

# Package ‘iCOBRA’

May 15, 2025

**Type** Package

**Title** Comparison and Visualization of Ranking and Assignment Methods

**Version** 1.37.0

**Date** 2024-12-31

**Description** This package provides functions for calculation and visualization of performance metrics for evaluation of ranking and binary classification (assignment) methods. Various types of performance plots can be generated programmatically. The package also contains a shiny application for interactive exploration of results.

**License** GPL (>=2)

**LazyData** TRUE

**Collate** 'AllGenerics.R' 'COBRADData.R' 'COBRAPerformance.R'  
'COBRAPlot.R' 'calculate\_performance.R' 'cobradata\_example.R'  
'cobradata\_example\_sval.R' 'helpers\_general.R' 'plot\_methods.R'  
'printHead.R' 'shiny.R'

**Depends** R (>= 4.4.0)

**Imports** shiny (>= 0.9.1.9008), shinydashboard, shinyBS, reshape2,  
ggplot2 (>= 3.4.0), scales, ROCR, dplyr, DT, limma, methods,  
UpSetR, markdown, utils, rlang

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

**biocViews** Classification, Visualization

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**URL** <https://github.com/csoneson/iCOBRA>

**BugReports** <https://github.com/csoneson/iCOBRA/issues>

**git\_url** <https://git.bioconductor.org/packages/iCOBRA>

**git\_branch** devel

**git\_last\_commit** 99223a3

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.22

**Date/Publication** 2025-05-15

**Author** Charlotte Soneson [aut, cre] (ORCID:  
<https://orcid.org/0000-0003-3833-2169>)

**Maintainer** Charlotte Soneson <charlottesoneson@gmail.com>

## Contents

basemethods . . . . .	3
calculate_adjp . . . . .	4
calculate_performance . . . . .	4
COBRAapp . . . . .	6
COBRADData-class . . . . .	7
cobradata_example . . . . .	9
cobradata_example_sval . . . . .	9
COBRAPerformance-class . . . . .	10
COBRAPlot-class . . . . .	12
coerce . . . . .	14
corr . . . . .	14
deviation . . . . .	15
Extract . . . . .	16
facetted . . . . .	17
fdrnbr . . . . .	18
fdrnbrcurve . . . . .	19
fdrtp . . . . .	20
fdrtpcurve . . . . .	21
fpc . . . . .	22
fpr . . . . .	23
fsrnbr . . . . .	24
fsrnbrcurve . . . . .	25
maxsplit . . . . .	26
onlyshared . . . . .	27
overlap . . . . .	28
padj . . . . .	29
plotcolors . . . . .	29
plot_corr . . . . .	30
plot_deviation . . . . .	31
plot_fdrnbrcurve . . . . .	32
plot_fdrtpcurve . . . . .	33
plot_fpc . . . . .	34
plot_fpr . . . . .	35
plot_fsrnbrcurve . . . . .	36
plot_overlap . . . . .	37
plot_roc . . . . .	38
plot_scatter . . . . .	39
plot_tpr . . . . .	40
plot_upset . . . . .	41
prepare_data_for_plot . . . . .	42
pval . . . . .	44
reorder_levels . . . . .	44
roc . . . . .	45
scatter . . . . .	46
score . . . . .	47

<i>basemethods</i>	3
splv . . . . .	48
stratiflevels . . . . .	49
sval . . . . .	49
tpr . . . . .	50
truth . . . . .	51
update_cobradata . . . . .	52
update_cobraperformance . . . . .	53
<b>Index</b>	<b>54</b>

---

basemethods	<i>Accessor function for basemethods</i>
-------------	--

---

### Description

Accessor function to extract the methods that are represented in an COBRAPerformance or COBRAPlot object.

### Usage

```
basemethods(x, ...)

## S4 method for signature 'COBRAPerformance'
basemethods(x)
```

### Arguments

x                   A COBRAPerformance or COBRAPlot object.  
 ...                 Additional arguments.

### Value

A character vector of all methods represented in the object.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)
cobraperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = "fdrtprcurve")
basemethods(cobraperf)
```

---

calculate\_adjp      *Calculate adjusted p-values*

---

### Description

Calculate adjusted p-values for methods where only nominal p-values are available in a COBRADData object.

### Usage

```
calculate_adjp(cobradata, method = "BH")
```

### Arguments

cobradata	A COBRADData object.
method	A character string giving the method (selected from <code>p.adjust.methods()</code> ) that will be used to perform the adjustment.

### Value

A COBRADData object, extended with the calculated adjusted p-values.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)
cobradata_example <- calculate_adjp(cobradata_example, method = "BH")
```

---

calculate\_performance      *Calculate performance measures*

---

### Description

Calculate performance measures from a given collection of p-values, adjusted p-values and scores provided in a COBRADData object.

### Usage

```
calculate_performance(
  cobradata,
  binary_truth = NULL,
  cont_truth = NULL,
  aspects = c("fdrtpr", "fdrtprcurve", "fdrnbr", "fdrnbrcurve", "tpr", "fpr", "roc",
             "fpc", "overlap", "corr", "scatter", "deviation", "fsrnbr", "fsrnbrcurve"),
  thrs = c(0.01, 0.05, 0.1),
  svalthrs = c(0.01, 0.05, 0.1),
  splv = "none",
```

```

    maxsplit = 3,
    onlyshared = FALSE,
    thr_venn = 0.05,
    type_venn = "adjp",
    topn_venn = 100,
    rank_by_abs = TRUE,
    prefer_pval = TRUE
)

```

### Arguments

cobradata	A COBRADData object.
binary_truth	A character string giving the name of the column of truth(cobradata) that contains the binary truth (true assignment of variables into two classes, represented by 0/1).
cont_truth	A character string giving the name of the column of truth(cobradata) that contains the continuous truth (a continuous value that the observations can be compared to).
aspects	A character vector giving the types of performance measures to calculate. Must be a subset of c("fdrtp", "fdrtpcurve", "fdrnbr", "fdrnbrcurve", "tpr", "fpr", "roc", "fpc", "overlap", "corr", "scatter", "deviation", "fsnbr", "fsnbrcurve").
thrs	A numeric vector of adjusted p-value thresholds for which to calculate the performance measures. Affects "fdrtp", "fdrnbr", "tpr" and "fpr".
svalthrs	A numeric vector of s-value thresholds for which to calculate the FSR. Affects "fsnbr".
splv	A character string giving the name of the column of truth(cobradata) that will be used to stratify the results. The default value is "none", indicating no stratification.
maxsplit	A numeric value giving the maximal number of categories to keep in the stratification. The largest categories containing both positive and negative features will be retained. By setting this argument to 'Inf' or 'NA_integer_', all categories (as well as the order of categories) will be retained.
onlyshared	A logical, indicating whether to only consider features for which both the true assignment and a result (p-value, adjusted p-value or score) is given. If FALSE, all features contained in the truth table are used.
thr_venn	A numeric value giving the adjusted p-value threshold to use to create Venn diagrams (if type_venn is "adjp").
type_venn	Either "adjp" or "rank", indicating whether Venn diagrams should be constructed based on features with adjusted p-values below a certain threshold, or based on the same number of top-ranked features by different methods.
topn_venn	A numeric value giving the number of top-ranked features to compare between methods (if type_venn is "rank").
rank_by_abs	Whether to take the absolute value of the score before using it to rank the variables for ROC, FPC, FDR/NBR and FDR/TPR curves.
prefer_pval	Whether to preferentially rank variables by p-values or adjusted p-values rather than score for ROC and FPC calculations. From version 1.5.5, this is the default behaviour. To obtain the behaviour of previous versions, set to FALSE.

**Details**

Depending on the collection of observations that are available for a given method, the appropriate one will be chosen for each performance measure. For `fpr`, `tpr`, `fdr_tpr`, `fdrnbr` and `overlap` aspects, results will only be calculated for methods where adjusted p-values are included in the `COBRADData` object, since these calculations make use of specific adjusted p-value cutoffs. For `fdr_tprcurve` and `fdrnbrcurve` aspects, the score observations will be preferentially used, given that they are monotonically associated with the adjusted p-values (if provided). If the score is not provided, the nominal p-values will be used, given that they are monotonically associated with the adjusted p-values (if provided). In other cases, the adjusted p-values will be used also for these aspects. For `roc` and `fpc`, the score observations will be used if they are provided, otherwise p-values and, as a last instance, adjusted p-values. Finally, for the `fdrnbr`, `corr`, `scatter` and `deviation` aspects, the score observations will be used if they are provided, otherwise no results will be calculated.

