Package 'structToolbox'

February 21, 2024

Type Package

Title Data processing & analysis tools for Metabolomics and other omics

Version 1.14.0

Description An extensive set of data (pre-)processing and analysis methods and tools for metabolomics and other omics, with a strong emphasis on statistics and machine learning. This toolbox allows the user to build extensive and standardised workflows for data analysis. The methods and tools have been implemented using class-based templates provided by the struct (Statistics in R Using Class-based Templates) package. The toolbox includes pre-processing methods (e.g. signal drift and batch correction, normalisation, missing value imputation and scaling), univariate (e.g. ttest, various forms of ANOVA, Kruskal–Wallis test and more) and multivariate statistical methods (e.g. PCA and PLS, including cross-validation and permutation testing) as well as machine learning methods (e.g. Support Vector Machines). The STATistics Ontology (STATO) has been integrated and implemented to provide standardised definitions for the different methods, inputs and outputs.

License GPL-3

Encoding UTF-8

Collate 'AUC_metric_class.R' 'entity_objects.R' 'DFA_class.R' 'zzz.R' 'anova_class.R' 'HSD_class.R' 'mixed_effect_class.R' 'HSDEM_class.R' 'MTBLS79_dataset_class.R' 'PCA_class.R' 'scatter_chart_class.R' 'PCA_plotfcns.R' 'PLSR_class.R' 'PLSDA_class.R' 'PLSDA_charts.R' 'as_data_frame_doc.R' 'autoscale_class.R' 'balanced_accuracy_class.R' 'blank_filter_class.R' 'bootstrap_class.R' 'calculate_doc.R' 'chart_plot_doc.R' 'classical_lsq_class.R' 'confounders_clsq_class.R' 'constant_sum_norm_class.R' 'corr_coef_class.R' 'd_ratio_filter_class.R' 'dataset_chart_classes.R' 'split_data_class.R' 'equal_split_class.R' 'factor_barchart_class.R' 'feature_plot_array_class.R' 'feature_profile_class.R' 'filter_by_name_class.R' 'filter_na_count.R' 'filter_smeta_class.R' 'fisher_exact_class.R'

'fold_change_class.R' 'fold_change_int_class.R'
'forward_selection_by_rank_class.R' 'ggplot_theme_pub.R'
'glog_class.R' 'grid_search_1d_class.R' 'hca_class.R'
'kfold_xval_class.R' 'kfold_xval_charts.R' 'knn_impute_class.R'
'kw_rank_sum_class.R' 'linear_model_class.R' 'log_transform.R'
'mean_centre_class.R' 'mean_of_medians.R' 'model_apply_doc.R'
'model_predict_doc.R' 'model_reverse_doc.R' 'model_train_doc.R'
'mv_feature_filter_class.R' 'mv_sample_filter_class.R'
'nroot_transform_class.R' 'oplsr_class.R' 'oplsda_class.R'
'pairs_filter_class.R' 'paretoscale_class.R'
'permutation_test_class.R' 'permute_sample_order_class.R'
'plsda_feature_significance_chart.R' 'pqn_norm_method_class.R'
'prop_na_class.R' 'r_squared_class.R' 'resample_class.R'
'rsd_filter.R' 'run_doc.R' 'sb_corr.R'
'stratified_split_class.R' 'structToolbox.R'
'svm_classifier_class.R' 'tSNE_class.R' 'tic_chart_class.R'
'ttest_class.R' 'vec_norm_class.R' 'wilcox_test_class.R'

Depends R (>= 4.0), struct (>= 1.5.1)

Imports ggplot2, ggthemes, grid, gridExtra, methods, scales, sp, stats, utils

RoxygenNote 7.2.3

Suggests agricolae, BiocFileCache, BiocStyle, car, covr, cowplot, e1071, emmeans, ggdendro, knitr, magick, nlme, openxlsx, pls, pmp, reshape2, ropls, rmarkdown, Rtsne, testthat, rappdirs

VignetteBuilder knitr

biocViews WorkflowStep, Metabolomics

URL https://github.com/computational-metabolomics/structToolbox

Roxygen list(markdown = TRUE)

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

git_branch RELEASE_3_18

git_last_commit f3b802f

git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-02-20

Author Gavin Rhys Lloyd [aut, cre] (<https://orcid.org/0000-0001-7989-6695>), Ralf Johannes Maria Weber [aut]

Maintainer Gavin Rhys Lloyd <g.r.lloyd@bham.ac.uk>

R topics documented:

`ANOVA` *Analysis of Variance*

Description

Analysis of Variance (ANOVA) is a univariate method used to analyse the difference among group means. Multiple test corrected p-values are computed to indicate significance for each feature.

Usage

```
ANOVA(alpha = 0.05, mtc = "fdr", formula, ss_type = "III", ...)
```

Arguments

`alpha`	(numeric) The p-value cutoff for determining significance. The default is `0.05`.
`mtc`	(character) Multiple test correction method. Allowed values are limited to the following: • `"bonferroni"`: Bonferroni correction in which the p-values are multiplied by the number of comparisons. • `"fdr"`: Benjamini and Hochberg False Discovery Rate correction. • `"none"`: No correction. The default is `"fdr"`.
`formula`	(formula) A symbolic description of the model to be fitted.

ss_type	(character) ANOVA sum of squares. Allowed values are limited to the following: • "I": Type I sum of squares. • "II": Type II sum of squares. • "III": Type III sum of squares. The default is "III".
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- car

Value

A ANOVA object with the following output slots:

f_statistic	(data.frame) The value of the calculated statistic.
p_value	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

References

Fox J, Weisberg S (2019). *An R Companion to Applied Regression*, Third edition. Sage, Thousand Oaks CA. https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Examples

```
D = iris_DatasetExperiment()
M = ANOVA(formula=y~Species)
M = model_apply(M,D)
```

---

as_data_frame *Convert to data.frame*

Description

Convert the outputs of the input model into a data.frame.

Usage

```
## S4 method for signature 'filter_na_count'
as_data_frame(M)

## S4 method for signature 'ttest'
as_data_frame(M)

## S4 method for signature 'wilcox_test'
as_data_frame(M)
```

Arguments

M	a model object

Value

A data.frame of model outputs

Examples

```
D = iris_DatasetExperiment()
M = filter_na_count(threshold=50,factor_name='Species')
M= model_apply(M,D)
df = as_data_frame(M)
```

AUC *Area under ROC curve*

Description

The area under the ROC curve of a classifier is estimated using the trapezoid method.

Usage

```
AUC(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A AUC object. This object has no output slots.

Examples

```
D = iris_DatasetExperiment()
XCV = kfold_xval(folds=5,factor_name='Species') *
      (mean_centre() + PLSDA(number_components=2,factor_name='Species'))
MET = AUC()
XCV = run(XCV,D,MET)
```

`autoscale` *Autoscaling*

Description

Each variable/feature is mean centred and scaled by the standard deviation. The transformed variables have zero-mean and unit-variance.

Usage

```
autoscale(mode = "data", ...)
```

Arguments

`mode`	(character) Mode of action. Allowed values are limited to the following: • `"data"`: Autoscaling is applied to the data matrix only. • `"sample_meta"`: Autoscaling is applied to the sample_meta data only. • `"both"`: Autoscaling is applied to both the data matrix and the meta data. The default is `"data"`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `autoscale` object with the following `output` slots:

`autoscaled`	(DatasetExperiment)
`mean_data`	(numeric)
`sd_data`	(numeric)
`mean_sample_meta`	(numeric)
`sd_sample_meta`	(numeric)

Examples

```
D = iris_DatasetExperiment()
M = autoscale()
M = model_train(M,D)
M = model_predict(M,D)
```

balanced_accuracy *Balanced Accuracy*

Description

Balanced Accuracy is the average proportion of correctly classified samples across all groups.

Usage

```
balanced_accuracy(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A balanced_accuracy object. This object has no output slots.

Examples

```
D = iris_DatasetExperiment()
XCV = kfold_xval(folds=5,factor_name='Species') *
      (mean_centre() + PLSDA(number_components=2,factor_name='Species'))
MET = balanced_accuracy()
XCV = run(XCV,D,MET)
```

blank_filter *Blank filter*

Description

A blank filter filters features by comparing the median intensity of blank samples to the median intensity of samples. Features where the relative intensity (fold change) is not large when compared to the blank are removed. The number of times a feature is detected across all blank samples may also be considered. If the feature is not detected in a high enough proportion of the blanks then it is not removed.

Usage

```
blank_filter(
  fold_change = 20,
  blank_label = "blank",
  qc_label = "QC",
  factor_name,
  fraction_in_blank = 0,
  ...
)
```

Arguments

fold_change	(numeric) Features with fold change less than this value are removed. The default is 20.
blank_label	(character) The label used to identify blank samples. The default is "blank".
qc_label	(character, NULL) The label used to identify QC samples. If set to NULL then the median of the samples is used. The default is "QC".
factor_name	(character) The name of a sample-meta column to use.
fraction_in_blank	(numeric) Features present in less than this proportion of the blanks are not considered for removal. The default is 0.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A blank_filter object with the following output slots:

filtered	(DatasetExperiment) A DatasetExperiment object containing the filtered data.
flags	(data.frame) A flag indicating whether the feature was rejected or not.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
D = iris_DatasetExperiment()
M = blank_filter(fold_change=2,
                 factor_name='Species',
                 blank_label='setosa',
                 qc_label='versicolor')
M = model_apply(M,D)
```

blank_filter_hist *Histogram of blank filter fold changes*

Description

A histogram of the calculated fold changes for the blank filter (median samples divided by median blanks)

Usage

```
blank_filter_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A blank_filter_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = blank_filter_hist()
```

bootstrap *Bootstrap resampling*

Description

In bootstrap resampling a subset of samples is selected at random with replacement to form a training set. Any sample not selected for training is included in the test set. This process is repeated many times, and performance metrics are computed for each repetition.

Usage

```
bootstrap(number_of_repetitions = 100, collect, ...)
```

Arguments

number_of_repetitions	(numeric, integer) The number of bootstrap repetitions. The default is 100.
collect	(character) The name of a model output to collect over all bootstrap repetitions, in addition to the input metric.
...	Additional slots and values passed to struct_class.

Value

A bootstrap object with the following output slots:

results	(data.frame)
metric	(data.frame)
collected	(logical, list)

Examples

```
I = bootstrap(number_of_repetitions = 10, collect = 'vip')
```

---

calculate,AUC-method *Calculate metric*

Description

Calculate metric

Usage

```
## S4 method for signature 'AUC'
calculate(obj, Y, Yhat)

## S4 method for signature 'balanced_accuracy'
calculate(obj, Y, Yhat)

## S4 method for signature 'r_squared'
calculate(obj, Y, Yhat)
```

Arguments

obj	a metric object
Y	the true values/group labels
Yhat	the predicted values/group labels

Value

a modified metric object

Examples

```
MET = metric()
calculate(MET)
```

chart_plot,dfa_scores_plot,DFA-method
chart_plot method

Description

Plots a chart object

Usage

```
## S4 method for signature 'dfa_scores_plot,DFA'
chart_plot(obj, dobj)

## S4 method for signature 'scatter_chart,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'pca_correlation_plot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'pca_scores_plot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'pca_biplot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'pca_loadings_plot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'pca_scree_plot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'pca_dstat_plot,PCA'
chart_plot(obj, dobj)

## S4 method for signature 'plsr_prediction_plot,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'plsr_residual_hist,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'plsr_qq_plot,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'plsr_cook_dist,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'pls_scores_plot,PLSR'
```

```
chart_plot(obj, dobj)

## S4 method for signature 'plsda_predicted_plot,PLSDA'
chart_plot(obj, dobj)

## S4 method for signature 'plsda_roc_plot,PLSDA'
chart_plot(obj, dobj)

## S4 method for signature 'pls_vip_plot,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'pls_regcoeff_plot,PLSR'
chart_plot(obj, dobj)

## S4 method for signature 'blank_filter_hist,blank_filter'
chart_plot(obj, dobj)

## S4 method for signature 'confounders_lsq_barchart,confounders_clsq'
chart_plot(obj, dobj)

## S4 method for signature 'confounders_lsq_boxplot,confounders_clsq'
chart_plot(obj, dobj)

## S4 method for signature 'feature_boxplot,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'mv_histogram,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'mv_boxplot,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'DatasetExperiment_dist,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'DatasetExperiment_boxplot,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'compare_dist,DatasetExperiment'
chart_plot(obj, dobj, eobj)

## S4 method for signature 'DatasetExperiment_heatmap,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'DatasetExperiment_factor_boxplot,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'feature_profile_array,DatasetExperiment'
```

```
chart_plot(obj, dobj)

## S4 method for signature 'feature_profile,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'fold_change_plot,fold_change'
chart_plot(obj, dobj)

## S4 method for signature 'fs_line,forward_selection_by_rank'
chart_plot(obj, dobj)

## S4 method for signature 'glog_opt_plot,glog_transform'
chart_plot(obj, dobj, gobj)

## S4 method for signature 'gs_line,grid_search_1d'
chart_plot(obj, dobj)

## S4 method for signature 'hca_dendrogram,HCA'
chart_plot(obj, dobj)

## S4 method for signature 'kfoldxcv_grid,kfold_xval'
chart_plot(obj, dobj)

## S4 method for signature 'kfoldxcv_metric,kfold_xval'
chart_plot(obj, dobj)

## S4 method for signature 'kw_p_hist,kw_rank_sum'
chart_plot(obj, dobj)

## S4 method for signature 'mv_feature_filter_hist,mv_feature_filter'
chart_plot(obj, dobj)

## S4 method for signature 'mv_sample_filter_hist,mv_sample_filter'
chart_plot(obj, dobj)

## S4 method for signature 'permutation_test_plot,permutation_test'
chart_plot(obj, dobj)

## S4 method for signature 'plsda_feature_importance_plot,PLSDA'
chart_plot(obj, dobj)

## S4 method for signature 'pqn_norm_hist,pqn_norm'
chart_plot(obj, dobj)

## S4 method for signature 'resample_chart,resample'
chart_plot(obj, dobj)

## S4 method for signature 'rsd_filter_hist,rsd_filter'
```

```
chart_plot(obj, dobj)

## S4 method for signature 'feature_profile,sb_corr'
chart_plot(obj, dobj, gobj)

## S4 method for signature 'svm_plot_2d,SVM'
chart_plot(obj, dobj, gobj)

## S4 method for signature 'tSNE_scatter,tSNE'
chart_plot(obj, dobj)

## S4 method for signature 'tic_chart,DatasetExperiment'
chart_plot(obj, dobj)

## S4 method for signature 'wilcox_p_hist,wilcox_test'
chart_plot(obj, dobj)
```

Arguments

obj	a chart object
dobj	a struct object
eobj	a second DatasetExperiment object to compare with the first
gobj	The DatasetExperiment object before signal correction was applied.

Value

a plot object

Examples

```
C = example_chart()
chart_plot(C,iris_DatasetExperiment())
```

---

classical_lsq *Univariate Classical Least Squares Regression*

Description

In univariate classical least squares regression a line is fitted between each feature/variable and a response variable. The fitted line minimises the sum of squared differences between the true response and the predicted response. The coefficients (offset, gradient) of the fit can be tested for significance.

Usage

```
classical_lsq(alpha = 0.05, mtc = "fdr", factor_names, intercept = TRUE, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_names	(character, list) The column names to regress against. If a character vector then the same list is used ofr all features. If a list of character vectors is provided it is assumed there is a different set of columns for each feature.
intercept	(logical) Model intercept. Allowed values are limited to the following: • "TRUE": An intercept term is included in the model. • "FALSE": An intercept term is not included in the model. The default is TRUE.
...	Additional slots and values passed to struct_class.

