

# Package ‘omXplore’

April 10, 2025

**Type** Package

**Title** Vizualization tools for 'omics' datasets with R

**Version** 1.1.2

**Description** This package contains a collection of functions (written as shiny modules) for the visualisation and the statistical analysis of omics data. These plots can be displayed individually or embedded in a global Shiny module.  
Additionally, it is possible to integrate third party modules to the main interface of the package omXplore.

**License** Artistic-2.0

**Depends** R (>= 4.4.0), methods

**Imports** DT, shiny, bs4Dash, waiter, thematic, MSnbase, PSMATCH, SummarizedExperiment, MultiAssayExperiment, shinyBS, shinyjs, shinyjqui, RColorBrewer, gplots, highcharter, visNetwork, tibble, grDevices, stats, utils, htmlwidgets, vioplot, graphics, FactoMineR, dendextend, dplyr, factoextra, tidyr, nipals

**Suggests** knitr, rmarkdown, BiocStyle, testthat, Matrix, graph

**biocViews** Software, ShinyApps, MassSpectrometry, DataRepresentation, GUI, QualityControl

**NeedsCompilation** no

**Collate** 'mod\_explode\_graphs.R' 'Infos\_adjacencyMatrix.R'  
'Prostar\_1x.R' 'convert\_to\_mae.R' 'doc-data.R'  
'external\_apps\_examples.R' 'get\_pep\_prot\_CC.R'  
'mae\_accessors.R' 'metacell\_utils.R' 'mod\_colorLegend.R'  
'modules.R' 'omXplore\_cc.R' 'omXplore\_corrmatrix.R'  
'omXplore\_density.R' 'omXplore\_format\_DT.R'  
'omXplore\_heatmap.R' 'omXplore\_intensity.R'  
'omXplore\_PCA\_nipals.R' 'omXplore\_pca.R'  
'omXplore\_plots\_tracking.R' 'omXplore\_tabExplorer.R'  
'omXplore\_variance.R' 'omXplore\_view\_dataset.R' 'palette.R'  
'plot\_boxplot.R' 'plot\_heatmap.R' 'plot\_pca.R' 'plot\_violin.R'  
'utils.R' 'zzz.R'

**RoxygenNote** 7.3.2  
**Encoding** UTF-8  
**LazyData** false  
**URL** <https://github.com/prostarproteomics/omXplore>,  
<https://prostarproteomics.github.io/omXplore/>  
**BugReports** <https://github.com/prostarproteomics/omXplore/issues>  
**Roxygen** list(markdown = TRUE)  
**VignetteBuilder** knitr  
**git\_url** <https://git.bioconductor.org/packages/omXplore>  
**git\_branch** devel  
**git\_last\_commit** 4524431  
**git\_last\_commit\_date** 2025-03-29  
**Repository** Bioconductor 3.21  
**Date/Publication** 2025-04-09  
**Author** Samuel Wieczorek [aut, cre] (ORCID:  
<https://orcid.org/0000-0002-5016-1203>),  
Thomas Burger [aut],  
Enora Fremy [ctb],  
Cyril Ariztegui [ctb]  
**Maintainer** Samuel Wieczorek <samuel.wieczorek@cea.fr>

## Contents

accessors . . . . .	3
BuildColorStyles . . . . .	5
Build_enriched_qdata . . . . .	5
color-legend . . . . .	6
converters . . . . .	7
corrmatrix . . . . .	9
customChart . . . . .	10
customExportMenu . . . . .	11
density-plot . . . . .	12
ds-cc . . . . .	13
ds-pca . . . . .	14
ds-view . . . . .	17
external_app . . . . .	19
FormatDataForDT . . . . .	20
format_DT . . . . .	21
GetPkgVersion . . . . .	22
globals . . . . .	23
intensity-plots . . . . .	23
is.listOf . . . . .	25

omXplore-modules . . . . .	26
omXplore_heatmap . . . . .	27
omXplore_tabExplorer . . . . .	29
palette . . . . .	30
pep_prot_CC . . . . .	31
pkgs.require . . . . .	32
plot-variance . . . . .	33
plots_tracking . . . . .	34
Prostar-1x-compatible . . . . .	35
q_metadata . . . . .	36
sub_R25 . . . . .	37
vdata . . . . .	38

## Index 39

---

accessors	<i>Accessors functions</i>
-----------	----------------------------

---

### Description

Functions used to access the additional plots in the instances of the class `MultiAssayExperiment`.

### Usage

```

get_adjacencyMatrix(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_adjacencyMatrix(object)

get_design(object, ...)

## S4 method for signature 'MultiAssayExperiment'
get_design(object)

get_group(object, ...)

## S4 method for signature 'MultiAssayExperiment'
get_group(object)

get_metacell(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_metacell(object, slot.name = c("metacell", "qMetacell"))

get_cc(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_cc(object)

```

```

get_parentProtId(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_parentProtId(object)

get_colID(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_colID(object)

get_type(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_type(object)

get_pkg_version(object, ...)

## S4 method for signature 'SummarizedExperiment'
get_pkg_version(object)

```

### Arguments

<code>object</code>	An instance of class <code>SummarizedExperiment</code> .
<code>...</code>	Additional parameters
<code>slot.name</code>	The name of the slot dedicated to cell metadata to search. Default values are 'metacell' and 'qMetacell'

### Value

See individual method description for the return value.

If exists, the slot value requested.

A `DataFrame` containing the adjacency matrix of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

A `data.frame` containing the metadata of the dataset

**Examples**

```
## -----  
## Accessing slots from a MSnSet dataset  
## -----  
data(sub_R25)  
se1 <- sub_R25[[1]]  
parentProtId <- get_parentProtId(se1)  
colID <- get_colID(se1)  
type <- get_type(se1)  
metacell <- get_metacell(se1)  
conds <- get_group(sub_R25)
```

---

BuildColorStyles      *Build color style for DT tables*

---

**Description**

This function builds a list which is used for styling DT tables with the function `DT:::styleEqual()`

**Usage**

```
BuildColorStyles(type)
```

**Arguments**

type                      The type of dataset. Available values are protein and peptide

**Value**

A list

---

Build\_enriched\_qdata      *Builds enriched assay with cell metadata info*

---

**Description**

If the cell metadata exists in the object of class `SummarizedExperiment`, then these information are added to the quantitative data so as to use styles with the functions of the package `DT`.

