Package 'proBatch'

September 18, 2019

Type Package

Title Tools for Diagnostics and Corrections of Batch Effects in Proteomics

Version 1.0.0

Author Jelena Cuklina <chuklina.jelena@gmail.com>, Chloe H. Lee <chloe.h.lee94@gmail.com>, Patrick Pedrioli <pedrioli@gmail.com>

Maintainer Chloe H. Lee <chloe.h.lee94@gmail.com>

Description The proBatch package facilitates batch effects analysis and correction in high-thoughput experiments. It was developed primarily for mass-spectrometry proteomics (DIA/SWATH), but could also be applicable to most omic data with minor adaptations. The package contains functions for diagnostics (proteome/genome-wide and feature-level), correction (normalization and batch effects correction) and quality control. Non-linear fitting based approaches were also included to deal with complex, mass spectrometry-specific signal drifts.

biocViews BatchEffect, Normalization, Preprocessing, Software, MassSpectrometry,Proteomics, QualityControl, GeneExpression

License GPL-3

URL https://github.com/symbioticMe/proBatch

BugReports https://github.com/symbioticMe/proBatch/issues

Depends R (>= 3.6)

Encoding UTF-8

LazyData true

Imports Biobase, corrplot, dplyr, data.table, ggfortify, ggplot2, grDevices, lazyeval, lubridate, magrittr, pheatmap, preprocessCore, purrr, pvca, RColorBrewer, readr, reshape2, rlang, scales, viridis, stats, sva, tidyr, tibble, utils, wesanderson, WGCNA

Suggests knitr, rmarkdown, devtools, roxygen2, testthat

VignetteBuilder knitr

RoxygenNote 6.1.1

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

git_branch RELEASE_3_9

git_last_commit 6640beb

git_last_commit_date 2019-05-02

Date/Publication 2019-09-17

R topics documented:

`adjust_batch_trend` *adjust batch signal trend with the custom (continuous) fit*

Description

adjust batch signal trend with the custom (continuous) fit

Usage

```
adjust_batch_trend(data_matrix, sample_annotation,
  batch_col = "MS_batch", feature_id_col = "peptide_group_label",
  sample_id_col = "FullRunName", measure_col = "Intensity",
  sample_order_col = "order", fit_func = fit_nonlinear,
  abs_threshold = 5, pct_threshold = 0.2, ...)
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
`sample_annotation`	data frame with sample ID, technical (e.g. MS batches) and biological (e.g. Diet) covariates
`batch_col`	column in `sample_annotation` that should be used for batch comparison
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`measure_col`	if `df_long` is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency
`sample_order_col`	column, determining the order of sample MS run, used as covariate to fit the non-linear fit
`fit_func`	function to fit the (non)-linear trend
`abs_threshold`	the absolute threshold to filter data for curve fitting
`pct_threshold`	the percentage threshold to filter data for curve fitting
`...`	other parameters, usually those of the `fit_func`

Value

list of two items: 1) `data_matrix`, adjusted with continious fit; 2) fit_df, used to examine the fitting curves

See Also

`fit_nonlinear`

Examples

```
trend_corrected_matrix <- adjust_batch_trend(example_proteome_matrix,
example_sample_annotation, span = 0.7,
abs_threshold = 5, pct_threshold = 0.20)
```

`center_peptide_batch_medians` *Median centering of the peptides (per batch median)*

Description

Median centering of the peptides (per batch median)

Usage

```
center_peptide_batch_medians(df_long, sample_annotation = NULL,
  sample_id_col = "FullRunName", batch_col = "MS_batch",
  feature_id_col = "peptide_group_label", measure_col = "Intensity")
```

Arguments

`df_long`	data frame where each row is a single feature in a single sample. It minimally has a `sample_id_col`, a `feature_id_col` and a `measure_col`, but usually also an `m_score` (in OpenSWATH output result file)
`sample_annotation`	data frame with sample ID, technical (e.g. MS batches) and biological (e.g. Diet) covariates
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`batch_col`	column in `sample_annotation` that should be used for batch comparison
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
`measure_col`	if `df_long` is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency

Value

`df_long`-size long format data with batch-effect corrected with per-feature batch median centering in Intensity_normalized column

Examples

```
median_centered_proteome <- center_peptide_batch_medians(
example_proteome, example_sample_annotation)
```

correct_batch_effects *Batch correction method allows correction of continuous sigal drift within batch and discrete difference across batches.*

Description

Batch correction method allows correction of continuous sigal drift within batch and discrete difference across batches.

Usage

```
correct_batch_effects(data_matrix, sample_annotation,
  fitFunc = "loess_regression", discreteFunc = c("MedianCentering",
  "ComBat"), batch_col = "MS_batch",
  feature_id_col = "peptide_group_label",
  sample_id_col = "FullRunName", measure_col = "Intensity",
  sample_order_col = "order", abs_threshold = 5, pct_threshold = 0.2,
  ...)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
sample_annotation	data frame with sample ID, technical (e.g. MS batches) and biological (e.g. Diet) covariates
fitFunc	function to use for the fit (currently only loess_regression available)
discreteFunc	function to use for discrete batch correction (MedianCentering or ComBat)
batch_col	column in sample_annotation that should be used for batch comparison
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency
sample_order_col	column, determining the order of sample MS run, used as covariate to fit the non-linear fit
abs_threshold	the absolute threshold to filter data for curve fitting
pct_threshold	the percentage threshold to filter data for curve fitting
...	other parameters, usually of normalize_custom_fit, and fit_func

Value

data_matrix-size data matrix with batch-effect corrected by fit and discrete functions

Examples

```
batch_corrected_matrix <- correct_batch_effects(
example_proteome_matrix, example_sample_annotation,
discreteFunc = 'MedianCentering',
batch_col = 'MS_batch',
span = 0.7,
abs_threshold = 5, pct_threshold = 0.20)
```

`correct_with_ComBat` *Adjusts for discrete batch effects using ComBat*

Description

Standardized input-output ComBat normalization ComBat allows users to adjust for batch effects in datasets where the batch covariate is known, using methodology described in Johnson et al. 2007. It uses either parametric or non-parametric empirical Bayes frameworks for adjusting data for batch effects. Users are returned an expression matrix that has been corrected for batch effects. The input data are assumed to be cleaned and normalized before batch effect removal.