Value

A classical_lsq object with the following output slots:

coefficients	(data.frame) The regression coefficients for each term in the model.
p_value	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold
r_squared	(data.frame) The value of R Squared for the fitted model.
adj_r_squared	(data.frame) The value ofAdjusted R Squared for the fitted model.

Examples

```
D = iris_DatasetExperiment()
M = classical_lsq(factor_names = 'Species')
M = model_apply(M,D)
```

---

compare_dist *Compare distributions*

Description

Histograms and boxplots computed across samples and features are used to visually compare two datasets e.g. before and after filtering and/or normalisation.

Usage

```
compare_dist(factor_name, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A compare_dist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D1=MTBLS79_DatasetExperiment(filtered=FALSE)
D2=MTBLS79_DatasetExperiment(filtered=TRUE)
C = compare_dist(factor_name='Class')
chart_plot(C,D1,D2)
```

---

confounders_clsq *Check for confounding factors*

Description

Univariate least squares regression models are used to compare models with and without potential confounding factors included. The change in coefficients (delta) is then computed for each potential confounding factor. Factors with a large delta are said to be having a large impact on the model and are therefore confounding. p-values are computed for models with confounders included to reduce potential false positives. Only suitable for main factors with 2 levels.

Usage

```
confounders_clsq(
  alpha = 0.05,
  mtc = "fdr",
  factor_name,
  confounding_factors,
  threshold = 0.15,
  ...
)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_name	(character) The name of the main factor with which other factors may be confounding.
confounding_factors	(character) The name(s) of factor(s) that are potential confounding factors.
threshold	(numeric) Factors with a delta greater than the the threshold are considered to be confounding. The default is 0.15.
...	Additional slots and values passed to struct_class.

Value

A confounders_clsq object with the following output slots:

coefficients	(data.frame)
p_value	(data.frame)
significant	(data.frame)
percent_change	(data.frame)
potential_confounders	(list)

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_by_name(mode='include',dimension='variable',
        names=colnames(D$data)[1:10]) + # first 10 features
    filter_smeta(mode='exclude',levels='QC',
        factor_name='Class') + # reduce to two group comparison
    confounders_clsq(factor_name = 'Class',
        confounding_factors=c('run_order','Batch'))
M = model_apply(M,D)
```

confounders_lsq_barchart *Confounding factor relative change barchart*

Description

A barchart of the relative change (delta) in regression coefficient when potential confounding factors are included, and excluded, from the model. Factors with a large delta are considered to be confounding factors.

Usage

```
confounders_lsq_barchart(feature_to_plot, threshold = 10, ...)
```

Arguments

feature_to_plot	(numeric, character, integer) The column name of the feature to be plotted.
threshold	(numeric) A horizontal line is plotted to indicate the threshold. The default is 10.
...	Additional slots and values passed to struct_class.

Value

A confounders_lsq_barchart object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_by_name(mode='include',dimension='variable',
        names=colnames(D$data)[1:10]) + # first 10 features
    filter_smeta(mode='exclude',levels='QC',
        factor_name='Class') + # reduce to two group comparison
    confounders_clsq(factor_name = 'Class',
        confounding_factors=c('run_order','Batch'))
M = model_apply(M,D)
C = C=confounders_lsq_barchart(feature_to_plot=1,threshold=15)
chart_plot(C,M[3])
```

confounders_lsq_boxplot *Confounding factor relative change boxplot*

Description

A boxplot of the relative change (delta) in regression coefficient when potential confounding factors are included, and excluded, from the model. Factors with a large delta are considered to be confounding factors.

Usage

```
confounders_lsq_boxplot(threshold = 10, ...)
```

Arguments

threshold	(numeric) A horizontal line is plotted to indicate the threshold. The default is 10.
...	Additional slots and values passed to struct_class.

Value

A confounders_lsq_boxplot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_by_name(mode='include',dimension='variable',
        names=colnames(D$data)[1:10]) + # first 10 features
    filter_smeta(mode='exclude',levels='QC',
        factor_name='Class') + # reduce to two group comparison
    confounders_clsq(factor_name = 'Class',
        confounding_factors=c('run_order','Batch'))
M = model_apply(M,D)
C = C=confounders_lsq_boxplot(threshold=15)
chart_plot(C,M[3])
```

constant_sum_norm *Normalisation to constant sum*

Description

Each sample is normalised such that the total signal is equal to one (or a scaling factor if specified).

Usage

```
constant_sum_norm(scaling_factor = 1, ...)
```

Arguments

scaling_factor	(numeric) The scaling factor applied after normalisation. The default is 1.
...	Additional slots and values passed to struct_class.

Value

A constant_sum_norm object with the following output slots:

normalised	(DatasetExperiment) A DatasetExperiment object containing the normalised data.
coeff	(data.frame) The sum of each row, used to normalise the samples.

Examples

```
M = constant_sum_norm()
```

---

corr_coef *Correlation coefficient*

Description

The correlation between features and a set of continuous factor are calculated. Multiple-test corrected p-values are used to indicate whether the computed coefficients may have occurred by chance.

Usage

```
corr_coef(alpha = 0.05, mtc = "fdr", factor_names, method = "spearman", ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_names	(character) The name of sample meta column(s) to use.

method	(character) Type of correlation. Allowed values are limited to the following: • "kendall": Kendall's tau is computed. • "pearson": Pearson product moment correlation is computed. • "spearman": Spearman's rho statistic is computed. The default is "spearman".
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- stats

Value

A corr_coef object with the following output slots:

coeff	(data.frame) The value of the calculate statistics which is converted to a p-value when compared to a t-distribu
p_value	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

References

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Examples

```
D = MTBLS79_DatasetExperiment(filtered=TRUE)

# subset for this example
D = D[,1:10]

# convert to numeric for this example
D$sample_meta$sample_order=as.numeric(D$sample_meta$run_order)
D$sample_meta$sample_rep=as.numeric(D$sample_meta$Sample_Rep)

M = corr_coef(factor_names=c('sample_order','sample_rep'))
M = model_apply(M,D)
```

`DatasetExperiment_boxplot`

Feature distribution histogram

Description

A boxplot to visualise the distribution of values within a subset of features.

Usage

```
DatasetExperiment_boxplot(
  factor_name,
  by_sample = TRUE,
  per_class = TRUE,
  number = 50,
  ...
)
```

Arguments

`factor_name`	(character) The name of a sample-meta column to use.
`by_sample`	(logical) Plot by sample. Allowed values are limited to the following: • `"TRUE"`: The data is plotted across features for a subset of samples. • `"FALSE"`: The data is plotted across samples for a subset of features. The default is `TRUE`.
`per_class`	(logical) Plot per class. Allowed values are limited to the following: • `"TRUE"`: The data is plotted for each class. • `"FALSE"`: The data is plotted for all samples. The default is `TRUE`.
`number`	(numeric, integer) The number of features/samples plotted. The default is `50`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `DatasetExperiment_boxplot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

struct object

Examples

```
D = MTBLS79_DatasetExperiment()
C = DatasetExperiment_boxplot(factor_name='Class',number=10,per_class=FALSE)
chart_plot(C,D)
```

DatasetExperiment_dist

Feature distribution histogram

Description

A histogram to visualise the distribution of values within features.

Usage

```
DatasetExperiment_dist(factor_name, per_class = TRUE, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
per_class	(logical) Plot per class. Allowed values are limited to the following: • "TRUE": The distributions are plotted for each class. • "FALSE": The distribution is plotted for all samples. The default is TRUE.
...	Additional slots and values passed to struct_class.

Value

A DatasetExperiment_dist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
C = DatasetExperiment_dist(factor_name='Class')
chart_plot(C,D)
```

DatasetExperiment_factor_boxplot

Factor boxplot

Description

Boxplot for a feature to visualise the distribution of values within each group

Usage

```
DatasetExperiment_factor_boxplot(feature_to_plot, factor_names, ...)
```

Arguments

feature_to_plot	(character, numeric, integer) The name of the plotted feature.
factor_names	(character) The name of sample meta column(s) to use.
...	Additional slots and values passed to struct_class.

Value

A DatasetExperiment_factor_boxplot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
C = DatasetExperiment_factor_boxplot(factor_names='Species',feature_to_plot='Petal.Width')
chart_plot(C,D)
```

DatasetExperiment_heatmap

DatasetExperiment heatmap

Description

A heatmap to visualise the measured values in a data matrix.

Usage

```
DatasetExperiment_heatmap(na_colour = "#FF00E4", ...)
```

Arguments

na_colour	(character) The hex colour code used to plot missing values. The default is "#FF00E4".
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- reshape2

Value

A DatasetExperiment_heatmap object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

Wickham H (2007). "Reshaping Data with the reshape Package." *Journal of Statistical Software*, *21*(12), 1-20. http://www.jstatsoft.org/v21/i12/.

Examples

```
D = iris_DatasetExperiment()
C = DatasetExperiment_heatmap()
chart_plot(C,D)
```

DFA *Discriminant Factor Analysis*

Description

Discriminant Factor Analysis (DFA) is a supervised classification method. Using a linear combination of the input variables, DFA finds new orthogonal axes (canonical values) to minimize the variance within each given class and maximize variance between classes.

Usage

```
DFA(factor_name, number_components = 2, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
number_components	(numeric, integer) The number of DFA components calculated. The default is 2.
...	Additional slots and values passed to struct_class.

Value

A DFA object with the following output slots:

scores	(DatasetExperiment)
loadings	(data.frame)
eigenvalues	(data.frame)
that	(DatasetExperiment)

References

Manly B (1986). *Multivariate Statistical Methods: A Primer*. Chapman and Hall, Boca Raton.

Examples

```
D = iris_DatasetExperiment()
M = DFA(factor_name='Species')
M = model_apply(M,D)
```

`dfa_scores_plot` *DFA scores plot*

Description

A scatter plot of the selected DFA components.

Usage

```
dfa_scores_plot(
  components = c(1, 2),
  points_to_label = "none",
  factor_name,
  ellipse = "all",
  label_filter = character(0),
  label_factor = "rownames",
  label_size = 3.88,
  ...
)
```

Arguments

`components`	(numeric) The components selected for plotting. The default is `c(1, 2)`.
`points_to_label`	(character) Points to label. Allowed values are limited to the following: • `"none"`: No samples labels are displayed. • `"all"`: The labels for all samples are displayed. • `"outliers"`: Labels for for potential outlier samples are displayed. The default is `"none"`.
`factor_name`	(character) The name of a sample-meta column to use.
`ellipse`	(character) Plot ellipses. Allowed values are limited to the following: • `"all"`: Hotelling T2 ellipses (p=0.95) are plotted for all groups and all samples. • `"group"`: Hotelling T2 ellipses (p=0.95) are plotted for all groups. • `"none"`: Ellipses are not included on the plot. • `"sample"`: A Hotelling T2 ellipse (p=0.95) is plotted for all samples (ignoring group). The default is `"all"`.
`label_filter`	(character) Labels are only plotted for the named groups. If zero-length then all groups are included. The default is `character(0)`.
`label_factor`	(character) The column name of sample_meta to use for labelling samples on the plot. "rownames" will use the row names from sample_meta. The default is `"rownames"`.

label_size	(numeric) The text size of labels. Note this is not in Font Units. The default is 3.88.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- scales
- ggplot2

Value

A dfa_scores_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

Wickham H, Seidel D (2022). *scales: Scale Functions for Visualization*. R package version 1.2.1, https://CRAN.R-project.org/package=scales.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre() + DFA(factor_name='Species')
M = model_apply(M,D)
C = dfa_scores_plot(factor_name = 'Species')
chart_plot(C,M[2])
```

---

dratio_filter *Dispersion ratio filter*

Description

The dispersion ratio (d-ratio) compares the standard deviation (or non-parametric equivalent) of the Quality Control (QC) samples relative to the standard deviation (or non-parametric equivalent) of the samples for each feature. If the d-ratio is greater than a predefined threshold then the observed sample variance could be due to technical variance and the feature is removed.

Usage

```
dratio_filter(
  threshold = 20,
  qc_label = "QC",
  factor_name,
  method = "ratio",
  dispersion = "sd",
  ...
)
```

Arguments

`threshold`	(numeric) The threshold above which features are removed. The default is `20`.
`qc_label`	(character) The label used to identify QC samples. The default is `"QC"`.
`factor_name`	(character) The name of a sample-meta column to use.
`method`	(character) dratio method. Allowed values are limited to the following: • `"ratio"`: Dispersion of the QCs divided by the dispersion of the samples. Corresponds to Eq 4 in Broadhurst et al (2018). • `"euclidean"`: Dispersion of the QCs divided by the euclidean length of the total dispersion. Total dispersion is estimated from the QC and Sample dispersion by assuming that they are orthogonal. Corresponds to Eq 5 in Broadhurst et al (2018). The default is `"ratio"`.
`dispersion`	(character) Dispersion method. Allowed values are limited to the following: • `"sd"`: Dispersion is estimated using the standard deviation. • `"mad"`: Dispersion is estimated using the median absolute deviation. The default is `"sd"`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `dratio_filter` object with the following `output` slots:

`filtered`	(DatasetExperiment) A DatasetExperiment object containing the filtered data.
`flags`	(data.frame) Flag indicating whether the feature was rejected by the filter or not.
`d_ratio`	(data.frame)

References

Broadhurst D, Goodacre R, Reinke SN, Kuligowski J, Wilson ID, Lewis MR, Dunn WB (2018). "Guidelines and considerations for the use of system suitability and quality control samples in mass spectrometry assays applied in untargeted clinical metabolomic studies." *Metabolomics*, *14*(6).