**Usage**

```
Build_enriched_qdata(obj)
```

**Arguments**

obj                        An instance of the class `SummarizedExperiment`

**Value**

A data.frame with new columns corresponding to the cell metadata (if exists)

---

color-legend

*Color legend for DaparToolshed*

---

**Description**

Shows a legend based on the tags in the package 'DaparToolshed'

**Usage**

```

custom_metacell_colors()

colorLegend_ui(id)

colorLegend_server(
  id,
  presentTags = reactive({
    NULL
  }),
  hide.white = TRUE
)

colorLegend(obj = SummarizedExperiment::SummarizedExperiment())

```

**Arguments**

id	A character(1) which is the id of the shiny module.
presentTags	A vector of character() which correspond to the tags.
hide.white	A boolean() to indicate whether the white cells must be hidden or not.
obj	An instance of the class SummarizedExperiment.

**Value**

A vector  
 NA  
 NA  
 A shiny app

**Examples**

```
## Not run:  
data(vdata)  
shiny::runApp(colorLegend(vdata[[1]]))  
  
## End(Not run)
```

---

converters

*Convert to enriched MultiAssayExperiment*

---

**Description**

The resulting object is an instance of the MultiAssayExperiment class.

**Usage**

```
convert_to_mae(obj)  
  
MSnSet_to_mae(obj)  
  
matrix_to_mae(obj)  
  
df_to_mae(obj)  
  
Compute_CC(obj)  
  
QFeatures_to_mae(obj)  
  
SE_to_mae(obj)  
  
MAE_to_mae(obj)  
  
Check_se_Consistency(obj)  
  
list_to_se(l1)  
  
Check_List_consistency(l1)  
  
listOfLists_to_mae(obj, colData = NULL)  
  
listOfSE_to_mae(obj)  
  
Check_MSnSet_Consistency(obj)  
  
matrix_to_se(obj)
```

```

df_to_se(obj)
MSnSet_to_se(obj)
Build_X_CC(se)
listOfMSnSet_to_mae(obj)
listOfmatrix_to_mae(obj)
listOfdf_to_mae(obj)

```

### Arguments

obj	An object compliant with the formats accepted by omXplore
ll	A list
colData	A data.frame()
se	AN instance of the class SummarizedExperiment

### Value

An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An instance of SimpleList  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 A boolean(1)  
 An enriched instance of the class SummarizedExperiment  
 A boolean(1)  
 An enriched instance of the class MultiAssayExperiment  
 An enriched instance of the class MultiAssayExperiment  
 A boolean(1)  
 An enriched instance of the class SummarizedExperiment  
 An enriched instance of the class SummarizedExperiment  
 An enriched instance of the class SummarizedExperiment  
 An enriched instance of the class SummarizedExperiment  
 An enriched instance of the class MultiAssayExperiment



An enriched instance of the class MultiAssayExperiment

An enriched instance of the class MultiAssayExperiment

### Examples

```
## Not run:

#-----
# Conversion of a MultiAssayExperiment instance
#-----
data(miniACC, package = 'MultiAssayExperiment')
convert_to_mae(miniACC)

## End(Not run)
```

---

corrmatrix	<i>Displays a correlation matrix of the quantitative data of a numeric matrix.</i>
------------	--

---

### Description

Displays a correlation matrix of the quantitative data of a numeric matrix.

### Usage

```
omXplore_corrmatrix_ui(id)

omXplore_corrmatrix_server(
  id,
  obj = reactive({
    NULL
  }),
  i = reactive({
    NULL
  })
)

corrMatrix(data, rate = 0.5, showValues = FALSE)

omXplore_corrmatrix(obj, i)
```

### Arguments

id	A character(1) which is the id of the shiny module.
obj	An instance of the class SummarizedExperiment
i	An integer which is the index of the assay in the param obj

data	An object of class 'matrix'
rate	The rate parameter to control the exponential law for the gradient of colors
showValues	A boolean which indicates whether to show values in the correlation plot.

**Value**

NA  
 NA  
 A plot  
 A shiny app

**Examples**

```
if (interactive()) {
  data(vdata)
  omXplore_corrmatrix(vdata, 1)
}
```

---

 customChart

---

*Customised resetZoom Button of highcharts plots*


---

**Description**

Customised resetZoom Button of highcharts plots

**Usage**

```
customChart(
  hc,
  chartType = "scatter",
  zoomType = "None",
  width = 0,
  height = 0
)
```

**Arguments**

hc	A highcharter object
chartType	The type of the plot
zoomType	The type of the zoom (one of "x", "y", "xy", "None")
width	The width of the plot
height	The height of the plot

**Value**

A highchart plot

**Author(s)**

Samuel Wieczorek

**Examples**

```
## Not run:
library(highcharter)
hc <- highchart()
hc_chart(hc, type = "line")
hc_add_series(hc, data = c(29, 71, 40))
customChart(hc)

## End(Not run)
```

---

customExportMenu      #' @title Customised contextual menu of highcharts plots

---

**Description**

#' @title Customised contextual menu of highcharts plots

**Usage**

```
customExportMenu(hc, fname)
```

**Arguments**

hc	A highcharter object
fname	The filename under which the plot has to be saved

**Value**

A contextual menu for highcharts plots

**Author(s)**

Samuel Wieczorek

**Examples**

```
NULL
```

---

density-plot	<i>Displays a correlation matrix of the quantitative data of a numeric matrix.</i>
--------------	--

---

### Description

Displays a correlation matrix of the quantitative data of a numeric matrix.