Usage

```
correct_with_ComBat(data_matrix, sample_annotation,
  sample_id_col = "FullRunName", batch_col = "MS_batch",
  par.prior = TRUE)
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
`sample_annotation`	data frame with sample ID, technical (e.g. MS batches) and biological (e.g. Diet) covariates
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`batch_col`	column in `sample_annotation` that should be used for batch comparison
`par.prior`	whether parametrical or non-parametrical prior should be used

Value

`data_matrix`-size data matrix with batch-effect corrected by `ComBat`

Examples

```
combat_corrected_matrix <- correct_with_ComBat(
example_proteome_matrix, example_sample_annotation)
```

create_peptide_annotation *Prepare peptide annotation from long format data frame Create light-weight peptide annotation data frame for selection of illustrative proteins*

Description

Prepare peptide annotation from long format data frame

Create light-weight peptide annotation data frame for selection of illustrative proteins

Usage

```
create_peptide_annotation(df_long,
  feature_id_col = "peptide_group_label",
  annotation_col = c("ProteinName"))
```

Arguments

df_long	data frame where each row is a single feature in a single sample. It minimally has a sample_id_col, a feature_id_col and a measure_col, but usually also an m_score (in OpenSWATH output result file)
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
annotation_col	one or more columns contatining protein ID

Value

data frame containing petpide annotations

See Also

plot_peptides_of_one_protein, plot_protein_corrplot

Examples

```
generated_peptide_annotation <- create_peptide_annotation(
example_proteome, feature_id_col = "peptide_group_label",
annotation_col = c("ProteinName" ))
```

dates_to_posix *Convert data/time to POSIXct*

Description

convert date/time column of sample_annotation to POSIX format required to keep number-like behaviour

Usage

```
dates_to_posix(sample_annotation, time_column = c("RunDate", "RunTime"),
  new_time_column = "DateTime", dateTimeFormat = c("%b_%d",
  "%H:%M:%S"))
```

Arguments

sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
time_column	name of the column(s) where run date & time are specified. These will be used to determine the run order
new_time_column	name of the new column to which date&time will be converted to
dateTimeFormat	POSIX format of the date and time. See as.POSIXct from base R for details

Value

sample annotation file with a new column new_time_column with POSIX-formatted date

Examples

```
date_to_posix <- dates_to_posix(example_sample_annotation,
time_column = c('RunDate','RunTime'),
new_time_column = 'DateTime',
dateTimeFormat = c("%b_%d", "%H:%M:%S"))
```

date_to_sample_order *Convert date/time to POSIXct and rank samples by it*

Description

Converts date/time columns fo sample_annotation to POSIXct format and calculates sample run rank in order column

Usage

```
date_to_sample_order(sample_annotation, time_column = c("RunDate",
  "RunTime"), new_time_column = "DateTime",
  dateTimeFormat = c("%b_%d", "%H:%M:%S"),
  new_order_col = "order", instrument_col = "instrument")
```

Arguments

sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
time_column	name of the column(s) where run date & time are specified. These will be used to determine the run order
new_time_column	name of the new column to which date&time will be converted to
dateTimeFormat	POSIX format of the date and time. See as.POSIXct from base R for details
new_order_col	name of column with generated the order of sample run based on time columns
instrument_col	column, denoting different instrument used for measurements

Value

sample annotation file with a new column new_time_column with POSIX-formatted date & new_order_col used in some diagnostic plots (e.g. plot_iRT, plot_sample_mean)

Examples

```
sample_annotation_wOrder <- date_to_sample_order(
example_sample_annotation,
time_column = c('RunDate','RunTime'),
new_time_column = 'new_DateTime',
dateTimeFormat = c("%b_%d", "%H:%M:%S"),
new_order_col = 'new_order',
instrument_col = NULL)
```

---

example_peptide_annotation

Peptide annotation data

Description

This is data from Aging study annotated with gene names

Usage

```
example_peptide_annotation
```

Format

A data frame with 535 rows and 10 variables:

peptide_group_label peptide group label ID, identical to `peptide_group_label` in `example_proteome`

Gene HUGO gene ID

ProteinName protein group name as specified in `example_proteome`

`example_proteome` *Example protein data in long format*

Description

This is data from Aging study with all iRT, spike-in peptides, few random peptides and QTL proteins for biological signal improvement demonstration

Usage

```
example_proteome
```

Format

A data frame with 124655 rows and 5 variables:

peptide_group_label peptide ID, which is regular feature level. This column is mostly used as `feature_id_col`

Intensity peptide group intensity in given sample. Used in function as `measure_col`

ProteinName Protein group ID, specified as N/UniProtID1|UniProtID2|..., where N is number of protein peptide group maps to. If 1/UniProtID, then this is proteotypic peptide

Gene conventional gene name of corresponding ProteinName

FullRunName name of the file, in most functions used for `sample_id_col` ...

Source

PRIDE ID will be added in future

`example_proteome_matrix` *Example protein data in matrix*

Description

This is measurement data from Aging study with columns representing samples and rows representing peptides

Usage

```
example_proteome_matrix
```

Format

A matrix with 534 rows and 233 columns:

Source

PRIDE ID will be added in future

`example_sample_annotation`

Sample annotation data version 1

Description

This is data from BXD aging study with mock instruments to show how instrument-specific functionality works

Usage

