

Package ‘MultiDataSet’

December 18, 2025

Type Package

Title Implementation of MultiDataSet and ResultSet

Version 1.39.0

Date 2021-10-04

Maintainer Xavier Escribano Montagut <xavier.escriba@isglobal.org>

Description Implementation of the BRGE's (Bioinformatic Research Group in Epidemiology from Center for Research in Environmental Epidemiology) MultiDataSet and ResultSet. MultiDataSet is designed for integrating multi omics data sets and ResultSet is a container for omics results. This package contains base classes for MEAL and rexposome packages.

License file LICENSE

LazyData TRUE

biocViews Software, DataRepresentation

Depends R (>= 4.1), Biobase

Imports BiocGenerics, GenomicRanges, IRanges, S4Vectors, SummarizedExperiment, methods, utils, ggplot2, ggrepel, qqman, limma

RoxygenNote 7.1.0

Suggests brgedata, minfi, minfiData, knitr, rmarkdown, testthat, omicade4, iClusterPlus, GEOquery, MultiAssayExperiment, BiocStyle, RaggedExperiment

VignetteBuilder knitr

git_url <https://git.bioconductor.org/packages/MultiDataSet>

git_branch devel

git_last_commit 5045884

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-12-17

Author Carlos Ruiz-Arenas [aut, cre],
Carles Hernandez-Ferrer [aut],
Juan R. Gonzalez [aut]

Contents

add_eset	2
add_genexp	3
add_methy	4
add_rnaseq	5
add_rse	5
add_se	7
add_snps	8
add_table	8
chrNumToChar	9
commonIds	10
commonSamples	11
getAssociation	12
lambdaClayton	13
mae2mds	13
mds2mae	14
MultiDataSet	14
MultiDataSet-class	15
opt	20
qq_plot	20
ResultSet	21
rowRangesElements	23
rset	24
volcano_plot	24
w_iclusterplus	25
w_mcia	26
Index	27

add_eset	<i>Method to add an eSet to MultiDataSet.</i>
----------	---

Description

This method adds or overwrites a slot of a MultiDataSet with the content of the given eSet.

Usage

```
add_eset(  
  object,  
  set,  
  dataset.type,  
  dataset.name = NULL,  
  sample.tables = NULL,  
  feature.tables = NULL,  
  warnings = TRUE,  
  overwrite = FALSE,
```

```
      GRanges
    )
```

Arguments

object	MultiDataSet that will be filled.
set	Object derived from eSet to be used to fill the slot.
dataset.type	Character with the type of data of the omic set (e.g. expression, methylation...)
dataset.name	Character with the specific name for this set (NULL by default). It is useful when there are several sets of the same type (e.g. multiple expression assays)
sample.tables	Character with the names of the slots with sample data besides phenoData.
feature.tables	Character with the names of the slots with feature data besides featureData.
warnings	Logical to indicate if warnings will be displayed.
overwrite	Logical to indicate if the set stored in the slot will be overwritten.
GRanges	GenomicRanges to be included in rowRanges slot.

Value

A new MultiDataSet with a slot filled.

See Also

[add_methy](#), [add_genexp](#), [add_rnaseq](#), [add_snps](#)

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(10), 5))
multi <- add_eset(multi, eset, "exampledata", GRanges = NA)
```

add_genexp	<i>Method to add an expression microarray dataset to MultiDataSet.</i>
------------	--

Description

This method adds or overwrites the slot "expression" of an MultiDataSet with the content of the given ExpressionSet. The fData of the ExpressionSet must contain the columns chromosome, start and end.

Usage

```
add_genexp(object, gexpSet, ...)
```

Arguments

object	MultiDataSet that will be filled.
gexpSet	ExpressionSet to be used to fill the slot.
...	Arguments to be passed to add_eset.

Value

A new MultiDataSet with the slot "expression" filled.

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(4), 2))
fData(eset) <- data.frame(chromosome = c("chr1", "chr2"), start = c(12414, 1234321),
  end = c(121241, 124124114), stringsAsFactors = FALSE)
multi <- add_genexp(multi, eset)
```

add_methy	<i>Method to add a slot of methylation to MultiDataSet.</i>
-----------	---

Description

This method adds or overwrites the slot "methylation" of an MultiDataSet with the content of the given MethylationSet or RatioSet. The fData of the input object must contain the columns chromosome and position.

Usage

```
add_methy(object, methySet, ...)
```

Arguments

object	MultiDataSet that will be filled.
methySet	MethylationSet or RatioSet to be used to fill the slot.
...	Further arguments to be passed to add_eset.

Value

A new MultiDataSet with the slot "methylation" filled.