**Value**

A `COBRAPerformance` object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
  binary_truth = "status",
  aspects = c("fdr_tpr", "fdr_tprcurve",
             "tpr", "roc"),
  thrs = c(0.01, 0.05, 0.1), splv = "none")
```

---

COBRAapp

*Interactive shiny app to visualize results*

---

**Description**

Interactive shiny app for visualization of results. The app can be initialized with a `COBRADData` object. If no object is provided, truth and results are loaded into the app from text files (see the Instructions tab of the app for formatting instructions). Properly formatted text files can also be obtained using the function [COBRADData\\_to\\_text](#).

**Usage**

```
COBRAapp(cobradata = NULL, autorun = FALSE, addStopButton = TRUE)
```

**Arguments**

<code>cobradata</code>	An (optional) <code>COBRADData</code> object. If not given, the user can load results from text files.
<code>autorun</code>	A logical indicating whether the app calculations should start automatically on launch, or wait for the user to press the 'Start calculation!' button.
<code>addStopButton</code>	Logical scalar. If <code>TRUE</code> (default), will add a button to stop the app (by calling <code>shiny::stopApp</code> ).

**Value**

Returns (and runs) an object representing the shiny app.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
app <- COBRAApp(cobradata_example)
if (interactive()) {
  shiny::runApp(app)
}
```

---

COBRADData-class

COBRADData *object and constructor*


---

**Description**

The COBRADData class contains slots to hold calculated p-values, adjusted p-values and general 'scores' for a set of features, as well as s-values (see Stephens (2017)). The slots can contain values from multiple methods, and each method can contribute to one or more slots. The class also contains a slot giving the 'truth' (a binary assignment and/or a continuous score) for each feature, as well as additional annotations that can be used to stratify the performance calculations.

**Usage**

```
COBRADData(
  pval = data.frame(),
  padj = data.frame(),
  score = data.frame(),
  sval = data.frame(),
  truth = data.frame(),
  object_to_extend = NULL
)
```

```
COBRADData_from_text(truth_file, result_files, feature_id)
```

```
COBRADData_to_text(cobradata, truth_file, result_files, feature_id)
```

**Arguments**

pval	A data frame with features as rows and methods as columns, containing nominal p-values. Missing values (NAs) are allowed. The row names should be feature names.
padj	A data frame with features as rows and methods as columns, containing adjusted p-values. Missing values (NAs) are allowed. The row names should be feature names.

score	A data frame with features as rows and methods as columns, containing generic scores. In case of comparison to a binary truth, larger values of the scores should correspond to 'more significant' features. Missing values (NAs) are allowed. The row names should be feature names.
sval	A data frame with features as rows and methods as columns, containing s-values (analogous to q-values, but for sign errors, see Stephens (2017)). Missing values (NAs) are allowed. The row names should be feature names.
truth	A data frame with features as rows columns containing feature annotations such as, e.g., binary and continuous truths and additional annotations that can be used to stratify the performance calculations. The row names should be feature names.
object_to_extend	A COBRADData object to extend with the provided information.
truth_file	A character string giving the path to a file with true labels and other feature annotations.
result_files	A character vector giving path(s) to file(s) with results (p-values, adjusted p-values, s-values, scores) for one or more methods. The column names of these files must be of the form "method:measure", where measure is one of P, adjP, S or score, depending on what is given in the column.
feature_id	A character string giving the name of the column in the truth and result files that encodes the feature identifier.
cobradata	A COBRADData object

### Details

If adjusted p-values are missing for some methods, for which nominal p-values are available, the adjusted p-values can be calculated using the [calculate\\_adjp](#) function.

The text files generated by COBRADData\_to\_text can be used as input to iCOBRAapp, when it is called without an input argument.

### Value

COBRADData and COBRADData\_from\_text return a COBRADData object.

### Author(s)

Charlotte Soneson

### Examples

```
## Empty COBRADData object:
COBRADData()

## COBRADData object from individual data frames
set.seed(123)
pval <- data.frame(m1 = runif(100), m2 = runif(100),
                  row.names = paste0("F", 1:100))
truth <- data.frame(status = round(runif(100)),
                  row.names = paste0("F", 1:100))
cobradata <- COBRADData(pval = pval, truth = truth)
```



---

cobradata_example	<i>Example data set with three differential gene expression methods</i>
-------------------	---

---

**Description**

A data set consisting of p-values, adjusted p-values and estimated absolute log fold changes (in the 'scores' slot) obtained by three methods for differential expression analysis of RNA-seq data, applied to a small synthetic data set of 3,858 human genes. The values are stored in a COBRADData object together with a 'truth' data frame containing the true differential expression status for each gene as well as various additional annotations such as the true log fold change, the number of isoforms of the gene and the average expression level.

**Usage**

```
cobradata_example
```

**Format**

A COBRADData object with five slots:

**pval** data frame with p-values for 2,399 genes, from three different methods.

**padj** data frame with adjusted p-values for 2,399 genes, from two different methods.

**sval** empty data frame

**score** data frame with estimated absolute log fold changes for 2,399 genes, from three different methods.

**truth** data frame with true differential expression status (status column), the number of isoforms (n\_isoforms column), the true absolute log fold change (logFC and logFC\_cat columns) and average expression level (expr and expr\_cat columns) for 3,858 genes.

**Value**

A COBRADData object.

---

cobradata_example_sval	<i>Example data set with three differential gene expression methods</i>
------------------------	---

---

**Description**

A data set consisting of p-values, adjusted p-values, s-values and estimated log fold changes (in the 'scores' slot) obtained by three methods for differential expression analysis of RNA-seq data, applied to a small synthetic data set of 3,858 human genes. The values are stored in a COBRADData object together with a 'truth' data frame containing the true differential expression status for each gene as well as various additional annotations such as the true log fold change, the number of isoforms of the gene and the average expression level.

**Usage**

```
cobradata_example_sval
```

**Format**

A COBRADData object with five slots:

**pval** data frame with p-values for 2,399 genes, from three different methods.

**padj** data frame with adjusted p-values for 2,399 genes, from two different methods.

**sval** data frame representing s-values for 2,399 genes, from two different methods. Since the purpose of this data set is only to demonstrate functionality, the s-values were obtained by copying the adjusted p-value slot, and thus these values do not represent true s-values.

**score** data frame with estimated log fold changes for 2,399 genes, from three different methods.

**truth** data frame with true differential expression status (status column), the number of isoforms (n\_isoforms column), the true log fold change (logFC and logFC\_cat columns) and average expression level (expr and expr\_cat columns) for 3,858 genes.

**Value**

A COBRADData object.

---

COBRAPerformance-class

COBRAPerformance *object and constructor*

---

**Description**

The COBRAPerformance class holds various types of calculated performance measures. Objects from this class are typically generated from COBRADData objects by means of the function [calculate\\_performance](#).

**Usage**

```
COBRAPerformance(
  fdrtpr = data.frame(),
  fdrtprcurve = data.frame(),
  fdnabr = data.frame(),
  fdnabrcurve = data.frame(),
  fsnabr = data.frame(),
  fsnabrcurve = data.frame(),
  tpr = data.frame(),
  fpr = data.frame(),
  splv = "",
  roc = data.frame(),
  fpc = data.frame(),
  deviation = data.frame(),
  onlyshared = NA,
  overlap = data.frame(),
  maxsplit = NA_integer_,
  corr = data.frame(),
  scatter = data.frame()
)
```

**Arguments**

<code>fdr tpr</code>	A data frame containing observed FDR and TPR values at various adjusted p-value thresholds.
<code>fdr tpr curve</code>	A data frame containing observed FDR and TPR values for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score).
<code>fdr nbr</code>	A data frame containing observed FDR and the number of features considered to be significant at various adjusted p-value thresholds.
<code>fdr nbr curve</code>	A data frame containing observed FDR and number of features considered to be significant for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score).
<code>fsr nbr</code>	A data frame containing observed false sign rate (FSR) and the number of features considered to be significant at various s-value thresholds
<code>fsr nbr curve</code>	A data frame containing observed false sign rate (FSR) and number of features considered to be significant for a (potentially large) number of cutoffs applied to the s-value
<code>tpr</code>	A data frame containing observed TPR values at various adjusted p-value thresholds.
<code>fpr</code>	A data frame containing observed FPR values at various adjusted p-value thresholds.
<code>splv</code>	A character string giving the name of the stratification factor, "none" if the results are not stratified.
<code>roc</code>	A data frame containing observed FPR and TPR values for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score), which can be used to generate a ROC curve.
<code>fpc</code>	A data frame containing observed numbers of false positive findings among the N top-ranked features (ranked by p-values, adjusted p-values or more general scores), for a (potentially large) number of Ns, which can be used to generate a false positive curve.
<code>deviation</code>	A data frame containing deviations between observed scores and true scores.
<code>only shared</code>	A logical value indicating whether only features shared between the results and the truth should be retained, or if all features present in the truth should be used.
<code>overlap</code>	A data frame or list of data frames with binary values indicating, for each of a number of methods and number of features, whether the method consider the feature 'positive' (significant, 1) or 'negative' (non-significant, 0). If it is a list of data frames, each list element corresponds to one level of a stratifying factor.
<code>max split</code>	A numeric value indicating the largest number of levels to retain if the results have been stratified by an annotation.
<code>corr</code>	A data frame containing observed (Pearson and Spearman) correlation values between observed and true scores.
<code>scatter</code>	A data frame containing observed 'scores' (p-values, adjusted p-values or more general scores) and true scores, which can be used to generate scatter plots.