### Usage

```
omXplore_density_ui(id)
```

```
omXplore_density_server(
  id,
  obj = reactive({
    NULL
  }),
  i = reactive({
    1
  }),
  pal.name = reactive({
    NULL
  })
)
```

```
densityPlot(data, conds = NULL, pal.name = NULL)
```

```
omXplore_density(obj, i)
```

### Arguments

id	A character(1) which is the id of the shiny module.
obj	An instance of the class SummarizedExperiment
i	An integer which is the index of the assay in the param obj
pal.name	A character(1) which is the name of the palette from the package <a href="#">RColorBrewer</a> from which the colors are taken. Default value is 'Set1'.
data	A data.frame() of quantitative data
conds	A vector indicating the name of each sample.

### Value

NA

NA

A plot

A shiny app

## Examples

```
## Not run:
  data(vdata)
  shiny::runApp(omXplore_density(vdata, 1))

## End(Not run)

## Not run:
data(vdata)
qdata <- SummarizedExperiment::assay(vdata[[1]])
conds <- get_group(vdata)
densityPlot(qdata, conds)

## End(Not run)
```

---

ds-cc

*plots\_cc\_ui and plots\_cc\_server*

---

## Description

A shiny Module.

## Usage

```
omXplore_cc_ui(id)

omXplore_cc_server(
  id,
  obj = reactive({
    NULL
  }),
  i = reactive({
    NULL
  })
)

omXplore_cc(obj, i)
```

## Arguments

id	A character(1) which is the id of the shiny module.
obj	An instance of SummarizedExperiment class
i	An integer which is the index of the assay in the param obj

**Value**

A shiny module

A shiny plot

A shiny module

A shiny app

A shiny app

**Examples**

```
if (interactive()) {  
  data(vdata)  
  omXplore_cc(vdata, 1)  
}
```

---

ds-pca

*my\_PCA*

---

**Description**

Process a PCA, using nipals or FactoMineR, on a quantitative dataset.

This method plots a bar plot which represents the distribution of the number of missing values (NA) per lines (ie proteins).

- `wrapper_pca()`
- `plotPCA_Eigen_hc()`: plots the eigen values of PCA with the highcharts library
- `plotPCA_Eigen()`: plots the eigen values of PCA
- `plotPCA_Var()`
- `plotPCA_Ind()`

**Usage**

```
my_PCA(  
  X,  
  scale.unit = TRUE,  
  ncp = min(12, nrow(X) - 1, ncol(X)),  
  ind.sup = NULL,  
  quanti.sup = NULL,  
  quali.sup = NULL,  
  row.w = NULL,  
  col.w = NULL,  
  graph = FALSE,  
  axes = c(1, 2),  
  approach = "FM",
```

```

    gramschmidt = TRUE
  )

omXplore_pca_ui(id)

omXplore_pca_server(id, obj, i)

omXplore_pca(obj, i)

wrapper_pca(
  qdata,
  group,
  var.scaling = TRUE,
  ncp = NULL,
  approach = "FM",
  gramschmidt = TRUE
)

plotPCA_Eigen(res.pca)

plotPCA_Var(res.pca, chosen.axes = c(1, 2))

plotPCA_Ind(res.pca, chosen.axes = c(1, 2))

plotPCA_Eigen_hc(res.pca)

```

### Arguments

<code>X</code>	a <code>data.frame()</code> of quantitative data
<code>scale.unit</code>	See <code>FactoMineR::PCA()</code>
<code>ncp</code>	See <code>FactoMineR::PCA()</code>
<code>ind.sup</code>	See <code>FactoMineR::PCA()</code>
<code>quanti.sup</code>	See <code>FactoMineR::PCA()</code>
<code>quali.sup</code>	See <code>FactoMineR::PCA()</code>
<code>row.w</code>	See <code>FactoMineR::PCA()</code>
<code>col.w</code>	See <code>FactoMineR::PCA()</code>
<code>graph</code>	See <code>FactoMineR::PCA()</code>
<code>axes</code>	See <code>FactoMineR::PCA()</code>
<code>approach</code>	a string corresponding to the package to use for PCA (if no NA, default is "FM" for <code>FactoMineR</code> )
<code>gramschmidt</code>	A boolean indicating whether to use Gram-Schmidt orthogonalization or not.
<code>id</code>	A character(1) which is the id of the shiny module.
<code>obj</code>	An instance of the class <code>MultiAssayExperiment</code> .
<code>i</code>	An integer which is the index of the assay in the param <code>obj</code>

qdata	A data.frame() of quantitative data
group	A vector with the name of samples
var.scaling	A boolean indicating whether to scale the data or not
res.pca	The result of the function FactoMineR::PCA()
chosen.axes	See the parameter 'axes' of the function factoextra::fviz_pca_var()

**Value**

res.pca a "PCA" "list" object  
 NA  
 NA  
 A shiny app  
 The result of the function FactoMineR::PCA()  
 A plot  
 A plot  
 A plot  
 A plot

**Author(s)**

Samuel Wieczorek, Enora Fremy

**Examples**

```
data(vdata)
obj <- vdata[[1]]
res.pca <- my_PCA(SummarizedExperiment::assay(obj), approach = "FM")
plotPCA_Eigen(res.pca)
plotPCA_Var(res.pca)
plotPCA_Eigen_hc(res.pca)
plotPCA_Ind(res.pca)

## Not run:
data(vdata)
library(shiny)
library(shinyWidgets)
library(dplyr)
library(highcharter)
# Replace missing values for the example
sel <- is.na(SummarizedExperiment::assay(vdata, 1))
SummarizedExperiment::assay(vdata[[1]])[sel] <- 0
# SummarizedExperiment::assay(vdata[[1]])[1,1] <- NA
omXplore_pca(vdata, 1)
shiny::runApp(omXplore_pca(vdata, 1))

## End(Not run)
```



```

data(vdata)
obj <- vdata[[1]]
res.pca <- wrapper_pca(qdata=SummarizedExperiment::assay(obj), group=get_group(obj))
plotPCA_Eigen(res.pca)
plotPCA_Var(res.pca)
plotPCA_Eigen_hc(res.pca)
plotPCA_Ind(res.pca)

```

---

ds-view

*Bar plot of missing values per lines using highcharter.*


---

### Description

This method plots a bar plot which represents the distribution of the number of missing values (NA) per lines (i.e. proteins).

