `factors_plot` (ggplot2 object), `loadings` (tibble), and `loa
# PLS
data %>%
  PomaPLS(method = "pls",
          y = NULL,
          ncomp = 5,
          labels = FALSE,
          ellipse = FALSE)

## Output is a list with objects `factors` (tibble), `factors_plot` (ggplot2 object), `vip_values` (tibble), and `v
```

```
# PLS-DA
data %>%
  PomaPLS(method = "plsda",
          y = NULL,
          ncomp = 5,
          labels = FALSE,
          ellipse = TRUE,
          cross_validation = FALSE,
          vip = 1)

# Alternative outcome (dependent variable)
data %>%
  PomaPLS(method = "plsda",
          y = "gender",
          ncomp = 5,
          labels = FALSE,
          ellipse = TRUE,
          cross_validation = FALSE,
          vip = 1)

## Output is a list with objects `factors` (tibble), `factors_plot` (ggplot2 object), `vip_values` (tibble), `vip_p
# PLS-DA with Cross-Validation
data %>%
  PomaPLS(method = "plsda",
          y = NULL,
          ncomp = 5,
          labels = FALSE,
          ellipse = TRUE,
          cross_validation = TRUE,
          validation = "Mfold",
          folds = 5,
          nrepeat = 10,
          vip = 1)

## Output is a list with objects `factors` (tibble), `factors_plot` (ggplot2 object), `selected_features` (tibble)
# sPLS-DA
data %>%
  PomaPLS(method = "splsda",
          y = NULL,
          ncomp = 5,
          labels = FALSE,
          ellipse = TRUE,
          cross_validation = FALSE,
          num_features = 10)

## Output is a list with objects `factors` (tibble), `factors_plot` (ggplot2 object), `selected_features` (tibble)
# sPLS-DA with Cross-Validation
data %>%
  PomaPLS(method = "splsda",
          y = NULL,
          ncomp = 3,
          labels = FALSE,
          ellipse = TRUE,
```

```
            cross_validation = TRUE,
            validation = "Mfold",
            folds = 5,
            nrepeat = 10,
            num_features = 10)
```

`PomaRandForest` *Classification Random Forest*

Description

`PomaRandForest` performs classification random forest. This method can be used both for prediction and variable selection.

Usage

```
PomaRandForest(
  data,
  ntest = NULL,
  ntree = 500,
  mtry = floor(sqrt(ncol(t(SummarizedExperiment::assay(data))))),
  nodesize = 1,
  nvar = 20
)
```

Arguments

`data`	A `SummarizedExperiment` object.
`ntest`	Numeric. Indicates the percentage of observations that will be used as test set. Default is NULL (no test set).
`ntree`	Numeric. Indicates the number of trees to grow.
`mtry`	Numeric. Indicates the number of variables randomly sampled as candidates at each split. This value is set sqrt(p) (where p is number of variables in data) by default.
`nodesize`	Numeric. Indicates the minimum size of terminal nodes. Default is 1.
`nvar`	Numeric. Indicates the number of variables to show in the Gini Index plot.

Value

A `list` with results including plots and tables.

Author(s)

Pol Castellano-Escuder

References

A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18–22.

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute() %>%
  PomaNorm()

## Output is a list with objects `MeanDecreaseGini` (tibble), `MeanDecreaseGini_plot` (ggplot2 object), `oob_error
data %>%
  PomaRandForest(ntree = 500,
                 mtry = floor(sqrt(ncol(t(SummarizedExperiment::assay(data))))),
                 nodesize = 1,
                 nvar = 20)
```

PomaRankProd *Rank Product/Rank Sum Analysis*

Description

PomaRankProd performs the Rank Product (or Rank Sum) method to identify differentially expressed genes.

Usage

```
PomaRankProd(data, logged = TRUE, paired = NA, cutoff = 1, method = "pfp")
```

Arguments

data	A SummarizedExperiment object.
logged	Logical. Indicates if data should be log transformed first.
paired	Numeric. Indicates the number of random pairs generated in the function, if set to NA (default), the odd integer closer to the square of the number of replicates is used.
cutoff	Numeric. Indicates the pfp/pvalue threshold value used to select features. Default is 1 to include all features.
method	Character. Indicates the method to identify features. "pfp" uses percentage of false prediction, which is a default setting. "pval" uses p-values which is less stringent than pfp.