**Value**

A COBRAPerformance object.

**Author(s)**

Charlotte Soneson

**Examples**

```
## Empty COBRAPerformance object
COBRAPerformance()
```

---

COBRAPlot-class

COBRAPlot *object and constructor*


---

**Description**

The COBRAPlot class is similar to the COBRAPerformance class in that it holds various types of calculated performance measures. However, it also contains other attributes that are necessary for plotting, such as color assignments. Several COBRAPlot objects can be generated from the same COBRAPerformance object, without having to go through the potentially time consuming task of recalculating all performance measures. Objects from this class are typically generated from a COBRAPerformance objects by means of the function [prepare\\_data\\_for\\_plot](#).

**Usage**

```
COBRAPlot(
  fdrtpr = data.frame(),
  fdrtprcurve = data.frame(),
  fsrnbr = data.frame(),
  fsrnbrcurve = data.frame(),
  fdnabr = data.frame(),
  corr = data.frame(),
  fdnabrcurve = data.frame(),
  tpr = data.frame(),
  fpr = data.frame(),
  roc = data.frame(),
  scatter = data.frame(),
  onlyshared = NA,
  fpc = data.frame(),
  overlap = data.frame(),
  plotcolors = "",
  splv = "",
  deviation = data.frame(),
  maxsplit = NA_integer_,
  faceted = NA
)
```

**Arguments**

fdrtpr	A data frame containing observed FDR and TPR values at various adjusted p-value thresholds.
fdrtprcurve	A data frame containing observed FDR and TPR values for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score).

fsrnbr	A data frame containing observed false sign rate (FSR) and the number of features considered to be significant at various s-value thresholds
fsrnbrcurve	A data frame containing observed false sign rate (FSR) and number of features considered to be significant for a (potentially large) number of cutoffs applied to the s-value
fdrnbr	A data frame containing observed FDR and the number of features considered to be significant at various adjusted p-value thresholds.
corr	A data frame containing observed (Pearson and Spearman) correlation values between observed and true scores.
fdrnbrcurve	A data frame containing observed FDR and number of features considered to be significant for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score).
tpr	A data frame containing observed TPR values at various adjusted p-value thresholds.
fpr	A data frame containing observed FPR values at various adjusted p-value thresholds.
roc	A data frame containing observed FPR and TPR values for a (potentially large) number of cutoffs applied to a 'score' (that can be p-value, adjusted p-value or a more general score), which can be used to generate a ROC curve.
scatter	A data frame containing observed 'scores' (p-values, adjusted p-values or more general scores) and true scores, which can be used to generate scatter plots.
onlyshared	A logical value indicating whether only features shared between the results and the truth should be retained, or if all features present in the truth should be used.
fpc	A data frame containing observed numbers of false positive findings among the N top-ranked features (ranked by p-values, adjusted p-values or more general scores), for a (potentially large) number of Ns, which can be used to generate a false positive curve.
overlap	A data frame or list of data frames with binary values indicating, for each of a number of methods and number of features, whether the method consider the feature 'positive' (significant, 1) or 'negative' (non-significant, 0). If it is a list of data frames, each list element corresponds to one level of a stratifying factor.
plotcolors	A character vector giving the color for each method (or method-stratification level combination).
splv	A character string giving the name of the stratification factor, "none" if the results are not stratified.
deviation	A data frame containing deviations between observed scores and true scores.
maxsplit	A numeric value indicating the largest number of levels to retain if the results have been stratified by an annotation.
facetted	A logical indicating whether the data is prepared for a facetted plot (separating different stratification levels into different panels) or for displaying all values in one plot panel.

**Value**

A COBRAPlot object.

**Author(s)**

Charlotte Sonesson

**Examples**

```
## Empty COBRAPlot object
cobraplot <- COBRAPlot()
```

---

coerce *Convert an object to another class*

---

**Description**

Convert object between COBRAPerformance and COBRAPlot classes.

**Arguments**

from            The object that is to be coerced into another class.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrtpr")
cobraplot <- prepare_data_for_plot(cobrapperf)

## Coerce COBRAPerformance object into COBRAPlot object
as(cobrapperf, "COBRAPlot")

## Coerce COBRAPlot object into COBRAPerformance object
as(cobraplot, "COBRAPerformance")
```

---

corr *Accessor and replacement functions for corr slot*

---

**Description**

Accessor and replacement functions for the corr slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
corr(x, ...)
```

```
corr(x, ...) <- value
```

```
## S4 method for signature 'COBRAPerformance'
corr(x)
```

```
## S4 replacement method for signature 'COBRAPerformance,data.frame'
```

```
corr(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
corr(x) <- value
```

### Arguments

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving correlation values for each method and each stratification level.

### Value

The accessor function returns a data frame giving correlation values for each method and each stratification level.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                   aspects = "corr")
head(corr(cobrapperf))
```

---

deviation

*Accessor and replacement functions for deviation slot*

---

### Description

Accessor and replacement functions for the deviation slot in a COBRAPerformance or COBRAPlot object.

### Usage

```
deviation(x, ...)

deviation(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
deviation(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
deviation(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
deviation(x) <- value
```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information necessary to plots of deviations between observed and true scores for each method and each stratification level.

**Value**

The accessor function returns a data frame giving information necessary to plots of deviations between observed and true scores for each method and each stratification level.

**Author(s)**

Charlotte Sonesson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                  aspects = "deviation")
head(deviation(cobrapperf))
```

---

Extract

*Subsetting COBRADData, COBRAPerformance or COBRAPlot objects*

---

**Description**

Functions to subset COBRADData, COBRAPerformance or COBRAPlot objects. COBRADData objects are subset by features (rows), while COBRAPerformance and COBRAPlot objects are subset by methods (columns). Numeric indices are not allowed, since not all slots may be arranged in the same order.

**Usage**

```
## S4 method for signature 'COBRADData'
x[i, j = "missing", drop = "missing"]

## S4 method for signature 'COBRAPerformance'
x[i = "missing", j, drop = "missing"]

## S4 method for signature 'COBRAPlot'
x[i = "missing", j, drop = "missing"]
```

**Arguments**

x	A COBRADData, COBRAPerformance or COBRAPlot object.
i	For COBRADData objects, a character vector of feature names to retain.
j	For COBRAPerformance and COBRAPlot objects, a character vector with method names to retain.
drop	not used.



**Value**

A subset of the original object, of the same class

**Examples**

```
data(cobradata_example)
cobradata_example[c("ENSG00000000457", "ENSG00000000971",
                    "ENSG00000000460"), ]
cobraperf <- calculate_performance(cobradata_example,
                                 binary_truth = "status",
                                 aspects = "fdrtpr")
cobraperf[, c("voom")]
cobraplot <- prepare_data_for_plot(cobraperf)
cobraplot[, c("voom")]
```

---

 faceted

*Accessor and replacement functions for faceted slot*


---

**Description**

Accessor and replacement functions for the faceted slot in an COBRAPlot object.

**Usage**

```
faceted(x, ...)

faceted(x, ...) <- value

## S4 method for signature 'COBRAPlot'
faceted(x)

## S4 replacement method for signature 'COBRAPlot,logical'
faceted(x) <- value
```

**Arguments**

x	A COBRAPlot object.
...	Additional arguments.
value	A logical value, indicating whether the object is formatted for faceted plots (visualizing each stratification level in a separate panel) or not.

**Value**

The accessor function returns a logical value, indicating whether the object is formatted for faceted plots (visualizing each stratification level in a separate panel) or not.