```
example_sample_annotation
```

Format

A data frame with 233 rows and 11 variables:

FullRunName name of the file, in most functions used for `sample_id_col`

MS_batch mass-spectrometry batch: 7-level factor of manually annotated batches

EarTag mouse ID, i.e. ID of the biological object

Strain mouse strain ID - biological covariate #1

Diet diet - either `HFD` = 'High Fat Diet' or `CD` = 'Chow Diet'. `Mix` stands for mixture of several samples

Sex mice sex - 3-level biological covariate. Possible values - ''

RunDate mass-spectrometry running date. In combination with `RunTime` used for running order determination

RunTime mass-spectrometry running time. In combination with `RunDate` used for running order determination

DateTime numeric date and time generated by `date_to_sample_order`

order order of samples generated by sorting DateTime in `date_to_sample_order`

digestion_batch peptide digestion batch: 5-level factor of manually annotated batches ...

log_transform — *Log transformation of the data, ensuring that the row and column names are retained*

Description

Log transformation of the data, ensuring that the row and column names are retained

Usage

```
log_transform(data_matrix, log_base = 2)
```

Arguments

data_matrix	raw data matrix (features in rows and samples in columns)
log_base	base of the logarithm for transformation

Value

data_matrix-size matrix, with columns log2 transformed

Examples

```
log_transformed_matrix <- log_transform(example_proteome_matrix)
```

long_to_matrix — *Long to wide conversion*

Description

Convert from a long data frame representation to a wide matrix representation

Usage

```
long_to_matrix(df_long, feature_id_col = "peptide_group_label",
  measure_col = "Intensity", sample_id_col = "FullRunName")
```

Arguments

df_long	data frame where each row is a single feature in a single sample. It minimally has a sample_id_col, a feature_id_col and a measure_col, but usually also an m_score (in OpenSWATH output result file)
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)

Value

data_matrix (proBatch) like matrix (features in rows, samples in columns)

See Also

Other matrix manipulation functions: matrix_to_long

Examples

```
proteome_matrix <- long_to_matrix(example_proteome)
```

---

matrix_to_long *Wide to long conversion*

Description

Convert from wide matrix to a long data frame representation

Usage

```
matrix_to_long(data_matrix, sample_annotation = NULL,
  feature_id_col = "peptide_group_label", measure_col = "Intensity",
  sample_id_col = "FullRunName", step = NULL)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
step	normalization step (e.g. Raw or Quantile_normalized or qNorm_ComBat). Useful if consecutive steps are compared in plots. Note that in plots these are usually ordered alphabetically, so it's worth naming with numbers, e.g. 1_raw, 2_quantile

Value

df_long (proBatch) like data frame

See Also

Other matrix manipulation functions: long_to_matrix

Examples

```
proteome_long <- matrix_to_long(example_proteome_matrix,
example_sample_annotation)
```

`normalize` *Data normalization methods*

Description

Data normalization methods

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
measure_col	if 'df_long' is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency

---

`normalize_data` *Normalization brings the samples to the same scale*

Description

Normalization brings the samples to the same scale

Usage

```
normalize_data(data_matrix, normalizeFunc = c("quantile",
  "medianCentering"), log_base = NULL)
```

Arguments

data_matrix	raw data matrix (features in rows and samples in columns)
normalizeFunc	global batch normalization method ('quantile' or 'MedianCentering')
log_base	whether to log transform data matrix before normalization ('NULL', '2' or '10')

Value

data_matrix-size matrix, with columns normalized

Examples

```
quantile_normalized_matrix <- normalize_data(example_proteome_matrix,
normalizeFunc = "quantile", log_base = 2)
```

`normalize_sample_medians` *Normalization by centering sample medians to global median of the data*

Description

Normalization by centering sample medians to global median of the data

Usage

```
normalize_sample_medians(df_long, sample_id_col = "FullRunName",
  measure_col = "Intensity")
```

Arguments

`df_long`	log transformed long format data matrix (see 'df_long')
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`measure_col`	if 'df_long' is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency

Value

'df_long'-size matrix, with intensity scaled to global median

Examples

```
median_normalized_matrix <- normalize_sample_medians(example_proteome)
```

`plot_heatmap` *Plot the heatmap of samples*

Description

Plot the heatmap of samples

Usage

```
plot_heatmap(data_matrix, sample_annotation = NULL,
  sample_id_col = "FullRunName", sample_annotation_col = NULL,
  sample_annotation_row = NULL, fill_the_missing = TRUE,
  cluster_rows = TRUE, cluster_cols = FALSE,
  annotation_color_list = NA,
  heatmap_color = colorRampPalette(rev(RColorBrewer::brewer.pal(n = 7,
  name = "RdYlBu")))(100), color_for_missing = "black", filename = NA,
  plot_title = NA, ...)
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. in most function, it is assumed that this is the log transformed version of the original data
`sample_annotation`	data matrix with 1. `sample_id_col` (this can be repeated as row names) 2. biological and 3. technical covariates (batches etc) ; each column of sample annotation will get it's own row. If `cluster_cols = T` this will indicate, whether sample proximity is driven by one of biolical or technical factors
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`sample_annotation_col`	biological or technical factors to be annotated in heatmap columns
`sample_annotation_row`	biological or technical factors to be annotated in heatmap rows
`fill_the_missing`	boolean value determining if missing values should be substituted with -1 (and colored with black)
`cluster_rows`	boolean value determining if rows should be clustered
`cluster_cols`	boolean value determining if columns should be clustered
`annotation_color_list`	list specifying colors for columns (samples). Best created by `sample_annotation_to_colors`
`heatmap_color`	vector of colors used in heatmap (typicall a gradient)
`color_for_missing`	special color to make missing values. Usually black or white, depending on `heatmap_color`
`filename`	filepath where to save the image
`plot_title`	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))
`...`	other parameters of `link[pheatmap]{pheatmap}`

Value

object returned by `link[pheatmap]{pheatmap}`

See Also

sample_annotation_to_colors, pheatmap

Examples