Examples

```
if (require(brgedata)){
  multi <- createMultiDataSet()
  multi <- add_methy(multi, brge_methy[1:100, ])
}
```

add_rnaseq	<i>Method to add an expression RNA seq dataset to MultiDataSet.</i>
------------	---

Description

This method adds or overwrites the slot "rnaseq" of an MultiDataSet with the content of the given ExpressionSet. The fData of the ExpressionSet must contain the columns chromosome, start and end.

Usage

```
add_rnaseq(object, rnaSet, ...)
```

Arguments

object	MultiDataSet that will be filled.
rnaSet	ExpressionSet to be used to fill the slot.
...	Arguments to be passed to add_eset.

Value

A new MultiDataSet with the slot "rnaseq" filled.

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(4), 2))
fData(eset) <- data.frame(chromosome = c("chr1", "chr2"), start = c(12414, 1234321),
  end = c(121241, 12122414), stringsAsFactors = FALSE)
multi <- add_genexp(multi, eset)
```

add_rse	<i>Method to add a RangedSummarizedExperiment to MultiDataSet.</i>
---------	--

Description

This method adds or overwrites a slot of a MultiDataSet with the content of the given RangedSummarizedExperiment.

Usage

```
add_rse(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  sample.tables = NULL,
  feature.tables = NULL,
  warnings = TRUE,
  overwrite = FALSE
)
```

Arguments

<code>object</code>	MultiDataSet that will be filled.
<code>set</code>	Object derived from RangedSummarizedExperiment to be used to fill the slot.
<code>dataset.type</code>	Character with the type of data of the omic set (e.g. expression, methylation...)
<code>dataset.name</code>	Character with the specific name for this set (NULL by default). It is useful when there are several sets of the same type (e.g. multiple expression assays)
<code>sample.tables</code>	Character with the names of the slots with sample data besides colData.
<code>feature.tables</code>	Character with the names of the slots with feature data besides rowData.
<code>warnings</code>	Logical to indicate if warnings will be displayed.
<code>overwrite</code>	Logical to indicate if the set stored in the slot will be overwritten.

Value

A new MultiDataSet with a slot filled.

Examples

```
if (require(GenomicRanges) & require(SummarizedExperiment)){
  multi <- createMultiDataSet()
  counts <- matrix(runif(200 * 6, 1, 1e4), 200)
  rowRanges <- GRanges(rep(c("chr1", "chr2"), c(50, 150)),
    IRanges(floor(runif(200, 1e5, 1e6)), width=100),
    strand=sample(c("+", "-"), 200, TRUE),
    feature_id=sprintf("ID%03d", 1:200))
  colData <- DataFrame(Treatment=rep(c("ChIP", "Input"), 3),
    row.names=LETTERS[1:6], id = LETTERS[1:6])
  names(rowRanges) <- 1:200
  rse <- SummarizedExperiment(assays=SimpleList(counts=counts),
    rowRanges=rowRanges, colData=colData)
  multi <- add_rse(multi, rse, "rseEx")
}
```

add_se

*Method to add a SummarizedExperiment to MultiDataSet.***Description**

This method adds or overwrites a slot of a `MultiDataSet` with the content of the given `SummarizedExperiment`.

Usage

```
add_se(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  sample.tables = NULL,
  feature.tables = NULL,
  warnings = TRUE,
  overwrite = FALSE,
  GRanges
)
```

Arguments

<code>object</code>	<code>MultiDataSet</code> that will be filled.
<code>set</code>	Object derived from <code>SummarizedExperiment</code> to be used to fill the slot.
<code>dataset.type</code>	Character with the type of data of the omic set (e.g. expression, methylation...)
<code>dataset.name</code>	Character with the specific name for this set (NULL by default). It is useful when there are several sets of the same type (e.g. multiple expression assays)
<code>sample.tables</code>	Character with the names of the slots with sample data besides <code>colData</code> .
<code>feature.tables</code>	Character with the names of the slots with feature data besides <code>rowData</code> .
<code>warnings</code>	Logical to indicate if warnings will be displayed.
<code>overwrite</code>	Logical to indicate if the set stored in the slot will be overwritten.
<code>GRanges</code>	GenomicRanges to be included in <code>rowRanges</code> slot.

Value

A new `MultiDataSet` with a slot filled.

Examples

```
multi <- createMultiDataSet()
se <- SummarizedExperiment::SummarizedExperiment(matrix(runif(10), 5))
multi <- add_se(multi, se, "exampledata", GRanges = NA)
```

add_snps	<i>Method to add a slot of SNPs to MultiDataSet.</i>
----------	--

Description

This method adds or overwrites the slot "snps" of an MultiDataSet with the content of the given SnpSet. The fData of the SnpSet must contain the columns chromosome and position.

Usage