Examples

```
D = MTBLS79_DatasetExperiment()
M = dratio_filter(threshold=20,qc_label='QC',factor_name='Class')
M = model_apply(M,D)
```

`equal_split` *Equal group sized sampling*

Description

Samples are randomly chosen from each level such that the training set has equal numbers of samples for all levels. The number of samples is based on the input proportion and the smallest group size.

Usage

```
equal_split(p_train = 1, factor_name, ...)
```

Arguments

`p_train`	(numeric) The proportion of samples selected for the training set. The default is 1.
`factor_name`	(character) The name of a sample-meta column to use.
`...`	Additional slots and values passed to `struct_class`.

Value

A `equal_split` object with the following `output` slots:

`training`	(DatasetExperiment) A DatasetExperiment object containing samples selected for the training set.
`testing`	(DatasetExperiment) A DatasetExperiment object containing samples selected for the testing set.

Examples

```
D = iris_DatasetExperiment()
M = equal_split(factor_name='Species')
M = model_apply(M,D)
```

feature_boxplot *Feature boxplot*

Description

A boxplot to visualise the distribution of values within a feature.

Usage

```
feature_boxplot(
  label_outliers = TRUE,
  feature_to_plot,
  factor_name,
  show_counts = TRUE,
  style = "boxplot",
  jitter = FALSE,
  fill = FALSE,
  ...
)
```

Arguments

label_outliers	(logical) Label outliers. Allowed values are limited to the following: • "TRUE": The index for outlier samples is included on the plot. • "FALSE": No labels are displayed. The default is TRUE.
feature_to_plot	(character, numeric, integer) The column name of the plotted feature.
factor_name	(character) The name of a sample-meta column to use.
show_counts	(logical) Show counts. Allowed values are limited to the following: • "TRUE": The number of samples for each box is displayed. • "FALSE": The number of samples for each box is not displayed. The default is TRUE.
style	(character) Plot style. Allowed values are limited to the following: • "boxplot": Boxplot style. • "violin": Violon plot style. The default is "boxplot".
jitter	(logical) Include points plotted with added jitter. The default is FALSE.
fill	(logical) Block fill the boxes or violins with the group colour. The default is FALSE.
...	Additional slots and values passed to struct_class.

Value

A feature_boxplot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment
C = feature_boxplot(factor_name='Species',feature_to_plot='Petal.Width')
chart_plot(C,D)
```

---

feature_profile *Feature profile*

Description

A plot visualising the change in intensity of a feature with a continuous variable such as time, dose, or run order.

Usage

```
feature_profile(
  run_order,
  qc_label,
  qc_column,
  colour_by,
  feature_to_plot,
  plot_sd = FALSE,
  ...
)
```

Arguments

run_order	(character) The sample-meta column name containing run order.
qc_label	(character) The label used to identify QC samples.
qc_column	(character) The sample-meta column name containing the labels used to identify QC samples.
colour_by	(character) The sample-meta column name to used to colour the plot.
feature_to_plot	(numeric, character, integer) The name or column id of the plotted feature.
plot_sd	(logical) Plot standard deviation. Allowed values are limited to the following: • "TRUE": Standard deviation of samples and QCs are included on the plot. • "FALSE": Standard deviation is not plotted. The default is FALSE.
...	Additional slots and values passed to struct_class.

Value

A `feature_profile` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
C = feature_profile(run_order='run_order',
    qc_label='QC',
    qc_column='Class',
    colour_by='Class',
    feature_to_plot=1)
chart_plot(C,D)
```

`feature_profile_array` *Feature profile*

Description

A plot visualising the change in intensity of a feature with a continuous variable such as time, dose, or run order.

Usage

```
feature_profile_array(
  run_order,
  qc_label,
  qc_column,
  colour_by,
  feature_to_plot,
  nrow = 5,
  log = TRUE,
  ...
)
```

Arguments

`run_order`	(character) The sample-meta column name containing run order.
`qc_label`	(character) The label used to identify QC samples.
`qc_column`	(character) The sample-meta column name containing the labels used to identify QC samples.
`colour_by`	(character) The sample-meta column name to used to colour the plot.
`feature_to_plot`	(numeric, character, integer) The name or column id of the plotted feature.
`nrow`	(numeric, integer) The number of rows in the plot. The default is `5`.
`log`	(logical) Log transform. Allowed values are limited to the following:

- "TRUE": The data is log tranformed before plotting.
- "FALSE": The data is not transformed before plotting.

The default is TRUE.

... Additional slots and values passed to struct_class.

Value

A feature_profile_array object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
C = feature_profile_array(
    run_order='run_order',
    qc_label='QC',
    qc_column='Class',
    colour_by='Class',
    feature_to_plot=1:3,
    nrow=1,
    log=TRUE)
chart_plot(C,D)
```

---

filter_by_name *Filter by name*

Description

Filter samples/variables by row/column name, index or logicals.

Usage

```
filter_by_name(mode = "exclude", dimension = "sample", names, ...)
```

Arguments

mode	(character) The filtering mode controls whether samples/features are mode="included" or mode="excluded" based on their name. The default is "exclude".
dimension	(character) The filtering dimensions controls whether dimension="sample" or dimension="variable" are filtered based on their name. The default is "sample".
names	(character, numeric, logical) The name of features/samples to be filtered. Must be an exact match. Can also provide indexes (numeric) or logical.
...	Additional slots and values passed to struct_class.

Value

A filter_by_name object with the following output slots:

filtered (DatasetExperiment)

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_by_name(mode='exclude',dimension='variable',names=c(1,2,3))
M = model_apply(M,D)
```

---

filter_na_count *Minimum number of measured values filter*

Description

The number of measured values is counted for each feature, and any feature with less than a predefined minimum number of values in each group is removed. If there are several factors, then the threshold is applied so that the minimum number of samples is present for all combinations (interactions) of groups.

Usage

```
filter_na_count(threshold, factor_name, ...)
```

Arguments

threshold	(numeric) The minimum number of samples in each group/interaction.
factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A filter_na_count object with the following output slots:

filtered	(DatasetExperiment) A DatasetExperiment object containing the filtered data.
count	(data.frame) The number of measured values in each group/interaction.
na_count	(data.frame) The number of missing values in each group/interaction.
flags	(data.frame) Flags to indicate which features were removed.

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_na_count(threshold=3,factor_name='Class')
M = model_apply(M,D)
```

filter_smeta *Filter by sample meta data*

Description

The data is filtered by so that the named levels of a factor are included/excluded from the dataset.

Usage

```
filter_smeta(mode = "include", levels, factor_name, ...)
```

Arguments

mode	(character) Mode of action. Allowed values are limited to the following: • "include": Samples in the specified levels are retained. • "exclude": Samples in the specified levels are excluded. The default is "include".
levels	(character) The level name(s) for filtering.
factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A filter_smeta object with the following output slots:

filtered (DatasetExperiment)

Examples

```
D = MTBLS79_DatasetExperiment()
M = filter_smeta(mode='exclude',levels='QC',factor_name='QC')
M = model_apply(M,D)
```

fisher_exact *Fisher Exact Test*

Description

A fisher exact test is used to analyse contingency tables by comparing the number of correctly/incorrectly predicted group labels. A multiple test corrected p-value indicates whether the number of measured values is significantly different between groups.

Usage

```
fisher_exact(alpha = 0.05, mtc = "fdr", factor_name, factor_pred, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_name	(character) The name of a sample-meta column to use.
factor_pred	(data.frame) A data.frame, where each column is a factor of predicted group labels to compare with the true groups labels.
...	Additional slots and values passed to struct_class.

Value

A fisher_exact object with the following output slots:

p_value	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

Examples

```
# load some data
D=MTBLS79_DatasetExperiment()

# prepare predictions based on NA
pred=as.data.frame(is.na(D$data))
pred=lapply(pred,factor,levels=c(TRUE,FALSE))
pred=as.data.frame(pred)

# apply method
M = fisher_exact(alpha=0.05,mtc='fdr',factor_name='Class',factor_pred=pred)
M=model_apply(M,D)
```

`fold_change` *Fold change*

Description

Fold change is the relative change in mean (or non-parametric equivalent) intensities of a feature between all pairs of levels in a factor.

Usage

```
fold_change(
  factor_name,
  paired = FALSE,
  sample_name = character(0),
  threshold = 2,
  control_group = character(0),
  method = "geometric",
  conf_level = 0.95,
  ...
)
```

Arguments

`factor_name`	(character) The name of a sample-meta column to use.
`paired`	(logical) Paired fold change. Allowed values are limited to the following: • `"TRUE"`: Fold change is calculated taking into account paired sampling. • `"FALSE"`: Fold change is calculated assuming there is no paired sampling. The default is `FALSE`.
`sample_name`	(character) The name of a sample_meta column containing sample identifiers for paired sampling. The default is `character(0)`.
`threshold`	(numeric) The fold change threshold for labelling features as significant. The default is `2`.
`control_group`	(character) The level name of the group used in the denominator (where possible) when computing fold change. The default is `character(0)`.
`method`	(character) Fold change method. Allowed values are limited to the following: • `"geometric"`: A log transform is applied before using group means to calculate fold change. In the non-tranformedspace this is equivalent to using geometric group means. Confidence intervals for independant and paired sampling are estimated using standard error of the mean in log transformed space before being transformed back to the original space. • `"median"`: The group medians and the method described by Price and Bonett is used to estimate confidence intervals. For paired data standard error of the median is used to estimate confidence intervals from the median fold change of all pairs.

- "mean": The group means and the method described by Price and Bonnet is used to estimate confidence intervals. For paired data standard error of the mean is used to estimate confidence intervals from the mean fold change of all pairs.

The default is "geometric".

conf_level	(numeric) The confidence level of the interval. The default is 0.95.
...	Additional slots and values passed to struct_class.

Value

A fold_change object with the following output slots:

fold_change	(data.frame) The fold change between groups.
lower_ci	(data.frame) Lower confidence interval for fold change.
upper_ci	(data.frame) Upper confidence interval for fold change.
significant	(data.frame) A logical indictor of whether the calculated fold change including the estimated confidence limit

References

Price Jr RM, Bonett DG (2020). "Confidence Intervals for Ratios of Means and Medians." *Journal of Educational and Behavioral Statistics*, *45*(6), 750-770.

Examples

```
D = MTBLS79_DatasetExperiment()
M = fold_change(factor_name='Class')
M = model_apply(M,D)
```

---

fold_change_int *Fold change for interactions between factors*

Description

For more than one factor the fold change calculation is extended to include all combinations of levels (interactions) of all factors. Paired fold changes are not possible for this computation.

Usage

```
fold_change_int(
  factor_name,
  threshold = 2,
  control_group = character(0),
  method = "geometric",
  conf_level = 0.95,
  ...
)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
threshold	(numeric) The fold change threshold for labelling features as significant. The default is 2.
control_group	(character) The level names of the groups used in the denominator (where possible) when computing fold change. One level for each factor, assumed to be in the same order as factor_name. The default is character(0).
method	(character) Fold change method. Allowed values are limited to the following: • "geometric": A log transform is applied before using group means to calculate fold change. In the non-tranformedspace this is equivalent to using geometric group means. Confidence intervals for independant and paired sampling are estimated using standard error of the mean in log transformed space before being transformed back to the original space. • "median": The group medians and the method described by Price and Bonett is used to estimate confidence intervals. For paired data standard error of the median is used to estimate confidence intervals from the median fold change of all pairs. • "mean": The group means and the method described by Price and Bonnet is used to estimate confidence intervals. For paired data standard error of the mean is used to estimate confidence intervals from the mean fold change of all pairs. The default is "geometric".
conf_level	(numeric) The confidence level of the interval. The default is 0.95.
...	Additional slots and values passed to struct_class.

Value

A fold_change_int object with the following output slots:

fold_change	(data.frame) The fold change between groups.
lower_ci	(data.frame) Lower confidence interval for fold change.
upper_ci	(data.frame) Upper confidence interval for fold change.
significant	(data.frame) A logical indictor of whether the calculated fold change including the estimated confidence limit

References

Lloyd GR, Jankevics A, Weber RJM (2020). "struct: an R/Bioconductor-based framework for standardized metabolomics data analysis and beyond." *Bioinformatics*, *36*(22-23), 5551-5552. https://doi.org/10.1093/bioinformatics/btaa1031.

Examples

```
D = MTBLS79_DatasetExperiment()
D=D[,1:10,drop=FALSE]
M = filter_smeta(mode='exclude',levels='QC',factor_name='Class') +
    fold_change_int(factor_name=c('Class','Batch'))
M = model_apply(M,D)
```

fold_change_plot *Fold change plot*

Description

A plot of fold changes calculated for a chosen subset of features. A predefined fold change threshold is indicated by shaded regions.

Usage

```
fold_change_plot(number_features = 20, orientation = "portrait", ...)
```

Arguments

number_features	(numeric) The number randomly selected features to plot, or a list of column numbers. The default is 20.
orientation	(character) Plot orientation. Allowed values are limited to the following: • "landscape": Features are plotted on the y-axis. • "portrait": Features are plotted on the x-axis. The default is "portrait".
...	Additional slots and values passed to struct_class.

Value

A fold_change_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = fold_change_plot()
```

forward_selection_by_rank *Forward selection by rank*

Description

A model is trained and performance metric computed by including increasing numbers of features in the model. The features to be included in each step are defined by their rank, which is computed from another variable e.g. VIP score. An "optimal"subset of features is suggested by minimising the input performance metric.

Usage

```
forward_selection_by_rank(
  min_no_vars = 1,
  max_no_vars = 100,
  step_size = 1,
  factor_name,
  variable_rank,
  ...
)
```

Arguments

min_no_vars	(numeric) The minimum number of variables to include in the model. The default is 1.
max_no_vars	(numeric) The maximum number of variables to include in the model. The default is 100.
step_size	(numeric) The incremental change in number of features in the model. The default is 1.
factor_name	(character) The name of a sample-meta column to use.
variable_rank	(numeric, integer) The values used to rank the features.
...	Additional slots and values passed to struct_class.

Value

A forward_selection_by_rank object with the following output slots:

metric	(data.frame) The value of the computed metric for each model. For nested models the metric is averaged.
results	(data.frame) The predicted outputs from collated from all models computed during forward selection.
chosen_vars	(numeric, integer) The column number of the variables chosen for the best performing model.
smoothed	(numeric) The value of the performance metric for each evaluated model after smoothing.
searchlist	(numeric) The maxmimum rank of features included in each model.