**Author(s)**

Charlotte Sonesson

**Examples**

```

data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrtrp")
cobraplot <- prepare_data_for_plot(cobrapperf)
faceted(cobraplot)

```

---

fdrnbr

*Accessor and replacement functions for fdrnbr slot*


---

**Description**

Accessor and replacement functions for the fdrnbr slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```

fdrnbr(x, ...)

fdrnbr(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
fdrnbr(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
fdrnbr(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
fdrnbr(x) <- value

```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information about the observed FPR and the number of features called positive for each method and each stratification level, at various adjusted p-value thresholds.

**Value**

The accessor function returns a data frame giving information about the observed FPR and the number of features called positive for each method and each stratification level, at various adjusted p-value thresholds.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrnbr")
head(fdrnbr(cobrapperf))
```

fdrnbrcurve

*Accessor and replacement functions for fdrnbrcurve slot***Description**

Accessor and replacement functions for the fdrnbrcurve slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
fdrnbrcurve(x, ...)

fdrnbrcurve(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
fdrnbrcurve(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
fdrnbrcurve(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
fdrnbrcurve(x) <- value
```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information necessary to generate curves of observed FDR vs number of features called positive for each method and each stratification level.

**Value**

The accessor function returns a data frame giving information necessary to generate curves of observed FDR vs number of features called positive for each method and each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrnbrcurve")
head(fdrnbrcurve(cobrapperf))
```

fdrtptr

*Accessor and replacement functions for fdrtptr slot***Description**

Accessor and replacement functions for the fdrtptr slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
fdrtptr(x, ...)
```

```
fdrtptr(x, ...) <- value
```

```
## S4 method for signature 'COBRAPerformance'
fdrtptr(x)
```

```
## S4 replacement method for signature 'COBRAPerformance,data.frame'
fdrtptr(x) <- value
```

```
## S4 replacement method for signature 'COBRAPlot,data.frame'
fdrtptr(x) <- value
```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information about the observed FPR and TPR for each method and each stratification level, at various adjusted p-value thresholds.

**Value**

The accessor function returns a data frame giving information about the observed FPR and TPR for each method and each stratification level, at various adjusted p-value thresholds.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrtptr")
head(fdrtptr(cobrapperf))
```

---

`fdrtpcurve`*Accessor and replacement functions for fdrtpcurve slot*

---

**Description**

Accessor and replacement functions for the `fdrtpcurve` slot in a `COBRAPerformance` or `COBRAPlot` object.

**Usage**

```
fdrtpcurve(x, ...)  
  
fdrtpcurve(x, ...) <- value  
  
## S4 method for signature 'COBRAPerformance'  
fdrtpcurve(x)  
  
## S4 replacement method for signature 'COBRAPerformance,data.frame'  
fdrtpcurve(x) <- value  
  
## S4 replacement method for signature 'COBRAPlot,data.frame'  
fdrtpcurve(x) <- value
```

**Arguments**

<code>x</code>	A <code>COBRAPerformance</code> or <code>COBRAPlot</code> object.
<code>...</code>	Additional arguments.
<code>value</code>	A data frame giving information necessary to generate curves of observed FDR vs TPR for each method and each stratification level.

**Value**

The accessor function returns a data frame giving information necessary to generate curves of observed FDR vs TPR for each method and each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                binary_truth = "status",  
                                aspects = "fdrtpcurve")  
  
head(fdrtpcurve(cobraperf))
```

---

fpc

*Accessor and replacement functions for fpc slot*

---

### Description

Accessor and replacement functions for the fpc slot in a COBRAPerformance or COBRAPlot object.

### Usage

```
fpc(x, ...)  
  
fpc(x, ...) <- value  
  
## S4 method for signature 'COBRAPerformance'  
fpc(x)  
  
## S4 replacement method for signature 'COBRAPerformance,data.frame'  
fpc(x) <- value  
  
## S4 replacement method for signature 'COBRAPlot,data.frame'  
fpc(x) <- value
```

### Arguments

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information necessary to generate false positive curves for each method and each stratification level.

### Value

The accessor function returns a data frame giving information necessary to generate false positive curves for each method and each stratification level.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                binary_truth = "status", aspects = "fpc")  
head(fpc(cobraperf))
```

---

fpr

*Accessor and replacement functions for fpr slot*

---

### Description

Accessor and replacement functions for the fpr slot in a COBRAPerformance or COBRAPlot object.

### Usage

```
fpr(x, ...)  
  
fpr(x, ...) <- value  
  
## S4 method for signature 'COBRAPerformance'  
fpr(x)  
  
## S4 replacement method for signature 'COBRAPerformance,data.frame'  
fpr(x) <- value  
  
## S4 replacement method for signature 'COBRAPlot,data.frame'  
fpr(x) <- value
```

### Arguments

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information about the observed FPR for each method and each stratification level, at various adjusted p-value thresholds.

### Value

The accessor function returns a data frame giving information about the observed FPR for each method and each stratification level, at various adjusted p-value thresholds.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                binary_truth = "status", aspects = "fpr")  
head(fpr(cobraperf))
```

---

`fsrnbr`*Accessor and replacement functions for fsrnbr slot*

---

**Description**

Accessor and replacement functions for the `fsrnbr` slot in a `COBRAPerformance` or `COBRAPlot` object.

**Usage**

```
fsrnbr(x, ...)
```

```
fsrnbr(x, ...) <- value
```

```
## S4 method for signature 'COBRAPerformance'  
fsrnbr(x)
```

```
## S4 replacement method for signature 'COBRAPerformance,data.frame'  
fsrnbr(x) <- value
```

```
## S4 replacement method for signature 'COBRAPlot,data.frame'  
fsrnbr(x) <- value
```

**Arguments**

<code>x</code>	A <code>COBRAPerformance</code> or <code>COBRAPlot</code> object.
<code>...</code>	Additional arguments.
<code>value</code>	A data frame giving information about the observed FSR and the number of features called positive for each method and each stratification level, at various s-value thresholds. If the object does not have an <code>fsrnbr</code> slot (older versions of the class did not have this slot), an empty data frame is returned for simplicity.

**Value**

The accessor function returns a data frame giving information about the observed FSR and the number of features called positive for each method and each stratification level, at various s-value thresholds.

**Author(s)**

Charlotte Soneson

**Examples**

```
cobradata <- cobradata_example_sval  
  
cobraperf <- calculate_performance(cobradata,  
                                 cont_truth = "logFC",  
                                 aspects = "fsrnbr")  
  
head(fsrnbr(cobraperf))
```



---

`fsrnbrcurve`*Accessor and replacement functions for fsrnbrcurve slot*

---

**Description**

Accessor and replacement functions for the `fsrnbrcurve` slot in a `COBRAPerformance` or `COBRAPlot` object.

**Usage**

```
fsrnbrcurve(x, ...)
```

```
fsrnbrcurve(x, ...) <- value
```

```
## S4 method for signature 'COBRAPerformance'  
fsrnbrcurve(x)
```

```
## S4 replacement method for signature 'COBRAPerformance,data.frame'  
fsrnbrcurve(x) <- value
```

```
## S4 replacement method for signature 'COBRAPlot,data.frame'  
fsrnbrcurve(x) <- value
```

**Arguments**

<code>x</code>	A <code>COBRAPerformance</code> or <code>COBRAPlot</code> object.
<code>...</code>	Additional arguments.
<code>value</code>	A data frame giving information necessary to generate curves of observed FSR vs number of features called positive for each method and each stratification level. If the object does not have an <code>fsrnbrcurve</code> slot (older versions of the class did not have this slot), an empty data frame is returned for simplicity.

**Value**

The accessor function returns a data frame giving information necessary to generate curves of observed FSR vs number of features called positive for each method and each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
cobradata <- cobradata_example_sval  
  
cobrapperf <- calculate_performance(cobradata,  
                                  cont_truth = "logFC",  
                                  aspects = "fsrnbrcurve")  
  
head(fsrnbrcurve(cobrapperf))
```



---

`onlyshared`*Accessor and replacement functions for onlyshared slot*

---

**Description**

Accessor and replacement functions for the onlyshared slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
onlyshared(x, ...)
```

```
onlyshared(x, ...) <- value
```

```
## S4 method for signature 'COBRAPerformance'  
onlyshared(x)
```

```
## S4 replacement method for signature 'COBRAPerformance,logical'  
onlyshared(x) <- value
```

```
## S4 replacement method for signature 'COBRAPlot,logical'  
onlyshared(x) <- value
```

**Arguments**

<code>x</code>	A COBRAPerformance or COBRAPlot object.
<code>...</code>	Additional arguments.
<code>value</code>	A logical indicating whether only features that are shared between result and truth are retained, or if all features in the truth are used.

**Value**

The accessor function returns a logical indicating whether only features that are shared between result and truth are retained, or if all features in the truth are used.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                binary_truth = "status",  
                                aspects = "fdrtpr")  
  
head(onlyshared(cobraperf))
```

---

`overlap`*Accessor and replacement functions for overlap slot*

---

**Description**

Accessor and replacement functions for the `overlap` slot in a `COBRAPerformance` or `COBRAPlot` object.

