Package ‘PLSDAbatch’

January 24, 2026

Type Package

Title PLSDA-batch

Version 1.99.1

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 (>= 4.5.0)

Imports ggplot2, ggpubr, grid, gridExtra, lmerTest, mixOmics, performance, scales, stats, Rdpack

Suggests SummarizedExperiment, TreeSummarizedExperiment, vegan, knitr, rmarkdown, BiocStyle, testthat, badger, pheatmap, Biobase

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

VignetteBuilder knitr

RdMacros Rdpack

RoxygenNote 7.3.3

Encoding UTF-8

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

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

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

git_branch devel

git_last_commit f1c3e0c

git_last_commit_date 2026-01-06

Repository Bioconductor 3.23

Date/Publication 2026-01-23

Author Yiwen (Eva) Wang [aut, cre] (ORCID:
<https://orcid.org/0000-0002-7067-9093>),
Kim-Anh Le Cao [aut]

Maintainer Yiwen (Eva) Wang <anjiwangyiwen@gmail.com>

Contents

`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')
```

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

Chapleur O, Madigou C, Civade R, Rodolphe Y, Mazéas L, Bouchez T (2016). “Increasing concentrations of phenol progressively affect anaerobic digestion of cellulose and associated microbial communities.” *Biodegradation*, **27**(1), 15–27.

`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 = NULL, ncomp = 20)
```

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 Le Cao

References

Butler A, Hoffman P, Smibert P, Papalexi E, Satija R (2018). "Integrating single-cell transcriptomic data across different conditions, technologies, and species." *Nature biotechnology*, **36**(5), 411–420.

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

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("sponge_data")
  X <- SummarizedExperiment::assays(sponge_data)$Clr_value # centered log ratio transformed data
    batch <- SummarizedExperiment::rowData(sponge_data)$Y.bat # batch information
    names(batch) <- rownames(sponge_data)

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

box_plot *Box Plot by Batch*

Description

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

Usage

```
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 with at least two columns. The first column contains the numeric values to be plotted on the y-axis, and the second column contains the batch information (categorical). Additional columns, if present, are ignored.
`title`	Character, the plot title.
`batch.legend.title`	Character, the legend title for the batch groups.
`ylab`	Character, the y-axis title.
`color.set`	A character vector specifying the colours to use for the batch groups. Colours can be given as hexadecimal codes or any values understood by `ggplot2`. If `NULL`, a default palette based on `pb_color()` is used. If the length is shorter than the number of batch levels, it will be recycled with a warning.
`x.angle`	Numeric, angle of the x-axis labels in degrees, in the range from 0 to 360.
`x.hjust`	Numeric, horizontal justification of the x-axis labels, in the range from 0 to 1.
`x.vjust`	Numeric, vertical justification of the x-axis labels, in the range from 0 to 1.

Value

A `ggplot` object representing the box plots.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

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

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("AD_data")

    # centered log-ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat
    names(ad.batch) <- rownames(AD_data$EgData)

    ad.df <- data.frame(
        value = ad.clr[, 1],
        batch = ad.batch
    )

    box_plot(df = ad.df, title = "OTU 12", x.angle = 30)

    # using a custom colour set
    colorlist <- rainbow(10)
    box_plot(
        df = ad.df, title = "OTU 12",
        color.set = colorlist, x.angle = 30
    )
}
```

darken *Darken Colors by Decreasing Brightness*

Description

This function takes one or more colors and decreases their brightness by subtracting a constant amount from their RGB values. The adjustment is performed in the RGB space, and the resulting values are truncated to stay within `[0, 1]`. Alpha channels are not preserved and the returned colors are fully opaque.

Usage

```
darken(colvec, amount = 0.15)
```

Arguments

colvec	A vector of colors specified as character strings (for example, hexadecimal colors or any color name recognized by R).
amount	A scalar numeric value indicating how much brightness to subtract. Must be between `0` and 1. Default is `0.15`.