Value

A list with the results. Objects in the list are up_regulated (tibble) and down_regulated (tibble).

Author(s)

Pol Castellano-Escuder

References

Breitling, R., Armengaud, P., Amtmann, A., and Herzyk, P.(2004) Rank Products: A simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments, FEBS Letter, 57383-92

Hong, F., Breitling, R., McEntee, W.C., Wittner, B.S., Nemhauser, J.L., Chory, J. (2006). RankProd: a bioconductor package for detecting differentially expressed genes in meta-analysis Bioinformatics. 22(22):2825-2827

Del Carratore, F., Jankevics, A., Eisinga, R., Heskes, T., Hong, F. & Breitling, R. (2017). RankProd 2.0: a refactored Bioconductor package for detecting differentially expressed features in molecular profiling datasets. Bioinformatics. 33(17):2774-2775

Examples

```
data <- POMA::st000336 %>% # Example SummarizedExperiment object included in POMA
  PomaImpute()

## Output is a list with objects `up_regulated` (tibble with up regulated features) and `down_regulated` (tibble wi
## Perform on no-scaled object to avoid negative values
data %>%
  PomaRankProd(method = "pfp")
```

PomaUMAP *Dimensionality Reduction with UMAP*

Description

PomaUMAP performs a dimension reduction of the data using the Uniform Manifold Approximation and Projection (UMAP) method. See ?uwot::umap() for more.

Usage

```
PomaUMAP(
  data,
  n_neighbors = floor(sqrt(nrow(data))),
  n_components = 2,
  metric = "euclidean",
  pca = NULL,
  min_dist = 0.01,
  spread = 1,
  hdbscan_minpts = floor(nrow(data) * 0.05),
  show_clusters = TRUE,
  hide_noise = TRUE,
  labels = FALSE,
```

```
  outcome = NULL,
  theme_params = list(legend_title = TRUE, legend_position = "bottom")
)
```

Arguments

data	A SummarizedExperiment object.
n_neighbors	Numeric. Indicates the size of local neighborhood (sample points) used for manifold approximation.
n_components	Numeric. Indicates the dimension of the space to embed into.
metric	Character. Indicates the distance measure method to find nearest neighbors. Options are "euclidean", "cosine", "manhattan", "hamming" and "correlation". See ?uwot::umap().
pca	If not NULL (default), reduce data to this number of columns using PCA before UMAP.
min_dist	Numeric. Indicates the effective minimum distance between embedded points.
spread	Numeric. Indicates the effective scale of embedded points.
hdbscan_minpts	Numeric. Indicates the minimum size of clusters. See ?dbscan::hdbscan().
show_clusters	Logical. Indicates if clusters computed with HDBSCAN method should be plotted or not.
hide_noise	Logical. Specifies whether to hide Cluster 0 in the plot. In HDBSCAN, Cluster 0 is typically regarded as "noise."
labels	Logical. Indicates if sample names should be plotted or not.
outcome	Character. Has no effect on the analysis. Indicates the name of the colData column to be added to the output table. Default is NULL (no extra column added).
theme_params	List. Indicates theme_poma parameters.

Value

A list with results including plots and tables.

Author(s)

Pol Castellano-Escuder

References

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.

Campello, R. J., Moulavi, D., & Sander, J. (2013, April). Density-based clustering based on hierarchical density estimates. In Pacific-Asia conference on knowledge discovery and data mining (pp. 160-172). Springer, Berlin, Heidelberg.

Examples

```
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

## Output is a list with objects `umap_embeddings` (tibble) and `umap_plot` (ggplot2 object)
data %>%
  PomaNorm() %>%
  PomaUMAP(metric = "euclidean",
           pca = NULL,
           min_dist = 0.01,
           spread = 1,
           hdbscan_minpts = floor(nrow(data) * 0.05),
           show_clusters = TRUE,
           hide_noise = TRUE,
           labels = FALSE,
           outcome = NULL)
```

PomaUnivariate *Univariate Statistical Test*

Description

PomaUnivariate performs parametric and non-parametric univariate statistical tests on a SummarizedExperiment object to compare groups or conditions. Available methods include T-test, ANOVA, ANCOVA, Mann Whitney U Test (Wilcoxon Rank Sum Test), and Kruskal-Wallis.