**Usage**

```
overlap(x, ...)  
  
overlap(x, ...) <- value  
  
## S4 method for signature 'COBRAPerformance'  
overlap(x)  
  
## S4 replacement method for signature 'COBRAPerformance,list_df'  
overlap(x) <- value  
  
## S4 replacement method for signature 'COBRAPlot,list_df'  
overlap(x) <- value
```

**Arguments**

<code>x</code>	A <code>COBRAPerformance</code> or <code>COBRAPlot</code> object.
<code>...</code>	Additional arguments.
<code>value</code>	A data frame or a list, giving information about which feature that are classified as 'positive' by each method and for each stratification level.

**Value**

The accessor function returns a data frame or a list, giving information about which feature that are classified as 'positive' by each method and for each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                 binary_truth = "status",  
                                 aspects = "overlap")  
  
head(overlap(cobraperf))
```

---

padj                      *Accessor and replacement functions for padj slot*

---

**Description**

Accessor and replacement functions for the padj slot in a COBRADData object.

**Usage**

```
padj(x, ...)  
  
padj(x, ...) <- value  
  
## S4 method for signature 'COBRADData'  
padj(x)  
  
## S4 replacement method for signature 'COBRADData,data.frame'  
padj(x) <- value
```

**Arguments**

x	A COBRADData object.
...	Additional arguments.
value	A data frame containing adjusted p-values for each feature and each method.

**Value**

The accessor function returns a data frame containing adjusted p-values for each feature and each method.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)  
head(padj(cobradata_example))
```

---

plotcolors                      *Accessor and replacement functions for plotcolors slot*

---

**Description**

Accessor and replacement functions for the plotcolors slot in an COBRAPlot object.

**Usage**

```

plotcolors(x, ...)

plotcolors(x, ...) <- value

## S4 method for signature 'COBRAPlot'
plotcolors(x)

## S4 replacement method for signature 'COBRAPlot,character'
plotcolors(x) <- value

```

**Arguments**

x	A COBRAPlot object.
...	Additional arguments.
value	A character vector giving the colors assigned to each of the methods (or method/stratification level combinations) represented in the COBRAPlot object.

**Value**

The accessor function returns a character vector giving the colors assigned to each of the methods (or method/stratification level combinations) represented in the COBRAPlot object.

**Author(s)**

Charlotte Soneson

**Examples**

```

data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   aspects = "fdrtpr")
cobraplot <- prepare_data_for_plot(cobrapperf)
plotcolors(cobraplot)

```

---

plot\_corr

*Plot correlations*

---

**Description**

Plot correlations between observations and a continuous truth value.

**Usage**

```

plot_corr(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,

```

```

    xaxisrange = c(-1, 1),
    corrtype = "pearson"
  )

```

### Arguments

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
corrtype	A character string giving the type of correlation to show. Either "pearson" or "spearman".

### Value

A ggplot object

### Author(s)

Charlotte Soneson

### Examples

```

data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                  aspects = "corr")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_corr(cobraplot, corrtype = "spearman")

```

---

plot_deviation	<i>Plot deviations</i>
----------------	------------------------

---

### Description

Plot the deviations between observed scores and the continuous truth variable.

### Usage

```

plot_deviation(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  xaxisrange = NULL,
  plottype = "boxplot",
  dojitter = TRUE,
  transf = "raw"
)

```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
plottype	Either "boxplot" or "violinplot", indicating what type of plot to make.
dojitter	A logical indicating whether to include jittered data points or not.
transf	A character indicating the transformation to apply to the deviations before plotting. Must be one of "raw", "absolute" or "squared"

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                  aspects = "deviation")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                  incltruth = TRUE)
plot_deviation(cobraplot)
```

---

plot\_fdrnbrcurve      *Plot number of significant features vs FDR*

---

**Description**

Plot the number of features considered significant vs observed false discovery rate (FDR), for given adjusted p-value thresholds and/or as curves traced out by considering all threshold values.

**Usage**

```
plot_fdrnbrcurve(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,
  xaxisrange = c(0, 1),
  plottype = c("curve", "points"),
  linewidth = 1
)
```



**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
plottype	A character vector giving the type of plot to construct. Can be any combination of the two elements "curve" and "points".
linewidth	The line width used for plotting

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = c("fdrnbr", "fdrnbrcurve"))
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_fdrnbrcurve(cobraplot, plottype = c("curve", "points"))
```

---

plot\_fdrtpcurve      *Plot TPR vs FDR*

---

**Description**

Plot observed true positive rate (TPR) vs observed false discovery rate (FDR), for given adjusted p-value thresholds and/or as curves traced out by considering all threshold values.

**Usage**

```
plot_fdrtpcurve(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,
  xaxisrange = c(0, 1),
  yaxisrange = c(0, 1),
  plottype = c("curve", "points"),
  linewidth = 1
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
yaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the y-axis, respectively.
plottype	A character vector giving the type of plot to construct. Can be any combination of the two elements "curve" and "points".
linewidth	The line width used for plotting

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = c("fdrtpr", "fdrtprcurve"))
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_fdrtprcurve(cobraplot, plottype = c("curve", "points"))
```

---

plot\_fpc

*Plot FP curves*

---

**Description**

Plot false positive curves, indicating the number of false positives among the top-ranked N variables, for varying values of N.

**Usage**

```
plot_fpc(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  maxnfdc = 500,
  linewidth = 1
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
maxnfdc	A numeric value giving the largest N to consider.
linewidth	The line width used for plotting

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status", aspects = "fpc")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_fpc(cobraplot, maxnfdc = 750)
```

---

plot\_fpr

*Plot FPR*

---

**Description**

Plot observed false positive rate (FPR) for given adjusted p-value thresholds.

**Usage**

```
plot_fpr(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,
  xaxisrange = c(0, 1)
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status", aspects = "fpr")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_fpr(cobraplot, xaxisrange = c(0, 0.25))
```

---

plot\_fsrnbrcurve

*Plot number of features with s-value below threshold vs FSR*

---

**Description**

Plot the number of features with an s-value below a threshold vs the observed false sign rate (FSR), for given adjusted p-value thresholds and/or as curves traced out by considering all threshold values.

**Usage**

```
plot_fsrnbrcurve(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,
  xaxisrange = c(0, 1),
  plottype = c("curve", "points"),
  linewidth = 1
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
plottype	A character vector giving the type of plot to construct. Can be any combination of the two elements "curve" and "points".
linewidth	The line width used for plotting

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example_sval)
cobrapperf <- calculate_performance(cobradata_example_sval,
                                  cont_truth = "logFC",
                                  aspects = c("fsrnbr", "fsrnbrcurve"))
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_fsrnbrcurve(cobraplot, plottype = c("curve", "points"))
```

---

plot\_overlap                      *Plot Venn diagram*

---

**Description**

Plot a Venn diagram showing the overlaps among sets of significant feature for a given adjusted p-value threshold (see `thr_venn` argument of `calculate_performance`). Optionally, the truth can be included as a "perfect" method. Note that maximally five methods (including the truth, if applicable) can be compared.

**Usage**

```
plot_overlap(cobraplot, ...)
```

**Arguments**

cobraplot	A COBRAPlot object.
...	Additional arguments to <code>limma::vennDiagram</code> .