Examples

```
# some data
D = MTBLS79_DatasetExperiment(filtered=TRUE)

# normalise, impute and scale then remove QCs
P = pqn_norm(qc_label='QC',factor_name='Class') +
    knn_impute(neighbours=5) +
    glog_transform(qc_label='QC',factor_name='Class') +
    filter_smeta(mode='exclude',levels='QC',factor_name='Class')
P = model_apply(P,D)
D = predicted(P)

# forward selection using a PLSDA model
M = forward_selection_by_rank(factor_name='Class',
```

```
                              min_no_vars=2,
                              max_no_vars=11,
                              variable_rank=1:2063) *
    (mean_centre() + PLSDA(number_components=1,
                           factor_name='Class'))
M = run(M,D,balanced_accuracy())
```

`fs_line` *Forward selection line plot*

Description

A line plot for forward selection. The computed model performance metric is plotted against the number of features included in the model.

Usage

```
fs_line(...)
```

Arguments

`...`	Additional slots and values passed to `struct_class`.

Value

A `fs_line` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

Examples

```
# some data
D = MTBLS79_DatasetExperiment(filtered=TRUE)

# normalise, impute and scale then remove QCs
P = pqn_norm(qc_label='QC',factor_name='Class') +
    knn_impute(neighbours=5) +
    glog_transform(qc_label='QC',factor_name='Class') +
    filter_smeta(mode='exclude',levels='QC',factor_name='Class')
P = model_apply(P,D)
D = predicted(P)

# forward selection using a PLSDA model
M = forward_selection_by_rank(factor_name='Class',
                              min_no_vars=2,
                              max_no_vars=11,
                              variable_rank=1:2063) *
    (mean_centre() + PLSDA(number_components=1,
                           factor_name='Class'))
M = run(M,D,balanced_accuracy())
```

```
# chart
C = fs_line()
chart_plot(C,M)
```

`glog_opt_plot` *Glog optimisation*

Description

A plot of the sum of squares error (SSE) vs different values of lambda for the glog transform. The indicated optimum value for lambda minimises the SSE.

Usage

```
glog_opt_plot(plot_grid = 100, ...)
```

Arguments

`plot_grid`	(numeric) The default is `100`.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A `glog_opt_plot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
D = iris_DatasetExperiment()
M = glog_transform(qc_label='versicolor',factor_name='Species')
M = model_apply(M,D)
C = glog_opt_plot()
chart_plot(C,M,D)
```

`glog_transform` *Generalised logarithmic transform*

Description

The generalised logarithm (glog) transformation applies a log transformation while applying an offset to account for technical variation.

Usage

```
glog_transform(qc_label = "QC", factor_name, lambda = NULL, ...)
```

Arguments

`qc_label`	(character) The label used to identify QC samples. The default is `"QC"`.
`factor_name`	(character) The name of a sample-meta column to use.
`lambda`	(numeric, NULL) The value of lambda to use. If NULL then the pmp package will be used to determine an "optimal" value for lambda. The default is `NULL`.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A `glog_transform` object with the following `output` slots:

`transformed`	(DatasetExperiment) A DatasetExperiment object containing the glog transformed data.
`error_flag`	(logical) A logical indicating whether the glog optimisation for lambda was successful. If not then PMP return

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Durbin B, Hardin J, Hawkins D, Rocke D (2002). "A variance-stabilizing transformation for gene-expression microarray data." *Bioinformatics*, *18*(Suppl 1), S105-S110.

Parsons HM, Ludwig C, Gunther UL, Viant MR (2007). "Improved classification accuracy in 1- and ', '2-dimensional NMR metabolomics data using the variance ', 'stabilising generalised logarithm transformation." *Bioinformatics*, *8*(1), 234.

Examples

```
D = iris_DatasetExperiment()
M = glog_transform(qc_label='versicolor',factor_name='Species')
M = model_apply(M,D)
```

`grid_search_1d` *One dimensional grid search*

Description

A one dimensional grid search calculates a performance metric for a model at evenly spaced values for a model input parameter. The "optimum" value for the parameter is suggested as the one which maximises performance, or minimises error (whichever is appropriate for the chosen metric)

Usage

```
grid_search_1d(
  param_to_optimise,
  search_values,
  model_index,
  factor_name,
  max_min = "min",
  ...
)
```

Arguments

`param_to_optimise`
: (character) The name of the model input parameter that is the focus of the search.

`search_values`
: (ANY) The values of the input parameter being optimised.

`model_index`
: (numeric, integer) The index of the model in the sequence that uses the parameter being optimised.

`factor_name`
: (character) The name of a sample-meta column to use.

`max_min`
: (character) Maximise or minimise. Allowed values are limited to the following:

  - `"max"`: The optimium parameter value is suggested based on maximising the performance metric.
  - `"min"`: The optimium parameter value is suggested based on minimising the performance metric.

  The default is `"min"`.

`...`
: Additional slots and values passed to `struct_class`.

Value

A grid_search_1d object with the following output slots:

results	(data.frame)
metric	(data.frame)
optimum_value	(numeric)

Examples

```
D = MTBLS79_DatasetExperiment()
# some preprocessing
M = pqn_norm(qc_label='QC',factor_name='Class') +
    knn_impute() +
    glog_transform(qc_label='QC',factor_name='Class') +
    filter_smeta(factor_name='Class',levels='QC',mode='exclude')
M=model_apply(M,D)
D=predicted(M)

# reduce number of features for this example
D=D[,1:10]

# optmise number of components for PLS model
I = grid_search_1d(param_to_optimise='number_components',search_values=1:5,
        model_index=2,factor_name='Class') *
        (mean_centre()+PLSDA(factor_name='Class'))
I = run(I,D,balanced_accuracy())
```

---

gs_line *Grid search line plot*

Description

A plot of the calculated performance metric against the model input parameter values used to train the model. The optimum parameter value is indicated based on minimising (or maximising) the chosen metric.

Usage

```
gs_line(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A gs_line object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = gs_line()
```

HCA *Hierarchical Cluster Analysis*

Description

Hierarchical Cluster Analysis is a numerical technique that uses agglomerative clustering to identify clusters or groupings of samples.

Usage

```
HCA(
  dist_method = "euclidean",
  cluster_method = "complete",
  minkowski_power = 2,
  factor_name,
  ...
)
```

Arguments

`dist_method`	(character) Distance measure. Allowed values are limited to the following: • `"euclidean"`: The euclidean distance (2 norm). • `"maximum"`: The maximum distance. • `"manhattan"`: The absolute distance (1 norm). • `"canberra"`: A weighted version of the mahattan distance. • `"minkowski"`: A generalisation of manhattan and euclidean distance to nth norm. The default is `"euclidean"`.
`cluster_method`	(character) Agglomeration method. Allowed values are limited to the following: • `"ward.D"`: Ward clustering. • `"ward.D2"`: Ward clustering using sqaured distances. • `"single"`: Single linkage. • `"complete"`: Complete linkage. • `"average"`: Average linkage (UPGMA). • `"mcquitty"`: McQuitty linkage (WPGMA). • `"median"`: Median linkage (WPGMC). • `"centroid"`: Centroid linkage (UPGMC). The default is `"complete"`.
`minkowski_power`	(numeric) The default is `2`.
`factor_name`	(character) The name of a sample-meta column to use.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- stats

Value

A HCA object with the following output slots:

dist_matrix	(dist) An object containing pairwise distance information between samples.
hclust	(hclust) An object of class hclust which describes the tree produced by the clustering process.
factor_df	(data.frame)

References

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Examples

```
D = iris_DatasetExperiment()
M = HCA(factor_name='Species')
M = model_apply(M,D)
```

---

hca_dendrogram *HCA dendrogram*

Description

A dendrogram visualising the clustering by HCA.

Usage

```
hca_dendrogram(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- ggdendro

Value

A hca_dendrogram object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

de Vries A, Ripley BD (2022). *ggdendro: Create Dendrograms and Tree Diagrams Using 'ggplot2'*. R package version 0.1.23, https://CRAN.R-project.org/package=ggdendro.

Examples

```
C = hca_dendrogram()
```

HSD *Tukey's Honest Significant Difference*

Description

Tukey's HSD post hoc test is a modified t-test applied for all features to all pairs of levels in a factor. It is used to determine which groups are different (if any). A multiple test corrected p-value is computed to indicate which groups are significantly different to the others for each feature.

Usage

```
HSD(alpha = 0.05, mtc = "fdr", formula, unbalanced = FALSE, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
formula	(formula) A symbolic description of the model to be fitted.
unbalanced	(logical) Unbalanced model. Allowed values are limited to the following: • "TRUE": A correction is applied for unbalanced designs. • "FALSE": No correction is applied for unbalanced designs. The default is FALSE.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- agricolae

Value

A HSD object with the following output slots:

difference	(data.frame)
UCL	(data.frame)
LCL	(data.frame)
p_value	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

References

de Mendiburu F (2023). *agricolae: Statistical Procedures for Agricultural Research*. R package version 1.3-6, https://CRAN.R-project.org/package=agricolae.

Examples

```
D = iris_DatasetExperiment()
M = HSD(formula=y~Species)
M = model_apply(M,D)
```

HSDEM *Tukey's Honest Significant Difference using estimated marginal means*

Description

Tukey's HSD post hoc test is a modified t-test applied for all features to all pairs of levels in a factor. It is used to determine which groups are different (if any). A multiple test corrected p-value is computed to indicate which groups are significantly different to the others for each feature. For mixed effects models estimated marginal means are used.

Usage

```
HSDEM(alpha = 0.05, mtc = "fdr", formula, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following:

- "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons.
- "fdr": Benjamini and Hochberg False Discovery Rate correction.
- "none": No correction.

The default is "fdr".

formula (formula) A symbolic description of the model to be fitted.

... Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- emmeans
- nlme

Value

A HSDEM object with the following output slots:

p_value (data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
significant (data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

References

Lenth R (2023). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.8.7, https://CRAN.R-project.org/package=emmeans.

Pinheiro J, Bates D, R Core Team (2023). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-162, https://CRAN.R-project.org/package=nlme.

Pinheiro JC, Bates DM (2000). *Mixed-Effects Models in S and S-PLUS*. Springer, New York. doi:10.1007/b98882 https://doi.org/10.1007/b98882.

Examples

```
D = iris_DatasetExperiment()
D$sample_meta$id=rownames(D) # dummy id column
M = HSDEM(formula = y~Species+ Error(id/Species))
M = model_apply(M,D)
```

kfoldxcv_grid *k-fold cross validation plot*

Description

A graphic for visualising the true class and the predicted class of samples in all groups for all cross-validation folds.

Usage

```
kfoldxcv_grid(factor_name, level, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
level	(character) The level/group to plot.
...	Additional slots and values passed to struct_class.

Value

A kfoldxcv_grid object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
I = kfold_xval(factor_name='Species') *
    (mean_centre() + PLSDA(factor_name='Species'))
I = run(I,D,balanced_accuracy())

C = kfoldxcv_grid(factor_name='Species',level='setosa')
chart_plot(C,I)
```

kfoldxcv_metric *kfoldxcv metric plot*

Description

A boxplot of the performance metric computed for each fold of a k-fold cross-validation.

Usage

```
kfoldxcv_metric(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A kfoldxcv_metric object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = kfoldxcv_metric()
```

---

kfold_xval *k-fold cross-validation*

Description

k-fold cross-validation is an iterative approach applied to validate models. The samples are divided into k "folds", or subsets. Each subset is excluded from model training and used for model validation once, resulting in a single left-out prediction for each sample. Model performance metrics are then computed for the training and test sets across all folds.

Usage

```
kfold_xval(folds = 10, method = "venetian", factor_name, collect = NULL, ...)
```

Arguments

folds	(numeric, integer) The number of cross-validation folds. The default is 10.
method	(character) Fold selection method. Allowed values are limited to the following: • "venetian": Every nth sample is assigned to the same fold, where n is the number of folds. • "blocks": Blocks of adjacent samples are assigned to the same fold. • "random": Samples are randomly assigned to a fold. The default is "venetian".
factor_name	(character) The name of a sample-meta column to use.
collect	(NULL, character) The name of a model output to collect over all bootstrap repetitions, in addition to the input metric. The default is NULL.
...	Additional slots and values passed to struct_class.

Value

A kfold_xval object with the following output slots:

results	(data.frame)
metric	(data.frame)
metric.train	(numeric)
metric.test	(numeric)
collected	(list)

Examples

```
D = iris_DatasetExperiment()
I = kfold_xval(factor_name='Species') *
    (mean_centre() + PLSDA(factor_name='Species'))
I = run(I,D,balanced_accuracy())
```

---

knn_impute *kNN missing value imputation*

Description

k-nearest neighbour missing value imputation replaces missing values in the data with the average of a predefined number of the most similar neighbours for which the value is present

Usage

```
knn_impute(
  neighbours = 5,
  sample_max = 50,
  feature_max = 50,
  by = "features",
  ...
)
```

Arguments

neighbours	(numeric) The number of neighbours (k) to use for imputation. The default is 5.
sample_max	(numeric) The maximum percent missing values per sample. The default is 50.
feature_max	(numeric) The maximum percent missing values per feature. The default is 50.
by	(character) Impute using similar "samples" or "features". Default features. The default is "features".
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A knn_impute object with the following output slots:

imputed (DatasetExperiment) A DatasetExperiment object containing the data where missing values have been imputed.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
M = knn_impute()
```

---

kw_p_hist *Histogram of p values*

Description

A histogram of the p-values computed by the kruskal-wallis method

Usage

```
kw_p_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A kw_p_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = kw_p_hist()
```

kw_rank_sum *Kruskal-Wallis rank sum test*

Description

The Kruskal-Wallis test is a univariate hypothesis testing method that allows multiple (n>=2) groups to be compared without making the assumption that values are normally distributed. It is the non-parametric equivalent of a 1-way ANOVA. The test is applied to all variables/features individually, and multiple test corrected p-values are computed to indicate the significance of variables/features.

Usage

```
kw_rank_sum(alpha = 0.05, mtc = "fdr", factor_names, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_names	(character) The name of sample meta column(s) to use.
...	Additional slots and values passed to struct_class.

Value

A kw_rank_sum object with the following output slots:

test_statistic	(data.frame) The value of the calculated statistic which is converted to a p-value when compared to a chi2-
p_value	(data.frame) The probability of observing the calculated statistic.
dof	(numeric) The number of degrees of freedom used to calculate the test statistic.
significant	(data.frame) TRUE if the calculated p-value is less than the supplied threhold (alpha).
estimates	(data.frame)

Examples

```
D = iris_DatasetExperiment()
M = kw_rank_sum(factor_names='Species')
M = model_apply(M,D)
```

linear_model *Linear model*

Description

Linear models can be used to carry out regression, single stratum analysis of variance and analysis of covariance.

