Package ‘MACSQuantifyR’

May 19, 2026

Type Package

Title Fast treatment of MACSQuantify FACS data

Date 08/10/2019

Version 1.27.0

Author Raphaël Bonnet [aut, cre], Marielle Nebout [dtc],Giulia Biondani [dtc], Jean-François Peyron[aut,ths], Inserm [fnd]

Maintainer Raphaël Bonnet `<raphael.bonnet@univ-cotedazur.fr>`

Description Automatically process the metadata of MACSQuantify FACS sorter.
It runs multiple modules: i) imports of raw file and graphical selection of duplicates in well plate, ii) computes statistics on data and iii) can compute combination index.

Imports readxl, graphics, tools, utils, grDevices, ggplot2, ggrepel, methods, stats, latticeExtra, lattice, rmarkdown, png, grid, gridExtra, prettydoc, rvest, xml2

biocViews DataImport, Preprocessing, Normalization, FlowCytometry, DataRepresentation, GUI

License Artistic-2.0

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests knitr, testthat, R.utils, spelling

VignetteBuilder knitr

Language en-US

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

git_branch devel

git_last_commit e19a424

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-18

Contents

`barplot_data` *Generate barplots*

Description

Uses the informations stored in the statistic tables of the MACSQuant object to generate barplots for the specified conditions. Plots can also be saved in the form of a png file in the output_MQ directory.

Usage

```
barplot_data(MACSQuant,plt.col=NULL,plt.conditions=NULL, plt.flavour=NULL,
plt.labels = NULL, plt.combo=FALSE,plt.3D.only = NULL,...)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table
`plt.col`	In case of manual run is used to define the bar colors. (must be the same for all conditions)
`plt.conditions`	user defined vector of conditions to plot (example: plt.conditions=c(1:9))
`plt.flavour`	Which barplot to produce, cell count or percentage of fluorochrome (i.e: "counts","percent")
`plt.labels`	xlabels of the plot
`plt.combo`	Does the experiment involves multiple variables (i.e some condition are a drug combination screening)
`plt.3D.only`	Whether to plot 3D barplot alone
`...`	params for lattice cloud namely z and x for parameter screen default for z and x are c(-110,-70) and argument xlab and ylab

Value

returns a barplot of the data

Examples

```
drugs_R_image <- system.file("extdata",
    "drugs.RDS",
    package = "MACSQuantifyR")
MACSQuant <- readRDS(drugs_R_image)
flavour <- "counts"
number_of_conditions <-
    slot(MACSQuant, "param.experiment")$number_of_conditions
plt.col <- c(rep(2, number_of_conditions), 1)
barplot_data(MACSQuant,
    plt.col = plt.col,
    plt.conditions = NULL,
    plt.flavour = flavour,
    plt.labels = NULL,
    plt.combo = FALSE)
```

`combination_index` *compute combination index*

Description

This function allows the user to compute combination index on the drug combinations. This function also generates intermediary plots and tables.

Usage

```
combination_index(MACSQuant, ...)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table
`...`	params for lattice cloud namely z and x for parameter screen default for z and x are c(-110,-70) and argument xlab and ylab

Value

Several plots and combination index

References

Chou, T. C. (2006). Theoretical basis, experimental design, and computerized simulation of synergism and antagonism in drug combination studies. *Pharmacological reviews*, 58(3), 621-681.

Examples

```
filepath <- system.file("extdata", "drugs.Rdata",
    package = "MACSQuantifyR")
load(filepath)
combination_index(MACSQuant)
```

`generate_report` *Word report generation*

Description

Allows the user to generate a report at every steps of the analysis

Also generates a complete report with the graphics

Usage

```
generate_report(MACSQuant)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table

Value

A formatted report

Examples

```
drugs_R_image <- system.file("extdata",
    "drugs.RDS",
    package = "MACSQuantifyR")
MACSQuant <- readRDS(drugs_R_image)
generate_report(MACSQuant)
```

`load_MACSQuant` *Load xls file from maxQuant 96-well-plate device*

Description

Load xls file from maxQuant 96-well-plate device

Usage

```
load_MACSQuant(filepath,sheet_name= NULL,MACSQuant.obj= NULL)
```

Arguments

`filepath`	path of the excel file
`sheet_name`	Name of the sheet to load (optional, first sheet is default)
`MACSQuant.obj`	object of class MACSQuant

Value

An object called MACSQuant of class MACSQuant containing variable my_data that corresponds to the data of the excel file in R

Examples

```
filepath <- system.file("extdata", "SingleDrugs.xlsx",
    package = "MACSQuantifyR")
# load_MACSQuant(filepath)
```

`MACSQuant` *The MACSQuant Class*

Description

MACSQuant object contains all the data and the user parameters

Slots

`my_data` Contains the raw data

`my_data_sorted` Contains the sorted data (according to replicates order)

`my_replicates_sorted` Contains the sorted matrix containing replicates names

`experiment_name` Contains the experiment name given by the user

`statistics` Contains the statistics for each condition

`combination.index.df` data.frame that contains the outputs of the combination_index function

`param.experiment` list that contains the parameters relative to the experiment (i.e, number of replicates, number of conditions...)

`param.output` List that contains the parameters relative to the report generation and outputs (i.e, save intermediary results, generate full report..)

`new_class_MQ` *Create a new MACSQuant object*

Description

The user can create a new custom MACSQuant object This function is also used in internal code for MACSQuant initialization

Usage