```
add_snps(object, snpSet, ...)
```

Arguments

object	MultiDataSet that will be filled.
snpSet	SnpSet to be used to fill the slot.
...	Arguments to be passed to add_eset.

Value

A new MultiDataSet with the slot "snps" filled.

Examples

```
multi <- createMultiDataSet()
geno <- matrix(c(3,1,2,1), ncol = 2)
colnames(geno) <- c("VAL0156", "VAL0372")
rownames(geno) <- c("rs3115860", "SNP1-1628854")
map <- AnnotatedDataFrame(data.frame(chromosome = c("chr1", "chr2"), position = c(12414, 1234321),
  stringsAsFactors = FALSE))
rownames(map) <- rownames(geno)
snpSet <- new("SnpSet", call = geno, featureData = map)
pheno <- data.frame(id = c("VAL0156", "VAL0372"))
rownames(pheno) <- c("VAL0156", "VAL0372")
pData(snpSet) <- pheno
multi <- add_snps(multi, snpSet)
```

add_table	<i>Method to add a matrix to MultiDataSet.</i>
-----------	--

Description

This method adds or overwrites a slot of a MultiDataSet with the content of the given matrix.

Usage

```
add_table(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  warnings = TRUE,
  overwrite = FALSE
)
```

Arguments

object	MultiDataSet that will be filled.
set	matrix used to fill the slot.
dataset.type	Character with the type of data
dataset.name	Character with the specific name for this set (NULL by default). It is useful when there are several sets of the same type.
warnings	Logical to indicate if warnings will be displayed.
overwrite	Logical to indicate if the set stored in the slot will be overwritten.

Value

A new MultiDataSet with a slot filled.

Examples

```
multi <- createMultiDataSet()
mat <- matrix(runif(12), nrow = 3)
colnames(mat) <- paste0("S", 1:4)
rownames(mat) <- paste0("F", 1:3)
multi <- add_table(multi, mat, "exampledata")
```

chrNumToChar

Convert chr numbers to chr strings

Description

Given a vector of number representing the chromosomes, convert them to string (e.g 1 to chr1). 23 is consider chrX, 24 is chrY, 25 is chrXY (probes shared between chromosomes X and Y) and 26 is chrMT.

Usage

```
chrNumToChar(vector)
```

Arguments

vector The vector with the chromosome numbers

Value

A vector with the chromosomes in string format.

Examples

```
chromosomes <- c(1, 3, 4, 23, 15)
stringChrs <- chrNumToChar(chromosomes)
stringChrs
```

commonIds	<i>Get the name of the ids common to all datasets</i>
-----------	---

Description

Get the name of the ids common to all datasets

Usage

```
commonIds(object)
```

Arguments

object MultiDataSet that will be filtered.

Value

Character vector with the common ids.

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(9), ncol = 3))
fData(eset) <- data.frame(chromosome = c("chr1", "chr1", "chr1"),
                          start = c(1, 5, 10), end = c(4, 6, 14),
                          stringsAsFactors = FALSE)
sampleNames(eset) <- c("S1", "S2", "S3")
pData(eset) <- data.frame(id = c("S1", "S2", "S3"))
rownames(pData(eset)) <- c("S1", "S2", "S3")
multi <- add_genexp(multi, eset, dataset.name = "g1")
eset <- new("ExpressionSet", exprs = matrix(runif(8), ncol = 2))
fData(eset) <- data.frame(chromosome = c("chr1", "chr1", "chr1", "chr1"),
                          start = c(1, 14, 25, 104), end = c(11, 16, 28, 115),
                          stringsAsFactors = FALSE)
sampleNames(eset) <- c("S1", "G2")
pData(eset) <- data.frame(id = c("S1", "G2"))
```

```
rownames(pData(eset)) <- c("S1", "G2")

multi <- add_genexp(multi, eset, dataset.name="g2")
commonIds(multi)
```

commonSamples

Method to select samples that are present in all datasets.

Description

This method subsets the datasets to only contain the samples that are in all datasets. All sets will have the samples in the same order, taking into account that there can be duplicates.

Usage

```
commonSamples(object, unify.names = FALSE)
```

Arguments

object	MultiDataSet that will be filtered.
unify.names	Logical indicating if sample names of the sets should be unified.

Details

If unify.names is TRUE, the sample names of the sets will be unified using the id column of phenodata. This option is only possible when there are no duplicated ids.

Value

A new MultiDataSet with only the common samples.