Value

A character vector of darkened colors in hexadecimal RGB format.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

lighten, pb_color

Examples

```
darken("#336699")
darken(c("red", "blue"), amount = 0.1)
darken(pb_color(seq_len(3)), amount = 0.2)
```

deflate_mtx *Matrix Deflation*

Description

This function removes the variance of a given component comp from the input matrix X.

$$\hat{X} = X - comp(comp^\top comp)^{-1} comp^\top X$$

It is mainly used internally in PLSDA_batch.

Usage

```
deflate_mtx(X, comp)
```

Arguments

X	A numeric matrix to be deflated. It assumes that samples are on the rows and variables are on the columns. NAs are not allowed.
comp	A numeric vector or single-column matrix representing the 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 Le Cao

References

Barker M, Rayens W (2003). “Partial least squares for discrimination.” *Journal of Chemometrics: A Journal of the Chemometrics Society*, **17**(3), 166–173.

Examples

```
NULL
```

`density_plot` *Density Plot by Batch*

Description

This function draws overlapping 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 with at least two columns. The first column contains the numeric values to be plotted on the x-axis, and the second column contains the batch information (categorical). Additional columns, if present, are ignored.

`title` Character, the plot title.

`batch.legend.title` Character, the legend title for the batch groups.

`xlab` Character, the x-axis title.

`color.set` A character vector specifying the colours to use for the batch groups. Colours can be given as hexadecimal codes or any values understood by `ggplot2`. If `NULL`, a default palette based on `pb_color()` is used. If the length is shorter than the number of batch levels, it will be recycled with a warning.

`title.hjust` Numeric, horizontal justification of the plot title, in the range from 0 to 1.

Value

A `ggplot` object representing the density plots.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

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

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("AD_data")

    # centered log-ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat
    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")

    # using a custom colour set
    colorlist <- rainbow(10)
    density_plot(
        df = ad.df, title = "OTU 12",
        color.set = colorlist
    )
}
```

lighten *Lighten Colors by Increasing Brightness*

Description

This function takes one or more colors and increases their brightness by adding a constant amount to their RGB values. The adjustment is performed in the RGB space, and the resulting values are truncated to stay within [0, 1]. Alpha channels are not preserved and the returned colors are fully opaque.

Usage

```
lighten(colvec, amount = 0.15)
```

Arguments

colvec — A vector of colors specified as character strings (for example, hexadecimal colors or any color name recognized by R).

amount — A scalar numeric value indicating how much brightness to add. Must be between `0` and `1`. Default is `0.15`.

Value

A character vector of lightened colors in hexadecimal RGB format.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

darken, pb_color

Examples

```
lighten("#336699")
lighten(c("red", "blue"), amount = 0.1)
lighten(pb_color(seq_len(3)), amount = 0.2)
```

linear_regres *Linear Regression*

Description

This function fits linear regression models (either a linear model or a linear mixed model) to each microbial variable, including treatment and batch covariates as specified. For each variable, two models are fitted: (i) a model including only the treatment effect (`trt.only`); (ii) a model including both treatment and batch effects (`trt.batch`). A selected criterion (e.g., AIC, BIC, RMSE, R2) is used to choose the better model for each variable, and only the p-value of the selected model is returned.

Usage

