Package ‘PLSDAbatch’

May 13, 2024

Type Package

Title PLSDA-batch

Version 1.0.0

Description A novel framework to correct for batch effects prior to any downstream analysis in microbiome data based on Projection to Latent Structures Discriminant Analysis. The main method is named “PLSDA-batch”. It first estimates treatment and batch variation with latent components, then subtracts batch-associated components from the data whilst preserving biological variation of interest. PLSDA-batch is highly suitable for microbiome data as it is non-parametric, multivariate and allows for ordination and data visualisation. Combined with centered log-ratio transformation for addressing uneven library sizes and compositional structure, PLSDA-batch addresses all characteristics of microbiome data that existing correction methods have ignored so far. Two other variants are proposed for 1/ unbalanced batch x treatment designs that are commonly encountered in studies with small sample sizes, and for 2/ selection of discriminative variables amongst treatment groups to avoid overfitting in classification problems. These two variants have widened the scope of applicability of PLSDA-batch to different data settings.

License GPL-3

Depends R (>= 4.3.0)

Imports mixOmics, scales, Rdpack, ggplot2, gridExtra, ggpubr, lmerTest, performance, grid, stats, pheatmap, vegan, Biobase, BiocStyle, TreeSummarizedExperiment

Suggests knitr, rmarkdown, testthat, badger

biocViews StatisticalMethod, DimensionReduction, PrincipalComponent, Classification, Microbiome, BatchEffect, Normalization, Visualization

VignetteBuilder knitr

RdMacros Rdpack

RdVersion 7.2.3

Encoding UTF-8

URL https://github.com/EvaYiwenWang/PLSDAbatch

BugReports https://github.com/EvaYiwenWang/PLSDAbatch/issues/
AD_data

Anaerobic digestion study

Description

This study explored the microbial indicators that could improve the efficacy of anaerobic digestion (AD) bioprocess and prevent its failure. The samples were treated with two different ranges of phenol concentration (effect of interest) and processed at five different dates (batch effect). This study includes a clear and strong batch effect with an approx. balanced batch x treatment design.

Usage

data('AD_data')
alignment_score

Format

A list containing three TreeSummarizedExperiment objects FullData, EgData and CorrectData:

FullData  A TreeSummarizedExperiment object containing the counts of 75 samples and 567 OTUs. The meta data information of each sample is stored in the rowData, while the taxonomy of each OTU is stored in the colData.

EgData  A TreeSummarizedExperiment object containing the values of 75 samples and 231 OTUs filtered and centered log ratio transformed from the FullData with raw counts. The rowData includes \( Y\.trt \) and \( Y\.bat \). \( Y\.trt \) is the effect of interest, which is a factor of phenol concentrations for each sample in the AD study; \( Y\.bat \) is the batch effect, which is a factor of sample processing dates for each sample. The taxonomy of each OTU is stored in the colData. The rowTree is built based on the \( Y\.bat \).

CorrectData  A TreeSummarizedExperiment object containing seven datasets before or after batch effect correction using different methods. Each assay includes 75 samples and 231 OTUs.

Value

None.

Source

The raw data were provided by Dr. Olivier Chapleur and published at the referenced article. Filtering and normalisation described in our package vignette.

References


alignment_score  Alignment Scores for Evaluating the Degree of Mixing Samples

Description

This function evaluates the degree of mixing samples from different batches in the batch corrected data. It is based on the dissimilarity matrix from Principal Component Analysis.

Usage

alignment_score(
  data,
  batch,
  var = 0.95,
  k = round(0.1 * nrow(data)),
  ncomp = 20
)
alignment_score

Arguments

- **data**: A numeric matrix. Samples are in rows, while variables are in columns. NAs are not allowed.
- **batch**: A factor or a class vector for the batch grouping information (categorical outcome variable). The length should be equal to the number of samples in the data.
- **var**: The proportion of data variance explained by the principal components, ranging from 0 to 1. Default value is 0.95.
- **k**: Integer, the number of nearest neighbours. By default 10% of the number of samples are used.
- **ncomp**: Integer, the number of components for principal component analysis. Default value is 20.

Value

A numeric alignment score that ranges from 0 to 1, representing poor to perfect performance of mixing the samples from different batches.