Usage

```
linear_model(formula, na_action = "na.omit", contrasts = list(), ...)
```

Arguments

formula	(formula) A symbolic description of the model to be fitted.
na_action	(character) NA action. Allowed values are limited to the following: • "na.omit": Incomplete cases are removed. • "na.fail": An error is thrown if NA are present. • "na.exclude": Incomplete cases are removed, and the output result is padded to the correct size using NA. • "na.pass": Does not apply a linear model if NA are present. The default is "na.omit".
contrasts	(list) The contrasts associated with a factor. The default is list.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- stats

Value

A linear_model object with the following output slots:

lm	(lm) The lm object for this model_.
coefficients	(numeric) The coefficients for the fitted model_.
residuals	(numeric) The residuals for the fitted model_.
fitted_values	(numeric) The fitted values for the data used to train the model_.
predicted_values	(numeric) The predicted values for new data using the fitted model_.
r_squared	(numeric) The value of R Squared for the fitted model_.
adj_r_squared	(numeric) The value ofAdjusted R Squared for the fitted model_.

References

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Examples

```
D = iris_DatasetExperiment()
M = linear_model(formula = y~Species)
```

`log_transform` *logarithm transform*

Description

A logarithmic transform is applied to all values in the data matrix.

Usage

```
log_transform(base = 10, ...)
```

Arguments

`base`	(numeric) The base of the logarithm used for the transform. The default is `10`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `log_transform` object with the following `output` slots:

`transformed`	(DatasetExperiment) A DatasetExperiment object containing the log transformed data.

struct object

Examples

```
M = log_transform()
```

mean_centre *Mean centre*

Description

The mean sample is subtracted from all samples in the data matrix. The features in the centred matrix all have zero mean.

Usage

```
mean_centre(mode = "data", ...)
```

Arguments

mode	(character) Mode of action. Allowed values are limited to the following: • "data": Centring is applied to the data block. • "sample_meta": Centring is applied to the sample_meta block. • "both": Centring is applied to both the data and the sample_meta blocks. The default is "data".
...	Additional slots and values passed to struct_class.

Value

A mean_centre object with the following output slots:

centred	(DatasetExperiment)
mean_data	(numeric)
mean_sample_meta	(numeric)

Examples

```
M = mean_centre()
```

mean_of_medians *Mean of medians*

Description

The data matrix is normalised by the mean of the median of each factor level.

Usage

```
mean_of_medians(factor_name, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A mean_of_medians object with the following output slots:

transformed	(DatasetExperiment) Data after the tranformation has been applied.

Examples

```
D = iris_DatasetExperiment()
M = mean_of_medians(factor_name='Species')
M = model_apply(M,D)
```

---

mixed_effect *Mixed effects model*

Description

A mixed effects model is an extension of ANOVA where there are both fixed and random effects.

Usage

```
mixed_effect(alpha = 0.05, mtc = "fdr", formula, ss_type = "marginal", ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
formula	(formula) A symbolic description of the model to be fitted.
ss_type	(character) Sum of squares type. Allowed values are limited to the following: • "marginal": Type III sum of squares. • "sequential": Type II sum of squares. The default is "marginal".
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- `nlme`
- `emmeans`

Value

A `mixed_effect` object with the following `output` slots:

`f_statistic`	(data.frame) The value of the calculated statistic.
`p_value`	(data.frame) The probability of observing the calculated statistic if the null hypothesis is true.
`significant`	(data.frame) True/False indicating whether the p-value computed for each variable is less than the threshold.

References

Pinheiro J, Bates D, R Core Team (2023). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-162, https://CRAN.R-project.org/package=nlme.

Pinheiro JC, Bates DM (2000). *Mixed-Effects Models in S and S-PLUS*. Springer, New York. doi:10.1007/b98882 https://doi.org/10.1007/b98882.

Lenth R (2023). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.8.7, https://CRAN.R-project.org/package=emmeans.

Fox J, Weisberg S (2019). *An R Companion to Applied Regression*, Third edition. Sage, Thousand Oaks CA. https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Examples

```
D = iris_DatasetExperiment()
D$sample_meta$id=rownames(D) # dummy id column
M = mixed_effect(formula = y~Species+ Error(id/Species))
M = model_apply(M,D)
```

`model_apply,ANOVA,DatasetExperiment-method`

Apply method

Description

Applies method to the input DatasetExperiment

Usage

```
## S4 method for signature 'ANOVA,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'HSD,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'mixed_effect,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'HSDEM,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'classical_lsq,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'confounders_clsq,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'constant_sum_norm,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'corr_coef,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'split_data,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'equal_split,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'filter_smeta,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'fisher_exact,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'fold_change,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'fold_change_int,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'HCA,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'knn_impute,DatasetExperiment'
model_apply(M, D)
```

```
## S4 method for signature 'kw_rank_sum,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'log_transform,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'mean_of_medians,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'nroot_transform,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'pairs_filter,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'prop_na,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'rsd_filter,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'sb_corr,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'stratified_split,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'tSNE,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'ttest,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'vec_norm,DatasetExperiment'
model_apply(M, D)

## S4 method for signature 'wilcox_test,DatasetExperiment'
model_apply(M, D)
```

Arguments

M	a method object
D	another object used by the first

Value

Returns a modified method object

Examples

```
M=model()
model_apply(M,DatasetExperiment())
```

`model_predict,DFA,DatasetExperiment-method`
Model prediction

Description

Apply a model using the input DatasetExperiment. Assumes the model is trained first.

Usage

```
## S4 method for signature 'DFA,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'PCA,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'PLSR,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'PLSDA,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'autoscale,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'blank_filter,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'constant_sum_norm,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'dratio_filter,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'filter_by_name,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'filter_na_count,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'filter_smeta,DatasetExperiment'
model_predict(M, D)
```

```
## S4 method for signature 'glog_transform,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'linear_model,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'mean_centre,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'mv_feature_filter,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'mv_sample_filter,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'OPLSR,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'OPLSDA,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'pareto_scale,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'pqn_norm,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'SVM,DatasetExperiment'
model_predict(M, D)

## S4 method for signature 'vec_norm,DatasetExperiment'
model_predict(M, D)
```

Arguments

M	a model object
D	a DatasetExperiment object

Value

Returns a modified model object

Examples

```
M = example_model()
M = model_predict(M,iris_DatasetExperiment())
```

model_reverse,autoscale,DatasetExperiment-method
Reverse preprocessing

Description

Reverse the effect of a preprocessing step on a DatasetExperiment.

Usage

```
## S4 method for signature 'autoscale,DatasetExperiment'
model_reverse(M, D)

## S4 method for signature 'mean_centre,DatasetExperiment'
model_reverse(M, D)
```

Arguments

M	a model object
D	a DatasetExperiment object

Value

Returns a modified DatasetExperiment object

Examples

```
M = example_model()
D = model_reverse(M,iris_DatasetExperiment())
```

model_train,DFA,DatasetExperiment-method
Train a model

Description

Trains a model using the input DatasetExperiment

Usage

```
## S4 method for signature 'DFA,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'PCA,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'PLSR,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'PLSDA,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'autoscale,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'blank_filter,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'constant_sum_norm,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'dratio_filter,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'filter_by_name,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'filter_na_count,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'filter_smeta,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'glog_transform,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'linear_model,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'mean_centre,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'mv_feature_filter,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'mv_sample_filter,DatasetExperiment'
model_train(M, D)
```

```
## S4 method for signature 'OPLSR,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'OPLSDA,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'pareto_scale,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'pqn_norm,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'SVM,DatasetExperiment'
model_train(M, D)

## S4 method for signature 'vec_norm,DatasetExperiment'
model_train(M, D)
```

Arguments

M	a model object
D	a DatasetExperiment object

Value

Returns a modified model object

Examples

```
M = example_model()
M = model_train(M,iris_DatasetExperiment())
```

`MTBLS79_DatasetExperiment` *MTBLS79: Direct infusion mass spectrometry metabolomics dataset: a benchmark for data processing and quality control*

Description

Direct-infusion mass spectrometry (DIMS) metabolomics is an important approach for characterising molecular responses of organisms to disease, drugs and the environment. Increasingly large-scale metabolomics studies are being conducted, necessitating improvements in both bioanalytical and computational workflows to maintain data quality. This dataset represents a systematic evaluation of the reproducibility of a multi-batch DIMS metabolomics study of cardiac tissue extracts. It comprises of twenty biological samples (cow vs. sheep) that were analysed repeatedly, in 8 batches across 7 days, together with a concurrent set of quality control (QC) samples. Data are presented from each step of the workflow and are available in MetaboLights (https://www.ebi.ac.uk/metabolights/MTBLS79)

Usage

```
MTBLS79_DatasetExperiment(filtered = FALSE)
```

Arguments

filtered	TRUE to load data with quality control filters already applied, or FALSE to load the unfiltered data. Default is FALSE. The raw data is available from (https://www.ebi.ac.uk/metabolights/MTBLS79) and as an R dataset in the pmp package, available on Bioconductor.

Value

DatasetExperiment object

Examples

```
D = MTBLS79_DatasetExperiment()
summary(D)
```

---

mv_boxplot *Missing value boxplots*

Description

Boxplots of the number of missing values per sample/feature.

Usage

```
mv_boxplot(
  label_outliers = TRUE,
  by_sample = TRUE,
  factor_name,
  show_counts = TRUE,
  ...
)
```

Arguments

label_outliers	(logical) Label outliers. Allowed values are limited to the following: • "TRUE": Sample labels for potential outliers are displayed on the plot. • "FALSE": Sample labels are not included on the plot. The default is TRUE.
by_sample	(logical) Plot by sample or by feature. Allowed values are limited to the following: • "TRUE": Missing values are plotted per sample. • "FALSE": Missing values are plotted per feature.

The default is TRUE.

factor_name (character) The name of a sample-meta column to use.

show_counts (logical) Show counts. Allowed values are limited to the following:

- "TRUE": The number of samples for each box is displayed.
- "FALSE": The number of samples for each box is not displayed.

The default is TRUE.

... Additional slots and values passed to struct_class.

Value

A mv_boxplot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = MTBLS79_DatasetExperiment()
C = mv_boxplot(factor_name='Class')
chart_plot(C,D)
```

---

mv_feature_filter *Filter features by missing values*

Description

Removes features where the percentage of non-missing values falls below a threshold.

Usage

```
mv_feature_filter(
  threshold = 20,
  qc_label = "QC",
  method = "QC",
  factor_name,
  ...
)
```

Arguments

threshold (numeric) The minimum percentage of non-missing values. The default is 20.

qc_label (character) The label used to identify QC/group samples when using the "QC" (within a named group) filtering method. The default is "QC".

method (character) Filtering method. Allowed values are limited to the following:

- "within_all": Features are removed if the threshold for non-missing values is not met for all groups.

- "within_one": Features are removed if the threshold for non-missing values is not met for at least one group.
- "QC": Features are removed if the threshold for non-missing values is not met for the named group.
- "across": The filter is applied ignoring sample group.

The default is "QC".

factor_name (character) The name of a sample-meta column to use.

... Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A mv_feature_filter object with the following output slots:

filtered (DatasetExperiment) A DatasetExperiment object containing the filtered data.
flags (data.frame) % missing values and a flag indicating whether the sample was rejected. 0 = rejected.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
D = iris_DatasetExperiment()
M = mv_feature_filter(factor_name='Species',qc_label='versicolor')
M = model_apply(M,D)
```

---

mv_feature_filter_hist

Histogram of missing values per feature

Description

A histogram of the proportion of missing values per feature.

Usage

```
mv_feature_filter_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A mv_feature_filter_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = mv_feature_filter_hist()
```

---

mv_histogram *Missing value histogram*

Description

A histogram of the numbers of missing values per sample/feature

Usage

```
mv_histogram(label_outliers = TRUE, by_sample = TRUE, ...)
```

Arguments

label_outliers	(logical) Label outliers. Allowed values are limited to the following: • "TRUE": Sample labels for potential outliers are displayed on the plot. • "FALSE": Sample labels are not included on the plot. The default is TRUE.
by_sample	(logical) Plot by sample or by feature. Allowed values are limited to the following: • "TRUE": Missing values are plotted per sample. • "FALSE": Missing values are plotted per feature. The default is TRUE.
...	additional slots and values passed to struct_class

Value

A mv_histogram object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

struct object

Examples

```
D = MTBLS79_DatasetExperiment()
C = mv_histogram(label_outliers=FALSE,by_sample=FALSE)
chart_plot(C,D)
```

`mv_sample_filter` *Missing value sample filter*

Description

Removes samples where the percent number of missing values exceeds a threshold.

Usage

```
mv_sample_filter(mv_threshold = 20, ...)
```

Arguments

`mv_threshold`	(numeric) The maximum percentage of features with missing values in a sample. The default is `20`.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A `mv_sample_filter` object with the following `output` slots:

`filtered`	(DatasetExperiment) A DatasetExperiment object containing the filtered data.
`flags`	(data.frame) A flag indicating whether the sample was rejected. 0 = rejected.
`percent_missing`	(data.frame) % missing values for each sample.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
C = mv_sample_filter()
```

mv_sample_filter_hist *Histogram of missing values per sample*

Description

A histogram of the the proportion of missing values per sample

Usage

```
mv_sample_filter_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A mv_sample_filter_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = mv_sample_filter_hist()
```

nroot_transform *nth root transform*

Description

All values in the data matrix are transformed by raising them to the power of 1/n.

Usage

```
nroot_transform(root = 2, ...)
```

Arguments

root	(numeric) The nth root used for the transform. The default is 2.
...	Additional slots and values passed to struct_class.

Value

A nroot_transform object with the following output slots:

transformed	(DatasetExperiment) A DatasetExperiment object containing the nth root transformed data.

Examples

```
M = nroot_transform()
```

`ontology_cache` *ontology cache*

Description

A cached list of ontology terms obtained from the ontology lookup service (OLS) for ontology terms specified for objects in `structToolbox`.

Usage

```
ontology_cache()
```

Value

list of cached ontology terms

See Also

ontology

Examples

```
cache = ontology_cache()
```

`OPLSDA` *Orthogonal Partial Least Squares regression*

Description

OPLS splits a data matrix into two parts. One part contains information orthogonal to the input vector, and the other is non-orthogonal.

Usage

```
OPLSDA(number_components = 1, factor_name, ...)
```

Arguments

`number_components`	(numeric, integer) The number of orthgonal components. The default is `1`.
`factor_name`	(character) The name of a sample-meta column to use.
`...`	Additional slots and values passed to `struct_class`.

Value

A OPLSDA object with the following output slots:

opls_model	(list)
filtered	(DatasetExperiment)
orthogonal	(DatasetExperiment)

Examples

```
M = OPLSR('number_components'=2,factor_name='Species')
```

OPLSR *Orthogonal Partial Least Squares regression*

Description

OPLS splits a data matrix into two parts. One part contains information orthogonal to the input vector, and the other is non-orthogonal.

Usage

```
OPLSR(number_components = 2, factor_name, ...)
```

Arguments

number_components	(numeric, integer) The number of orthgonal components. The default is 2.
factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A OPLSR object with the following output slots:

opls_model	(list)
filtered	(DatasetExperiment)
orthogonal	(DatasetExperiment)

Examples

```
M = OPLSR('number_components'=2,factor_name='Species')
```

pairs_filter *Pairs filter*

Description

This filter is used for study designs with paired sampling to ensure that measurements from the same source (e.g. patient) are represented in all factor levels and interactions.

Usage

```
pairs_filter(factor_name, sample_id, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
sample_id	(character) Name of sample meta column containing sample identifiers.
...	Additional slots and values passed to struct_class.