```
#initialize empty MACSQuant object
new_class_MQ(my_data=NULL,my_data_sorted=NULL,my_replicates_sorted=NULL,
experiment_name=NULL,statistics=NULL,combination.index.df=NULL,
number_of_replicates=NULL,number_of_conditions=NULL,doses=NULL,
doses.alt=NULL,c_names=NULL, control=NULL,plt.title=NULL,
plt.labels=NULL,plt.col=NULL,save.files=NULL,path=tempdir())
#initialize custom MACSQuant object
new_class_MQ(my_data,my_data_sorted=NULL,my_replicates_sorted=NULL,
experiment_name=NULL,statistics=NULL,combination.index.df=NULL,
number_of_replicates=NULL,number_of_conditions=NULL,
doses=NULL,doses.alt=NULL, c_names=NULL,control=NULL,plt.title=NULL,
plt.labels=NULL,plt.col=NULL, save.files=NULL,path=tempdir())
```

Arguments

`my_data`	Contains the raw data
`my_data_sorted`	Contains the sorted data (according to replicates order)
`my_replicates_sorted`	Contains the sorted matrix containing replicates names
`experiment_name`	User defined experiment named. Used for the report generation
`statistics`	Contains the statistics for each condition
`combination.index.df`	data.frame that contains the outputs of the combination_index function
`number_of_replicates`	For each condition, the number of duplicates (must be the same for all conditions)
`number_of_conditions`	The number of conditions tested (eg: Drug 1 alone, Drug 2 alone)
`doses`	Numeric vector representing doses for each conditions
`doses.alt`	In case of interaction test second dose vector
`c_names`	Vector containing experiment names
`control`	logical: is there a control in this experiment (eg: Staurosporin)
`plt.title`	Title of the experiment to add to the barplot
`plt.labels`	legend labels for the bar plot
`plt.col`	color vector for the graphical representations
`save.files`	Used to save the image in the output folder
`path`	path of the experiment output folder (default '.')

Value

A formatted report file along with intermediates results

Examples

```
# initialize empty MACSQuant object
new_class_MQ()
# initialize custom MACSQuant object
my_data <- data.frame(character(length = 10), character(length = 10),
    numeric(length = 10), numeric(length = 10))
names(my_data) <- c("Full path", "WID", "%-#", "Count/mL")
new_class_MQ(my_data)
```

`on_plate_selection` *Sort experimental design on graphical 96-well-plate*

Description

Loads user's data, prompt a graphical representation of a 96 well plate and let the user select where the duplicates of each condition were placed. Use for reordering excel file. Plate image with selection can also be saved in the form of a png file in the "specified_path/output_MQ" directory.

Usage

```
on_plate_selection(MACSQuant,number_of_replicates,number_of_conditions,
control=FALSE,save.files=FALSE)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table
`number_of_replicates`	For each condition, the number of duplicates (must be the same for all conditions)
`number_of_conditions`	The number of conditions tested (eg: Drug 1 alone, Drug 2 alone)
`control`	Is there a control in this experiment (eg: Staurosporin)
`save.files`	Used to save the image in the output folder

Value

A formatted report file along with intermediates results

Examples

```
print("run manually, requires user input")
# on_plate_selection(MACSQuant,3,5)
# let you select 5 conditions of 3 replicates each
```

`pipeline` *pipeline with report generation*

Description

Loads user's data, prompt a graphical representation of a 96 well plate and let the user select where the duplicates of each condition were placed. Use for reordering excel file. Plate image with selection can also be saved in the form of a png file in the output_MQ directory.

Also generates a complete report with the graphics

Usage

```
pipeline(filepath,sheet_name=NULL,number_of_replicates,number_of_conditions,
control=FALSE,save.files=TRUE,MACSQuant.obj=NULL)
```

Arguments

`filepath`	path of the excel file
`sheet_name`	Name of the sheet to load (optional, first sheet is default)
`number_of_replicates`	For each condition, the number of duplicates (must be the same for all conditions)
`number_of_conditions`	The number of conditions tested (eg: Drug 1 alone, Drug 2 alone)
`control`	Is there a control in this experiment (eg: Staurosporin)
`save.files`	Used to save the image in the output folder
`MACSQuant.obj`	object of class MACSQuant

Value

A formatted report file along with intermediates results

Examples

```
print("run manually, requires user input")
filepath <- system.file("extdata", "SingleDrugs.xlsx",
    package = "MACSQuantifyR")
# pipeline(filepath,3,5)
# let you select 5 conditions of 3 replicates each
```

`rData` *accessor function to access raw data*

Description

This function allows the user to access raw data table

Usage

```
rData(MACSQuant)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table

Value

the raw data table

Examples

```
filepath <- system.file("extdata", "SingleDrugs.xlsx",
    package = "MACSQuantifyR")
#MACSQuant = load_MACSQuant(filepath)

#rData(MACSQuant)
```

`setPath` *sets custom output path*

Description

This function allows the user to set the output directory path

Usage

```
setPath(MACSQuant,path=NULL)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table
`path`	user defined path, default is tempd

Value

object of class MACSQuant with updated path

Examples

```
filepath <- system.file("extdata", "SingleDrugs.xlsx",
    package = "MACSQuantifyR")
#MACSQuant = load_MACSQuant(filepath)
user_path="."
#MACSQuant = setPath(MACSQuant,path=user_path)
```

`sorted` *accessor function to access sorted data*

Description

This function allows the user to access sorted data table

Usage

```
sorted(MACSQuant)
```

Arguments

`MACSQuant`	object of class MACSQuant resulting of the function load_maxQuant(). Contains the original data table

Value

the raw data table

Examples

```
filepath <- system.file("extdata", "SingleDrugs.xlsx",
    package = "MACSQuantifyR")
#MACSQuant = load_MACSQuant(filepath)

#sorted(MACSQuant)
```

Index