```
color_scheme <- sample_annotation_to_colors (example_sample_annotation,
factor_columns = c('MS_batch','EarTag', "Strain",
"Diet", "digestion_batch", "Sex"),
not_factor_columns = 'DateTime',
numeric_columns = c('order'))

heatmap_plot <- plot_heatmap(example_proteome_matrix,
example_sample_annotation,
sample_annotation_col = c("MS_batch",  "digestion_batch", "Diet"),
cluster_cols = TRUE,
annotation_color_list = color_scheme$list_of_colors,
show_rownames = FALSE, show_colnames = FALSE)
```

---

plot_hierarchical_clustering

cluster the data matrix to visually inspect which confounder dominates

---

Description

cluster the data matrix to visually inspect which confounder dominates

Usage

```
plot_hierarchical_clustering(data_matrix, color_df,
  distance = "euclidean", agglomeration = "complete",
  label_samples = TRUE, label_font = 0.2, plot_title = NULL, ...)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. in most function, it is assumed that this is the log transformed version of the original data
color_df	data frame of colors, as created by sample_annotation_to_colors
distance	distance metric used for clustering
agglomeration	agglomeration methods as used by hclust
label_samples	if TRUE sample IDs (column names of data_matrix) will be printed
label_font	size of the font. Is active if label_samples is TRUE, ignored otherwise
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))
...	other parameters of plotDendroAndColors from WGCNA package

Value

No return

See Also

hclust, sample_annotation_to_colors, plotDendroAndColors

Examples

```
color_scheme <- sample_annotation_to_colors (example_sample_annotation,
factor_columns = c('MS_batch','EarTag', "Strain", "Diet", "digestion_batch", "Sex"),
not_factor_columns = 'DateTime',
numeric_columns = c('order'))

color_annotation <- color_scheme$color_df

hiarchical_clustering_plot <- plot_hierarchical_clustering(
example_proteome_matrix, color_annotation,
distance = "euclidean", agglomeration = 'complete',
label_samples = FALSE)
```

---

plot_iRT *Plot iRT measurements*

Description

Creates a iRT facetted ggplot2 plot of the value in `measure_col` vs `order_col` using `plot_single_feature`. Additionally, the resulting plot can also be facetted by batch.

Usage

```
plot_iRT(df_long, sample_annotation, peptide_annotation = NULL,
  protein_col = "ProteinName", order_col = "order",
  irt_pattern = "iRT", sample_id_col = "FullRunName",
  batch_col = "MS_batch", measure_col = "Intensity",
  feature_id_col = "peptide_group_label", color_by_batch = FALSE,
  color_scheme = "brewer", facet_by_batch = FALSE,
  color_by_col = NULL, color_by_value = NULL,
  plot_title = "iRT peptide profile", ...)
```

Arguments

`df_long` data frame where each row is a single feature in a single sample. It minimally has a `sample_id_col`, a `feature_id_col` and a `measure_col`, but usually also an `m_score` (in OpenSWATH output result file)

`sample_annotation` data matrix with:

1. `sample_id_col` (this can be repeated as row names)
2. biological covariates
3. technical covariates (batches etc)

`peptide_annotation` long format data with peptide ID and their corresponding protein annotations

`protein_col` column where protein names are specified

order_col	column in sample_annotation that determines sample order. It is used for certain diagnostics and normalisations.
irt_pattern	substring used to identify irts proteins in the column 'ProteinName'
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
batch_col	column in sample_annotation that should be used for batch comparison
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
color_by_batch	(logical) whether to color points by batch
color_scheme	color scheme for ggplot representation
facet_by_batch	(logical) whether to plot each batch in its own facet
color_by_col	column to color by certain value denoted by color_by_value
color_by_value	value in color_by_col to color
plot_title	the string indicating the source of the peptides
...	additional arguments to plot_single_feature function

Value

ggplot2 type plot of measure_col vs order_col, faceted by irt_pattern containing proteins and (optionally) by batch_col

See Also

Other feature-level diagnostic functions: plot_peptides_of_one_protein, plot_single_feature, plot_spike_in, plot_with_fitting_curve

Examples

```
irt_plot <- plot_iRT(example_proteome,
example_sample_annotation,
protein_col = 'Gene', irt_pattern = "BOVINE_A1ag")
```

---

plot_PCA *plot PCA plot*

Description

plot PCA plot

Usage

```
plot_PCA(data_matrix, sample_annotation,
  feature_id_col = "peptide_group_label", color_by = "MS_batch",
  PC_to_plot = c(1, 2), fill_the_missing = 0,
  colors_for_factor = NULL, theme = "classic", plot_title = NULL)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. in most function, it is assumed that this is the log transformed version of the original data
sample_annotation	data matrix with 1) sample_id_col (this can be repeated as row names) 2) biological and 3) technical covariates (batches etc)
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
color_by	column name (as in sample_annotation) to color by
PC_to_plot	principal component numbers for x and y axis
fill_the_missing	boolean value determining if missing values should be substituted with -1 (and colored with black)
colors_for_factor	named vector of colors for the color_by variable
theme	ggplot theme, by default classic. Can be easily overriden (see examples)
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))

Value

ggplot scatterplot colored by factor levels of column specified in factor_to_color

See Also

autoplot.pca_common, ggplot

Examples

```
pca_plot <- plot_PCA(example_proteome_matrix, example_sample_annotation,
color_by = 'MS_batch', plot_title = "PCA colored by MS batch")
```

---

plot_peptides_of_one_protein

Plot peptides of one protein

Description

Creates a spike-in facetted ggplot2 plot of the value in measure_col vs order_col using plot_single_feature. Additionally, the resulting plot can also be facetted by batch.

Usage