Value

A pairs_filter object with the following output slots:

filtered	(DatasetExperiment) A DatasetExperiment object after the filter has been applied.
flags	(data.frame) A data.frame indicating whether features were filtered from the DatasetExperiment.

struct object

Examples

```
M=pairs_filter(factor_name='Class',sample_id='ids')
```

pareto_scale *Pareto scaling*

Description

The mean sample is subtracted from all samples and then scaled by the square root of the standard deviation. The transformed data has zero mean.

Usage

```
pareto_scale(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A pareto_scale object with the following output slots:

scaled	(DatasetExperiment)
mean	(numeric)
sd	(numeric)

Examples

```
D = iris_DatasetExperiment()
M = pareto_scale()
M = model_train(M,D)
M = model_predict(M,D)
```

PCA *Principal Component Analysis (PCA)*

Description

PCA is a multivariate data reduction technique. It summarises the data in a smaller number of Principal Components that maximise variance.

Usage

```
PCA(number_components = 2, ...)
```

Arguments

number_components	(numeric, integer) The number of Principal Components calculated. The default is 2.
...	Additional slots and values passed to struct_class.

Value

A PCA object with the following output slots:

scores	(DatasetExperiment) A matrix of PCA scores where each column corresponds to a Principal Component.
loadings	(data.frame)
eigenvalues	(data.frame)
ssx	(numeric)
correlation	(data.frame)

`that` (DatasetExperiment)

Examples

```
M = PCA()
```

`pca_biplot` *PCA biplot*

Description

A scatter plot of the selected principal component scores overlaid with the corresponding principal component loadings.

Usage

```
pca_biplot(
  components = c(1, 2),
  points_to_label = "none",
  factor_name,
  scale_factor = 0.95,
  style = "points",
  label_features = FALSE,
  ...
)
```

Arguments

`components`	(numeric) The principal components used to generate the plot. The default is `c(1, 2)`.
`points_to_label`	(character) points_to_label. Allowed values are limited to the following: • `"none"`: No samples are labelled on the plot. • `"all"`: All samples are labelled on the plot. • `"outliers"`: Potential outliers are labelled on the plot. The default is `"none"`.
`factor_name`	(character) The name of a sample-meta column to use.
`scale_factor`	(numeric) The scaling factor applied to the loadings. The default is `0.95`.
`style`	(character) Plot style. Allowed values are limited to the following: • `"points"`: Loadings and scores are plotted as a scatter plot. • `"arrows"`: The loadings are plotted as arrow vectors. The default is `"points"`.
`label_features`	(logical) Add feature labels. Allowed values are limited to the following:

- "TRUE": Features are labelled.
- "FALSE": Features are not labelled.

The default is FALSE.

... Additional slots and values passed to struct_class.

Value

A pca_biplot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = pca_biplot(factor_name='Species')
```

---

pca_correlation_plot *PCA correlation plot*

Description

A plot of the correlation between the variables/features and the selected principal component scores. Features with high correlation are well represented by the selected component(s)

Usage

```
pca_correlation_plot(components = c(1, 2), ...)
```

Arguments

components (numeric) The Principal Components used to generate the plot. The default is c(1, 2).

... Additional slots and values passed to struct_class.

Value

A pca_correlation_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = pca_correlation_plot()
```

pca_dstat_plot *d-statistic plot*

Description

A bar chart of the d-statistics for samples in the input PCA model. Samples above the indicated threshold are considered to be outlying.

Usage

```
pca_dstat_plot(number_components = 2, alpha = 0.05, ...)
```

Arguments

number_components	(numeric) The number of principal components to use. The default is 2.
alpha	(numeric) A confidence threshold for rejecting samples based on the d-statistic. The default is 0.05.
...	Additional slots and values passed to struct_class.

Value

A pca_dstat_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = pca_dstat_plot()
```

pca_loadings_plot *PCA loadings plot*

Description

A barchart (one component) or scatter plot (two components) of the selected principal component loadings.

Usage

```
pca_loadings_plot(
  components = c(1, 2),
  style = "points",
  label_features = NULL,
  ...
)
```

Arguments

components	(numeric) The principal components used to generate the plot. The default is c(1, 2).
style	(character) Plot style. Allowed values are limited to the following: • "points": Loadings and scores are plotted as a scatter plot. • "arrows": The loadings are plotted as arrow vectors. The default is "points".
label_features	(character, NULL) Feature labels. Allowed values are limited to the following: • "character()": A vector of labels for the features. • "NULL": No labels. • "row.names": Labels will be extracted from the column names of the data matrix. The default is NULL.
...	Additional slots and values passed to struct_class.

Value

A pca_loadings_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = pca_loadings_plot()
```

---

pca_scores_plot *PCA scores plot*

Description

Plots a 2d scatter plot of the selected components

Usage

```
pca_scores_plot(
  xcol = "PC1",
  ycol = "PC2",
  points_to_label = "none",
  factor_name,
  ellipse = "all",
  ellipse_type = "norm",
  ellipse_confidence = 0.95,
  label_filter = character(0),
  label_factor = "rownames",
  label_size = 3.88,
  components = NULL,
  ...
)
```

Arguments

xcol	(numeric, integer, character) The column name, or index, of data to plot on the x-axis. The default is "PC1".
ycol	(numeric, integer, character) The column name, or index, of data to plot on the y-axis. The default is "PC2".
points_to_label	(character) Points to label. Allowed values are limited to the following: • "none": No samples labels are displayed. • "all": The labels for all samples are displayed. • "outliers": Labels for for potential outlier samples are displayed. The default is "none".
factor_name	(character) The name of a sample-meta column to use.
ellipse	(character) Plot ellipses. Allowed values are limited to the following: • "all": Ellipses are plotted for all groups and all samples. • "group": Ellipses are plotted for all groups. • "none": Ellipses are not included on the plot. • "sample": An ellipse is plotted for all samples (ignoring group). The default is "all".
ellipse_type	(character) Type of ellipse. Allowed values are limited to the following: • "norm": Multivariate normal (p = 0.95). • "t": Multivariate t (p = 0.95). The default is "norm".
ellipse_confidence	(numeric) The confidence level for plotting ellipses. The default is 0.95.
label_filter	(character) Labels are only plotted for the named groups. If zero-length then all groups are included. The default is character(0).
label_factor	(character) The column name of sample_meta to use for labelling samples on the plot. "rownames" will use the row names from sample_meta. The default is "rownames".
label_size	(numeric) The text size of labels. Note this is not in Font Units. The default is 3.88.
components	(numeric, integer, NULL) The principal components used to generate the plot. If provided this parameter overrides xcol and ycol params. The default is NULL.
...	Additional slots and values passed to struct_class.

Value

A pca_scores_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre() + PCA()
M = model_apply(M,D)
C = pca_scores_plot(factor_name = 'Species')
chart_plot(C,M[2])
```

`pca_scree_plot` *Scree plot*

Description

A plot of the percent variance and cumulative percent variance for the components of a PCA model.

Usage

```
pca_scree_plot(max_pc = 15, ...)
```

Arguments

`max_pc`	(numeric, integer) The maximum number of components to include in the plot. The default is 15.
`...`	Additional slots and values passed to `struct_class`.

Value

A `pca_scree_plot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

struct object

Examples

```
C = pca_scree_plot()
```

`permutation_test` *Permutation test*

Description

A permutation test generates a "null" model by randomising the response (for regression models) or group labels (for classification models). This is repeated many times to generate a distribution of performance metrics for the null model. This distribution can then be compared to the performance of the true model. If there is overlap between the true and null model performances then the model is overfitted.

Usage

```
permutation_test(number_of_permutations = 50, factor_name, ...)
```

Arguments

number_of_permutations	(numeric, integer) The number of permutations. The default is 50.
factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A permutation_test object with the following output slots:

results.permuted	(data.frame)
results.unpermuted	(data.frame)
metric	(data.frame)

Examples

```
I=permutation_test(factor_name='Species')
```

---

permutation_test_plot *permutation_test_plot class*

Description

Plots the results of a permutation test.

Usage

```
permutation_test_plot(style = "boxplot", binwidth = 0.05, ...)
```

Arguments

style	The plot style. One of 'boxplot', 'violin', 'histogram', 'density' or 'scatter'.
binwidth	Binwidth for the "histogram" style. Ignored for all other styles.
...	additional slots and values passed to struct_class

Value

struct object

Examples

```
C = permutation_test_plot(style='boxplot')
```

permute_sample_order *Permute Sample Order*

Description

The order of samples in the data matrix is randomly permuted. The relationship between the samples and the sample meta data is maintained.

Usage

```
permute_sample_order(number_of_permutations = 10, ...)
```

Arguments

number_of_permutations	(numeric, integer) The number of times the sample order is permuted. The default is 10.
...	Additional slots and values passed to struct_class.

Value

A permute_sample_order object with the following output slots:

results	(data.frame)
metric	(data.frame)
metric.train	(numeric)

Examples

```
C = permute_sample_order()
```

PLSDA *Partial least squares discriminant analysis*

Description

PLS is a multivariate regression technique that extracts latent variables maximising covariance between the input data and the response. The Discriminant Analysis variant uses group labels in the response variable. For >2 groups a 1-vs-all approach is used. Group membership can be predicted for test samples based on a probability estimate of group membership, or the estimated y-value.

Usage

```
PLSDA(number_components = 2, factor_name, pred_method = "max_prob", ...)
```

Arguments

number_components
: (numeric, integer) The number of PLS components. The default is 2.

factor_name
: (character) The name of a sample-meta column to use.

pred_method
: (character) Prediction method. Allowed values are limited to the following:

  - "max_yhat": The predicted group is selected based on the largest value of y_hat.
  - "max_prob": The predicted group is selected based on the largest probability of group membership.

  The default is "max_prob".

...
: Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pls

Value

A PLSDA object with the following output slots:

scores	(DatasetExperiment)
loadings	(data.frame)
yhat	(data.frame)
design_matrix	(data.frame)
y	(data.frame)
reg_coeff	(data.frame)
probability	(data.frame)
vip	(data.frame)
pls_model	(list)
pred	(data.frame)
threshold	(numeric)
sr	(data.frame) Selectivity ratio for a variable represents a measure of a variable's importance in the PLS mod
sr_pvalue	(data.frame) A p-value computed from the Selectivity Ratio based on an F-distribution.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Perez NF, Ferre J, Boque R (2009). "Calculation of the reliability of classification in discriminant partial least-squares binary classification." *Chemometrics and Intelligent Laboratory Systems*, *95*(2), 122-128.

Barker M, Rayens W (2003). "Partial least squares for discrimination." *Journal of Chemometrics*, *17*(3), 166-173.

Examples

```
M = PLSDA('number_components'=2,factor_name='Species')
```

`plsda_feature_importance_plot` *PLSDA feature importance summary plot*

Description

A plot of the selected feature significance metric for a PLSDA model for the top selected features.

Usage

```
plsda_feature_importance_plot(n_features = 30, metric = "vip", ...)
```

Arguments

`n_features`	(numeric, integer) The number of features to include in the summary. The default is `30`.
`metric`	(character) Metric to plot. Allowed values are limited to the following: • `"sr"`: Plot Selectivity Ratio. • `"sr_pvalue"`: Plot SR p-values. • `"vip"`: Plot Variable Importance in Projection scores. The default is `"vip"`.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- `pls`
- `ggplot2`
- `reshape2`
- `cowplot`

Value

A `plsda_feature_importance_plot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Wickham H (2007). "Reshaping Data with the reshape Package." *Journal of Statistical Software*, *21*(12), 1-20. http://www.jstatsoft.org/v21/i12/.

Wilke C (2020). *cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'*. R package version 1.1.1, https://CRAN.R-project.org/package=cowplot.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = plsda_feature_importance_plot(n_features=30,metric='vip')
chart_plot(C,M[2])
```

`plsda_predicted_plot` *PLSDA predicted plot*

Description

A plot of the regression coefficients from a PLSDA model.

Usage

```
plsda_predicted_plot(factor_name, style = "boxplot", ycol = 1, ...)
```

Arguments

`factor_name`	(character) The name of a sample-meta column to use.
`style`	(character) Plot style. Allowed values are limited to the following: • `"boxplot"`: A boxplot. • `"violin"`: A violin plot. • `"density"`: A density plot. The default is `"boxplot"`.
`ycol`	(character, numeric, integer) The column of the Y block to be plotted. The default is `1`.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- pls
- ggplot2

Value

A plsda_predicted_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = plsda_predicted_plot(factor_name='Species')
chart_plot(C,M[2])
```

---

plsda_roc_plot *PLSDA ROC plot*

Description

A Receiver Operator Characteristic (ROC) plot for PLSDA models computed by adjusting the threshold for assigning group labels from PLS predictions.

Usage

```
plsda_roc_plot(factor_name, ycol = 1, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
ycol	(character, numeric, integer) The column of the Y block to be plotted. The default is 1.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pls
- ggplot2

Value

A plsda_roc_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = plsda_roc_plot(factor_name='Species')
chart_plot(C,M[2])
```

PLSR *Partial least squares regression*

Description

PLS is a multivariate regression technique that extracts latent variables maximising covariance between the input data and the response. For regression the response is a continuous variable.

Usage

```
PLSR(number_components = 2, factor_name, ...)
```

Arguments

number_components	(numeric, integer) The number of PLS components. The default is 2.
factor_name	(character) The name of sample meta column(s) to use.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pls

Value

A PLSR object with the following output slots:

scores	(DatasetExperiment)
loadings	(data.frame)
yhat	(data.frame)
y	(data.frame)
reg_coeff	(data.frame)
vip	(data.frame)
pls_model	(list)
pred	(data.frame)
sr	(data.frame) Selectivity ratio for a variable represents a measure of a variable's importance in the PLS model. Tl
sr_pvalue	(data.frame) A p-value computed from the Selectivity Ratio based on an F-distribution.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Examples

```
M = PLSR(factor_name='run_order')
```

---

plsr_cook_dist *Cook's distance barchart*

Description

A barchart of Cook's distance for each sample used to train a PLSR model. Cook's distance is used to estimate the influence of a sample on the model and can be used to identify potential outliers.