Author(s)

Yiwen Wang, Kim-Anh Lê Cao

References


See Also

- **Scatter_Density**, **box_plot**, **density_plot** and **partVar_plot** as the other methods for batch effect detection and batch effect removal assessment.

Examples

```r
library(TreeSummarizedExperiment) # for functions assays(),rowData()
data('sponge_data')
X <- assays(sponge_data)$Clr_value # centered log ratio transformed data
batch <- rowData(sponge_data)$Y.bat # batch information
names(batch) <- rownames(sponge_data)

alignment_score(data = X, batch = batch, var = 0.95, k = 3, ncomp = 20)
```
Description

This function draws side-by-side box plots for each batch.

Usage

```r
box_plot(
  df, 
  title = NULL, 
  batch.legend.title = "Batch", 
  ylab = "Value", 
  color.set = NULL, 
  x.angle = 0, 
  x.hjust = 0.5, 
  x.vjust = 0.5
)
```

Arguments

- `df` A data frame used to draw the box plots.
- `title` Character, the plot title.
- `batch.legend.title` Character, the legend title of batches.
- `ylab` Character, y-axis title.
- `color.set` A vector of character, indicating the set of colors to use. The colors are represented by hexadecimal color code.
- `x.angle` Numeric, angle of x axis, in the range of 0 to 360.
- `x.hjust` Numeric, horizontal justification of x axis, in the range of 0 to 1.
- `x.vjust` Numeric, vertical justification of x axis, in the range of 0 to 1.

Value

None.

Author(s)

Yiwen Wang, Kim-Anh Lê Cao

See Also

`Scatter_Density, density_plot, alignment_score` and `partVar_plot` as the other methods for batch effect detection and batch effect removal assessment.
deflate_mtx

### Description

This function removes the variance of given component \( t \) from the input matrix \( X \).

\[
\hat{X} = X - t(t^\top t)^{-1}t^\top X
\]

It is a built-in function of PLSDA_batch.

### Usage

```r
deflate_mtx(X, t)
```

### Arguments

- **X**: A numeric matrix to be deflated. It assumes that samples are on the row, while variables are on the column. NAs are not allowed.
- **t**: A component to be deflated out from the matrix.

### Value

A deflated matrix with the same dimension as the input matrix.

### Author(s)

Yiwen Wang, Kim-Anh Lê Cao

### References

Examples

# A built-in function of PLSDA_batch, not separately used.
# Not run
data('AD_data')
library(mixOmics)
library(TreeSummarizedExperiment)

X <- assays(AD_data$EgData)$Clr_value
ad_pca <- pca(X, ncomp = 3)
# the matrix without the information of PC1:
ad.def.mtx <- deflate_mtx(X, ad_pca$variates$X[,1])

density_plot  
Density Plot

Description
This function draws an overlap of multiple density plots for each batch.

Usage

density_plot(
  df,
  title = NULL,
  batch.legend.title = "Batch",
  xlab = "Value",
  color.set = NULL,
  title.hjust = 0.5
)

Arguments

df  A data frame used to draw the density plots.
title Character, the plot title.
batch.legend.title Character, the legend title of batches.
xlab Character, x-axis title.
color.set A vector of character, indicating the set of colors to use. The colors are represented by hexadecimal color code.
title.hjust Numeric, horizontal justification of the plot title, in the range of 0 to 1.

Value
None.
Author(s)
Yiwen Wang, Kim-Anh Lê Cao

See Also
Scatter_Density, box_plot, alignment_score and partVar_plot as the other methods for batch effect detection and batch effect removal assessment.

Examples

# The first example
library(TreeSummarizedExperiment) # for functions assays(),rowData()
data('AD_data')
# centered log ratio transformed data
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.batch <- rowData(AD_data$EgData)$Y.bat # batch information
names(ad.batch) <- rownames(AD_data$EgData)
ad.df <- data.frame(value = ad.clr[,1], batch = ad.batch)
density_plot(df = ad.df, title = 'OTU 12')

# The second example
colorlist <- rainbow(10)
density_plot(df = ad.df, title = 'OTU 12', color.set = colorlist)

linear_regres Linear Regression

Description
This function fits linear regression (linear model or linear mixed model) on each microbial variable and includes treatment and batch effects as covariates. It generates p-values, adjusted p-values for multiple comparisons, and evaluation metrics of model quality.

