

Package ‘regioneReloaded’

March 15, 2023

Type Package

Title RegioneReloaded: Multiple Association for Genomic Region Sets

Version 1.0.0

URL <https://github.com/RMalinverni/regioneReload>

Description RegioneReloaded is a package that allows simultaneous analysis of associations between genomic region sets, enabling clustering of data and the creation of ready-to-publish graphs. It takes over and expands on all the features of its predecessor regioneR. It also incorporates a strategy to improve p-value calculations and normalize z-scores coming from multiple analysis to allow for their direct comparison. RegioneReloaded builds upon regioneR by adding new plotting functions for obtaining publication-ready graphs.

License Artistic-2.0

Encoding UTF-8

LazyData false

RoxygenNote 7.2.1

Roxygen list(markdown = TRUE)

Depends R (>= 4.2), regioneR

Imports stats, RColorBrewer, Rtsne, umap, ggplot2, ggrepel, reshape2, methods, scales, cluster, grid, grDevices

Suggests rmarkdown, BiocStyle, GenomeInfoDb, knitr, testthat (>= 3.0.0)

biocViews Genetics, ChIPSeq, DNaseSeq, MethylSeq, CopyNumberVariation, Clustering, MultipleComparison

VignetteBuilder knitr

Config/testthat/edition 3

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

git_branch RELEASE_3_16

git_last_commit ac84607

git_last_commit_date 2022-11-01

Date/Publication 2023-03-15

Author Roberto Malinverni [aut, cre] (<<https://orcid.org/0000-0002-0113-3417>>),
 David Corujo [aut],
 Bernat Gel [aut]

Maintainer Roberto Malinverni <roberto.malinverni@gmail.com>

R topics documented:

AlienGenome	3
AlienRSList_broad	3
AlienRSList_narrow	4
chooseHclustMet	4
createUniverse	5
crosswisePermTest	6
cw_Alien	7
cw_Alien_RaR	8
cw_Alien_ReG	8
cw_Alien_ReG_no_Square	9
cw_Alien_ReR	9
genoMatriXeR-class	10
getHClust	10
getMatrix	11
getMultiEvaluation	12
getParameters	13
makeCrosswiseMatrix	13
makeLZMatrix	15
mLZ_regA_ReG	16
mLZ_regA_ReG_br	17
mLZ_regD_ReG	17
multiLocalZscore	18
multiLocalZScore-class	19
plotCrosswiseDimRed	20
plotCrosswiseMatrix	22
plotLocalZScoreMatrix	23
plotSingleLZ	24
plotSinglePT	26
randomizeRegionsPerc	27
similarRegionSet	28

Index

30

AlienGenome

AlienGenome

Description

The Alien Genome is an artificial genomic coordinates system for the purposes of testing and demonstrating the functions of `regioneReload` with a low computing time.

Usage

```
data(cw_Alien)
```

Format

An objects of class [GRanges](#).

Details

The Alien Genome consists of four chromosomes and is generated by the following code:

```
AlienGenome <-  
toGRanges(data.frame(  
  chr = c("A1Chr1", "A1Chr2", "A1Chr3", "A1Chr4"),  
  start = c(rep(1, 4)),  
  end = c(2e6, 1e6, 5e5, 1e5)  
))
```

AlienRSList_broad

AlienRSList_broad

Description

List of region sets (as [GRanges](#)) on the [AlienGenome](#).

Usage

```
data(cw_Alien)
```

Format

A list of [GRanges](#) objects.

Details

This region sets are generated for the purpose of demonstrating the functions of RegioneReloaded with a low computing time and "predictable" associations. The regions are generated with by combining `createRandomRegions()` and `similarRegionSet()` so that there is a known overlap between certain region sets. To see a full description of this sample data and the code used to generate it, see the RegioneReloaded vignette.

AlienRSList_narrow	<i>AlienRSList_narrow</i>
--------------------	---------------------------

Description

List of region sets (as `GRanges`) on the `AlienGenome`.

Usage

```
data(cw_Alien)
```

Format

A list of `GRanges` objects.

Details

This region sets are generated for the purpose of demonstrating the functions of RegioneReloaded with a low computing time and "predictable" associations. The regions are generated with by combining `createRandomRegions()` and `similarRegionSet()` so that there is a known overlap between certain region sets. To see a full description of this sample data and the code used to generate it, see the RegioneReloaded vignette.

chooseHclustMet	<i>chooseHclustMet</i>
-----------------	------------------------

Description

Evaluate and choose the best method for clustering a matrix using the `hclust()` function.