Usage

```
plsr_cook_dist(ycol = 1, ...)
```

Arguments

ycol	(numeric, integer, character) The y-block column to plot. The default is 1.
...	Additional slots and values passed to struct_class.

Value

A plsr_cook_dist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = plsr_cook_dist()
```

---

plsr_prediction_plot *PLSR prediction plot*

Description

A scatter plot of the true response values against the predicted values for a PLSR model.

Usage

```
plsr_prediction_plot(ycol = 1, ...)
```

Arguments

ycol	(numeric, integer, character) The y-block column to plot. The default is 1.
...	Additional slots and values passed to struct_class.

Value

A plsr_prediction_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = plsr_prediction_plot()
```

---

plsr_qq_plot *PLSR QQ plot*

Description

A plot of the quantiles of the residuals from a PLSR model against the quantiles of a normal distribution.

Usage

```
plsr_qq_plot(ycol = 1, ...)
```

Arguments

ycol	(numeric, integer, character) The y-block column to plot. The default is 1.
...	Additional slots and values passed to struct_class.

Value

A plsr_qq_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = plsr_qq_plot()
```

---

plsr_residual_hist *PLSR residuals histogram*

Description

A histogram of the residuals for a PLSR model.

Usage

```
plsr_residual_hist(ycol = 1, ...)
```

Arguments

ycol	(numeric, integer, character) The y-block column to plot. The default is 1.
...	Additional slots and values passed to struct_class.

Value

A plsr_residual_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = plsr_residual_hist()
```

`pls_regcoeff_plot` *pls_regcoeff_plot class*

Description

Plots the regression coefficients of a PLSDA model.

Plots the regression coefficient scores of a PLSDA model

Usage

```
pls_regcoeff_plot(ycol = 1, ...)
```

Arguments

`ycol`	(character, numeric, integer) The Y column to plot. The default is 1.
`...`	additional slots and values passed to struct_class

Details

This object makes use of functionality from the following packages:

- `pls`
- `ggplot2`

Value

A `pls_regcoeff_plot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

struct object

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = pls_regcoeff_plot(ycol='setosa')
chart_plot(C,M[2])
```

`pls_scores_plot` *PLSDA scores plot*

Description

A scatter plot of the selected PLSDA scores.

Usage

```
pls_scores_plot(
  xcol = "LV1",
  ycol = "LV2",
  points_to_label = "none",
  factor_name,
  ellipse = "all",
  ellipse_type = "norm",
  ellipse_confidence = 0.95,
  label_filter = character(0),
  label_factor = "rownames",
  label_size = 3.88,
  components = NULL,
  ...
)

plsda_scores_plot(
  xcol = "LV1",
  ycol = "LV2",
  points_to_label = "none",
  factor_name,
  ellipse = "all",
  ellipse_type = "norm",
  ellipse_confidence = 0.95,
  label_filter = character(0),
  label_factor = "rownames",
  label_size = 3.88,
  components = NULL,
  ...
)
```

Arguments

`xcol`	(numeric, integer, character) The column name, or index, of data to plot on the x-axis. The default is `"LV1"`.
`ycol`	(numeric, integer, character) The column name, or index, of data to plot on the y-axis. The default is `"LV2"`.
`points_to_label`	(character) Points to label. Allowed values are limited to the following:

- "none": No samples labels are displayed.
- "all": The labels for all samples are displayed.
- "outliers": Labels for for potential outlier samples are displayed.

The default is "none".

factor_name (character) The name of a sample-meta column to use.

ellipse (character) Plot ellipses. Allowed values are limited to the following:

- "all": Ellipses are plotted for all groups and all samples.
- "group": Ellipses are plotted for all groups.
- "none": Ellipses are not included on the plot.
- "sample": An ellipse is plotted for all samples (ignoring group).

The default is "all".

ellipse_type (character) Type of ellipse. Allowed values are limited to the following:

- "norm": Multivariate normal (p = 0.95).
- "t": Multivariate t (p = 0.95).

The default is "norm".

ellipse_confidence (numeric) The confidence level for plotting ellipses. The default is 0.95.

label_filter (character) Labels are only plotted for the named groups. If zero-length then all groups are included. The default is character(0).

label_factor (character) The column name of sample_meta to use for labelling samples on the plot. "rownames" will use the row names from sample_meta. The default is "rownames".

label_size (numeric) The text size of labels. Note this is not in Font Units. The default is 3.88.

components (numeric, integer, NULL) The principal components used to generate the plot. If provided this parameter overrides xcol and ycol params. The default is NULL.

... Additional slots and values passed to struct_class.

Value

A pls_scores_plot object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = pls_scores_plot(factor_name='Species')
chart_plot(C,M[2])
```

`pls_vip_plot` *PLSDA VIP plot*

Description

A plot of the Variable Importance for Projection (VIP) scores for a PLSDA model.

Usage

```
pls_vip_plot(threshold = 1, ycol = 1, ...)
```

Arguments

`threshold`	(numeric, integer) The threshold for indicating significant features. The default is 1.
`ycol`	(character, numeric, integer) The column of the Y block to be plotted. The default is 1.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- `pls`
- `ggplot2`

Value

A `pls_vip_plot` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

References

Liland K, Mevik B, Wehrens R (2023). *pls: Partial Least Squares and Principal Component Regression*. R package version 2.8-2, https://CRAN.R-project.org/package=pls.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

Examples

```
D = iris_DatasetExperiment()
M = mean_centre()+PLSDA(factor_name='Species')
M = model_apply(M,D)

C = pls_vip_plot(ycol='setosa')
chart_plot(C,M[2])
```

pqn_norm *Probabilistic Quotient Normalisation (PQN)*

Description

PQN is used to normalise for differences in concentration between samples. It makes use of Quality Control (QC) samples as a reference. PQN scales by the median change relative to the reference in order to be more robust against changes caused by response to perturbation.

Usage

```
pqn_norm(
  qc_label = "QC",
  factor_name,
  qc_frac = 0,
  sample_frac = 0,
  ref_method = "mean",
  ref_mean = NULL,
  ...
)
```

Arguments

qc_label	(character) The label used to identify QC samples. The default is "QC".
factor_name	(character) The name of a sample-meta column to use.
qc_frac	(numeric) A value between 0 and 1 to indicate the minimum proportion of QC samples a feature must be present in for it to be included when computing the reference. Default qc_frac = 0. . The default is 0.
sample_frac	(numeric) A value between 0 and 1 to indicate the minimum proportion of samples a feature must be present in for it to be considered when computing the normalisation coefficients. . The default is 0.
ref_method	(character) Reference computation method. Allowed values are limited to the following: • "mean": The reference is computed as the mean of the samples matching the qc_label input. • "median": The reference is computed as the median of the samples matching the qc_label_input. The default is "mean".
ref_mean	(numeric, NULL) A single sample to use as the reference for normalisation. If set to NULL then the reference will be computed based on the other input parameters (ref_mean, qc_label etc). . The default is NULL.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A pqn_norm object with the following output slots:

normalised	(DatasetExperiment) A DatasetExperiment object containing the normalised data.
coeff	(data.frame) The normalisation coefficients calculated by PQN.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
D = iris_DatasetExperiment()
M = pqn_norm(factor_name='Species',qc_label='all')
M = model_apply(M,D)
```

---

pqn_norm_hist *PQN coefficient histogram*

Description

A histogram of the PQN coefficients for all features

Usage

```
pqn_norm_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A pqn_norm_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = pqn_norm_hist()
```

prop_na *Fisher's exact test for missing values*

Description

A Fisher's exact test is used to compare the number of missing values in each group. Multiple test corrected p-values are computed to indicate whether there is a significant difference in the number of missing values across groups for each feature.

Usage

```
prop_na(alpha = 0.05, mtc = "fdr", factor_name, ...)
```

Arguments

alpha	(numeric) The p-value cutoff for determining significance. The default is 0.05.
mtc	(character) Multiple test correction method. Allowed values are limited to the following: • "bonferroni": Bonferroni correction in which the p-values are multiplied by the number of comparisons. • "fdr": Benjamini and Hochberg False Discovery Rate correction. • "none": No correction. The default is "fdr".
factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Value

A prop_na object with the following output slots:

p_value	(data.frame) The probability of observing the calculated statistic.
significant	(data.frame) TRUE if the calculated p-value is less than the supplied threshold (alpha).
na_count	(data.frame) The number of NA values per group of the chosen factor.

struct object

Examples

```
M = prop_na(factor_name='Species')
```

`resample` *Data resampling*

Description

New training sets are generated from the original data by selecting samples at random. This can be based on levels in a factor or on the whole dataset.

Usage

```
resample(
  number_of_iterations = 10,
  method = "split_data",
  factor_name,
  p_train = 0.8,
  collect = NULL,
  ...
)
```

Arguments

`number_of_iterations`
: (numeric, integer) The number of training sets to generate. The default is `10`.

`method`
: (character) Resampling method. Allowed values are limited to the following:

  - `"split_data"`: Samples for the training set are selected at random from the full dataset.
  - `"stratified_split"`: Samples for the training set are randomly selected from each level of the chosen factor.
  - `"equal_split"`: Samples for the training set are selected at random from each level of the main factor such that all group sizes are equal.

  The default is `"split_data"`.

`factor_name`
: (character) The name of a sample-meta column to use.

`p_train`
: (numeric) The proportion of samples selected for the training set. The default is `0.8`.

`collect`
: (NULL, character) The name of a model output to collect over all bootstrap repetitions, in addition to the input metric. The default is `NULL`.

`...`
: Additional slots and values passed to `struct_class`.

Value

A `resample` object with the following `output` slots:

`results.training`	(data.frame)
`results.testing`	(data.frame)

metric	(data.frame)
collected	(list)
metric.train	(numeric)
metric.test	(numeric)

Examples

```
I = resample(
    number_of_iterations = 10,
    factor_name = 'Species',
    method = 'split_data',
    p_train = 0.8)
```

---

resample_chart *resample_chart class*

Description

Plots the results of a resampling.

Usage

```
resample_chart(style = "boxplot", binwidth = 0.05, ...)
```

Arguments

style	The plot style. One of 'boxplot', 'violin', 'histogram', 'density' or 'scatter'.
binwidth	Binwidth for the "histogram" style. Ignored for all other styles.
...	additional slots and values passed to struct_class

Value

struct object

Examples

```
C = resample_chart(style='boxplot')
```

`rsd_filter` *RSD filter*

Description

An RSD filter calculates the relative standard deviation (the ratio of the standard deviation to the mean) for all features. Any feature with an RSD greater than a predefined threshold is excluded.

Usage

```
rsd_filter(rsd_threshold = 20, qc_label = "QC", factor_name, ...)
```

Arguments

`rsd_threshold`	(numeric) The RSD threshold above which features are removed. The default is `20`.
`qc_label`	(character) The label used to identify QC samples. The default is `"QC"`.
`factor_name`	(character) The name of a sample-meta column to use.
`...`	Additional slots and values passed to `struct_class`.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A `rsd_filter` object with the following `output` slots:

`filtered`	(DatasetExperiment) A DatasetExperiment object containing the filtered data.
`flags`	(data.frame) RSD and a flag indicating whether the feature was rejected by the filter or not.
`rsd_qc`	(data.frame) The calculated RSD of the QC class.

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Examples

```
M = rsd_filter(factor_name='Class')
```

rsd_filter_hist *RSD histogram*

Description

A histogram of the calculated RSD values.

Usage

```
rsd_filter_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A rsd_filter_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
C = rsd_filter_hist()
```

run,bootstrap,DatasetExperiment,metric-method *Runs an iterator, applying the chosen model multiple times.*

Description

Running an iterator will apply the iterator a number of times to a DatasetExperiment. For example, in cross-validation the same model is applied multiple times to the same data, splitting it into training and test sets. The input metric object can be calculated and collected for each iteration as an output.

Usage

```
## S4 method for signature 'bootstrap,DatasetExperiment,metric'
run(I, D, MET = NULL)

## S4 method for signature 'forward_selection_by_rank,DatasetExperiment,metric'
run(I, D, MET)

## S4 method for signature 'grid_search_1d,DatasetExperiment,metric'
run(I, D, MET)
```

```
## S4 method for signature 'kfold_xval,DatasetExperiment,metric'
run(I, D, MET = NULL)

## S4 method for signature 'permutation_test,DatasetExperiment,metric'
run(I, D, MET = NULL)

## S4 method for signature 'permute_sample_order,DatasetExperiment,metric'
run(I, D, MET)

## S4 method for signature 'resample,DatasetExperiment,metric'
run(I, D, MET)
```

Arguments

I	an iterator object
D	a DatasetExperiment object
MET	a metric object

Value

Modified iterator object

Examples

```
D = iris_DatasetExperiment() # get some data
MET = metric()  # use a metric
I = example_iterator() # initialise iterator
models(I) = example_model() # set the model
I = run(I,D,MET) # run
```

---

r_squared *Coefficient of determination (R-squared)*

Description

R-squared is a metric used to assess the goodness of fit for regression models. It measures how much variance of one variable can be explained by another variable.

Usage

```
r_squared(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A `r_squared` object. This object has no `output` slots.

Examples

```
MET = r_squared()
```

`sb_corr` *Signal/batch correction for mass spectrometry data*

Description

Applies Quality Control Robust Spline (QC-RSC) method to correct for signal drift and batch differences in mass spectrometry data.

Usage

```
sb_corr(
  order_col,
  batch_col,
  qc_col,
  smooth = 0,
  use_log = TRUE,
  min_qc = 4,
  qc_label = "QC",
  spar_lim = c(-1.5, 1.5),
  ...
)
```

Arguments

`order_col`	(character) The column name of sample_meta indicating the run order of the samples.
`batch_col`	(character) The column name of sample_meta indicating the batch each sample was measured in.
`qc_col`	(character) The column name of sample_meta indicating the group each sample is a member of.
`smooth`	(numeric) The amount of smoothing applied (0 to 1). If set to 0 the smoothing parameter will be estimated using leave-one-out cross-validation. The default is `0`.
`use_log`	(logical) Log tranformation. Allowed values are limited to the following: • `"TRUE"`: The data is log transformed prior to performing signal correction. • `"FALSE"`: Signal correction is applied to the input data. The default is `TRUE`.

min_qc	(numeric) The minimum number of QC samples required for signal correction. The default is 4.
qc_label	(character) The label used to identify QC samples. The default is "QC".
spar_lim	(numeric) A two element vector specifying the upper and lower limits when spar = 0. Allows the value of spar to be constrained within these limits to prevent overfitting. The default is c(-1.5, 1.5).
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- pmp

Value

A sb_corr object with the following output slots:

corrected	(DatasetExperiment) The DatasetExperiment after signal/batch correction has been applied.
fitted	(data.frame) The fitted splines for each feature.

struct object

References

Jankevics A, Lloyd GR, Weber RJM (2023). *pmp: Peak Matrix Processing and signal batch correction for metabolomics datasets*. doi:10.18129/B9.bioc.pmp https://doi.org/10.18129/B9.bioc.pmp, R package version 1.12.0, https://bioconductor.org/packages/pmp.