```
plot_peptides_of_one_protein(protein_name, protein_col = "ProteinName",
  df_long, sample_annotation, peptide_annotation = NULL,
  order_col = "order", sample_id_col = "FullRunName",
  batch_col = "MS_batch", measure_col = "Intensity",
  feature_id_col = "peptide_group_label", color_by_batch = FALSE,
  color_scheme = "brewer", facet_by_batch = FALSE,
  color_by_col = NULL, color_by_value = NULL,
  plot_title = sprintf("Peptides of %s protein", protein_name), ...)
```

Arguments

protein_name	name of the protein as defined in ProteinName
protein_col	column where protein names are specified
df_long	data frame where each row is a single feature in a single sample. It minimally has a sample_id_col, a feature_id_col and a measure_col, but usually also an m_score (in OpenSWATH output result file)
sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
peptide_annotation	long format data with peptide ID and their corresponding protein annotations
order_col	column in sample_annotation that determines sample order. It is used for certain diagnostics and normalisations.
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
batch_col	column in sample_annotation that should be used for batch comparison
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
color_by_batch	(logical) whether to color points by batch
color_scheme	color scheme for ggplot representation
facet_by_batch	(logical) whether to plot each batch in its own facet
color_by_col	column to color by certain value denoted by color_by_value
color_by_value	value in color_by_col to color
plot_title	the string indicating the source of the peptides
...	additional arguments to plot_single_feature function

Value

ggplot2 type plot of measure_col vs order_col, faceted by spike_ins containing proteins and (optionally) by batch_col

See Also

Other feature-level diagnostic functions: plot_iRT, plot_single_feature, plot_spike_in, plot_with_fitting_cur

Examples

```
peptides_of_one_protein_plot <- plot_peptides_of_one_protein (
protein_name = "Haao",
protein_col = "Gene", df_long = example_proteome,
example_sample_annotation,
order_col = 'order', sample_id_col = 'FullRunName',
batch_col = 'MS_batch')
```

`plot_peptide_corr_distribution` *Plot distribution of peptide correlations within one protein and between proteins*

Description

Plot distribution of peptide correlations within one protein and between proteins

Usage

```
plot_peptide_corr_distribution(data_matrix, peptide_annotation,
  protein_col = "ProteinName", feature_id_col = "peptide_group_label",
  plot_title = "Distribution of peptide correlation",
  theme = "classic")
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
`peptide_annotation`	long format data with peptide ID and their corresponding protein annotations
`protein_col`	the column name in `peptide_annotation` with protein names
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation `df_long`. In the wide formatted representation `data_matrix` this corresponds to the row names.
`plot_title`	Title of the plot, usually processing step
`theme`	ggplot theme, by default `classic`. Can be easily overriden
`...`	parameters for the `ggplot` visualisation

Value

`ggplot` type object with violin plot for each `plot_param`

Examples

```
peptide_corr_distribution <- plot_peptide_corr_distribution(
example_proteome_matrix,
example_peptide_annotation, protein_col = 'Gene')
```

`plot_protein_corrplot` *Peptide correlation matrix (heatmap)*

Description

Plots correlation plot of peptides from a single protein

Usage

```
plot_protein_corrplot(data_matrix, protein_name, peptide_annotation,
  protein_col = "ProteinName", feature_id_col = "peptide_group_label",
  flavor = c("pheatmap", "corrplot"), filename = NULL, width = NA,
  height = NA, unit = c("cm", "in", "mm"),
  plot_title = sprintf("Peptide correlation matrix of %s protein",
  protein_name), ...)
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
`protein_name`	the name of the protein
`peptide_annotation`	df with peptides and their corresponding proteins
`protein_col`	the column name in `peptide_annotation` with protein names
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation `df_long`. In the wide formatted representation `data_matrix` this corresponds to the row names.
`flavor`	either corrplot from 'corrplot' package or heatmap, as in 'pheatmap'
`filename`	path where the results are saved. If null the object is returned to the active window; otherwise, the object is save into the file. Currently only pdf and png format is supported
`width`	option determining the output image width
`height`	option determining the output image width
`unit`	units: 'cm', 'in' or 'mm'
`plot_title`	The title of the plot
`...`	parameters for the corrplot visualisation

Value

`corrplot` or `pheatmap` object depending on `flavor`

Examples

```
protein_corrplot_plot <- plot_protein_corrplot(example_proteome_matrix, protein_name = 'Haao',
               peptide_annotation = example_peptide_annotation,
               protein_col = 'Gene', flavor = "pheatmap")
```

`plot_PVCA` *Plot variance distribution by variable*

Description

Plot variance distribution by variable

Usage

```
plot_PVCA(data_matrix, sample_annotation, sample_id_col = "FullRunName",
  feature_id_col = "peptide_group_label",
  technical_covariates = c("MS_batch", "instrument"),
  biological_covariates = c("cell_line", "drug_dose"),
  fill_the_missing = 0, threshold_pca = 0.6, threshold_var = 0.01,
  colors_for_bars = NULL, theme = "classic", plot_title = NULL)
```

Arguments

`data_matrix`	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. in most function, it is assumed that this is the log transformed version of the original data
`sample_annotation`	data matrix with 1) `sample_id_col` (this can be repeated as row names) 2) biological and 3) technical covariates (batches etc)
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation `df_long`. In the wide formatted representation `data_matrix` this corresponds to the row names.
`technical_covariates`	vector `sample_annotation` column names that are technical covariates
`biological_covariates`	vector `sample_annotation` column names, that are biologically meaningful covariates
`fill_the_missing`	numeric value that the missing values are substituted with
`threshold_pca`	the percentile value of the minimum amount of the variabilities that the selected principal components need to explain
`threshold_var`	the percentile value of weight each of the covariates needs to explain (the rest will be lumped together)
`colors_for_bars`	four-item color vector, specifying colors for the following categories: c('residual', 'biological', 'biol:techn', 'technical')

theme	ggplot theme, by default classic. Can be easily overriden (see examples)
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))

Value

list of two items: plot =gg, df = pvca_res

See Also

sample_annotation_to_colors, ggplot

Examples