Usage

linear_regres(
data,
trt,
batch.fix = NULL,
batch.fix2 = NULL,
batch.random = NULL,
type = "linear model",
p.adjust.method = "fdr"
)
Arguments

data A data frame that contains the response variables for the linear regression. Samples as rows and variables as columns.

trt A factor or a class vector for the treatment grouping information (categorical outcome variable).

batch.fix A factor or a class vector for the batch grouping information (categorical outcome variable), treated as a fixed effect in the model.

batch.fix2 A factor or a class vector for a second batch grouping information (categorical outcome variable), treated as a fixed effect in the model.

batch.random A factor or a class vector for the batch grouping information (categorical outcome variable), treated as a random effect in the model.

type The type of model to be used for fitting, either 'linear model' or 'linear mixed model'.

p.adjust.method The method to be used for p-value adjustment, either "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr" or "none".

Value

linear_regres returns a list that contains the following components:

type The type of model used for fitting.

model Each object fitted.

raw.p The p-values for each response variable.

adj.p The adjusted p-values for each response variable.

p.adjust.method The method used for p-value adjustment.

R2 The proportion of variation in the response variable that is explained by the predictor variables. A higher R2 indicates a better model. Results for 'linear model' only.

adj.R2 Adjusted R2 for many predictor variables in the model. Results for 'linear model' only.

cond.R2 The proportion of variation in the response variable that is explained by the "complete" model with all covariates. Results for 'linear mixed model' only. Similar to R2 in linear model.

marg.R2 The proportion of variation in the response variable that is explained by the fixed effects part only. Results for 'linear mixed model' only.

RMSE The average error performed by the model in predicting the outcome for an observation. A lower RMSE indicates a better model.

RSE also known as the model sigma, is a variant of the RMSE adjusted for the number of predictors in the model. A lower RSE indicates a better model.

AIC A penalisation value for including additional predictor variables to a model. A lower AIC indicates a better model.

BIC is a variant of AIC with a stronger penalty for including additional variables to the model.
Note

R2, adj.R2, cond.R2, marg.R2, RMSE, RSE, AIC, BIC all include the results of two models: (i) the full input model; (ii) a model without batch effects. It can help to decide whether it is better to include batch effects.

Author(s)

Yiwen Wang, Kim-Anh Lê Cao

References


See Also

percentile_norm and PLSDA_batch as the other methods for batch effect management.

Examples

library(TreeSummarizedExperiment) # for functions assays(),rowData()
data('AD_data')

# centered log ratio transformed data
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.batch <- rowData(AD_data$EgData)$Y.bat # batch information
ad.trt <- rowData(AD_data$EgData)$Y.trt # treatment information
names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)
ad.lm <- linear_regres(data = ad.clr, trt = ad.trt,
                         batch.fix = ad.batch,
                         type = 'linear model')
ad.p.adj <- ad.lm$adj.p
head(ad.lm$AIC)
partVar_plot

Usage

partVar_plot(
  prop.df,  # A data frame that contains the proportion of variance explained by different sources.
  text.cex = 3,  # Numeric, the size of text on the plot.
  x.angle = 60,  # Numeric, angle of x axis, in the range of 0 to 360.
  x.hjust = 1,  # Numeric, horizontal justification of x axis, in the range of 0 to 1.
  title = NULL,  # Character, the plot title.
  color.set = NULL
)

Arguments

prop.df  # A data frame that contains the proportion of variance explained by different sources.
text.cex  # Numeric, the size of text on the plot.
x.angle  # Numeric, angle of x axis, in the range of 0 to 360.
x.hjust  # Numeric, horizontal justification of x axis, in the range of 0 to 1.
title  # Character, the plot title.
color.set  # A vector of characters, indicating the set of colors to use. The colors are represented by hexadecimal color code.

Value

None.

Author(s)

Yiwen Wang, Kim-Anh Lê Cao

See Also

Scatter_Density, box_plot, density_plot and alignment_score as the other methods for batch effect detection and batch effect removal assessment.