```
linear_regres(
  data,
  trt,
  batch.fix = NULL,
  batch.fix2 = NULL,
  batch.random = NULL,
  type = "linear model",
  p.adjust.method = "fdr",
  criterion = "AIC",
  return.model = TRUE
)
```

Arguments

`data` A data frame containing microbial variables. Rows correspond to samples and columns to features.

`trt` A factor or class vector representing the treatment groups. This argument is mandatory and is coerced to a factor internally. The p-values correspond to the global treatment effect extracted from `anova()`.

`batch.fix` A factor or vector representing a batch effect treated as a fixed effect. Required when `type = "linear model"`.

`batch.fix2` A second fixed batch effect. Can only be used when `batch.fix` is provided.

`batch.random` A factor or vector representing a batch effect treated as a random effect. Required when `type = "linear mixed model"`.

`type` Either `"linear model"` or `"linear mixed model"`.

`p.adjust.method`
Method for p-value adjustment. One of: `"holm"`, `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, or `"none"`.

`criterion` A character string indicating the model selection criterion used to choose between the treatment-only model and the treatment+batch model for each microbial variable. One of `"R2"`, `"RMSE"`, `"RSE"`, `"AIC"` or `"BIC"`. When `criterion = "R2"`:

- For `type = "linear model"`, the comparison is based on the **adjusted R2** of the two models (treatment-only versus treatment+batch).
- For `type = "linear mixed model"`, the comparison is based on the **conditional R2** of the mixed model versus the R2 of the corresponding treatment-only linear model.

A larger R2 indicates a better model. For all other criteria (`"RMSE"`, `"RSE"`, `"AIC"`, `"BIC"`), smaller values indicate a better model.

`return.model` Logical. If `TRUE`, fitted model objects (`lm` or `lmerMod`) are returned. If `FALSE`, model objects are replaced with `NULL` to save memory.

Value

A list containing:

`type` The type of model used ("linear model" or "linear mixed model").

`model` A list of fitted `lm` or `lmerMod` objects, or `NULL` if `return.model = FALSE`.

`raw.p` A vector of p-values corresponding to the selected model (based on `criterion`) for each microbial variable.

`adj.p` Adjusted p-values.

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

`criterion` The criterion used to select between the two models.

`best.model` For each feature, either `"trt.only"` or `"trt.batch"`, indicating which model was selected.

`raw.R2` For `type = "linear model"`, the R2 values for the treatment-only and treatment+batch models. For `type = "linear mixed model"`, this field contains `NA`.

`adj.R2` Adjusted R2 values (linear model only). For mixed models, this field contains `NA`.

`cond.R2` Conditional R2 for mixed models: first column corresponds to the treatment-only linear model, second column to the mixed model including batch.random. For linear models, this field is `NA`.

`RMSE` The root mean squared error for both models (two columns: `trt.only` and `trt.batch`).

`RSE` Residual standard error for both models.

`AIC` AIC values for both models.

`BIC` BIC values for both models.

Note

For each microbial variable, two models are always fitted:

1. `trt.only`: `y ~ trt`
2. `trt.batch`: `y ~ trt + batch` (or with random effects)

The selected model is determined solely by `criterion`. Only the p-value corresponding to the selected model is returned in `raw.p`.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

`percentile_norm`, `PLSDA_batch`

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("AD_data")

    # centered log ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat # batch information
    ad.trt <- SummarizedExperiment::rowData(AD_data$EgData)$Y.trt # treatment information
    names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)
    ad.lm <- linear_regres(
        data = ad.clr, trt = ad.trt,
        batch.fix = ad.batch,
        type = "linear model",
        criterion = "AIC"
    )
    ad.p.adj <- data.frame(best_model = ad.lm$best.model, adjust_p = ad.lm$adj.p)
    head(ad.p.adj)
    head(ad.lm$AIC)
    table(ad.lm$best.model)
}
```

partVar_plot *Partitioned Variance Plot*

Description

This function draws a partitioned variance plot explained by different sources.