**Value**

Nothing, displays a graph

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = "overlap")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_overlap(cobraplot)
```

---

plot\_roc

*Plot ROC curves*

---

**Description**

Plot receiver operating characteristics (ROC) curves.

**Usage**

```
plot_roc(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  xaxisrange = c(0, 1),
  yaxisrange = c(0, 1),
  linewidth = 1
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.
yaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the y-axis, respectively.
linewidth	The line width used for plotting

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status", aspects = "roc")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)

plot_roc(cobraplot)
```

---

plot\_scatter

*Plot scatter plots*

---

**Description**

Plot scatter plots, indicating the relationship between observed values and a continuous truth.

**Usage**

```
plot_scatter(
  cobraplot,
  title = "",
  stripsize = 10,
  titlecol = "black",
  pointsize = 3,
  doflip = FALSE,
  dolog = FALSE
)
```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
doflip	A logical indicating whether to flip the axes when results are stratified by an annotation. By default (doflip = FALSE), stratification levels are shown as columns and methods as rows in the plot.
dolog	A logical indicating whether to log <sub>10</sub> -transform values before plotting.

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson

**Examples**

```

data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                  aspects = "scatter")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
plot_scatter(cobraplot)

```

plot\_tpr

*Plot TPR***Description**

Plot observed true positive rate (TPR) for given adjusted p-value thresholds.

**Usage**

```

plot_tpr(
  cobraplot,
  title = "",
  stripsize = 15,
  titlecol = "black",
  pointsize = 5,
  xaxisrange = c(0, 1)
)

```

**Arguments**

cobraplot	A COBRAPlot object.
title	A character string giving the title of the plot.
stripsize	A numeric value giving the size of the strip text, when the results are stratified by an annotation.
titlecol	A character string giving the color of the title.
pointsize	A numeric value giving the size of the plot characters.
xaxisrange	A numeric vector with two elements, giving the lower and upper boundary of the x-axis, respectively.

**Value**

A ggplot object

**Author(s)**

Charlotte Soneson



**Examples**

```

data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status", aspects = "tpr")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                  incltruth = TRUE)
plot_tpr(cobraplot)

```

plot\_upset

*Create UpSet plots***Description**

Generate UpSet plots showing the overlaps among sets of significant feature for a given adjusted p-value threshold (see `thr_venn` argument of `calculate_performance`). Optionally, the truth can be included as a "perfect" method. Note that if the results are stratified, only one category at a time can be displayed.

**Usage**

```

plot_upset(
  cobraplot,
  stratum = NULL,
  nsets = NULL,
  nintersects = NULL,
  sets.bar.color = NULL,
  ...
)

```

**Arguments**

<code>cobraplot</code>	A COBRAPlot object.
<code>stratum</code>	If results are stratified, the category to plot results for. Can be numeric or categorical (the name of the category).
<code>nsets</code>	The number of methods to include. By default, it is determined automatically from the <code>cobraplot</code> object.
<code>nintersects</code>	The number of set intersections to display. By default, it is determined automatically from the <code>cobraplot</code> object.
<code>sets.bar.color</code>	The colors to use for the bars in the UpSet plot. By default, they are extracted from the <code>plotcolors</code> slot of the <code>cobraplot</code> object.
<code>...</code>	Additional arguments to <code>UpSetR::upset</code> .

**Value**

Nothing, displays a graph

**Author(s)**

Charlotte Soneson

## References

- Lex and Gehlenborg (2014): Points of view: Sets and intersections. *Nature Methods* 11, 779.
- Lex et al (2014): UpSet: Visualization of intersecting sets. *IEEE Transactions on Visualization and Computer Graphics* 20(12), 1983-1992.

## Examples

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = "overlap")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                  incltruth = TRUE)

plot_upset(cobraplot)
plot_upset(cobraplot, order.by = "freq", decreasing = TRUE)

cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status",
                                  aspects = "overlap",
                                  splv = "expr_cat")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                  incltruth = TRUE)
plot_upset(cobraplot, stratum = "[2.85e+00,1.45e+01)")
```

---

prepare\_data\_for\_plot *Prepare data for plotting*

---

## Description

Prepare performance data provided in a COBRAPerformance object (obtained by [calculate\\_performance](#)) for plotting.

## Usage

```
prepare_data_for_plot(
  cobrapperf,
  keepmethods = NULL,
  inclooverall = TRUE,
  colorscheme = "hue_pal",
  faceted = TRUE,
  incltruth = TRUE,
  conditionalfill = TRUE
)
```

## Arguments

- |              |  |
|--------------|--|
| cobrapperf   | A COBRAPerformance object.   |
| keepmethods  | A character vector consisting of methods to retain for plotting (these should be a subset of basemethods(cobrapperf)), or NULL (indicating that all methods represented in cobrapperf should be retained). |
| inclooverall | A logical indicating whether the "overall" results should be included if the results are stratified by an annotation.  |

colorscheme	<p>Either a character string giving the color palette to use to define colors for the different methods, or a character vector with colors to use. The available pre-defined palettes depend on the number of different methods to distinguish. The choices are:</p> <ul style="list-style-type: none"> <li>- Accent (max 8 methods)</li> <li>- Dark2 (max 8 methods)</li> <li>- Paired (max 12 methods)</li> <li>- Pastel1 (max 9 methods)</li> <li>- Pastel2 (max 8 methods)</li> <li>- Set1 (max 9 methods)</li> <li>- Set2 (max 8 methods)</li> <li>- Set3 (max 12 methods)</li> <li>- hue_pal</li> <li>- rainbow</li> <li>- heat</li> <li>- terrain</li> <li>- topo</li> <li>- cm</li> </ul> <p>If the number of allowed methods is exceeded, the colorscheme defaults to hue_pal.</p>
facetted	A logical indicating whether the results should be split into subpanels when stratified by an annotation (TRUE), or kept in the same panel but shown with different colors (FALSE).
incltruth	A logical indicating whether the truth should be included in Venn diagrams.
conditionalfill	A logical indicating whether the points (in FDR/TPR, FDR/NBR, FSR/NBR plots) should be filled conditional on whether they satisfy the imposed criterion (e.g., false discovery rate control at imposed threshold).

**Value**

A COBRAPlot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status",
                                   cont_truth = "none",
                                   aspects = c("fdrtpr", "fdrtprcurve",
                                               "tpr", "roc"),
                                   thrs = c(0.01, 0.05, 0.1), splv = "none")
cobraplot <- prepare_data_for_plot(cobrapperf, keepmethods = NULL,
                                   colorscheme = "Dark2")

## User-specified colors
cobraplot2 <- prepare_data_for_plot(cobrapperf, keepmethods = NULL,
                                   colorscheme = c("blue", "red", "green"))
```

---

pval *Accessor and replacement functions for pval slot*

---

### Description

Accessor and replacement functions for the pval slot in a COBRADData object.

### Usage

```
pval(x, ...)
```

```
pval(x, ...) <- value
```

```
## S4 method for signature 'COBRADData'
```

```
pval(x)
```

```
## S4 replacement method for signature 'COBRADData,data.frame'
```

```
pval(x) <- value
```

### Arguments

x                    A COBRADData object.

...                  Additional arguments.

value                A data frame containing p-values for each feature and each method.

### Value

The accessor function returns a data frame containing p-values for each feature and each method.

### Author(s)

Charlotte Soneson

### Examples

```
data(cobradata_example)
```

```
head(pval(cobradata_example))
```

---

reorder\_levels *Reorder levels in COBRAPlot object*

---

### Description

Reorder levels in COBRAPlot object to achieve desired ordering in figure legends etc. If `facetted(cobraplot)` is TRUE, the releveling will be applied to the "method" column. If `facetted(cobraplot)` is FALSE, it will be applied to the "fullmethod" column.

### Usage

```
reorder_levels(cobraplot, levels)
```

**Arguments**

`cobraplot` A COBRAPlot object

`levels` A character vector giving the order of the levels. Any values not present in the COBRAPlot object will be removed. Any methods present in the COBRAPlot object but not contained in this vector will be added at the end.

**Value**

A COBRAPlot object

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example_sval)
cobrapperf <- calculate_performance(cobradata_example_sval,
                                   binary_truth = "status", aspects = "fpr")
cobraplot <- prepare_data_for_plot(cobrapperf, colorscheme = "Dark2",
                                   incltruth = TRUE)
cobraplot <- reorder_levels(cobraplot, c("Method2", "Method1"))
```

---

roc

*Accessor and replacement functions for roc slot*

---

**Description**

Accessor and replacement functions for the roc slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
roc(x, ...)

roc(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
roc(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
roc(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
roc(x) <- value
```

**Arguments**

`x` A COBRAPerformance or COBRAPlot object.

`...` Additional arguments.

`value` A data frame giving information necessary to generate ROC curves for each method and each stratification level.

**Value**

The accessor function returns a data frame giving information necessary to generate ROC curves for each method and each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                   binary_truth = "status", aspects = "roc")
head(roc(cobrapperf))
```

---

 scatter

---

*Accessor and replacement functions for scatter slot*


---

**Description**

Accessor and replacement functions for the scatter slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```
scatter(x, ...)

scatter(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
scatter(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
scatter(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
scatter(x) <- value
```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information necessary to generate scatter plots of observed vs true values for each method and each stratification level.

**Value**

The accessor function returns a data frame giving information necessary to generate scatter plots of observed vs true values for each method and each stratification level.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example, cont_truth = "logFC",
                                  aspects = "scatter")
head(scatter(cobrapperf))
```

---

score *Accessor and replacement functions for score slot*

---

**Description**

Accessor and replacement functions for the score slot in a COBRADData object.