Examples

```r
## First example
library(vegan)  # for function varpart()
library(TreeSummarizedExperiment)  # for functions assays(),rowData()
data('AD_data')  # centered log ratio transformed data
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.batch <- rowData(AD_data$EgData)$Y.bat # batch information
ad.trt <- rowData(AD_data$EgData)$Y.trt # treatment information
names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)

ad.factors.df <- data.frame(trt = ad.trt, batch = ad.batch)
rda.res <- varpart(ad.clr, ~ trt, ~ batch,  # data = ad.factors.df, scale = TRUE)
```
ad.prop.df <- data.frame(Treatment = NA, Batch = NA,
                      Intersection = NA,
                      Residuals = NA)
ad.prop.df[1,] <- rda.res$part$indfract$Adj.R.squared
ad.prop.df <- ad.prop.df[, c(1,3,2,4)]
ad.prop.df[ad.prop.df < 0] <- 0
ad.prop.df <- as.data.frame(t(apply(ad.prop.df, 1, function(x){x/sum(x)})))
partVar_plot(prop.df = ad.prop.df)

## Second example
# a list of data corrected from different methods
ad.corrected.list <- assays(AD_data$CorrectData)
ad.prop.df <- data.frame(Treatment = NA, Batch = NA,
                      Intersection = NA,
                      Residuals = NA)
for(i in seq_len(length(ad.corrected.list))){
    rda.res <- varpart(ad.corrected.list[[i]], ~ trt, ~ batch,
                      data = ad.factors.df, scale = TRUE)
ad.prop.df[i, ] <- rda.res$part$indfract$Adj.R.squared}
rownames(ad.prop.df) <- names(ad.corrected.list)
ad.prop.df <- ad.prop.df[, c(1,3,2,4)]
ad.prop.df[ad.prop.df < 0] <- 0
ad.prop.df <- as.data.frame(t(apply(ad.prop.df, 1, function(x){x/sum(x)})))
partVar_plot(prop.df = ad.prop.df)

---

pb_color  

### pb_color  

**Color Palette for PLSDA-batch**  

**Description**  
The function outputs a vector of colors.

**Usage**  

```r
pb_color(num.vector)
```

**Arguments**  

- `num.vector`: An integer vector specifying which color to use in the palette (there are only 25 colors available).
percentileofscore

**Value**

A vector of colors (25 colors max.)

**Author(s)**

Yiwen Wang, Kim-Anh Lê Cao

**Examples**

```r
pb_color(seq_len(5))
```

---

<table>
<thead>
<tr>
<th>percentileofscore</th>
<th>Percentile score</th>
</tr>
</thead>
</table>

**Description**

This function converts the relative abundance of microbial variables (i.e. bacterial taxa) in case (i.e. disease) samples to percentiles of the equivalent variables in control (i.e. healthy) samples. It is a built-in function of `percentile_norm`.

**Usage**

```r
percentileofscore(df, control.index)
```

**Arguments**

- `df` A data frame that contains the microbial variables and required to be converted into percentile scores. Samples as rows and variables as columns.
- `control.index` A numeric vector that contains the indexes of control samples.

**Value**

A data frame of percentile scores for each microbial variable and each sample.

**References**

## Examples

```r
# A built-in function of percentile_norm, not separately used.
# Not run
library(TreeSummarizedExperiment)
data('AD_data')
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.batch <- rowData(AD_data$EgData)$Y.bat
ad.trt <- rowData(AD_data$EgData)$Y.trt
names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)
trt.first.b <- ad.trt[ad.batch == '09/04/2015']
ad.first.b.pn <- percentileofscore(ad.clr[ad.batch == '09/04/2015'],
                                 which(trt.first.b == '0-0.5'))
```

---

### percentile_norm

**Percentile Normalisation**

### Description

This function corrects for batch effects in case-control microbiome studies. Briefly, the relative abundance of microbial variables (i.e. bacterial taxa) in case (i.e. disease) samples are converted to percentiles of the equivalent variables in control (i.e. healthy) samples within a batch prior to pooling data across batches. Pooled batches must have similar case and control cohort definitions.

### Usage

```r
percentile_norm(data = data, batch = batch, trt = trt, ctrl.grp)
```

### Arguments

- **data**: A data frame that contains the microbial variables and required to be corrected for batch effects. Samples as rows and variables as columns.
- **batch**: A factor or a class vector for the batch grouping information (categorical outcome variable).
- **trt**: A factor or a class vector for the treatment grouping information (categorical outcome variable).
- **ctrl.grp**: Character, the name of control samples (i.e. healthy).