Usage

```
PomaUnivariate(
  data,
  method = "ttest",
  covs = NULL,
  error = NULL,
  paired = FALSE,
  var_equal = FALSE,
  adjust = "fdr",
  run_post_hoc = TRUE
)
```

Arguments

data	A SummarizedExperiment object.
method	Character. The univariate statistical test to be performed. Available options include "ttest" (T-test), "anova" (analysis of variance), "mann" (Wilcoxon rank-sum test), and "kruskal" (Kruskal-Wallis test).
covs	Character vector. Indicates the names of colData columns to be included as covariates. Default is NULL (no covariates). If not NULL, an ANCOVA model will be fitted using the specified covariates. Note: The order of the covariates is important and should be listed in increasing order of importance in the experimental design.

error	Character vector. Indicates the name of a colData column to be included as an error term (e.g., replicates). Default is NULL (no error term).
paired	Logical. Indicates if the data is paired or not. Default is FALSE.
var_equal	Logical. Indicates if the data variances are assumed to be equal or not. Default is FALSE.
adjust	Character. Multiple comparisons correction method to adjust p-values. Available options are: "fdr" (false discovery rate), "holm", "hochberg", "hommel", "bonferroni", "BH" (Benjamini-Hochberg), and "BY" (Benjamini-Yekutieli).
run_post_hoc	Logical. Indicates if computing post-hoc tests or not. Setting this parameter to FALSE can save time for large datasets.

Value

A tibble for "ttest" and "mann". A list for "anova" and "kruskal".

Author(s)

Pol Castellano-Escuder

Examples

```
# Two groups
## Output columns: feature, fold_change, diff_means, pvalue, adj_pvalue, mean_xxx (group 1) mean_yyy (group 2), sd_
data <- POMA::st000336 # Example SummarizedExperiment object included in POMA

## Perform T-test
ttest_results <- st000336 %>%
  PomaImpute() %>%
  PomaUnivariate(method = "ttest",
                 paired = FALSE,
                 var_equal = FALSE,
                 adjust = "fdr")

ttest_results %>%
  dplyr::slice(1:10)

## Volcano plot
ttest_results %>%
  dplyr::select(feature, fold_change, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
           outcome = "group", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = ttest_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% ttest_results$feature[1:10]] %>%
  PomaHeatmap(covs = c("group"), # covariates to plot (e.g., treatment, sex, etc)
```

```
              feature_names = TRUE)

## Perform Mann-Whitney U test
mann_whitney_results <- st000336 %>%
  PomaImpute() %>%
  PomaUnivariate(method = "mann",
                 paired = FALSE,
                 var_equal = FALSE,
                 adjust = "fdr")

mann_whitney_results %>%
  dplyr::slice(1:10)

## Volcano plot
mann_whitney_results %>%
  dplyr::select(feature, fold_change, pvalue) %>%
  PomaVolcano(labels = TRUE)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
           outcome = "group", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = mann_whitney_results$feature[1:10])

## Heatmap of top features
data[rownames(data) %in% mann_whitney_results$feature[1:10]] %>%
  PomaHeatmap(covs = c("group"), # covariates to plot (e.g., treatment, sex, etc)
              feature_names = TRUE)

# More than 2 groups
## Output is a list with objects `result` and `post_hoc_tests`
data <- POMA::st000284 # Example SummarizedExperiment object included in POMA

## Perform Two-Way ANOVA
anova_results <- data %>%
  PomaUnivariate(method = "anova",
                 covs = c("gender"),
                 error = NULL,
                 adjust = "fdr",
                 run_post_hoc = TRUE)

anova_results$result %>%
  dplyr::slice(1:10)

anova_results$post_hoc_tests %>%
  dplyr::slice(1:10)

## Boxplot of top features
data %>%
  PomaBoxplots(x = "features",
           outcome = "factors", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = anova_results$result$feature[1:10])
```

```
## Boxplot of top significant pairwise features (after posthoc test)
data %>%
  PomaBoxplots(x = "features",
           outcome = "factors", # factorial variable to group by (e.g., treatment, sex, etc)
               feature_name = unique(anova_results$post_hoc_tests$feature)[1:10])

## Heatmap of top features
data[rownames(data) %in% anova_results$result$feature[1:10]] %>%
  PomaHeatmap(covs = c("factors"), # covariates to plot (e.g., treatment, sex, etc)
              feature_names = TRUE)

## Perform Three-Way ANOVA
data %>%
  PomaUnivariate(method = "anova",
                 covs = c("gender", "smoking_condition"))

## Perform ANCOVA with one numeric covariate and one factor covariate
data %>%
  PomaUnivariate(method = "anova",
                 covs = c("age_at_consent", "smoking_condition"))

# Perform Kruskal-Wallis test
data %>%
  PomaUnivariate(method = "kruskal",
                 adjust = "holm",
                 run_post_hoc = TRUE)
```