**Usage**

```
score(x, ...)
score(x, ...) <- value

## S4 method for signature 'COBRADData'
score(x)

## S4 replacement method for signature 'COBRADData,data.frame'
score(x) <- value
```

**Arguments**

x                    A COBRADData object.  
...                   Additional arguments.  
value                A data frame containing scores for each feature and each method.

**Value**

The accessor function returns a data frame containing scores for each feature and each method.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
head(score(cobradata_example))
```

---

`splv`*Accessor and replacement functions for splv slot*

---

**Description**

Accessor and replacement functions for the `splv` slot in a `COBRAPerformance` or `COBRAPlot` object.

**Usage**

```
splv(x, ...)  
  
splv(x, ...) <- value  
  
## S4 method for signature 'COBRAPerformance'  
splv(x)  
  
## S4 replacement method for signature 'COBRAPerformance,character'  
splv(x) <- value  
  
## S4 replacement method for signature 'COBRAPlot,character'  
splv(x) <- value
```

**Arguments**

<code>x</code>	A <code>COBRAPerformance</code> or <code>COBRAPlot</code> object.
<code>...</code>	Additional arguments.
<code>value</code>	A character string giving the name of a feature annotation to use for stratification.

**Value**

The accessor function returns a character string giving the name of a feature annotation to use for stratification.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)  
cobraperf <- calculate_performance(cobradata_example,  
                                 binary_truth = "status",  
                                 aspects = "fdrtpr", splv = "expr_cat")  
  
splv(cobraperf)
```



---

stratiflevels	<i>Accessor function for stratification levels</i>
---------------	--

---

**Description**

Accessor function to extract the stratification levels that are represented in a COBRAPerformance or COBRAPlot object.

**Usage**

```
stratiflevels(x, ...)  
  
## S4 method for signature 'COBRAPerformance'  
stratiflevels(x)
```

**Arguments**

x	A COBRAPerformance or COBRAPlot object
...	Additional arguments

**Value**

A character vector of all stratification levels represented in the object

**Author(s)**

Charlotte Sonesson

**Examples**

```
data(cobradata_example)  
cobrapperf <- calculate_performance(cobradata_example,  
                                   binary_truth = "status",  
                                   aspects = "fdrtpr", splv = "expr_cat",  
                                   maxsplit = 4)  
  
stratiflevels(cobrapperf)
```

---

sval	<i>Accessor and replacement functions for sval slot</i>
------	---

---

**Description**

Accessor and replacement functions for the sval slot in a COBRADData object.

**Usage**

```

sval(x, ...)

sval(x, ...) <- value

## S4 method for signature 'COBRADData'
sval(x)

## S4 replacement method for signature 'COBRADData,data.frame'
sval(x) <- value

```

**Arguments**

x	A COBRADData object.
...	Additional arguments.
value	A data frame containing s-values for each feature and each method. If the object does not have an s-value slot (older versions of the class did not have this slot), an empty data frame is returned for simplicity.

**Value**

The accessor function returns a data frame containing s-values for each feature and each method.

**Author(s)**

Charlotte Soneson

**Examples**

```

data(cobradata_example)
head(sval(cobradata_example))

```

---

tpr

---

*Accessor and replacement functions for tpr slot*


---

**Description**

Accessor and replacement functions for the tpr slot in a COBRAPerformance or COBRAPlot object.

**Usage**

```

tpr(x, ...)

tpr(x, ...) <- value

## S4 method for signature 'COBRAPerformance'
tpr(x)

## S4 replacement method for signature 'COBRAPerformance,data.frame'
tpr(x) <- value

## S4 replacement method for signature 'COBRAPlot,data.frame'
tpr(x) <- value

```

**Arguments**

x	A COBRAPerformance or COBRAPlot object.
...	Additional arguments.
value	A data frame giving information about the observed TPR for each method and each stratification level, at various adjusted p-value thresholds.

**Value**

The accessor function returns a data frame giving information about the observed TPR for each method and each stratification level, at various adjusted p-value thresholds.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
cobrapperf <- calculate_performance(cobradata_example,
                                  binary_truth = "status", aspects = "tpr")
head(tpr(cobrapperf))
```

---

truth	<i>Accessor and replacement functions for truth slot</i>
-------	--

---

**Description**

Accessor and replacement functions for the truth slot in a COBRADData object.

**Usage**

```
truth(x, ...)
truth(x, ...) <- value

## S4 method for signature 'COBRADData'
truth(x)

## S4 replacement method for signature 'COBRADData,data.frame'
truth(x) <- value
```

**Arguments**

x	A COBRADData object.
...	Additional arguments.
value	A data frame containing true assignments and/or scores for features, together with other feature annotations to use for stratification of performance calculations.

**Value**

The accessor function returns a data frame containing true assignments and/or scores for features, together with other feature annotations to use for stratification of performance calculations.

**Author(s)**

Charlotte Soneson

**Examples**

```
data(cobradata_example)
head(truth(cobradata_example))
```

---

update_cobradata	<i>Update COBRADData object to the current version of the class format</i>
------------------	--

---

**Description**

Update a COBRADData object generated by a previous version of the package to the latest version.

**Usage**

```
update_cobradata(object, quiet = FALSE)
```

**Arguments**

object	A COBRADData object
quiet	Set to TRUE to disable messages listing the modifications that are applied to the object

**Value**

An updated COBRADData object

**Author(s)**

Charlotte Soneson

**Examples**

```
## Generate COBRADData object
set.seed(123)
pval <- data.frame(m1 = runif(100), m2 = runif(100),
                  row.names = paste0("F", 1:100))
truth <- data.frame(status = round(runif(100)),
                   row.names = paste0("F", 1:100))
cobradata <- COBRADData(pval = pval, truth = truth)

## Update object if needed
cobradata <- update_cobradata(cobradata)
```

---

update\_cobraperformance

*Update COBRAPerformance or COBRAPlot object to the current version of the class format*

---

### **Description**

Update a COBRAPerformance or COBRAPlot object generated by a previous version of the package to the latest version.

### **Usage**

```
update_cobraperformance(object, quiet = FALSE)
```

### **Arguments**

object	A COBRAPerformance or COBRAPlot object
quiet	Set to TRUE to disable messages listing the modifications that are applied to the object

### **Value**

An updated COBRAPerformance or COBRAPlot object

### **Author(s)**

Charlotte Soneson

### **Examples**