### Value

A data frame that corrected for batch effects.

### Author(s)

Yiwen Wang, Kim-Anh Lê Cao
### References

### See Also
- `linear_regres` and `PLSDA_batch` as the other methods for batch effect management.

### Examples
```r
library(TreeSummarizedExperiment)  # for functions assays(),rowData()
data('AD_data')

# centered log ratio transformed data
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.batch <- rowData(AD_data$EgData)$Y.bat # batch information
ad.trt <- rowData(AD_data$EgData)$Y.trt # treatment information
names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)
ad.PN <- percentile_norm(data = ad.clr, batch = ad.batch,
                         trt = ad.trt, ctrl.grp = '0-0.5')
```

---

### PLSDA

**Partial Least Squares Discriminant Analysis**

### Description
This function estimates latent dimensions from the explanatory matrix $X$. The latent dimensions are maximally associated with the outcome matrix $Y$. It is a built-in function of `PLSDA_batch`.

### Usage
```r
PLSDA(X, Y, ncomp, keepX = rep(ncol(X), ncomp), tol = 1e-06, max.iter = 500)
```

### Arguments
- **X**: A numeric matrix that is centered and scaled as an explanatory matrix. NAs are not allowed.
- **Y**: A dummy matrix that is centered and scaled as an outcome matrix.
- **ncomp**: Integer, the number of dimensions to include in the model.
- **keepX**: A numeric vector of length `ncomp`, the number of variables to keep in `X`-loadings. By default all variables are kept in the model. A valid input of `keepX` extends `PLSDA` to a sparse version.
- **tol**: Numeric, convergence stopping value.
- **max.iter**: Integer, the maximum number of iterations.
Value

PLSDA returns a list that contains the following components:

- `original_data`: The original explanatory matrix $X$ and outcome matrix $Y$.
- `defl_data`: The centered and scaled deflated matrices ($\hat{X}$ and $\hat{Y}$) after removing the variance of latent components calculated with estimated latent dimensions.
- `latent_comp`: The latent components calculated with estimated latent dimensions.
- `loadings`: The estimated latent dimensions.
- `iters`: Number of iterations of the algorithm for each component.
- `exp_var`: The amount of data variance explained per component (note that contrary to PCA, this amount may not decrease as the aim of the method is not to maximise the variance, but the covariance between $X$ and the dummy matrix $Y$).

Author(s)

Yiwen Wang, Kim-Anh Lê Cao

References


Examples

```r
# A built-in function of PLSDA_batch, not separately used.
# Not run
data('AD_data')
library(mixOmics)
library(TreeSummarizedExperiment)

X <- assays(AD_data$EgData)$Clr_value
Y.trt <- rowData(AD_data$EgData)$Y.trt
names(Y.trt) <- rownames(AD_data$EgData)
X.scale <- scale(X, center = TRUE, scale = TRUE)
Y.trt.mat <- unmap(as.numeric(Y.trt))
Y.trt.scale <- scale(Y.trt.mat, center = TRUE, scale = TRUE)
ad_plsda.trt <- PLSDA(X.scale, Y.trt.scale, ncomp = 1)
X.compnt <- ad_plsda.trt$latent_comp$t
```
PLSDA_batch

Partial Least Squares Discriminant Analysis for Batch Effect Correction

Description

This function removes batch variation from the input data given the batch grouping information and the number of associated components with PLSDA-batch. For sparse PLSDA-batch, the number of variables to keep for each treatment related component is needed (keepX.trt). For weighted PLSDA-batch, the balance should be set to FALSE, and it cannot deal with the nested batch x treatment design.