### Usage

```

view_dataset_ui(id)

view_dataset_server(
  id,
  obj = reactive({
    NULL
  }),
  addons = list(),
  useModal = TRUE,
  verbose = FALSE
)

view_dataset(obj = NULL, addons = NULL, useModal = TRUE)

```

### Arguments

id	A character(1) for the 'id' of the shiny module. It must be the same as for the '*_ui' function.
obj	An instance of the class MultiAssayExperiment.
addons	A list to configure the other shiny apps to integrate. Each item correspond to one package: <ul style="list-style-type: none"> <li>the name of the slot is the name of the package</li> <li>the content of the slot is a vector composed of the generic name of the shiny app. Each of the apps listed here must be an exported app of the package. For example, given the value addons = list(testPkg = c('foo', 'foo2')). That means that the package called "testPkg" must provide the four functions: foo1_ui(), foo1_server() and foo2_ui(), foo2_server()</li> </ul>

useModal	A boolean(1) that indicates whether to open plot modules in a modal window or not. Default is TRUE.
verbose	A boolean for verbose mode. Default is FALSE.

### Details

- distribution of the missing values per line,
- a bar plot which represents the distribution of the number of missing values (NA) per lines (i.e. proteins) and per conditions,
- Histogram of missing values.
- Variance : Builds a densityplot of the CV of entities in numeric matrix. The CV is calculated for each condition present in the dataset (see the slot 'Condition' in the colData() DataFrame)
- Heatmap:

The function [heatmapD\(\)](#)

The function `[]` is inspired from the function 'heatmap.2' that displays a numeric matrix. For more information, please refer to the help of the heatmap.2 function.

### Value

NA

NA

NA

A shiny application which wraps the functions `view_dataset_ui()` and the `view_dataset_server()`

### Missing values

#' - distribution of the missing values per line,

- a bar plot which represents the distribution of the number of missing values (NA) per lines (ie proteins) and per conditions,
- Histogram of missing values.

### Author(s)

Samuel Wieczorek, Enora Fremy

### Examples

```
## Not run:
data(vdata)
addons <- list(omXplore = c("extFoo1", "extFoo2"))
runApp(view_dataset(vdata, addons, useModal = FALSE))

shiny::runApp(view_dataset(vdata))

## End(Not run)
```

```
if (interactive()) {  
  data(vdata)  
  view_dataset(vdata)  
}
```

---

external\_app

*External module example*

---

### Description

Example for an external shiny module, well structured to be run within a workflow for Mage11anNTK

### Usage

```
extFoo1_ui(id)  
  
extFoo1_server(  
  id,  
  obj = reactive({  
    NULL  
  }),  
  i = reactive({  
    NULL  
  })  
)  
  
extFoo1(obj, i)  
  
extFoo2_ui(id)  
  
extFoo2_server(  
  id,  
  obj = reactive({  
    NULL  
  }),  
  i = reactive({  
    NULL  
  })  
)  
  
extFoo2(obj, i)
```

**Arguments**

id	A character(1) which is the id of the shiny module.
obj	An object of instance MultiAssayExperiment
i	An integer which is the index of the assay in the param obj

**Value**

NA  
 NA  
 NA  
 A shiny app  
 NA  
 NA  
 A shiny app

**Examples**

```
## Not run:
data(vdata)
app1 <- extFoo1(vdata, 1)
app2 <- extFoo2(vdata, 1)
shiny::runApp(app1)
shiny::runApp(app2)

## End(Not run)
```

---

FormatDataForDT

*Constructs a dataset suitable to use with the module format\_DT.*


---

**Description**

This function builds the skeleton of a dataset which can be used by the module formatDT. It creates additional columns to be used to style the table. to colors cells.

**Usage**

```
FormatDataForDT(se, digits = 2)
```

**Arguments**

se	An instance of the class SummarizedExperiment
digits	An 'integer(1)' to specify the number of digits to display in the tables for numerical values. Default is 2.

**Value**

A data.frame

---

format_DT	<i>formatDT_ui and formatDT_server</i>
-----------	--

---

**Description**

A shiny Module.

See DT package homepage for more details about styling tables. If no style is precised, this module show the raw data. If any style is given, then the dataset must be well configured (I.e. it must contain the correct columns )

**Usage**

```
formatDT_ui(id)

formatDT_server(
  id,
  data = reactive({
    NULL
  }),
  data_nostyle = reactive({
    NULL
  }),
  withDLBtns = FALSE,
  showRownames = FALSE,
  dt_style = reactive({
    NULL
  }),
  filename = "Prostar_export",
  selection = "single"
)

formatDT(data)
```

**Arguments**

id	shiny id
data	A data.frame
data_nostyle	A data.frame() to be bind to the main data with no custom style
withDLBtns	A boolean to indicate whether to display download buttons or not.
showRownames	A boolean to indicate whether to show rownames.
dt_style	A list composed of:

- data : a data.frame
- colors : a named vector

filename      A character(1) which is the default filename for download.

selection     A character(1) which indicates the type of selection. Default is 'single'.

**Value**

NA

NA

NA

NA

**Examples**

```
## Not run:
  data(vdata)
  formatDT(vdata)

## End(Not run)
```

---

GetPkgVersion	<i>Package version</i>
---------------	------------------------

---

**Description**

Gets the version number of a package

**Usage**

```
GetPkgVersion(pkg)
```

**Arguments**

pkg            The name of the package

**Value**

A character(1) with the name of the package and its version number.