```
matrix <- example_proteome_matrix[1:50, ]
pvca_plot <- plot_PVCA(matrix, example_sample_annotation,
technical_covariates = c('MS_batch', 'digestion_batch'),
biological_covariates = c("Diet", "Sex", "Strain"))
```

---

plot_sample_corr_distribution

Create violin plot of correlation distribution

Description

Useful to visualize within batch vs within replicate vs non-related sample correlation

Usage

```
plot_sample_corr_distribution(data_matrix, sample_annotation,
  repeated_samples = NULL, sample_id_col = "FullRunName",
  batch_col = "MS_batch", biospecimen_id_col = "EarTag",
  plot_title = "Correlation distribution",
  plot_param = "batch_replicate")
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
sample_annotation	data matrix with 1) sample_id_col (this can be repeated as row names) 2) biological and 3) technical covariates (batches etc)
repeated_samples	if NULL, only repeated sample correlation is plotted
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix) are found
batch_col	column in sample_annotation that should be used for batch comparison

biospecimen_id_col
: column in sample_annotation that captures the biological sample, that (possibly) was profiled several times as technical replicates. Tip: if such ID is absent, but can be defined from several columns, create new biospecimen_id column

plot_title
: Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))

plot_param
: columns, defined in correlation_df, which is output of get_sample_corr_distrib, specifically, #'

1. replicate
2. batch_the_same
3. batch_replicate
4. batches

;

Value

ggplot type object with violin plot for each plot_param

See Also

get_sample_corr_distrib, ggplot

Examples

```
sample_corr_distribution_plot <- plot_sample_corr_distribution(
example_proteome_matrix,
example_sample_annotation, batch_col = 'MS_batch',
biospecimen_id_col = "EarTag",
plot_param = 'batch_replicate')
```

---

plot_sample_corr_heatmap

Sample correlation matrix (heatmap)

Description

Plot correlation of selected samples

Usage

```
plot_sample_corr_heatmap(data_matrix, samples_to_plot = NULL,
  flavor = c("pheatmap", "corrplot"), filename = NULL, width = NA,
  height = NA, unit = c("cm", "in", "mm"),
  plot_title = sprintf("Correlation matrix of sample %s",
  samples_to_plot), ...)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
samples_to_plot	string vector of samples in data_matrix to be used in the plot
flavor	either corrplot from 'corrplot' package or heatmap, as in 'pheatmap'
filename	path where the results are saved. If null the object is returned to the active window; otherwise, the object is save into the file. Currently only pdf and png format is supported
width	option determining the output image width
height	option determining the output image width
unit	units: 'cm', 'in' or 'mm'
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))
...	parameters for the corrplot.mixed or pheatmap visualisation, for details see examples and help to corresponding functions

Value

corrplot or pheatmap object depending on flavor

See Also

pheatmap, corrplot.mixed

Examples

```
specified_samples = example_sample_annotation$FullRunName[
which(example_sample_annotation$order %in% 110:115)]

sample_corr_heatmap <- plot_sample_corr_heatmap(example_proteome_matrix,
samples_to_plot = specified_samples,
 flavor = 'pheatmap',
 cluster_rows= FALSE, cluster_cols=FALSE,
 annotation_names_col = TRUE, annotation_legend = FALSE,
 show_colnames = FALSE)
```

---

plot_sample_mean_or_boxplot

Plot per-sample mean or boxplot (showing median and quantiles) vs order (if the real running order available)

Description

Plot per-sample mean or boxplot (showing median and quantiles) vs order (if the real running order available)

Usage

```
plot_sample_mean(data_matrix, sample_annotation = NULL,
  sample_id_col = "FullRunName", order_col = "order",
  batch_col = "MS_batch", facet_col = NULL, color_by_batch = FALSE,
  color_scheme = "brewer", theme = "classic", plot_title = NULL,
  order_per_facet = FALSE, vline_color = "grey", ylimits = NULL)

plot_boxplot(df_long, sample_annotation = NULL,
  sample_id_col = "FullRunName", measure_col = "Intensity",
  order_col = "order", batch_col = "MS_batch", facet_col = NULL,
  color_by_batch = TRUE, color_scheme = "brewer", theme = "classic",
  plot_title = NULL, order_per_facet = FALSE)
```

Arguments

data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. in most function,
sample_annotation	data matrix with 1) sample_id_col (this can be repeated as row names) 2) biological and 3) technical covariates (batches etc)
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
order_col	column where running order is specified.
batch_col	column in sample_annotation that should be used for batch comparison
facet_col	recommended if more than one batch covariate is present. Faceting is most suited to examine instruments separately
color_by_batch	should the each batch be represented with its own color?
color_scheme	named vector, names corresponding to unique batch values as specified in sample_annotation
theme	ggplot theme, by default classic. Can be easily overriden (see examples)
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))
order_per_facet	if order is defined ignoring facets (usually instrument), re-define order per-batch
vline_color	color of vertical lines, typically denoting different MS batches in ordered runs; should be NULL for experiments without intrinsic order
ylimits	range of y-axis to plot feature-level trends
df_long	data frame where each row is a single feature in a single sample, thus it has minimally, sample_id_col, feature_id_col and measure_col, but usually also m_score (in OpenSWATH output result file)
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity, otherwise, it is used internally for consistency

Details

functions for quick visual assessment of trends associated, overall or specific covariate-associated (see batch_col and facet_col)

Value

ggplot2 class object. Thus, all aesthetics can be overriden

See Also

ggplot

Examples

```
mean_plot <- plot_sample_mean(example_proteome_matrix, example_sample_annotation,
order_col = 'order', batch_col = "MS_batch")

boxplot <- plot_boxplot(example_proteome, example_sample_annotation,
batch_col = "MS_batch")
```

---

plot_single_feature *Plot peptide measurements*

Description

Creates a peptide facetted ggplot2 plot of the value in `measure_col` vs `order_col`. Additionally, the resulting plot can also be facetted by batch.

Usage