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(9), ncol = 3))
fData(eset) <- data.frame(chromosome = c("chr1", "chr1", "chr1"),
                          start = c(1, 5, 10), end = c(4, 6, 14),
                          stringsAsFactors = FALSE)
sampleNames(eset) <- c("S1", "S2", "S3")
pData(eset) <- data.frame(id = c("S1", "S2", "S3"))
rownames(pData(eset)) <- c("S1", "S2", "S3")
multi <- add_genexp(multi, eset, dataset.name = "g1")
eset <- new("ExpressionSet", exprs = matrix(runif(8), ncol = 2))
fData(eset) <- data.frame(chromosome = c("chr1", "chr1", "chr1", "chr1"),
                          start = c(1, 14, 25, 104), end = c(11, 16, 28, 115),
                          stringsAsFactors = FALSE)
sampleNames(eset) <- c("S1", "G2")
pData(eset) <- data.frame(id = c("S1", "G2"))
rownames(pData(eset)) <- c("S1", "G2")
```

```
multi <- add_genexp(multi, eset, dataset.name="g2")
commonSamples(multi)
```

getAssociation	<i>Method to extrat feature result from a ResultSet</i>
----------------	---

Description

Homologous methods from limma, getAssociation returns a data.frame with the logFC and PValue per featrue for the selcted coef and for given result (rid).

Usage

```
getAssociation(object, rid = 1, coef = 2, contrast = NULL, fNames = NULL, ...)
```

Arguments

object	A ResultSet object.
rid	The name or index of the result to be extracted.
coef	(default 2) Index of the coefficient to be extracted.
contrast	(default 1) When code corresponds to a multicategorical variable, contast selects the comparison.
fNames	(default c("chromosome", "start", "end", "genesymbol")) Corresponds to the columns selected from fData that will be incorporated into the resulting data.frame.
...	Further arguments passed to topTable

Value

A data.frame with the result of the association study, including P-Value and Fold Change.

Examples

```
data(rset)
getAssociation(rset, rid=1, fNames = c("chromosome", "position"))
```

lambdaClayton	<i>Lambda Calculation for a vector of P-Values</i>
---------------	--

Description

Implementation of Clayton's lambda score for a vector of P-Values

Usage

```
lambdaClayton(x, trim = 0.5)
```

Arguments

x	Vector of P-Value
trim	(default 0.5)

Value

A lambda value, indicating the inflation/deflation of the analysis.

Author(s)

Juan R. Gonzalez

Examples

```
lambdaClayton(runif(30))
```

mae2mds	<i>Convert a MultiAssayExperiment to a MultiDataSet</i>
---------	---

Description

This function creates a MultiDataSet using the data of a MultiAssayExperiment.

Usage

```
mae2mds(MAE, warnings = TRUE)
```

Arguments

MAE	a MultiAssayExperiment
warnings	Logical to indicate if warnings will be displayed.

Value

MultiDataSet with the of the incoming MultiAssayExperiment.

mds2mae	<i>Convert a MultiDataSet to a MultiAssayExperiment</i>
---------	---

Description

This function creates a MultiAssayExperiment using the data of a MultiDataSet.

Usage

mds2mae(MDS)

Arguments

MDS a MultiDataSet

Value

MultiAssayExperiment with the of the incoming MultiDataSet.

MultiDataSet	<i>MultiDataSet: Implementation of the BRGE's basic classes</i>
--------------	---

Description

Implementation of the BRGE's (Bioinformatic Research Group in Epidemiology from Center for Research in Environmental Epidemiology) MultiDataSet and MethylationSet. MultiDataSet is designed for integrating multi omics data sets and MethylationSet to contain normalized methylation data. MultiDataSet for integrating multi omics data sets

See Also

[MultiDataSet](#)

MultiDataSet-class *MultiDataSet instances*

Description

The class MultiDataSet is a superior class to store multiple datasets in form of triplets (assayData-phenoData-featureData). The datasets can be eSet or SummarizedExperiment derived or matrices.

Usage

```
## S4 method for signature 'MultiDataSet,eSet'
add_eset(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  sample.tables = "protocolData",
  feature.tables = NULL,
  warnings = TRUE,
  overwrite = FALSE,
  GRanges
)

## S4 method for signature 'MultiDataSet,ExpressionSet'
add_genexp(object, gexpSet, ...)

## S4 method for signature 'MultiDataSet,ExpressionSet'
add_rnaseq(object, rnaSet, ...)

## S4 method for signature 'MultiDataSet,GenomicRatioSet'
add_methy(object, methySet, ...)

## S4 method for signature 'MultiDataSet,RangedSummarizedExperiment'
add_rse(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  sample.tables = NULL,
  feature.tables = "elementMetadata",
  warnings = TRUE,
  overwrite = FALSE
)