Usage

PLSDA_batch(
  X,
  Y.trt = NULL,
  Y.bat,
  ncomp.trt = 2,
  ncomp.bat = 2,
  keepX.trt = rep(ncol(X), ncomp.trt),
  keepX.bat = rep(ncol(X), ncomp.bat),
  max.iter = 500,
  tol = 1e-06,
  near.zero.var = TRUE,
  balance = TRUE
)

Arguments

X  A numeric matrix as an explanatory matrix. NAs are not allowed.
Y.trt  A factor or a class vector for the treatment grouping information (categorical outcome variable). Without the input of Y.trt, treatment variation cannot be preserved before correcting for batch effects.
Y.bat  A factor or a class vector for the batch grouping information (categorical outcome variable).
ncomp.trt  Integer, the number of treatment associated dimensions to include in the model.
ncomp.bat  Integer, the number of batch associated dimensions to include in the model.
keepX.trt  A numeric vector of length ncomp.trt, the number of variables to keep in X-loadings. By default all variables are kept in the model. A valid input of keepX.trt extends PLSDA-batch to a sparse version.
keepX.bat  A numeric vector of length ncomp.bat, the number of variables to keep in X-loadings. By default all variables are kept in the model. We usually use the default setting.
max.iter  Integer, the maximum number of iterations.
PLSDA\_batch returns a list that contains the following components:

- **X**
  The original explanatory matrix $X$.

- **X.nobatch**
  The batch corrected matrix with the same dimension as the input matrix.

- **X.notrt**
  The matrix from which treatment variation is removed.

- **Y**
  The original outcome variables $Y.trt$ and $Y.bat$.

- **latent\_var.trt**
  The treatment associated latent components calculated with corresponding latent dimensions.

- **latent\_var.bat**
  The batch associated latent components calculated with corresponding latent dimensions.

- **loadings.trt**
  The estimated treatment associated latent dimensions.

- **loadings.bat**
  The estimated batch associated latent dimensions.

- **tol**
  The tolerance used in the iterative algorithm, convergence stopping value.

- **max.iter**
  The maximum number of iterations.

- **iter.trt**
  Number of iterations of the algorithm for each treatment associated component.

- **iter.bat**
  Number of iterations of the algorithm for each batch associated component.

- **explained\_variance.trt**
  The amount of data variance explained per treatment associated component.

- **explained\_variance.bat**
  The amount of data variance explained per batch associated component.

- **weight**
  The sample weights, all 1 for a balanced batch $\times$ treatment design.

**Value**

**Author(s)**

Yiwen Wang, Kim-Anh Lê Cao

**References**


See Also

linear_regres and percentile_norm as the other methods for batch effect management.

Examples

```r
## First example
## PLSDA-batch
library(TreeSummarizedExperiment) # for functions assays(),rowData()
data('AD_data')
X <- assays(AD_data$EgData)$Clr_value # centered log ratio transformed data
Y.trt <- rowData(AD_data$EgData)$Y.trt # treatment information
Y.bat <- rowData(AD_data$EgData)$Y.bat # batch information
names(Y.bat) <- names(Y.trt) <- rownames(AD_data$EgData)
ad_plsda_batch <- PLSDA_batch(X, Y.trt, Y.bat, ncomp.trt = 1, ncomp.bat = 5)
ad_X.corrected <- ad_plsda_batch$X.nobatch # batch corrected data

## Second example
## sparse PLSDA-batch
ad_splsda_batch <- PLSDA_batch(X, Y.trt, Y.bat, ncomp.trt = 1, keepX.trt = 30, ncomp.bat = 5)
```

---

**PreFL**

Prefiltering for Microbiome Data

**Description**

This function prefilters the data to remove samples or microbial variables with excess zeroes.

**Usage**

`PreFL(data, keep.spl = 10, keep.var = 0.01)`

**Arguments**

- `data`: The data to be prefiltered. The samples in rows and variables in columns.
- `keep.spl`: The minimum counts of a sample to be kept.
- `keep.var`: The minimum proportion of counts of a variable to be kept.

**Value**

`PreFL` returns a list that contains the following components:

- `data.filter`: The filtered data matrix.
- `sample.idx`: The indexes of samples kept.
- `var.idx`: The indexes of variables kept.
- `zero.prob`: The proportion of zeros of the input data.
Author(s)
Yiwen Wang, Kim-Anh Lê Cao

References

Examples

```r
library(TreeSummarizedExperiment) # for functions assays()
data('AD_data')
ad.count <- assays(AD_data$FullData)$Count # microbial count data
ad.filter.res <- PreFL(data = ad.count)

# The proportion of zeroes of the AD count data
ad.zero.prob <- ad.filter.res$zero.prob

# The filtered AD count data
ad.filter <- ad.filter.res$data.filter
```

---

**Scatter_Density**

*Principal Component Analysis (PCA) with Density Plots per Component*

**Description**

This function draws a PCA sample plot with density plots per principal component.