```
plot_single_feature(pep_name, df_long, sample_annotation,
  order_col = "order", sample_id_col = "FullRunName",
  batch_col = "MS_batch", measure_col = "Intensity",
  feature_id_col = "peptide_group_label", geom = c("point", "line"),
  color_by_batch = FALSE, color_scheme = "brewer",
  facet_by_batch = FALSE, color_by_col = NULL, color_by_value = NULL,
  plot_title = NULL, vline_color = "red", theme = "classic")
```

Arguments

`pep_name`	name of the peptide for diagnostic profiling
`df_long`	data frame where each row is a single feature in a single sample. It minimally has a `sample_id_col`, a `feature_id_col` and a `measure_col`, but usually also an `m_score` (in OpenSWATH output result file)
`sample_annotation`	data matrix with: 1. `sample_id_col` (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
`order_col`	column in `sample_annotation` that determines sample order. It is used for certain diagnostics and normalisations.
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`batch_col`	column in `sample_annotation` that should be used for batch comparison
`measure_col`	if `df_long` is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency

feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
geom	whether to show the feature as points and/or connect by lines
color_by_batch	(logical) whether to color points by batch
color_scheme	color scheme for ggplot representation
facet_by_batch	(logical) whether to plot each batch in its own facet
color_by_col	column to color by certain value denoted by color_by_value
color_by_value	value in color_by_col to color
plot_title	the string indicating the source of the peptides
vline_color	color of vertical lines, typically denoting different MS batches in ordered runs; should be NULL for experiments without intrinsic order
theme	plot theme (default is 'classical'; other options not implemented)

Value

ggplot2 type plot of measure_col vs order_col, faceted by pep_name and (optionally) by batch_col

See Also

Other feature-level diagnostic functions: plot_iRT, plot_peptides_of_one_protein, plot_spike_in, plot_with_fitting_curve

Examples

```
single_feature_plot <- plot_single_feature(
pep_name = "46213_NVGVSFYADKPEVTQEQK_2",
df_long = example_proteome, example_sample_annotation,
color_by_col = NULL)
```

---

plot_spike_in *Plot spike-in measurements*

Description

Creates a spike-in facetted ggplot2 plot of the value in measure_col vs order_col using plot_single_feature. Additionally, the resulting plot can also be facetted by batch.

Usage

```
plot_spike_in(df_long, sample_annotation, peptide_annotation = NULL,
  protein_col = "ProteinName", order_col = "order",
  spike_ins = "BOVIN", sample_id_col = "FullRunName",
  batch_col = "MS_batch", measure_col = "Intensity",
  feature_id_col = "peptide_group_label", color_by_batch = FALSE,
  color_scheme = "brewer", facet_by_batch = FALSE,
  color_by_col = NULL, color_by_value = NULL,
  plot_title = "Spike-in BOVINE protein peptides", ...)
```

Arguments

df_long	data frame where each row is a single feature in a single sample. It minimally has a sample_id_col, a feature_id_col and a measure_col, but usually also an m_score (in OpenSWATH output result file)
sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
peptide_annotation	long format data with peptide ID and their corresponding protein annotations
protein_col	column where protein names are specified
order_col	column in sample_annotation that determines sample order. It is used for certain diagnostics and normalisations.
spike_ins	substring used to identify spike-in proteins in the column 'ProteinName'
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
batch_col	column in sample_annotation that should be used for batch comparison
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
color_by_batch	(logical) whether to color points by batch
color_scheme	color scheme for ggplot representation
facet_by_batch	(logical) whether to plot each batch in its own facet
color_by_col	column to color by certain value denoted by color_by_value
color_by_value	value in color_by_col to color
plot_title	the string indicating the source of the peptides
...	additional arguments to plot_single_feature function

Value

ggplot2 type plot of measure_col vs order_col, faceted by spike_ins containing proteins and (optionally) by batch_col

See Also

Other feature-level diagnostic functions: plot_iRT, plot_peptides_of_one_protein, plot_single_feature, plot_with_fitting_curve

Examples

```
spike_in_plot <- plot_spike_in(example_proteome, example_sample_annotation,
protein_col = 'Gene', spike_ins = "BOVINE_A1ag",
plot_title = "Spike-in BOVINE protein peptides")
```

`plot_with_fitting_curve` *Plot peptide measurements across multi-step analysis*

Description

Plot Intensity of a few representative peptides for each step of the analysis including the fitting curve

Usage

```
plot_with_fitting_curve(pep_name, df_long, sample_annotation, fit_df,
  fit_value_var = "fit", order_col = "order",
  sample_id_col = "FullRunName", batch_col = "MS_batch",
  measure_col = "Intensity", feature_id_col = "peptide_group_label",
  geom = c("point", "line"), color_by_batch = FALSE,
  color_scheme = "brewer", facet_by_batch = FALSE,
  plot_title = sprintf("Fitting curve of %s peptide", pep_name),
  color_by_col = NULL, color_by_value = NULL, theme = "classic",
  vline_color = "grey", ...)
```

Arguments

`pep_name`	name of the peptide for diagnostic profiling
`df_long`	data frame where each row is a single feature in a single sample. It minimally has a `sample_id_col`, a `feature_id_col` and a `measure_col`, but usually also an `m_score` (in OpenSWATH output result file)
`sample_annotation`	data matrix with: 1. `sample_id_col` (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
`fit_df`	data frame typically output generated from nonlinear curve fitting by `normalize_custom_fit`
`fit_value_var`	column denoting intensity values, typically fitted to curve
`order_col`	column in `sample_annotation` that determines sample order. It is used for certain diagnostics and normalisations.
`sample_id_col`	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
`batch_col`	column in `sample_annotation` that should be used for batch comparison
`measure_col`	if `df_long` is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
`feature_id_col`	name of the column with feature/gene/peptide/protein ID used in the long format representation `df_long`. In the wide formatted representation `data_matrix` this corresponds to the row names.
`geom`	for the intensity `measure_col` profile
`color_by_batch`	(logical) whether to color points by batch
`color_scheme`	color scheme for `ggplot` representation

facet_by_batch	(logical) whether to plot each batch in its own facet
plot_title	the string indicating the source of the peptides
color_by_col	column to color by certain value denoted by color_by_value
color_by_value	value in color_by_col to color
theme	plot theme (default is 'classical'; other options not implemented)
vline_color	color of vertical lines, typically denoting different MS batches in ordered runs; should be NULL for experiments without intrinsic order
...	additional arguments to plot_single_feature function

Value

ggplot-class plot with minimally two facets (before and after non-linear fit) with measure_col (Intensity) vs order_col (injection order) for selected peptides (specified in pep_name)

See Also

Other feature-level diagnostic functions: plot_iRT, plot_peptides_of_one_protein, plot_single_feature, plot_spike_in

Examples