## S4 method for signature 'MultiDataSet,SummarizedExperiment'
add_se(
  object,
```

```

    set,
    dataset.type,
    dataset.name = NULL,
    sample.tables = NULL,
    feature.tables = "elementMetadata",
    warnings = TRUE,
    overwrite = FALSE,
    GRanges
)

## S4 method for signature 'MultiDataSet,SnpSet'
add_snps(object, snpSet, ...)

## S4 method for signature 'MultiDataSet,matrix'
add_table(
  object,
  set,
  dataset.type,
  dataset.name = NULL,
  warnings = TRUE,
  overwrite = FALSE
)

## S4 method for signature 'MultiDataSet'
as.list(x)

## S4 method for signature 'MultiDataSet'
commonIds(object)

## S4 method for signature 'MultiDataSet'
commonSamples(object, unify.names = FALSE)

createMultiDataSet()

## S4 method for signature 'MultiDataSet'
dims(x)

## S4 method for signature 'MultiDataSet'
w_iclusterplus(object, commonSamples = TRUE, ...)

## S4 method for signature 'MultiDataSet'
length(x)

## S4 method for signature 'MultiDataSet'
w_mcia(object, ...)

## S4 method for signature 'MultiDataSet'
names(x)

```

```

## S4 method for signature 'MultiDataSet'
ncol(x)

## S4 method for signature 'MultiDataSet'
nrow(x)

## S4 method for signature 'MultiDataSet'
rowRangesElements(object)

## S4 method for signature 'MultiDataSet'
sampleNames(object)

## S4 method for signature 'MultiDataSet'
assayData(object)

## S4 method for signature 'MultiDataSet'
fData(object)

## S4 method for signature 'MultiDataSet'
featureData(object)

## S4 method for signature 'MultiDataSet'
pData(object)

## S4 method for signature 'MultiDataSet'
phenoData(object)

## S4 method for signature 'MultiDataSet'
rowRanges(x)

## S4 method for signature 'MultiDataSet,ANY,ANY'
x[[i]]

## S4 method for signature 'MultiDataSet,ANY,ANY,ANY'
x[i, j, k, ..., drop = FALSE]

## S4 method for signature 'MultiDataSet'
subset(x, feat, phe, warnings = TRUE, keep = TRUE)

```

Arguments

<code>object</code>	<code>MultiDataSet</code>
<code>set</code>	Object derived from <code>eSet</code> to be used to fill the slot.
<code>dataset.type</code>	Character with the type of data of the omic set (e.g. expression, methylation...)
<code>dataset.name</code>	Character with the specific name for this set (NULL by default). It is useful when there
<code>sample.tables</code>	Character with the names of the slots with sample data besides <code>phenoData</code> .

<code>feature.tables</code>	Character with the names of the slots with feature data besides <code>featureData</code> .
<code>warnings</code>	Logical to indicate if warnings will be displayed.
<code>overwrite</code>	Logical to indicate if the set stored in the slot will be overwritten.
<code>GRanges</code>	<code>GenomicRanges</code> to be included in <code>rowRanges</code> slot.
<code>gexpSet</code>	<code>ExpressionSet</code> to be used to fill the slot.
<code>...</code>	Further arguments passed to <code>add_rse</code> or <code>add_se</code>
<code>rnaSet</code>	<code>ExpressionSet</code> to be used to fill the slot.
<code>methySet</code>	<code>GenomicRatioSet</code> to be used to fill the slot.
<code>snpSet</code>	<code>SnpsSet</code> to be used to fill the slot.
<code>x</code>	<code>MultiDataSet</code>
<code>unify.names</code>	Logical indicating if sample names of the sets should be unified.
<code>commonSamples</code>	Logical to indicate if common samples are selected
<code>i</code>	Character corresponding to selected sample names. They should match the <code>id</code> column of <code>phenoData</code> .
<code>j</code>	Character with the name of the selected tables.
<code>k</code>	<code>GenomicRange</code> used to filter the features.
<code>drop</code>	If <code>TRUE</code> , sets with no samples or features will be discarded
<code>feat</code>	Logical expression indicating features to keep
<code>phe</code>	Logical expression indicating the phenotype of the samples to keep
<code>keep</code>	If <code>FALSE</code> , sets where the expression cannot be evaluated will be discarded.

Details

The names of the three lists (`assayData`, `phenoData` and `featureData`) must be the same.

Value

`MultiDataSet`

`MultiDataSet`

Methods (by generic)

- `add_eset`: Method to add an `eSet` to `MultiDataSet`.
- `add_genexp`: Method to add a slot of expression to `MultiDataSet`.
- `add_rnaseq`: Method to add a slot of (RNASeq) expression to `MultiDataSet`.
- `add_methy`: Method to add a slot of methylation to `MultiDataSet` from a `GenomicRatioSet`.
- `add_rse`: Method to add a `RangedSummarizedExperiment` to `MultiDataSet`.
- `add_se`: Method to add a `SummarizedExperiment` to `MultiDataSet`.
- `add_snps`: Method to add a slot of SNPs to `MultiDataSet`.
- `add_table`: Method to add a matrix to `MultiDataSet`.
- `as.list`: Returns a list with the first matrix of each dataset.

- `commonIds`: Get the name of the ids common to all datasets
- `commonSamples`: Get a MultiDataSet only with the samples present in all the tables
- `dims`: Returns the dimensions of the sets
- `w_iclusterplus`: Apply iClusterPlus clustering method to a MultiDataSet object
- `length`: Returns the number of sets into the object.
- `w_mcia`: Apply mcia integration method to a MultiDataSet object
- `names`: Get the names of the slots.
- `ncol`: Get number of samples of each set
- `nrow`: Get number of features of each set
- `rowRangesElements`: Get the name of the datasets that have rowRanges
- `sampleNames`: Get sample names
- `assayData`: Retrieve all assay data blocks.
- `fData`: Retrieve information on features.
- `featureData`: Retrieve information on features.
- `pData`: Retrieve information on experimental phenotypes
- `phenoData`: Retrieve information on experimental phenotypes
- `rowRanges`: Retrieve information on feature ranges.
- `[[]`: Get a set from a slot
- `[[]`: Subset a MultiDataSet
- `subset`: Filter a subset using feature ids or phenotypes

Slots

`assayData` List of assayData elements.

`phenoData` List of AnnotatedDataFrame containing the phenoData of each assayData.

`featureData` List of AnnotatedDataFrame containing the featureData of each assayData.

`rowRanges` List of GenomicRanges containing the rowRanges of each assayData.

`extraData` List of other slots of the original object.

`return_method` List of functions used to create the original object.

See Also

[add_eset](#), [add_rse](#)

Examples