Usage

```
chooseHclustMet(GM, scale = FALSE, vecMet = NULL, distHC = "euclidean")
```

Arguments

GM	matrix, numerical matrix.
scale	logical, if TRUE, the clustering will be performed using the scaled matrix. (default = FALSE)
vecMet	character, vector of methods that will be tested in the function. If NULL, the following methods will be tested: "complete", "average", "single", "ward.D2", "median", "centroid" and "mcquitty. (default = NULL)
distHC	character, the distance measure to be used from those available in <code>dist()</code> . (default = "euclidean")

Value

An object of class `hclust`

See Also

`hclust()`

Examples

```
M1 <- matrix(1:18, nrow = 6, ncol = 3)
set.seed(42)
M2 <- matrix(sample(100, 18), nrow = 6, ncol = 3)
GM <- cbind(M1, M2)

chooseHclustMet(GM)
```

createUniverse *createUniverse*

Description

Create the universe parameter for `regioneR::resampleRegions()` using all unique regions present in Alist.

Usage

```
createUniverse(Alist, joinR = TRUE)
```

Arguments

Alist	list of regions set in a format accepted for <code>regioneR</code>
joinR	logical, if TRUE all the regions will be joined using the function <code>regioneR::joinRegions()</code> . (default == TRUE)

Value

A list of [GRanges](#) objects

Examples

```
data("cw_Alien")

universe <- createUniverse(AlienRSList_narrow)
```

crosswisePermTest *crosswisePermTest*

Description

Perform multiple permutation tests between each element in two lists of region sets.

Usage

```
crosswisePermTest(Alist, Blist = NULL, sampling = FALSE, fraction = 0.15,
  min_sampling = 5000, ranFUN = "randomizeRegions", evFUN = "numOverlaps",
  ntimes = 100, universe = NULL, adj_pv_method = "BH",
  genome = "hg19", ...)
```

Arguments

Alist, Blist	GRangesList or list of region sets in any accepted formats by regioneR package (GRanges , data.frame etc.).
sampling	logical, if TRUE the function will use only a sample of each element of Alist to perform the test as specified in fraction. (default = FALSE)
fraction	logical, if sampling=TRUE, defines the fraction of the region sets used to perform the test. (default = 0.15)
min_sampling	numeric, minimum number of regions accepted after sampling is performed with the specified fraction. If the number of sampled regions is less than min_sampling, the number specified by min_sampling will be used as number of regions sampled instead. (default = 5000)
ranFUN	character, the randomization strategy used for the test, see regioneR . (default = "randomizeRegions")
evFUN	character, the evaluation strategy used for the test, see regioneR . (default = "numOverlaps")
ntimes	numeric, number of permutations used in the test. (default = 100)
universe	region set to use as universe, used only when regioneR::resampleRegions() function is selected. (default = NULL)
adj_pv_method	character, the method used for the calculation of the adjusted p-value, to choose between the options of p.adjust() . (default = "BH")

genome character or [GRanges](#), genome used to compute the randomization. (default = "hg19")

... further arguments to be passed to other methods.

Details

This function performs multiple permutation tests for all pairwise combinations of the elements in two lists of region sets. Essentially, it uses the [regioneR::permTest\(\)](#) function and its associated randomization and evaluation functions. It creates and returns a [genoMatriXeR](#) object with the result of the permutation tests stored in the `multiOverlaps` slot. In addition, all the parameters used for the test are stored in the `parameters` slot.

Value

A object of class [genoMatriXeR](#) containing three slots

- `@parameters`
- `@multioverlaps`
- `@matrix`

See Also

[genoMatriXeR](#), [regioneR](#), [regioneR::permTest\(\)](#), [regioneR::overlapPermTest\(\)](#)

Examples

```
fakeGenome <- regioneR::toGRanges("chrF", 1, 1000)
regA <- regioneR::createRandomRegions(nregions = 10, length.mean = 10,
length.sd = 2, genome = fakeGenome)
regB <- regioneR::createRandomRegions(nregions = 10, length.mean = 10,
length.sd = 2, genome = fakeGenome)
regAs <- similarRegionSet(GR = regA, genome = fakeGenome, name = "A",
vectorPerc = seq(0.1, 0.3, by = 0.1))
regBs <- similarRegionSet(GR = regB, genome = fakeGenome, name = "B",
vectorPerc = seq(0.1, 0.3, by = 0.1))
ABList <- c(regAs, regBs)
cw_ptAB <- crosswisePermTest(ABList, genome = fakeGenome, ntimes = 10)
print(cw_ptAB)
```

cw_Alien

cw_Alien

Description

Alien Genome crosswise matrix using [regioneR::randomizeRegions](#), [regioneR::circularRandomizeRegions](#), [regioneR::resampleRegions](#), [regioneR::resampleGenome](#) functions as permutation strategies.