**Examples**

```
GetPkgVersion('omXplore')
```

---

globals	<i>Global variables</i>
---------	-------------------------

---

**Description**

Defines the global variables for the package omXplore

**Usage**

```
globals()
```

**Value**

A list

**Examples**

```
globals()
```

---

intensity-plots	<i>Displays different intensity plots.</i>
-----------------	--

---

**Description**

Displays different intensity plots.

**Usage**

```
omXplore_intensity_ui(id)

omXplore_intensity_server(
  id,
  obj = reactive({
    NULL
  }),
  i = reactive({
    1
  }),
  track.indices = reactive({
    NULL
  }),
  remoteReset = reactive({
    NULL
  }),
  is.enabled = reactive({
```

```

      TRUE
    })
  )

  omXplore_intensity(obj, i = NULL, withTracking = FALSE)

  boxPlot(obj, conds, legend = NULL, pal = NULL, subset = NULL)

  violinPlot(data, conds, subset = NULL, pal.name = "Set1")

```

### Arguments

<code>id</code>	A character(1) which is the id of the shiny module.
<code>obj</code>	A instance of the class <code>MultiAssayExperiment</code>
<code>i</code>	An integer which is the index of the assay in the param <code>obj</code>
<code>track.indices</code>	A vector of integers which are the indices of lines to track.
<code>remoteReset</code>	An integer to activate the reset functions
<code>is.enabled</code>	A boolean that indicates whether the widgets should be enabled or disabled. Default is TRUE
<code>withTracking</code>	A boolean(1) indicating whether the tracking option is activated or not.
<code>conds</code>	A vector indicating the name of each sample.
<code>legend</code>	A vector of the conditions (one condition per sample).
<code>pal</code>	A basis palette for the boxes which length must be equal to the number of unique conditions in the dataset.
<code>subset</code>	A integer() vector of index indicating the indices of rows in the dataset to highlight
<code>data</code>	A data.frame() of quantitative data
<code>pal.name</code>	A character(1) which is the name of the palette from the package <a href="#">RColorBrewer</a> from which the colors are taken. Default value is 'Set1'.

### Value

NA  
 NA  
 A shiny app  
 A boxplot  
 A violin plot

### Author(s)

Samuel Wieczorek, Anais Courtier, Enora Fremy



**Examples**

```
## Not run:
  data(vdata)
  shiny::runApp(omXplore_intensity(vdata, 1))

data(sub_R25)
conds <- legend <- SummarizedExperiment::colData(sub_R25)$group
pal <- ExtendPalette(length(unique(conds)))
boxPlot(sub_R25[[1]], conds, legend, pal, seq_len(10))

shiny::runApp(omXplore_intensity(sub_R25, 1, withTracking = TRUE))

## End(Not run)
```

---

is.listOf                      *Checks the class of a list's slots*

---

**Description**

Checks if all slots of the given list are of the same class.

**Usage**

```
is.listOf(object, obj.class = NULL)
```

**Arguments**

object	A list
obj.class	The name of the class to search in items of the list.

**Value**

A character(1) with the name of the package or NULL

**Examples**

```
ll <- as.list(LETTERS[1:3])
is.listOf(ll, "data.frame")
is.listOf(ll, "character")
```

---

omXplore-modules      *Shiny modules used by omXplore*

---

## Description

These functions are relative to external modules that can be added into omXplore UI:

- `listShinyApps()`: Show the shiny modules recognized by omXplore and ready to be integrated in the UI of the function `view_dataset()`
- `listPlotModules()`: Show the shiny modules function names (only prefixes) recognized by omXplore and ready to use in the UI.
- `addModules()`: Add external shiny module(s) to the R global environment in such a way (specific prefix renaming of the functions) that it can be discovered by the function `view_dataset()` of the package omXplore during its launch.

## Usage

```
addModules(addons = list())
```

```
listShinyApps(location = "both")
```

```
listPlotModules(location = "both")
```

## Arguments

addons	A list in which each item: <ul style="list-style-type: none"> <li>• is named by the name of a package containing the modules to add,</li> <li>• contains the name of the shiny modules to integrate (without <code>'_ui'</code> nor <code>'_server'</code> suffixes)</li> </ul>
location	A character ( $\emptyset$ ) to indicate which modules to list. Available values are: <code>'builtin'</code> , <code>'external'</code> and <code>'both'</code> (default).

## Value

NA

A vector

A vector

## Examples

```
listShinyApps()
listPlotModules()
```

```
#####
# Integration of a module in the package 'mypackage'
#####
```

```
## Not run:  
addons <- list(omXplore = c("extFoo1", "extFoo2"))  
addModules(addons)  
  
## End(Not run)
```

---

omXplore_heatmap	<i>Displays a correlation matrix of the quantitative data of a numeric matrix.</i>
------------------	--

---

### Description

This function is a wrapper to `heatmap.2()` that displays assay data in an instance of `SummarizedExperiment`. For more details, see `heatmap.2()`.