```
createMultiDataSet()
```

opt	<i>Method to get the options sued to create the ResultSet</i>
-----	---

Description

Method that returns a list with the options used to create the ResultSet.

Usage

```
opt(object)
```

Arguments

object A [ResultSet](#) object.

Value

A list with the options used to create the ResultSet.

Examples

```
data(rset)
opt(rset)
```

qq_plot	<i>Function to draw a QQ Plot from a vector of numbers</i>
---------	--

Description

Function to draw a QQ Plot from a vector of numbers

Usage

```
qq_plot(values, show.lambda = TRUE)
```

Arguments

values Numeric vector of P.Values
show.lambda (default: TRUE) If TRUE shows lambda score for the given model.

Value

An object obtained from [ggplot](#).

Examples

```
data(rset)
rst <- getAssociation(rset, rid = 1, fName = NULL)
qq_plot(rst$P.Value)
```

ResultSet

Class ResultSet

Description

Class ResultSet used to encapsulate results from MEAL and omicrexposome.

Usage

```
## S4 method for signature 'ResultSet'
fData(object)

## S4 method for signature 'ResultSet'
getAssociation(object, rid = 1, coef = 2, contrast = NULL, fName = NULL, ...)

## S4 method for signature 'ResultSet'
length(x)

## S4 method for signature 'ResultSet'
names(x)

## S4 method for signature 'ResultSet'
opt(object)

## S4 method for signature 'ResultSet,ANY'
plot(
  x,
  y,
  rid = 1,
  coef = 2,
  contrast = NULL,
  type,
  tFC = 2,
  tPV = -log10(0.001),
  show.labels = TRUE,
  show.effect = FALSE,
  show.lambda = TRUE,
  fName = c("chromosome", "start"),
  subset,
  highlight,
  ...
)