Usage

```
data(cw_Alien)
```

Format

An objects of class `genoMatriXeR`; see `makeCrosswiseMatrix()`.

cw_Alien_RaR	<i>cw_Alien_RaR</i>
--------------	---------------------

Description

Alien Genome crosswise matrix using `regioneR::randomizeRegions()` function a permutation strategy. Alist = AlienRSList_narrow, Blist = AlienRSList_narrow

Usage

```
data(cw_Alien)
```

Format

An objects of class `genoMatriXeR`; see `makeCrosswiseMatrix()`.

cw_Alien_ReG	<i>cw_Alien_ReG</i>
--------------	---------------------

Description

Alien Genome crosswise matrix using `regioneR::resampleGenome()` function as permutations strategy. Alist = AlienRSList_narrow, Blist = AlienRSList_narrow

Usage

```
data(cw_Alien)
```

Format

An objects of class `genoMatriXeR`; see `makeCrosswiseMatrix()`.

cw_Alien_ReG_no_Square
cw_Alien_ReG_no_Square

Description

Alien Genome crosswise matrix using `regioneR::resampleGenome()` function as permutations strategy. Alist = AlienRSList_narrow, Blist = AlienRSList_broad

Usage

```
data(cw_Alien)
```

Format

An objects of class `genoMatriXeR`; see `makeCrosswiseMatrix()`.

cw_Alien_ReR *cw_Alien_ReR*

Description

Alien Genome crosswise matrix using `regioneR::resampleRegions()` function a permutation strategy. Alist = AlienRSList_narrow, Blist = AlienRSList_narrow

Usage

```
data(cw_Alien)
```

Format

An objects of class `genoMatriXeR`; see `makeCrosswiseMatrix()`.

genoMatriXeR-class *genoMatriXeR Class*

Description

An S4 class for "genoMatriXeR" object.

Slots

parameters List of parameters used to create the object.

multiOverlaps Results of multiple pairwise permutation tests generated with `crosswisePermTest()`.

matrix List of numerical matrices containing z-score, pvalues and correlation values generated with `makeCrosswiseMatrix()`

Examples

```
data("cw_Alien")

AlienRSList_narrow_small <- AlienRSList_narrow[c("regA","regB","regC")]

cw_test <- crosswisePermTest(Alist = AlienRSList_narrow_small,Blist = AlienRSList_narrow_small,
                             sampling = FALSE, genome = AlienGenome, per.chromosome = TRUE,
                             ranFUN = "resampleGenome", evFUN = "numOverlaps",
                             ntimes = 10, mc.cores = 2)

class(cw_test)
```

getHClust *getHClust*

Description

get Object of class `hclust` from `genoMatriXeR` or `multiLocalZScore`

Usage

```
getHClust( rR, hctype = "rows")
```

Arguments

rR A `genoMatriXeR` or `multiLocalZScore` object.
hctype character. Can be "rows" or "cols". (default= "cols")

Value

an object of class [hclust](#)

See Also

[genoMatriXeR](#), [multiLocalZScore](#), [hclust](#)

Examples

```
data("cw_Alien")

cw_Alien_ReG <- makeCrosswiseMatrix(cw_Alien_ReG)
hc <- getHClust(cw_Alien_ReG)

plot(hc)
```

getMatrix

Get Matrix

Description

Returns the matrix from an [genoMatriXeR](#) or [multiLocalZScore](#) object.

Usage

```
getMatrix(rR)
```

Arguments

rR [genoMatriXeR](#) or [multiLocalZScore](#) object

Value

a numerical matrix from a

See Also

[genoMatriXeR](#), [multiLocalZScore](#), [makeCrosswiseMatrix](#), [makeLZMatrix](#)

Examples

```
data("cw_Alien")

cw_Alien_ReG <- makeCrosswiseMatrix(cw_Alien_ReG)
mtx <- getMatrix(cw_Alien_ReG)

mtx
```

```
data("cw_Alien")

cw_Alien_RaR <- makeCrosswiseMatrix(cw_Alien_RaR)
GM <- getMatrix(cw_Alien_RaR)

GM
```

getMultiEvaluation *getMultiEvaluation*

Description

Get multiEvaluation slot from [genoMatrixeR](#) or [multiLocalZScore](#) class.