### Usage

```
omXplore_heatmap_ui(id)  
  
omXplore_heatmap_server(  
  id,  
  obj = reactive({  
    NULL  
  }),  
  i = reactive({  
    NULL  
  })  
)  
  
omXplore_heatmap(obj, i)  
  
heatmapD(  
  qdata,  
  conds,  
  distance = "euclidean",  
  cluster = "complete",  
  dendro = FALSE  
)  
  
mv.heatmap(  
  x,  
  col = grDevices::heat.colors(100),  
  srtCol = NULL,  
  labCol = NULL,  
  labRow = NULL,  
  key = TRUE,  
  key.title = NULL,
```

```

    main = NULL,
    ylab = NULL
  )

heatmapForMissingValues(
  x,
  col = NULL,
  srtCol = NULL,
  labCol = NULL,
  labRow = NULL,
  key = TRUE,
  key.title = NULL,
  main = NULL,
  ylab = NULL
)

```

### Arguments

id	A character(1) which is the id of the shiny module.
obj	An instance of a class MultiAssayExperiment.
i	An integer which is the index of the assay in the param obj
qdata	A data.frame() of quantitative data.
conds	A vector indicating the name of each sample.
distance	The distance used by the clustering algorithm to compute the dendrogram.
cluster	the clustering algorithm used to build the dendrogram.
dendro	A boolean to indicate if the dendrogram has to be displayed
x	A matrix or array containing the quantitative data.
col	Colors used for the image. Defaults to heat colors (heat.colors).
srtCol	Angle of column conds, in degrees from horizontal
labCol	Character vectors with column conds to use.
labRow	Character vectors with row conds to use.
key	Logical indicating whether a color-key should be shown.
key.title	Main title of the color key. If set to NA no title will be plotted.
main	Main title; default to none.
ylab	y-axis title; default to none.

### Value

NA  
 NA  
 A shiny app  
 A heatmap  
 A heatmap  
 A heatmap

**Author(s)**

Florence Combes, Samuel Wiczorek, Enora Fremy

**Examples**

```
## Not run:
  data(vdata)
  omXplore_heatmap(vdata, 1)

## End(Not run)
```

---

omXplore\_tabExplorer *Explore MultiAssayExperiment objects.*

---

**Description**

Explore MultiAssayExperiment objects.

**Usage**

```
omXplore_tabExplorer_ui(id)

omXplore_tabExplorer_server(
  id,
  obj = reactive({
    NULL
  }),
  i = reactive({
    NULL
  }),
  digits = reactive({
    3
  })
)

omXplore_tabExplorer(obj, i)
```

**Arguments**

id	A character(1) which is the id of the shiny module.
obj	An instance of the class MultiAssayExperiment
i	An integer which is the index of the assay in the param obj
digits	An integer for the number of digits shown in the table

**Value**

NA  
 NA  
 NA  
 A shiny app

**Examples**

```
## Not run:
  data(vdata)
  shiny::runApp(omXplore_tabExplorer(vdata, 1))

## End(Not run)
```

---

 palette

*Palette for samples.*


---

**Description**

Builds a vector of #conditions colors for a set of samples. One color is given for a given condition. This function extends a base palette from the package [RColorBrewer](#) to 'n' colors. The colors in the returned palette are always in the same order

**Usage**

```
SampleColors(conds, pal.name = "Set1")

ExtendPalette(n, pal.name = "Set1")

GetColorsForConditions(conds, pal = NULL)
```

**Arguments**

conds	A character() of conditions. The length of the vector is the number of samples in the dataset.
pal.name	A character(1) which is the name of the palette from the package <a href="#">RColorBrewer</a> from which the colors are taken. Default value is 'Set1'.
n	The number of desired colors in the palette
pal	A vector of HEX color code that form the basis palette from which to build the complete color vector for the conditions.

**Value**

A vector  
 A vector  
 A vector

**Examples**

```

#-----
# Builds a palette for a dataset with 3 conditions
# of 3 samples each.
#-----

conds <- c(rep("A", 3), rep("B", 3), rep("C", 3))
SampleColors(conds)
SampleColors(conds, "Dark2")

#-----
# Extend the default palette to 12 colors
#-----

ExtendPalette(12)

data(vdata)
conds <- get_group(vdata)
GetColorsForConditions(conds, ExtendPalette(2))

```

---

pep\_prot\_CC

*Display a CC*


---

**Description**

Display a CC  
 Connected Components infos

**Usage**

```

buildGraph(cc = NULL, meta = NULL)

display.CC.visNet(g = NULL, layout = "layout_with_fr")

plotCCJitter(df, clickFunction = NULL)

GetCCInfos(cc)

```

**Arguments**

cc	A list of connected component
meta	A data.frame()
g	An instance of a graph

layout	A character(1) which is the layout used in visNetwork. Default value is 'layout_with_fr'
df	A data.frame()
clickFunction	A JS function to determine the behaviour of a click

**Value**

A list

A plot

A plot

A list of three items:

- One\_One: the number of cc composed of one protein and one peptide
- One\_Multi: the number of cc composed of one protein and several peptides
- Multi\_Multi: the number of cc composed of several proteins and several (shared) peptides.

**Author(s)**

Thomas Burger, Samuel Wiczorek

**Examples**

```
data(sub_R25)
se <- sub_R25[[1]]
g <- buildGraph(get_cc(se)[[1]])
display.CC.visNet(g)
```

```
data(sub_R25)
GetCCInfos(get_cc(sub_R25[[1]]))
```

---

pkgs.require

*Loads packages*

---

**Description**

Checks if a package is available to load it

**Usage**

```
pkgs.require(ll.deps)
```

**Arguments**

ll.deps            A character() vector which contains packages names



**Value**

NA

**Author(s)**

Samuel Wieczorek

**Examples**

```
pkgs.require('omXplore')
```

---

plot-variance	<i>Variance plot</i>
---------------	----------------------

---

**Description**

A shiny module which plots the variance of samples

**Usage**

```
omXplore_variance_ui(id)
```

```
omXplore_variance_server(id, obj, i, pal.name = NULL)
```

```
CVDist(obj, conds, pal.name = NULL)
```

```
omXplore_variance(obj, i)
```

**Arguments**

id	A character(1) which is the id of the shiny module.
obj	An matrix
i	An integer which is the index of the assay in the param obj
pal.name	A character(1) which is the name of the palette from the package <a href="#">RColorBrewer</a> from which the colors are taken. Default value is 'Set1'.
conds	A vector indicating the name of each sample.