```
cobradata <- cobradata_example_sval  
  
cobraperf <- calculate_performance(cobradata,  
                                binary_truth = "status",  
                                aspects = "fpr")  
cobraperf <- update_cobraperformance(cobraperf)
```

# Index

## \* datasets

- cobradata\_example, 9
- cobradata\_example\_sval, 9
- .COBRADData (COBRADData-class), 7
- .COBRAPerformance
  - (COBRAPerformance-class), 10
- .COBRAPlot (COBRAPlot-class), 12
- [ (Extract), 16
- [, COBRADData-method (Extract), 16
- [, COBRAPerformance-method (Extract), 16
- [, COBRAPlot-method (Extract), 16
  
- basemethods, 3
- basemethods, COBRAPerformance-method
  - (basemethods), 3
- basemethods, COBRAPlot-method
  - (basemethods), 3
  
- calculate\_adj, 4, 8
- calculate\_performance, 4, 10, 42
- COBRAapp, 6
- COBRADData (COBRADData-class), 7
- COBRADData-class, 7
- cobradata\_example, 9
- cobradata\_example\_sval, 9
- COBRADData\_from\_text (COBRADData-class), 7
- COBRADData\_to\_text, 6
- COBRADData\_to\_text (COBRADData-class), 7
- COBRAPerformance
  - (COBRAPerformance-class), 10
- COBRAPerformance-class, 10
- COBRAPlot (COBRAPlot-class), 12
- COBRAPlot, COBRAPerformance-method
  - (coerce), 14
- COBRAPlot-class, 12
- coerce, 14
- coerce, (coerce), 14
- coerce, COBRAPerformance, COBRAPlot-method
  - (coerce), 14
- coerce, COBRAPlot, COBRAPerformance-method
  - (coerce), 14
- corr, 14
- corr, COBRAPerformance-method (corr), 14
- corr, COBRAPlot-method (corr), 14
  
- corr<- (corr), 14
- corr<-, COBRAPerformance, data.frame-method
  - (corr), 14
- corr<-, COBRAPlot, data.frame-method
  - (corr), 14
  
- deviation, 15
- deviation, COBRAPerformance-method
  - (deviation), 15
- deviation, COBRAPlot-method (deviation), 15
- deviation<- (deviation), 15
- deviation<-, COBRAPerformance, data.frame-method
  - (deviation), 15
- deviation<-, COBRAPlot, data.frame-method
  - (deviation), 15
  
- Extract, 16
  
- faceted, 17
- faceted, COBRAPlot-method (faceted), 17
- faceted<- (faceted), 17
- faceted<-, COBRAPlot, logical-method
  - (faceted), 17
- fdrnbr, 18
- fdrnbr, COBRAPerformance-method
  - (fdrnbr), 18
- fdrnbr, COBRAPlot-method (fdrnbr), 18
- fdrnbr<- (fdrnbr), 18
- fdrnbr<-, COBRAPerformance, data.frame-method
  - (fdrnbr), 18
- fdrnbr<-, COBRAPlot, data.frame-method
  - (fdrnbr), 18
- fdrnbrcurve, 19
- fdrnbrcurve, COBRAPerformance-method
  - (fdrnbrcurve), 19
- fdrnbrcurve, COBRAPlot-method
  - (fdrnbrcurve), 19
- fdrnbrcurve<- (fdrnbrcurve), 19
- fdrnbrcurve<-, COBRAPerformance, data.frame-method
  - (fdrnbrcurve), 19
- fdrnbrcurve<-, COBRAPlot, data.frame-method
  - (fdrnbrcurve), 19
- fdrtp, 20

- fdrtp, COBRAPerformance-method (fdrtp), 20
- fdrtp, COBRAPlot-method (fdrtp), 20
- fdrtp<- (fdrtp), 20
- fdrtp<-, COBRAPerformance, data.frame-method (fdrtp), 20
- fdrtp<-, COBRAPlot, data.frame-method (fdrtp), 20
- fdrtpcurve, 21
- fdrtpcurve, COBRAPerformance-method (fdrtpcurve), 21
- fdrtpcurve, COBRAPlot-method (fdrtpcurve), 21
- fdrtpcurve<- (fdrtpcurve), 21
- fdrtpcurve<-, COBRAPerformance, data.frame-method (fdrtpcurve), 21
- fdrtpcurve<-, COBRAPlot, data.frame-method (fdrtpcurve), 21
- fpc, 22
- fpc, COBRAPerformance-method (fpc), 22
- fpc, COBRAPlot-method (fpc), 22
- fpc<- (fpc), 22
- fpc<-, COBRAPerformance, data.frame-method (fpc), 22
- fpc<-, COBRAPlot, data.frame-method (fpc), 22
- fpr, 23
- fpr, COBRAPerformance-method (fpr), 23
- fpr, COBRAPlot-method (fpr), 23
- fpr<- (fpr), 23
- fpr<-, COBRAPerformance, data.frame-method (fpr), 23
- fpr<-, COBRAPlot, data.frame-method (fpr), 23
- fsrc, 24
- fsrc, COBRAPerformance-method (fsrc), 24
- fsrc, COBRAPlot-method (fsrc), 24
- fsrc<- (fsrc), 24
- fsrc<-, COBRAPerformance, data.frame-method (fsrc), 24
- fsrc<-, COBRAPlot, data.frame-method (fsrc), 24
- fsrccurve, 25
- fsrccurve, COBRAPerformance-method (fsrccurve), 25
- fsrccurve, COBRAPlot-method (fsrccurve), 25
- fsrccurve<- (fsrccurve), 25
- fsrccurve<-, COBRAPerformance, data.frame-method (fsrccurve), 25
- fsrccurve<-, COBRAPlot, data.frame-method (fsrccurve), 25
- fsrcnbc, 25
- fsrcnbc, COBRAPerformance-method (fsrcnbc), 25
- fsrcnbc, COBRAPlot-method (fsrcnbc), 25
- fsrcnbc<- (fsrcnbc), 25
- fsrcnbc<-, COBRAPerformance, data.frame-method (fsrcnbc), 25
- fsrcnbc<-, COBRAPlot, data.frame-method (fsrcnbc), 25
- fsrcnbcrcurve, 25
- fsrcnbcrcurve, COBRAPerformance-method (fsrcnbcrcurve), 25
- fsrcnbcrcurve, COBRAPlot-method (fsrcnbcrcurve), 25
- fsrcnbcrcurve<- (fsrcnbcrcurve), 25
- fsrcnbcrcurve<-, COBRAPerformance, data.frame-method (fsrcnbcrcurve), 25
- fsrcnbcrcurve<-, COBRAPlot, data.frame-method (fsrcnbcrcurve), 25
- maxsplit, 26
- maxsplit, COBRAPerformance-method (maxsplit), 26
- maxsplit, COBRAPlot-method (maxsplit), 26
- maxsplit<- (maxsplit), 26
- maxsplit<-, COBRAPerformance, numeric-method (maxsplit), 26
- maxsplit<-, COBRAPlot, numeric-method (maxsplit), 26
- onlyshared, 27
- onlyshared, COBRAPerformance-method (onlyshared), 27
- onlyshared, COBRAPlot-method (onlyshared), 27
- onlyshared<- (onlyshared), 27
- onlyshared<-, COBRAPerformance, logical-method (onlyshared), 27
- onlyshared<-, COBRAPlot, logical-method (onlyshared), 27
- overlap, 28
- overlap, COBRAPerformance-method (overlap), 28
- overlap, COBRAPlot-method (overlap), 28
- overlap<- (overlap), 28
- overlap<-, COBRAPerformance, list\_df-method (overlap), 28
- overlap<-, COBRAPlot, list\_df-method (overlap), 28
- padj, 29
- padj, COBRADData-method (padj), 29
- padj<- (padj), 29
- padj<-, COBRADData, data.frame-method (padj), 29
- plot\_corr, 30
- plot\_deviation, 31
- plot\_fdrnbcrcurve, 32
- plot\_fdrtpcurve, 33
- plot\_fpc, 34
- plot\_fpr, 35
- plot\_fsrcnbcrcurve, 36
- plot\_overlap, 37
- plot\_roc, 38
- plot\_scatter, 39
- plot\_tpr, 40
- plot\_upset, 41
- plotcolors, 29
- plotcolors, COBRAPlot-method (plotcolors), 29
- plotcolors<- (plotcolors), 29

plotcolors<- ,COBRAPlot,character-method  
     (plotcolors), 29  
 prepare\_data\_for\_plot, 12, 42  
 pval, 44  
 pval, COBRADData-method (pval), 44  
 pval<- (pval), 44  
 pval<- ,COBRADData,data.frame-method  
     (pval), 44  
  
 reorder\_levels, 44  
 roc, 45  
 roc, COBRAPerformance-method (roc), 45  
 roc, COBRAPlot-method (roc), 45  
 roc<- (roc), 45  
 roc<- ,COBRAPerformance,data.frame-method  
     (roc), 45  
 roc<- ,COBRAPlot,data.frame-method  
     (roc), 45  
  
 scatter, 46  
 scatter, COBRAPerformance-method  
     (scatter), 46  
 scatter, COBRAPlot-method (scatter), 46  
 scatter<- (scatter), 46  
 scatter<- ,COBRAPerformance,data.frame-method  
     (scatter), 46  
 scatter<- ,COBRAPlot,data.frame-method  
     (scatter), 46  
 score, 47  
 score, COBRADData-method (score), 47  
 score<- (score), 47  
 score<- ,COBRADData,data.frame-method  
     (score), 47  
 splv, 48  
 splv, COBRAPerformance-method (splv), 48  
 splv, COBRAPlot-method (splv), 48  
 splv<- (splv), 48  
 splv<- ,COBRAPerformance,character-method  
     (splv), 48  
 splv<- ,COBRAPlot,character-method  
     (splv), 48  
 stratiflevels, 49  
 stratiflevels, COBRAPerformance-method  
     (stratiflevels), 49  
 stratiflevels, COBRAPlot-method  
     (stratiflevels), 49  
 sval, 49  
 sval, COBRADData-method (sval), 49  
 sval<- (sval), 49  
 sval<- ,COBRADData,data.frame-method  
     (sval), 49  
  
 tpr, 50  
 tpr, COBRAPerformance-method (tpr), 50  
 tpr, COBRAPlot-method (tpr), 50  
 tpr<- (tpr), 50  
 tpr<- ,COBRAPerformance,data.frame-method  
     (tpr), 50  
 tpr<- ,COBRAPlot,data.frame-method  
     (tpr), 50  
 truth, 51  
 truth, COBRADData-method (truth), 51  
 truth<- (truth), 51  
 truth<- ,COBRADData,data.frame-method  
     (truth), 51  
  
 update\_cobradata, 52  
 update\_cobraperformance, 53