## S4 method for signature 'ResultSet'
varLabels(object)

create_resultset(fOrigin, lResults, fData, lOptions = list())
```

Arguments

object	A <code>ResultSet</code> object.
rid	Name or index of the internal result to be used
coef	Coefficient to be returned, usually 2
contrast	Numeric matrix with the contrasts used to perform the analyses
fNames	Character vector with the names of the <code>fData</code> columns that will be added to the results <code>data.frame</code> .
...	Further arguments passed to topTable
x	A <code>ResultSet</code> object.
y	-
type	Type of plot to be drawn
tFC	Threshold for log FC of effect
tPV	Threshold for P-Value
show.labels	(default TRUE) If set to TRUE, features are labelled.
show.effect	(default: TRUE). Used in volcano plot. If TRUE, effect is shown as FC instead of logFC.
show.lambda	(default: TRUE) If TRUE shows lambda score for the given model.
subset	<code>GenomicRanges</code> used to zoom a region in Manhattan plot
highlight	<code>GenomicRanges</code> used to highlight a region in Manhattan plot
fOrigin	Character with the function used to run the analysis.
lResults	List with the results
fData	List with the feature data.
lOptions	List with additional options

Value

An object of class `ResultSet`

Methods (by generic)

- `fData`: Returns `data.frame` with feature's data.
- `getAssociation`: Getter to obtain the raw `data.frame` from association and integration analysis.
- `length`: Returns the amount of analyses stored in the `ResultSet`.
- `names`: Returns the names of the omics data used to create the `ResultSet`.
- `opt`: Returns a list with the options used to create the `ResultSet`
- `plot`: Allows to plot a series of plots (QQ plot, Manhattan plot and Volcano plot) depending on the results stored in the `ResultSet`.
- `varLabels`: Returns the names of the variables of the models used in a `ResultSet`.

Slots

`fun_origin` Character containing the function that creates the object.

`results` List containing the results of the association/integration.

`fData` List containing the feature-data of the original objects.

`options` list of options used to create the `ResultSet`.

Examples

```
create_resultset("hello", list(), list(), list())
```

rowRangesElements	<i>Get the name of the datasets that have rowRanges</i>
-------------------	---

Description

Get the name of the datasets that have rowRanges

Usage

```
rowRangesElements(object)
```

Arguments

`object` `MultiDataSet`

Value

Character vector with the slots that have rowRanges.

Examples

```
multi <- createMultiDataSet()
eset <- new("ExpressionSet", exprs = matrix(runif(10), 5))
eset2 <- new("ExpressionSet", exprs = matrix(runif(8), ncol = 2))
fData(eset2) <- data.frame(chromosome = c("chr1", "chr1", "chr1", "chr1"),
                           start = c(1, 14, 25, 104), end = c(11, 16, 28, 115),
                           stringsAsFactors = FALSE)
multi <- add_eset(multi, eset, "exampledata", GRanges = NA)
multi <- add_genexp(multi, eset2)
rowRangesElements(multi)
```

rset

*Example ResultSet***Description**

Example `ResultSet` used in the functions examples and in the tests. The script used to generate it can be found in `inst/scripts`.

Usage

rset

Format`ResultSet`

volcano_plot

*Function to draw a Volcano Plot***Description**

Function that takes two numeric vectors (P-Value and fold change) and draws a volcano plot using `ggplot2`

Usage

```
volcano_plot(
  pval,
  fc,
  names,
  size = 2,
  tFC = 2,
  tPV = -log10(0.001),
  show.labels = TRUE,
  show.effect = FALSE
)
```

Arguments

pval	numeric vector of P.Values
fc	numeric vector of fold change
names	character vector with the feature's names.
size	(default 2) Size of the labels in case they are placed.
tFC	(default 2) fold change threshold. It can be set to <code>NULL</code> to not filter.

tPV	(default $-\log_{10}(0.001)$) P-Value threshold. It can be set to NULL to not filter.
show.labels	(default TRUE) If set to TRUE, features are labelled.
show.effect	(default FALSE) If set to TRUE, the X-axis will should $2^{\log FC}$ instead to the default logFC.

Value

A ggplot object

Examples

```
data(rset)
w1 <- getAssociation(rset, rid = 1, fName = NULL)
volcano_plot(w1$P.Value, w1$logFC, rownames(w1))
```

w_iclusterplus	<i>Apply iClusterPlus clustering method to a MultiDataSet object</i>
----------------	--

Description

Method [iClusterPlus](#) is applied on a [MultiDataSet](#) object after getting the common samples along all the contained datasets.

Usage

```
w_iclusterplus(object, commonSamples = TRUE, ...)
```

Arguments

object	MultiDataSet
commonSamples	Logical to indicate if common samples are selected
...	Arguments passed to function iClusterPlus

Value

A list of results from [iClusterPlus](#)

Note

Argument type for [iClusterPlus](#) is filled within the method.

`w_mcia`*Apply mcia integration method to a MultiDataSet object*

Description

Method `mcia` is applied on a `MultiDataSet` object after getting the common samples along all the contained datasets.

Usage