**Value**

NA

NA

A plot

A shiny app

**Examples**

```
if (interactive()) {
  data(vdata)
  omXplore_variance(vdata, 1)
}
```

---

plots\_tracking      *plots\_tracking\_ui and plots\_tracking\_server*

---

**Description**

This shiny module provides a tool to select

**Usage**

```
plots_tracking_ui(id)

plots_tracking_server(
  id,
  obj = reactive({
    NULL
  }),
  remoteReset = reactive({
    NULL
  }),
  is.enabled = reactive({
    TRUE
  })
)

plots_tracking(obj)
```

**Arguments**

id	shiny id
obj	An instance of the class MultiAssayExperiment
remoteReset	A boolean(1) which indicates whether to show the 'Reset' button or not.
is.enabled	xxx

**Value**

NA  
 A list (same structure as the parameter params)  
 A shiny app

**Examples**

```
## Not run:  
  data(vdata)  
  shiny::runApp(plots_tracking(vdata[[1]]))  
  
## End(Not run)
```

---

Prostar-1x-compatible *Convert datasets exported by the package Prostar*

---

**Description**

Convert datasets exported by the package Prostar

**Usage**

```
SE_Compatibility_with_Prostar_1.x(obj, se)  
MAE_Compatibility_with_Prostar_1x(obj, mae)
```

**Arguments**

obj	An instance of the class MSnSet
se	An instance of the class SummarizedExperiment
mae	An instance of the class MultiAssayExperiment

**Value**

An enriched instance of the class SummarizedExperiment  
An enriched instance of the class MultiAssayExperiment

**Examples**

```
data(sub_R25)
```

---

q\_metadata

*Quantitative metadata vocabulary for entities*


---

## Description

This function gives the vocabulary used for the quantitative metadata of each entity in each condition.

## Usage

```
metacell.def(level)
```

```
Parent(level, node = NULL)
```

```
Children(level, parent = NULL)
```

```
GetMetacellTags(metacells = NULL, level = NULL, onlyPresent = FALSE)
```

## Arguments

level	A string corresponding to the type of object
node	The name of the node for which one wants its parent # @examples Parent('protein', 'Missing') Parent('protein', 'Missing POV') Parent('protein', c('Missing POV', 'Missing MEC')) Parent('protein', c('Missing', 'Missing POV', 'Missing MEC'))
parent	The name of the parent node
metacells	A data.frame() representing the cell metadata
onlyPresent	A boolean(1)

## Value

A data.frame containing the different tags and corresponding colors for the level given in parameter

A list

A vector

A vector

A vector

## Glossary

Peptide-level vocabulary

├─ 'Any' ||||─ 1.0 'Quantified' |||||─ 1.1 "Quant. by direct id" (color 4, white) |||||─ 1.2 "Quant. by recovery" (color 3, lightgrey) ||||─ 2.0 "Missing" (no color) |||||─ 2.1 "Missing POV" (color 1) |||||─ 2.2 'Missing MEC' (color 2) ||||─ 3.0 'Imputed' |||||─ 3.1 'Imputed POV' (color 1) ||||─ 3.2 'Imputed MEC' (color 2)

Protein-level vocabulary: |– 'Any' | | | |– 1.0 'Quantified' | | | | | |– 1.1 "Quant. by direct id" (color 4, white) | | | | | |– 1.2 "Quant. by recovery" (color 3, lightgrey) | | | |– 2.0 "Missing" | | | | | |– 2.1 "Missing POV" (color 1) | | | | | |– 2.2 'Missing MEC' (color 2) | | | |– 3.0 'Imputed' | | | | | |– 3.1 'Imputed POV' (color 1) | | | | | |– 3.2 'Imputed MEC' (color 2) | | | |– 4.0 'Combined tags' (color 3bis, lightgrey)

### Conversion to the glossary

A generic conversion  
 Conversion for Proline datasets  
 Conversion from Maxquant datasets

### Author(s)

Thomas Burger, Samuel Wieczorek  
 Samuel Wieczorek

### Examples

```
metacell.def("protein")
metacell.def("peptide")

Children("protein", "Missing")
Children("protein", "Missing POV")
Children("protein", c("Missing POV", "Missing MEC"))
Children("protein", c("Missing", "Missing POV", "Missing MEC"))
data(vdata)
metacells <- get_metacell(vdata[[1]])
level <- get_type(vdata[[1]])
GetMetacellTags(metacells, level)
```

---

sub\_R25

*Feature example data*


---

### Description

sub\_R25 is a protein subset of the dataset 'Exp1\_R25\_pept' in the package DAPARdata.

### Format

An instance of the class `MultiAssayExperiment`

### Value

An enriched instance of the class `MultiAssayExperiment`

**Source**

sub\_R25 was built from the source code available in `inst/scripts/build_datasets.R`  
The DAPARdata package: <https://github.com/prostarproteomics/DAPARdata>

---

vdata

*Feature example data*

---

**Description**

vdata is a small object for testing and demonstration.

**Format**

An instance of the class `MultiAssayExperiment`

**Value**

An enriched instance of the class `MultiAssayExperiment`

**Source**

vdata was built from the source code available in `inst/scripts/build_datasets.R`