Kirwan JA, Broadhurst DI, Davidson RL, Viant MR (2013). "Characterising and correcting batch variation in an automated direct infusion mass spectrometry (DIMS) metabolomics workflow." *Analytical and Bioanalytical Chemistry*, *405*(15), 5147-5157.

Examples

```
M = sb_corr(order_col='run_order',batch_col='batch_no',qc_col='class')
```

---

scatter_chart *Group scatter chart*

Description

Plots a 2d scatter plot of the input data.

Usage

```
scatter_chart(
  xcol = 1,
  ycol = 2,
  points_to_label = "none",
  factor_name = "none",
  ellipse = "all",
  ellipse_type = "norm",
  ellipse_confidence = 0.95,
  label_filter = character(0),
  label_factor = "rownames",
  label_size = 3.88,
  ...
)
```

Arguments

`xcol`	(numeric, integer, character) The column name, or index, of data to plot on the x-axis. The default is `1`.
`ycol`	(numeric, integer, character) The column name, or index, of data to plot on the y-axis. The default is `2`.
`points_to_label`	(character) Points to label. Allowed values are limited to the following: • `"none"`: No samples labels are displayed. • `"all"`: The labels for all samples are displayed. • `"outliers"`: Labels for for potential outlier samples are displayed. The default is `"none"`.
`factor_name`	(character) The name of a sample-meta column to use. The default is `"none"`.
`ellipse`	(character) Plot ellipses. Allowed values are limited to the following: • `"all"`: Ellipses are plotted for all groups and all samples. • `"group"`: Ellipses are plotted for all groups. • `"none"`: Ellipses are not included on the plot. • `"sample"`: An ellipse is plotted for all samples (ignoring group). The default is `"all"`.
`ellipse_type`	(character) Type of ellipse. Allowed values are limited to the following: • `"norm"`: Multivariate normal (p = 0.95). • `"t"`: Multivariate t (p = 0.95). The default is `"norm"`.
`ellipse_confidence`	(numeric) The confidence level for plotting ellipses. The default is `0.95`.
`label_filter`	(character) Labels are only plotted for the named groups. If zero-length then all groups are included. The default is `character(0)`.

label_factor	(character) The column name of sample_meta to use for labelling samples on the plot. "rownames" will use the row names from sample_meta. The default is "rownames".
label_size	(numeric) The text size of labels. Note this is not in Font Units. The default is 3.88.
...	Additional slots and values passed to struct_class.

Value

A scatter_chart object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
C = scatter_chart(
        xcol = 'Petal.Width',
        ycol = 'Sepal.Width',
        factor_name = 'Species'
    )
chart_plot(C,D)
```

---

split_data *Split data*

Description

The data matrix is divided into two subsets.A predefined proportion of the samples are randomly selected for a training set, and the remaining samples are used for the test set.

Usage

```
split_data(p_train, ...)
```

Arguments

p_train	(numeric) The proportion of samples selected for the training set.
...	Additional slots and values passed to struct_class.

Value

A split_data object with the following output slots:

training	(DatasetExperiment) A DatasetExperiment object containing samples selected for the training set.
testing	(DatasetExperiment) A DatasetExperiment object containing samples selected for the testing set.

Examples

```
M = split_data(p_train=0.75)
```

`stratified_split` *Stratified sampling*

Description

The dataset is divided into two subsets. A predefined proportion of samples from each level of a factor is selected for the training set, and the remaining samples are used for the test set. The stratification by factor level means that the relative number of samples per level is approximately equal to the original dataset.

Usage

```
stratified_split(p_train, factor_name, ...)
```

Arguments

`p_train`	(numeric) The proportion of samples selected for the training set.
`factor_name`	(character) The name of a sample-meta column to use.
`...`	Additional slots and values passed to `struct_class`.

Value

A `stratified_split` object with the following `output` slots:

`training`	(DatasetExperiment) A DatasetExperiment object containing samples selected for the training set.
`testing`	(DatasetExperiment) A DatasetExperiment object containing samples selected for the testing set.

Examples

```
D = iris_DatasetExperiment()
M = stratified_split(p_train=0.75,factor_name='Species')
M = model_apply(M,D)
```

`structToolbox` *structToolbox: Examples of tools built using the Statistics in R Using Class Templates (struct) package*

Description

This package extends the classes defined in the struct package

SVM *Support Vector Machine Classifier*

Description

Support Vector Machines (SVM) are a machine learning algorithm for classification. They can make use of kernel functions to generate highly non-linear boundaries between groups.

Usage

```
SVM(
  factor_name,
  kernel = "linear",
  degree = 3,
  gamma = 1,
  coef0 = 0,
  cost = 1,
  class_weights = NULL,
  ...
)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
kernel	(character) Kernel type. Allowed values are limited to the following: • "linear": . • "polynomial": . • "radial": . • "sigmoid": . The default is "linear".
degree	(numeric) The polynomial degree. The default is 3.
gamma	(numeric) The gamma parameter. The default is 1.
coef0	(numeric) The offset coefficient. The default is 0.
cost	(numeric) The cost of violating the constraints. The default is 1.
class_weights	(numeric, character, NULL) A named vector of weights for the different classes. Specifying "inverse" will choose the weights inversely proportional to the class distribution. The default is NULL.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- e1071

Value

A SVM object with the following output slots:

SV	(matrix)
index	(numeric)
coefs	(matrix)
pred	(data.frame)
decision_values	(data.frame)

struct object

References

Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2023). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-13, https://CRAN.R-project.org/package=e1071.

Brereton RG, Lloyd GR (2010). "Support Vector Machines for classification and regression." *The Analyst*, *135*(2), 230-267.

Examples

```
M = SVM(factor_name='Species',gamma=1)
```

---

svm_plot_2d *SVM scatter plot*

Description

A scatter plot of the input data by group and the calculated boundary of a SVM model.

Usage

```
svm_plot_2d(factor_name, npoints = 100, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
npoints	(numeric) The number of grid points used to plot the boundary. The default is 100.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- e1071

Value

A svm_plot_2d object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

References

Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2023). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-13, https://CRAN.R-project.org/package=e1071.

Examples

```
D = iris_DatasetExperiment()
M = filter_smeta(mode='exclude',levels='setosa',factor_name='Species') +
    mean_centre()+PCA(number_components=2)+
    SVM(factor_name='Species',kernel='linear')
M = model_apply(M,D)

C = svm_plot_2d(factor_name='Species')
chart_plot(C,M[4],predicted(M[3]))
```

---

tic_chart *Total Ion Count chart.*

Description

A scatter plot of Total Ion Count (sum of each sample) versus run order.

Usage

```
tic_chart(run_order, factor_name, connected = FALSE, ...)
```

Arguments

run_order	(character) The column name of sample_meta indicating the run order of the samples.
factor_name	(character) The name of a sample-meta column to use.
connected	(logical) Plot samples connected by a grey line. The default is FALSE.
...	Additional slots and values passed to struct_class.

Value

A tic_chart object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
D = iris_DatasetExperiment()
D$sample_meta$run_order=1:nrow(D)
C = tic_chart(factor_name='Species',run_order='run_order')
chart_plot(C,D)
```

---

tSNE *tSNE*

Description

t-Distributed Stochastic Neighbor Embedding.

Usage

```
tSNE(
  dims = 2,
  perplexity = 30,
  max_iter = 100,
  theta = 0.5,
  check_duplicates = FALSE,
  init = NULL,
  eta = 200,
  ...
)
```

Arguments

dims	(numeric) The number of tSNE dimensions computed. The default is 2.
perplexity	(numeric) Perplexity parameter. The default is 30.
max_iter	(numeric) The maximum number of tSNE iterations. The default is 100.
theta	(numeric) Speed/accuracy trade-off. A value of 0 gives an exact tSNE. The default is 0.5.
check_duplicates	(logical) Check for duplicates. Allowed values are limited to the following: • "TRUE": Checks for the presence of exact duplicate samples. • "FALSE": Does not check for exact duplicate samples. The default is FALSE.
init	(NULL, data.frame, DatasetExperiment) A set of coordinates for initialising the tSNE algorithm. NULL uses random initialisation. The default is NULL.
eta	(numeric) The learning rate parameter. The default is 200.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- Rtsne

Value

A tSNE object with the following output slots:

Y (DatasetExperiment)

References

van der Maaten L, Hinton G (2008). "Visualizing High-Dimensional Data Using t-SNE." *Journal of Machine Learning Research*, *9*, 2579-2605.

van der Maaten L (2014). "Accelerating t-SNE using Tree-Based Algorithms." *Journal of Machine Learning Research*, *15*, 3221-3245.

Krijthe JH (2015). *Rtsne: T-Distributed Stochastic Neighbor Embedding using Barnes-Hut Implementation*. R package version 0.16, https://github.com/jkrijthe/Rtsne.

Examples

```
M = tSNE()
```

---

tSNE_scatter *Feature boxplot*

Description

plots the new representation of data after applying tSNE.

Usage

```
tSNE_scatter(factor_name, ...)
```

Arguments

factor_name	(character) The name of a sample-meta column to use.
...	Additional slots and values passed to struct_class.

Details

This object makes use of functionality from the following packages:

- Rtsne

Value

A `tSNE_scatter` object. This object has no `output` slots. See `chart_plot` in the `struct` package to plot this chart object.

References

van der Maaten L, Hinton G (2008). "Visualizing High-Dimensional Data Using t-SNE." *Journal of Machine Learning Research*, *9*, 2579-2605.

van der Maaten L (2014). "Accelerating t-SNE using Tree-Based Algorithms." *Journal of Machine Learning Research*, *15*, 3221-3245.

Krijthe JH (2015). *Rtsne: T-Distributed Stochastic Neighbor Embedding using Barnes-Hut Implementation*. R package version 0.16, https://github.com/jkrijthe/Rtsne.

Examples

```
M = tSNE_scatter(factor_name='Species')
```

`ttest` *t-test*

Description

A t-test compares the means of two factor levels. Multiple-test corrected p-values are used to indicate the significance of the computed difference for all features.

Usage

```
ttest(
  alpha = 0.05,
  mtc = "fdr",
  factor_names,
  paired = FALSE,
  paired_factor = character(0),
  equal_variance = FALSE,
  conf_level = 0.95,
  ...
)
```

Arguments

`alpha`	(numeric) The p-value cutoff for determining significance. The default is `0.05`.
`mtc`	(character) Multiple test correction method. Allowed values are limited to the following: • `"bonferroni"`: Bonferroni correction in which the p-values are multiplied by the number of comparisons.

- `"fdr"`: Benjamini and Hochberg False Discovery Rate correction.
- `"none"`: No correction.

The default is `"fdr"`.

`factor_names`	(character) The name of sample meta column(s) to use.
`paired`	(logical) Apply a paired t-test. The default is `FALSE`.
`paired_factor`	(character) The factor name that encodes the sample id for pairing. The default is `character(0)`.
`equal_variance`	(logical) Equal variance. Allowed values are limited to the following: • `"TRUE"`: The variance of each group is treated as being equal using the pooled variance to estimate the variance. • `"FALSE"`: The variance of each group is not assumed to be equal and the Welch (or Satterthwaite) approximation is used. The default is `FALSE`.
`conf_level`	(numeric) The confidence level of the interval. The default is `0.95`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `ttest` object with the following `output` slots:

`t_statistic`	(data.frame) The value of the calculate statistics which is converted to a p-value when compared to a t-distribu
`p_value`	(data.frame) The probability of observing the calculated t-statistic.
`dof`	(numeric) The number of degrees of freedom used to calculate the test statistic.
`significant`	(data.frame) TRUE if the calculated p-value is less than the supplied threhold (alpha).
`conf_int`	(data.frame) Confidence interval for t statistic.
`estimates`	(data.frame) The group means estimated when computing the t-statistic.

Examples

```
M = ttest(factor_name='Class')
```

`vec_norm` *Vector normalisation*

Description

The samples in the data matrix are normalised to account for differences in concentration by scaling each sample such that the sum of squares is equal to 1.

Usage

```
vec_norm(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A vec_norm object with the following output slots:

normalised	(DatasetExperiment) A DatasetExperiment object containing the normalised data.
coeff	(data.frame) The normalisation coefficients calculated by PQN.

struct object

Examples

```
M = vec_norm()
```

---

wilcox_p_hist *Histogram of p values*

Description

A histogram of p values for the wilcoxon signed rank test

Usage

```
wilcox_p_hist(...)
```

Arguments

...	Additional slots and values passed to struct_class.

Value

A wilcox_p_hist object. This object has no output slots. See chart_plot in the struct package to plot this chart object.

Examples

```
M = wilcox_p_hist()
```

`wilcox_test` *wilcoxon signed rank test*

Description

A Mann-Whitney-Wilcoxon signed rank test compares ,the ranks of values in two groups. It is the non-parametric equivalent of a t-test. Multiple test corrected p-values are computed as indicators of significance for each variable/feature.

Usage

```
wilcox_test(
  alpha = 0.05,
  mtc = "fdr",
  factor_names,
  paired = FALSE,
  paired_factor = character(0),
  conf_level = 0.95,
  ...
)
```

Arguments

`alpha`	(numeric) The p-value cutoff for determining significance. The default is `0.05`.
`mtc`	(character) Multiple test correction method. Allowed values are limited to the following: • `"bonferroni"`: Bonferroni correction in which the p-values are multiplied by the number of comparisons. • `"fdr"`: Benjamini and Hochberg False Discovery Rate correction. • `"none"`: No correction. The default is `"fdr"`.
`factor_names`	(character) The name of a sample-meta column to use.
`paired`	(logical) Apply a paired test. The default is `FALSE`.
`paired_factor`	(character) The factor name containing sample ids for paired data. The default is `character(0)`.
`conf_level`	(numeric) The confidence level of the interval. The default is `0.95`.
`...`	Additional slots and values passed to `struct_class`.

Value

A `wilcox_test` object with the following `output` slots:

`statistic`	(data.frame) The value of the calculated statistic which is converted to a p-value.
`p_value`	(data.frame) The probability of observing the calculated t-statistic.

`dof`	(numeric) The number of degrees of freedom used to calculate the test statistic.
`significant`	(data.frame) TRUE if the calculated p-value is less than the supplied threhold (alpha).
`conf_int`	(data.frame) Confidence interval for t statistic.
`estimates`	(data.frame) The group estimates used when computing the statistic.

struct object

Examples

```
M = wilcox_test(factor_name='Class')
```

Index