**Usage**

```r
Scatter_Density(
  object,
  batch = NULL,
  trt = NULL,
  xlim = NULL,
  ylim = NULL,
  color.set = NULL,
  batch.legend.title = "Batch",
  trt.legend.title = "Treatment",
  density.lwd = 0.2,
  title = NULL,
  title.cex = 1.5,
  legend.cex = 0.7,
)```
legend.title.cex = 0.75

Arguments

object
The object of class PCA.

batch
A factor or a class vector for the batch grouping information (categorical outcome variable).

trt
A factor or a class vector for the treatment grouping information (categorical outcome variable).

xlim
A numeric vector of length 2, indicating the x coordinate ranges.

ylim
A numeric vector of length 2, indicating the y coordinate ranges.

color.set
A vector of character, indicating the set of colors to use. The colors are represented by hexadecimal color code.

batch.legend.title
Character, the legend title of batches.

trt.legend.title
Character, the legend title of treatments.

density.lwd
Numeric, the thickness of density lines.

title
Character, the plot title.

title.cex
Numeric, the size of plot title.

legend.cex
Numeric, the size of legends.

legend.title.cex
Numeric, the size of legend title.

Value
None.

Author(s)
Yiwen Wang, Kim-Anh Lê Cao

See Also

box_plot, density_plot, alignment_score and partVar_plot as the other methods for batch effect detection and batch effect removal assessment.

Examples

# The first example
library(mixOmics) # for function pca()
library(TreeSummarizedExperiment) # for functions assays(),rowData()
data('AD_data')
# centered log ratio transformed data
ad.clr <- assays(AD_data$EgData)$Clr_value
ad.pca.before <- pca(ad.clr, ncomp = 3, scale = TRUE)
ad.batch <- rowData(AD_data$EgData)$Y.bat # batch information  
ad.trt <- rowData(AD_data$EgData)$Y.trt # treatment information  
names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)  
Scatter_Density(object = ad.pca.before, batch = ad.batch, trt = ad.trt)  

# The second example  
colorlist <- rainbow(10)  
Scatter_Density(object = ad.pca.before, batch = ad.batch, trt = ad.trt,  
               color.set = colorlist)

---

**sponge_data**

**Sponge A. aerophoba study**

**Description**

This study investigated the relationship between metabolite concentration and microbial abundance of specific sponge tissues. The samples were collected from two types of tissues (Ectosome vs. Choanosome) and processed on two separate denaturing gradient gels in electrophoresis. This study includes relative abundance data only and a completely balanced batch x treatment design.

**Usage**

```r
data('sponge_data')
```

**Format**

A TreeSummarizedExperiment object containing the relative abundance (Tss_value) and centered log ratio transformed values (Clr_value) of 32 samples and 24 OTUs. The rowData includes Y.trt and Y.bat. Y.trt is the effect of interest, which is a factor of sponge tissues for each sample in the sponge study; Y.bat is the batch effect, which is a factor of electrophoresis gels where each sample processed. The rowTree is built based on the Y.bat.

**Value**

None.

**Source**

The raw data were downloaded from the referenced article. Filtering and normalisation described in [https://evayiwenwang.github.io/PLSDAbatch_workflow/](https://evayiwenwang.github.io/PLSDAbatch_workflow/).

**References**

Index

* Internal
  deflate_mtx, 6
  percentileofscore, 13
  PLSDA, 15
* datasets
  AD_data, 2
  sponge_data, 22

AD_data, 2
alignment_score, 3, 5, 8, 11, 21

box_plot, 4, 5, 8, 11, 21

deflate_mtx, 6
density_plot, 4, 5, 7, 11, 21

linear_regres, 8, 15, 19

partVar_plot, 4, 5, 8, 10, 21
pb_color, 12
percentile_norm, 10, 14, 19
percentileofscore, 13
PLSDA, 15
PLSDA_batch, 10, 15, 17
PreFL, 19

Scatter_Density, 4, 5, 8, 11, 20
sponge_data, 22