Usage

```
partVar_plot(
  prop.df,
  text.cex = 3,
  x.angle = 60,
  x.hjust = 1,
  title = NULL,
  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. Use the size rule of ggplot2::geom_text().
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

A ggplot object.

Author(s)

Yiwen Wang, Kim-Anh Le 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

```
if (requireNamespace("vegan", quietly = TRUE) &&
    requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    ## First example
    data("AD_data")
    # centered log ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat # batch information
    ad.trt <- SummarizedExperiment::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 <- vegan::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)]

    partVar_plot(prop.df = ad.prop.df)

    ## Second example
    # a list of data corrected from different methods
    ad.corrected.list <- SummarizedExperiment::assays(AD_data$CorrectData)
    ad.prop.df <- data.frame(
        Treatment = NA, Batch = NA,
        Intersection = NA,
        Residuals = NA
    )
    for (i in seq_len(length(ad.corrected.list))) {
        rda.res <- vegan::varpart(ad.corrected.list[[i]], ~trt, ~batch,
            data = ad.factors.df, scale = TRUE
        )
        ad.prop.df[i, ] <- rda.res$part$indfract$Adj.R.squared
    }

    rownames(ad.prop.df) <- names(ad.corrected.list)

    ad.prop.df <- ad.prop.df[, c(1, 3, 2, 4)]

    partVar_plot(prop.df = ad.prop.df)
}
```

pb_color *Color Palette for PLSDAbatch Plots*

Description

This function returns a discrete color palette used in **PLSDAbatch** plots. The palette combines hues from `color.mixo` and `hue_pal` and contains at most 25 distinct colors.

Usage

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

Arguments

num.vector An integer vector of indices specifying which colors to extract from the internal palette. Valid values are between 1 and 25.

Value

A character vector of hexadecimal color codes.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

`lighten`, `darken`, `color.mixo`, `hue_pal`

Examples

```
pb_color(seq_len(5))
pb_color(c(1, 3, 5))
```

`percentileofscore` *Percentile score*

Description

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

Usage

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

Arguments

`df`	A data frame or matrix that contains the microbial variables to be converted into percentile scores. Samples are in rows and variables are in columns.
`control.index`	A numeric vector that contains the indices of control samples (row indices of `df`).

Value

A data frame of percentile scores (between 0 and 1) for each microbial variable and each sample.

References

Gibbons SM, Duvallet C, Alm EJ (2018). "Correcting for batch effects in case-control microbiome studies." *PLoS Computational Biology*, **14**(4), e1006102.

Examples

```
NULL
```

`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 is converted to percentiles of the corresponding variables in control (i.e. healthy) samples within each batch, before pooling data across batches. Pooled batches must share comparable case and control cohort definitions.

Usage

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

Arguments

`data` A data frame or matrix that contains the microbial variables to be corrected for batch effects. Samples are in rows and variables are in columns.

`batch` A factor or vector for the batch grouping information (categorical outcome variable). Its length must match the number of rows in `data`.

`trt` A factor or vector for the treatment grouping information (categorical outcome variable). Its length must match the number of rows in `data`.

`ctrl.grp` Character, the label of control samples (i.e. healthy) in `trt`.

Value

A data frame with batch effects corrected by percentile normalisation.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

References

Gibbons SM, Duvallet C, Alm EJ (2018). “Correcting for batch effects in case-control microbiome studies.” *PLoS Computational Biology*, **14**(4), e1006102.

See Also

`linear_regres` and `PLSDA_batch` as other methods for batch effect management.

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("AD_data")

    # centered log ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat # batch information
    ad.trt <- SummarizedExperiment::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

```
PLSDA(
  X,
  Y,
  ncomp = 2,
  keepX = NULL,
  mode = c("regression", "canonical"),
  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.
mode	Character, either "regression" or "canonical". Default mode is 'regression'.
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 algorthm 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 Le Cao

References

Barker M, Rayens W (2003). "Partial least squares for discrimination." *Journal of Chemometrics: A Journal of the Chemometrics Society*, **17**(3), 166–173.