PomaVolcano *Volcano Plot*

Description

PomaVolcano creates a volcano plot from a given dataset. This function is designed to visualize the statistical significance (p-value) against the magnitude of change (log2 fold change) for features.

Usage

```
PomaVolcano(
  data,
  pval_cutoff = 0.05,
  log2fc_cutoff = NULL,
  labels = FALSE,
  x_label = "log2 (Fold Change)",
  y_label = "-log10 (P-value)"
)
```

Arguments

data	A data frame with at least three columns: feature names, effect size (e.g., logFC), and statistical significance values. These should be the first three columns of the data, in this order.

pval_cutoff	Numeric. Specifies the p-value threshold for significance in the volcano plot. The default is set to 0.05. This parameter determines the horizontal line in the plot indicating the threshold for statistical significance.
log2fc_cutoff	Numeric. Specifies the log2 fold change cutoff for the volcano plot. If not provided, the cutoff is set to the 75th percentile of the absolute log2 fold changes in the data. This parameter determines the vertical lines in the plot indicating the magnitude of change threshold.
labels	Logical. Indicates whether to plot labels for significant features.
x_label	Character. Custom label for the x-axis.
y_label	Character. Custom label for the y-axis.

Value

A ggplot object representing the volcano plot.

Author(s)

Pol Castellano-Escuder

Examples

```
data <- POMA::st000336 # Example SummarizedExperiment object included in POMA

# T-test
results <- data %>%
  PomaImpute() %>%
  PomaUnivariate() %>%
  dplyr::select(feature, fold_change, pvalue)

results %>%
  PomaVolcano(pval_cutoff = 0.05,
              log2fc_cutoff = NULL,
              labels = FALSE,
              x_label = "log2 (Fold Change)",
              y_label = "-log10 (P-value)")

# Limma
results <- data %>%
  PomaImpute() %>%
  PomaNorm() %>%
  PomaLimma(contrast = "DMD-Controls") %>%
  dplyr::select(feature, log2FC, pvalue)

results %>%
  PomaVolcano(pval_cutoff = 0.05,
              log2fc_cutoff = NULL,
              labels = FALSE,
              x_label = "log2 (Fold Change)",
              y_label = "-log10 (P-value)")
```

poma_pal_c *Return function to interpolate a continuous POMA color palette*

Description

Return function to interpolate a continuous POMA color palette

Usage

```
poma_pal_c(palette = "nature")
```

Arguments

palette	Character name of palette in poma_palettes

poma_pal_d *Return function to interpolate a discrete POMA color palette*

Description

Return function to interpolate a discrete POMA color palette

Usage

```
poma_pal_d(palette = "nature")
```

Arguments

palette	Character name of palette in poma_palettes

quantile_norm *Sample Quantile Normalization*

Description

Compute quantile normalization.

Usage

```
quantile_norm(data)
```

Arguments

data	A data matrix (samples in rows).

scale_color_poma_c *Color scale constructor for continuous* viridis *"plasma" palette*

Description

Color scale constructor for continuous viridis "plasma" palette

Usage

```
scale_color_poma_c()
```

scale_color_poma_d *Color scale constructor for discrete* viridis *"plasma" palette*

Description

Color scale constructor for discrete viridis "plasma" palette

Usage

```
scale_color_poma_d()
```

scale_fill_poma_c *Fill scale constructor for continuous* viridis *"plasma" palette*

Description

Fill scale constructor for continuous viridis "plasma" palette

Usage

```
scale_fill_poma_c()
```

scale_fill_poma_d *Fill scale constructor for discrete* viridis *"plasma" palette*