# Index

- \* **datasets**
  - vdata, 38
- \* **data**
  - vdata, 38
- \* **internal**
  - ds-cc, 13
  - plots\_tracking, 34
- accessors, 3
- addModules (omXplore-modules), 26
- boxPlot (intensity-plots), 23
- Build\_enriched\_qdata, 5
- Build\_X\_CC (converters), 7
- BuildColorStyles, 5
- buildGraph (pep\_prot\_CC), 31
- Check\_List\_consistency (converters), 7
- Check\_MSDataSet\_Consistency (converters), 7
- Check\_se\_Consistency (converters), 7
- Children (q\_metadata), 36
- color-legend, 6
- colorLegend (color-legend), 6
- colorLegend\_server (color-legend), 6
- colorLegend\_ui (color-legend), 6
- Compute\_CC (converters), 7
- convert\_to\_mae (converters), 7
- converters, 7
- corrMatrix (corrmatrix), 9
- corrmatrix, 9
- custom\_metacell\_colors (color-legend), 6
- customChart, 10
- customExportMenu, 11
- CVDist (plot-variance), 33
- density-plot, 12
- densityPlot (density-plot), 12
- df\_to\_mae (converters), 7
- df\_to\_se (converters), 7
- display.CC.visNet (pep\_prot\_CC), 31
- ds-cc, 13
- ds-pca, 14
- ds-view, 17
- ExtendPalette (palette), 30
- external\_app, 19
- extFoo1 (external\_app), 19
- extFoo1\_server (external\_app), 19
- extFoo1\_ui (external\_app), 19
- extFoo2 (external\_app), 19
- extFoo2\_server (external\_app), 19
- extFoo2\_ui (external\_app), 19
- format\_DT, 21
- FormatDataForDT, 20
- formatDT (format\_DT), 21
- formatDT\_server (format\_DT), 21
- formatDT\_ui (format\_DT), 21
- get\_adjacencyMatrix (accessors), 3
- get\_adjacencyMatrix, SummarizedExperiment-method (accessors), 3
- get\_cc (accessors), 3
- get\_cc, SummarizedExperiment-method (accessors), 3
- get\_colID (accessors), 3
- get\_colID, SummarizedExperiment-method (accessors), 3
- get\_design (accessors), 3
- get\_design, MultiAssayExperiment-method (accessors), 3
- get\_group (accessors), 3
- get\_group, MultiAssayExperiment-method (accessors), 3
- get\_metacell (accessors), 3
- get\_metacell, SummarizedExperiment-method (accessors), 3
- get\_parentProtId (accessors), 3
- get\_parentProtId, SummarizedExperiment-method (accessors), 3

- get\_pkg\_version (accessors), 3
- get\_pkg\_version, SummarizedExperiment-method (accessors), 3
- get\_type (accessors), 3
- get\_type, SummarizedExperiment-method (accessors), 3
- GetCCInfos (pep\_prot\_CC), 31
- GetColorsForConditions (palette), 30
- GetMetacellTags (q\_metadata), 36
- GetPkgVersion, 22
- globals, 23
  
- heatmapD (omXplore\_heatmap), 27
- heatmapD(), 18
- heatmapForMissingValues (omXplore\_heatmap), 27
  
- intensity-plots, 23
- is.listOf, 25
  
- list\_to\_se (converters), 7
- listOfdf\_to\_mae (converters), 7
- listOfLists\_to\_mae (converters), 7
- listOfmatrix\_to\_mae (converters), 7
- listOfMSnSet\_to\_mae (converters), 7
- listOfSE\_to\_mae (converters), 7
- listPlotModules (omXplore-modules), 26
- listShinyApps (omXplore-modules), 26
  
- MAE\_Compatibility\_with\_Prostar\_1x (Prostar-1x-compatible), 35
- MAE\_to\_mae (converters), 7
- matrix\_to\_mae (converters), 7
- matrix\_to\_se (converters), 7
- metacell.def (q\_metadata), 36
- MSnSet\_to\_mae (converters), 7
- MSnSet\_to\_se (converters), 7
- mv.heatmap (omXplore\_heatmap), 27
- my\_PCA (ds-pca), 14
  
- omXplore-modules, 26
- omXplore\_cc (ds-cc), 13
- omXplore\_cc\_server (ds-cc), 13
- omXplore\_cc\_ui (ds-cc), 13
- omXplore\_corrmatrix (corrmatrix), 9
- omXplore\_corrmatrix\_server (corrmatrix), 9
- omXplore\_corrmatrix\_ui (corrmatrix), 9
- omXplore\_density (density-plot), 12
- omXplore\_density\_server (density-plot), 12
- omXplore\_density\_ui (density-plot), 12
- omXplore\_heatmap, 27
- omXplore\_heatmap\_server (omXplore\_heatmap), 27
- omXplore\_heatmap\_ui (omXplore\_heatmap), 27
- omXplore\_intensity (intensity-plots), 23
- omXplore\_intensity\_server (intensity-plots), 23
- omXplore\_intensity\_ui (intensity-plots), 23
- omXplore\_pca (ds-pca), 14
- omXplore\_pca\_server (ds-pca), 14
- omXplore\_pca\_ui (ds-pca), 14
- omXplore\_tabExplorer, 29
- omXplore\_tabExplorer\_server (omXplore\_tabExplorer), 29
- omXplore\_tabExplorer\_ui (omXplore\_tabExplorer), 29
- omXplore\_variance (plot-variance), 33
- omXplore\_variance\_server (plot-variance), 33
- omXplore\_variance\_ui (plot-variance), 33
  
- palette, 30
- Parent (q\_metadata), 36
- pep\_prot\_CC, 31
- pkgs.require, 32
- plot-variance, 33
- plotCCJitter (pep\_prot\_CC), 31
- plotPCA\_Eigen (ds-pca), 14
- plotPCA\_Eigen\_hc (ds-pca), 14
- plotPCA\_Ind (ds-pca), 14
- plotPCA\_Var (ds-pca), 14
- plots\_tracking, 34
- plots\_tracking\_server (plots\_tracking), 34
- plots\_tracking\_ui (plots\_tracking), 34
- Prostar-1x-compatible, 35
  
- q\_metadata, 36
- QFeatures\_to\_mae (converters), 7
  
- RColorBrewer, 12, 24, 30, 33
  
- SampleColors (palette), 30
- SE\_Compatibility\_with\_Prostar\_1.x (Prostar-1x-compatible), 35



SE\_to\_mae (converters), [7](#)  
sub\_R25, [37](#)

vdata, [38](#)

view\_dataset (ds-view), [17](#)

view\_dataset\_server (ds-view), [17](#)

view\_dataset\_ui (ds-view), [17](#)

violinPlot (intensity-plots), [23](#)

wrapper\_pca (ds-pca), [14](#)