Examples

```
NULL
```

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 = NULL,
  keepX.bat = NULL,
  max.iter = 500,
  tol = 1e-06,
  near.zero.var = TRUE,
  balance = TRUE,
  mode = c("regression", "canonical")
)
```

Arguments

`X` A numeric matrix as an explanatory matrix. `NA`s 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.

`tol` Numeric, convergence stopping value.

`near.zero.var` Logical, should be set to `TRUE` in particular for data with many zero values. Setting this argument to `FALSE` (when appropriate) will speed up the computations. Default value is `TRUE`.

`balance` Logical, should be set to `TRUE`, if the `batch x treatment design` is balanced (or complete). Setting this argument to `FALSE` extends `PLSDA-batch` to `weighted PLSDA-batch`. `wPLSDA-batch` can deal with highly unbalanced designs but not the nested design. Default value is `TRUE`.

`mode` Character, either `"regression"` or `"canonical"`. Default mode is 'regression', which is recommended for most use cases. If you want to reproduce previous results obtained from PLSDAbatch <= 1.6.0, please explicitly set mode = 'canonical'.

Value

`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 algorthm for each treatment associated component.

`iter.bat` Number of iterations of the algorthm 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 x treatment design`.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

References

Wang Y, LêCao K (2020). "Managing batch effects in microbiome data." *Briefings in bioinformatics*, **21**(6), 1954–1970.

Wang Y, Lê Cao K (2023). "PLSDA-batch: a multivariate framework to correct for batch effects in microbiome data." *Briefings in Bioinformatics*, **24**(2), bbac622.

See Also

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

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    ## First example
    ## PLSDA-batch
    data("AD_data")

   X <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value # centered log ratio transformed data
    Y.trt <- SummarizedExperiment::rowData(AD_data$EgData)$Y.trt # treatment information
    Y.bat <- SummarizedExperiment::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 = 4, mode = "regression"
    )
    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 = 100, ncomp.bat = 4,
        mode = "regression"
    )

    ## Third example
    ## weighted PLSDA-batch
    ad_wplsda_batch <- PLSDA_batch(X, Y.trt, Y.bat,
        ncomp.trt = 1,
        ncomp.bat = 4, balance = FALSE,
        mode = "regression"
    )
}
```

PreFL *Prefiltering for Microbiome Count Data*

Description

This function prefilters microbiome count data to remove samples or microbial variables with very low abundance.

Usage

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

Arguments

`data` A numeric matrix or data frame with samples in rows and variables in columns.

`keep.spl` Numeric, the minimum total count of a sample to be kept. Samples with a total count smaller than `keep.spl` are removed. Default is `10`.

`keep.var` Numeric, the minimum percentage (between `0` and `100`) of total counts a variable must contribute to be kept. For example, `keep.var = 0.01` keeps variables that account for at least `0.01%` of the total counts. Default is `0.01`.

Value

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

`data.filter` The filtered data matrix.

`sample.idx` The indices of samples kept.

`var.idx` The indices of variables kept.

`zero.prob.before` The proportion of zeros in the input data.

`zero.prob.after` The proportion of zeros after filtering.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

References

Le Cao K, Costello M, Lakis VA, Bartolo F, Chua X, Brazeilles R, Rondeau P (2016). “MixMC: a multivariate statistical framework to gain insight into microbial communities.” *PloS One*, **11**(8), e0160169.

Examples

```
if (requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    data("AD_data")

    ad.count <- SummarizedExperiment::assays(AD_data$FullData)$Count
    ad.filter.res <- PreFL(data = ad.count)

    ad.filter <- ad.filter.res$data.filter
    ad.zero.before <- ad.filter.res$zero.prob.before
    ad.zero.after <- ad.filter.res$zero.prob.after
}
```

Scatter_Density *Scatter Plot of Components with Marginal Density Plots*

Description

This function draws a sample scatter plot for two selected components, together with marginal density plots along each axis. It is generic in the sense that it only requires a matrix or dataframe of component scores, and can therefore be used with PCA, PLS or any other multivariate method that returns component scores.