Description

Fill scale constructor for discrete viridis "plasma" palette

Usage

```
scale_fill_poma_d()
```

st000284 *Colorectal Cancer Detection Using Targeted Serum Metabolic Profiling*

Description

Colorectal cancer (CRC) is one of the most prevalent and deadly cancers in the world. Despite an expanding knowledge of its molecular pathogenesis during the past two decades, robust biomarkers to enable screening, surveillance, and therapy monitoring of CRC are still lacking. In this study, we present a targeted liquid chromatography-tandem mass spectrometry-based metabolic profiling approach for identifying biomarker candidates that could enable highly sensitive and specific CRC detection using human serum samples. In this targeted approach, 158 metabolites from 25 metabolic pathways of potential significance were monitored in 234 serum samples from three groups of patients (66 CRC patients, 76 polyp patients, and 92 healthy controls). Partial least squares-discriminant analysis (PLS-DA) models were established, which proved to be powerful for distinguishing CRC patients from both healthy controls and polyp patients. Receiver operating characteristic curves generated based on these PLS-DA models showed high sensitivities (0.96 and 0.89, respectively, for differentiating CRC patients from healthy controls or polyp patients); good specificities (0.80 and 0.88), and excellent areas under the curve (0.93 and 0.95) were also obtained. Monte Carlo cross validation (MCCV) was also applied, demonstrating the robust diagnostic power of this metabolic profiling approach.

Usage

```
st000284
```

Format

A `SummarizedExperiment` object: 224 samples, 113 metabolites, 4 covariables and 3 groups (CRC, Healthy and Polyp).

metabolites 113 serum metabolites.

covariables Age at consent, Gender, Smoking Condition and Alcohol Consumption.

Source

https://www.metabolomicsworkbench.org/data/DRCCMetadata.php?Mode=Study&StudyID=ST000284&StudyType=MS&ResultType=1%20target=_blank

References

Colorectal Cancer Detection Using Targeted Serum Metabolic Profiling, J. Proteome. Res., 2014, 13, 4120-4130.

st000336 *Targeted LC/MS of urine from boys with DMD and controls*

Description

Duchenne Muscular Dystrophy (DMD) is an X-linked recessive form of muscular dystrophy that affects males via a mutation in the gene for the muscle protein, dystrophin. Progression of the disease results in severe muscle loss, ultimately leading to paralysis and death. Steroid therapy has been a commonly employed method for reducing the severity of symptoms. This study aims to quantify the urine levels of amino acids and organic acids in patients with DMD both with and without steroid treatment. Track the progression of DMD in patients who have provided multiple urine samples.

Usage

```
st000336
```

Format

A `SummarizedExperiment` object: 57 samples, 31 metabolites, 1 covariable and 2 groups (Controls and DMD).

metabolites 31 urine metabolites.

covariables Steroid status.

Source

https://www.metabolomicsworkbench.org/data/DRCCMetadata.php?Mode=Study&DataMode=AllData&StudyID=ST000336&StudyType=MS&ResultType=1#DataTabs

sum_norm *Sample Sum Normalization*

Description

Compute sum normalization. Final unit is a percentage.

Usage

```
sum_norm(data)
```

Arguments

`data`	A data matrix (samples in rows).

theme_poma *A ggplot theme which allow custom yet consistent styling of plots in the POMA package and web app.*

Description

A ggplot theme which allow custom yet consistent styling of plots in the POMA package and web app.

Usage

```
theme_poma(
  base_size = 15,
  axistitle = "xy",
  axistext = "xy",
  legend_position = "bottom",
  legend_title = TRUE,
  axis_x_rotate = FALSE,
  margin = 2
)
```

Arguments

base_size	(integer) Base point size
axistitle	(string) Axis titles. Options include "none" or any combination of "X", "Y", "x" and "y".
axistext	(string) Axis text labels for values or groups. Options include "none" or any combination of "X", "Y", "x" and "y".
legend_position	Character. Legend position. See ggplot2 documentation.
legend_title	Logical. Include legend title.
axis_x_rotate	Logical. Rotate x-axis 45 degrees.
margin	(numeric) Should a margin of x be added to the plot? Defaults to 0 (no margin by default).

Examples

```
## Not run:
library(ggplot2)
ggplot(diamonds, aes(cut)) + geom_bar() + theme_poma()

## End(Not run)
```

`%>%` *Pipe operator*

Description

See `magrittr::%>%` for details.

Usage

```
lhs %>% rhs
```

Value

Nothing. Just allow the use of magrittr pipe "%>%"

Index