```
loess_fit_70 <- adjust_batch_trend(example_proteome_matrix,
example_sample_annotation, span = 0.7)

fitting_curve_plot <- plot_with_fitting_curve(
pep_name = "10231_QDVDVWLWQQEGSSK_2",
df_long = example_proteome, example_sample_annotation,
fit_df = loess_fit_70$fit_df, plot_title = "Curve fitting with 70% span")
```

---

proBatch *proBatch: A package for diagnostics and correction of batch effects, primarily in proteomics*

Description

The proBatch package contains functions for analyzing and correcting batch effects and other unwanted technical variation from high-thoughput experiments. Although the package has primarily been developed for mass spectrometry proteomics (DIA/SWATH), it should also be applicable to most omic data with minor adaptations. It addresses the following needs:

- prepare the data for analysis
- Visualize batch effects in sample-wide and feature-level;
- Normalize and correct for batch effects.

Arguments

df_long	data frame where each row is a single feature in a single sample. It minimally has a sample_id_col, a feature_id_col and a measure_col, but usually also an m_score (in OpenSWATH output result file)
data_matrix	features (in rows) vs samples (in columns) matrix, with feature IDs in rownames and file/sample names as colnames. Usually the log transformed version of the original data
sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
batch_col	column in sample_annotation that should be used for batch comparison
order_col	column in sample_annotation that determines sample order. It is used for certain diagnostics and normalisations.
measure_col	if df_long is among the parameters, it is the column with expression/abundance/intensity; otherwise, it is used internally for consistency
feature_id_col	name of the column with feature/gene/peptide/protein ID used in the long format representation df_long. In the wide formatted representation data_matrix this corresponds to the row names.
plot_title	Title of the plot (usually, processing step + representation level (fragments, transitions, proteins))
theme	ggplot theme, by default classic. Can be easily overriden

Details

To learn more about proBatch, start with the vignettes: browseVignettes(package = "proBatch")

Section

Common arguments to the functions.

---

quantile_normalize *Quantile normalization of the data, ensuring that the row and column names are retained*

Description

Quantile normalization of the data, ensuring that the row and column names are retained

Usage

```
quantile_normalize(data_matrix)
```

Arguments

data_matrix	log transformed data matrix (features in rows and samples in columns)

Value

data_matrix-size matrix, with columns quantile-normalized

Examples

```
quantile_normalized_matrix <- quantile_normalize(example_proteome_matrix)
```

---

sample_annotation_to_colors

Generate colors for sample annotation

Description

Convert the sample annotation data frame to list of colors the list is named as columns included to use in potting functions

Usage

```
sample_annotation_to_colors(sample_annotation,
  columns_for_plotting = NULL, sample_id_col = "FullRunName",
  factor_columns = c("MS_batch", "EarTag", "Strain", "Diet", "Sex"),
  not_factor_columns = "DateTime", numeric_columns = "order",
  rare_categories_to_other = TRUE, numeric_palette_type = "brewer",
  granularity = 10)
```

Arguments

sample_annotation	data matrix with: 1. sample_id_col (this can be repeated as row names) 2. biological covariates 3. technical covariates (batches etc)
columns_for_plotting	only consider these columns from sample_annotation
sample_id_col	name of the column in sample_annotation file, where the filenames (colnames of the data matrix are found)
factor_columns	columns of sample_annotation to be treated as factors. Note that factor and character columns are treated as factors by default.
not_factor_columns	don't treat these columns as factors. This can be used to override the default behaviour of considering factors and character columns as factors.
numeric_columns	columns of sample_annotation to be treated as continuous numeric values.
rare_categories_to_other	if True rare categories will be merged as 'other'
numeric_palette_type	palette to be used for numeric values coloring
granularity	number of colors to map to the number vector (equally spaced between minimum and maximum)

Value

list of colors

Examples

```
color_scheme <- sample_annotation_to_colors (example_sample_annotation,
factor_columns = c('MS_batch','EarTag', "Strain",
"Diet", "digestion_batch", "Sex"),
not_factor_columns = 'DateTime',
numeric_columns = c('order'))
```

---

sample_color_scheme *Sample color annotation*

Description

This is an color scheme generated from example sample annotation

Usage

```
sample_color_scheme
```

Format

A list of 3 components: `list_of_colors`, `color_df` and `sample_annotation`

list_of_colors a list of colors for 11 variables, including `MS_batch`, `EarTag`, `Strain`, `Diet`, `digestion_batch`, `Sex`, `FullRunName`, `RunDate`, `RunTime`, `DateTime`, `order`

color_df a data frame with 233 samples and 11 variables describing a color for each component

sample_anotation a data frame containing 233 samples and 11 variables annotating samples to facilitate conversion to a color scheme

Index