```
w_mcia(object, ...)
```

Arguments

<code>object</code>	<code>MultiDataSet</code>
<code>...</code>	Arguments passed to function <code>mcia</code>

Value

A list of results from `mcia`

Index

- * **datasets**
 - rset, [24](#)
 - [(MultiDataSet-class), [15](#)
 - [, MultiDataSet, ANY, ANY, ANY-method (MultiDataSet-class), [15](#)
 - [[, MultiDataSet, ANY, ANY-method (MultiDataSet-class), [15](#)
- add_eset, [2, 19](#)
- add_eset, MultiDataSet, eSet-method (MultiDataSet-class), [15](#)
- add_genexp, [3, 3](#)
- add_genexp, MultiDataSet, ExpressionSet-method (MultiDataSet-class), [15](#)
- add_methy, [3, 4](#)
- add_methy, MultiDataSet, GenomicRatioSet-method (MultiDataSet-class), [15](#)
- add_rnaseq, [3, 5](#)
- add_rnaseq, MultiDataSet, ExpressionSet-method (MultiDataSet-class), [15](#)
- add_rse, [5, 19](#)
- add_rse, MultiDataSet, RangedSummarizedExperiment-method (MultiDataSet-class), [15](#)
- add_se, [7](#)
- add_se, MultiDataSet, SummarizedExperiment-method (MultiDataSet-class), [15](#)
- add_snps, [3, 8](#)
- add_snps, MultiDataSet, SnpSet-method (MultiDataSet-class), [15](#)
- add_table, [8](#)
- add_table, MultiDataSet, matrix-method (MultiDataSet-class), [15](#)
- as.list (MultiDataSet-class), [15](#)
- as.list, MultiDataSet-method (MultiDataSet-class), [15](#)
- assayData (MultiDataSet-class), [15](#)
- assayData, MultiDataSet-method (MultiDataSet-class), [15](#)
- chrNumToChar, [9](#)
- commonIds, [10](#)
- commonIds, MultiDataSet-method (MultiDataSet-class), [15](#)
- commonSamples, [11](#)
- commonSamples, MultiDataSet-method (MultiDataSet-class), [15](#)
- create_resultset (ResultSet), [21](#)
- createMultiDataSet (MultiDataSet-class), [15](#)
- dims (MultiDataSet-class), [15](#)
- dims, MultiDataSet-method (MultiDataSet-class), [15](#)
- fData (MultiDataSet-class), [15](#)
- fData, MultiDataSet-method (MultiDataSet-class), [15](#)
- fData, ResultSet-method (ResultSet), [21](#)
- featureData (MultiDataSet-class), [15](#)
- featureData, MultiDataSet-method (MultiDataSet-class), [15](#)
- getAssociation, [12](#)
- getAssociation, ResultSet-method (ResultSet), [21](#)
- ggplot, [20](#)
- iClusterPlus, [25](#)
- lambdaClayton, [13](#)
- length (MultiDataSet-class), [15](#)
- length, MultiDataSet-method (MultiDataSet-class), [15](#)
- length, ResultSet-method (ResultSet), [21](#)
- mae2mds, [13](#)
- mcia, [26](#)
- mds2mae, [14](#)
- MultiDataSet, [14, 14, 25, 26](#)
- MultiDataSet-class, [15](#)

MultiDataSet-methods
 (MultiDataSet-class), 15
 MultiDataSet-methods,
 (MultiDataSet-class), 15

 names (MultiDataSet-class), 15
 names, MultiDataSet-method
 (MultiDataSet-class), 15
 names, ResultSet-method (ResultSet), 21
 ncol (MultiDataSet-class), 15
 ncol, MultiDataSet-method
 (MultiDataSet-class), 15
 nrow (MultiDataSet-class), 15
 nrow, MultiDataSet-method
 (MultiDataSet-class), 15

 opt, 20
 opt, ResultSet-method (ResultSet), 21

 pData (MultiDataSet-class), 15
 pData, MultiDataSet-method
 (MultiDataSet-class), 15
 phenoData (MultiDataSet-class), 15
 phenoData, MultiDataSet-method
 (MultiDataSet-class), 15
 plot, ResultSet, ANY-method (ResultSet),
 21

 qq_plot, 20

 ResultSet, 12, 20, 21
 ResultSet-class (ResultSet), 21
 ResultSet-methods (ResultSet), 21
 rowRanges (MultiDataSet-class), 15
 rowRanges, MultiDataSet-method
 (MultiDataSet-class), 15
 rowRangesElements, 23
 rowRangesElements, MultiDataSet-method
 (MultiDataSet-class), 15
 rset, 24

 sampleNames, MultiDataSet-method
 (MultiDataSet-class), 15
 subset, MultiDataSet-method
 (MultiDataSet-class), 15

 topTable, 12, 22

 varLabels, ResultSet-method (ResultSet),
 21

 volcano_plot, 24

 w_iclusterplus, 25
 w_iclusterplus, MultiDataSet-method
 (MultiDataSet-class), 15
 w_mcia, 26
 w_mcia, MultiDataSet-method
 (MultiDataSet-class), 15