Usage

```
Scatter_Density(
  components,
  comp = c(1, 2),
  expl.var = NULL,
  batch = NULL,
  trt = NULL,
  xlim = NULL,
  ylim = NULL,
  color.set = NULL,
  shape.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

`components`	A numeric matrix or data frame containing component scores. Samples are in rows and components in columns.

`comp` Integer vector of length two indicating which components to plot on the x and y axes respectively, for example `c(1, 2)` or `c(1, 3)`.

`expl.var` Optional numeric vector giving the proportion of variance explained by each component. If provided, it should have length at least `max(comp)` and is used to annotate the axis labels.

`batch` Optional factor or vector giving batch grouping information (categorical outcome). If provided, its length must match the number of samples in `components`. When present, points and density plots are coloured by batch.

`trt` Optional factor or vector giving treatment grouping information (categorical outcome). If provided, its length must match the number of samples in `components`. When present, points and density plots are further distinguished by treatment-specific shapes or line types.

`xlim` Optional numeric vector of length two giving the x-axis limits for the scatter plot. If `NULL`, limits are chosen automatically from the data.

`ylim` Optional numeric vector of length two giving the y-axis limits for the scatter plot. If `NULL`, limits are chosen automatically from the data.

`color.set` Optional character vector of colours (hexadecimal codes) used to represent batch levels. If `NULL` and `batch` is provided, an internal palette is used and extended when the number of batches exceeds the base palette size. If `batch` is `NULL`, this argument is ignored.

`shape.set` Optional numeric vector of plotting characters used to represent treatment levels. If `NULL` and `trt` is provided, an internal sequence of hollow and solid shapes is used. If `trt` is `NULL`, this argument is ignored and a fixed shape is used.

`batch.legend.title`
Character string giving the legend title for batch.

`trt.legend.title`
Character string giving the legend title for treatment.

`density.lwd` Numeric value giving the line width for the density curves in the marginal plots.

`title` Character string giving the main title.

`title.cex` Numeric value controlling the relative size of the main title.

`legend.cex` Numeric value controlling the relative size of legend text.

`legend.title.cex`
Numeric value controlling the relative size of legend titles.

Value

A grob object containing the combined scatter and density plots.

Author(s)

Yiwen Wang, Kim-Anh Le Cao

See Also

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

Examples

```
if (requireNamespace("mixOmics", quietly = TRUE) &&
    requireNamespace("SummarizedExperiment", quietly = TRUE)) {
    ## Example using a PCA object from mixOmics

    data("AD_data")

    ## centered log-ratio transformed data
    ad.clr <- SummarizedExperiment::assays(AD_data$EgData)$Clr_value
    ad.pca <- mixOmics::pca(ad.clr, ncomp = 3, scale = TRUE)

    ad.batch <- SummarizedExperiment::rowData(AD_data$EgData)$Y.bat # batch information
   ad.trt <- SummarizedExperiment::rowData(AD_data$EgData)$Y.trt # treatment information
    names(ad.batch) <- names(ad.trt) <- rownames(AD_data$EgData)

    ## components and explained variance extracted from the PCA object
    comp.mat <- ad.pca$variates$X
    expl.var <- ad.pca$prop_expl_var$X

    ## Scatter plot of the first two components
    Scatter_Density(
        components = comp.mat,
        comp = c(1, 2),
        expl.var = expl.var,
        batch = ad.batch,
        trt = ad.trt
    )

    ## Scatter plot of components 1 and 3, with a user-defined colour set
    cols <- rainbow(10)
    Scatter_Density(
        components = comp.mat,
        comp = c(1, 3),
        expl.var = expl.var,
        batch = ad.batch,
        trt = ad.trt,
        color.set = cols
    )
}
```

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

```
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/.

References

Sacristán-Soriano O, Banaigs B, Casamayor EO, Becerro MA (2011). “Exploring the links between natural products and bacterial assemblages in the sponge Aplysina aerophoba.” *Appl. Environ. Microbiol.*, **77**(3), 862–870.

Index