Usage

```
getMultiEvaluation( rR, namesRS = NULL)
```

Arguments

rR	A genoMatrixeR or multiLocalZScore object.
namesRS	a vector of names. (default = NA)

Value

If rR is a [genoMatrixeR](#) object, a list of data frames resuming the associations results. If rR is a [multiLocalZScore](#) object, a list of two elements: "resumeTable" that is a data frame summarizing the associations and "shifts", a list of shifts computed from [multiLocalZscore\(\)](#) function for the elements indicated in the nameRS vector.

See Also

[genoMatrixeR](#), [multiLocalZScore](#)

Examples

```
data("cw_Alien")

mevs <- getMultiEvaluation(cw_Alien_ReG, names = "regA")

mevs
```

getParameters	<i>getParameters</i>
---------------	----------------------

Description

Get parameters from a `genoMatriXeR` or `multiLocalZScore` class object.

Usage

```
getParameters(rR, show_err = FALSE)
```

Arguments

<code>rR</code>	A <code>genoMatriXeR</code> or <code>multiLocalZScore</code> class object.
<code>show_err</code>	logical, if TRUE the function returns a list with two dataframes: one containing the parameter values and one with any error messages that have been generated during the permutation test iterations when running crosswisePermTest .

Value

A dataframe with parameters and values, or a list with two dataframes with parameters and errors information.

See Also

[genoMatriXeR](#), [multiLocalZScore](#)

Examples

```
data("cw_Alien")  
prm <- getParameters(cw_Alien_ReG)  
prm
```

makeCrosswiseMatrix	<i>makeCrosswiseMatrix</i>
---------------------	----------------------------

Description

Populate the `matrix` slot in a [genoMatriXeR](#) object.

Usage

```
makeCrosswiseMatrix(mPT, clusterize = TRUE, hc.method = NULL, dist.method = "euclidean",
  transform = FALSE, scale = FALSE, zs.type = 'norm_zscore', symm_matrix = TRUE,
  selectRow = NULL, selectCol = NULL, pvcut = 1, subEX = 0, GM_diag = TRUE, ...)
```

Arguments

mPT	an object of class genoMatriXeR .
clusterize	logical, if TRUE the matrix will be clustered using the method specified by hc.method (default = TRUE)
hc.method	character, select the hclust() method to use for clustering the matrix. If NULL, the clustering method will be automatically selected by the function chooseHclustMet() . (default = NULL)
dist.method	character, the distance measure to be used from those available in dist() . (default = "euclidean")
transform	logical, if TRUE the matrix will be transformed using the function t() . (default = FALSE)
scale	logical, if TRUE the matrix will be scaled. (default = FALSE)
zs.type	character, z-score type to use to generate the matrix, either raw z-score ("zscore") or normalized z-score ("norm_zscore"). (default = "norm_zscore")
symm_matrix	logical, if TRUE the matrix will be treated as symmetrical (same clustering for rows and columns). (default = TRUE)
selectRow, selectCol	vector, the matrix will be reduced selecting the rows and/or columns in this vector. (default = NULL)
pvcut	numeric, the z-score value is substituted by subEX (0 by default) for all the associations with an adj.pvalue (as calculated in crosswisePermTest()) higher than pvcut. (default = 0.05)
subEX	numeric, value used to substitute the z-score values when the associated pvalue is higher than pvcut. (default = 0)
GM_diag	logic, if FALSE the values of the diagonal will be set to 0. (default = TRUE)
...	further arguments to be passed to other methods.

Details

This function will create a series of matrices of z-scores, adj.pvalues and pearson correlation values from all the pairwise permutation tests stored in the `multiOverlaps` slot of a [genoMatriXeR](#) as calculated with [multiPermTest\(\)](#). These matrices will then be stored in the `matrix` slot of the [genoMatriXeR](#) object. In addition, clustering will be performed on the association matrices using [hclust](#).

Value

An object of class [genoMatriXeR](#) containing three slots, with a populated `matrix` slot.

- @parameters
- @multioverlaps
- @matrix

See Also

[crosswisePermTest\(\)](#), [chooseHclustMet\(\)](#), [plotCrosswiseMatrix\(\)](#)

Examples

```
data("cw_Alien")

cw_Alien_ReG <- makeCrosswiseMatrix(cw_Alien_ReG)

summary(cw_Alien_ReG)
```

makeLZMatrix	<i>Make Local Z-Score Matrix</i>
--------------	----------------------------------

Description

Create a local z-score matrix from a [multiLocalZScore](#) object and save it in its matrix slot.

Usage

```
makeLZMatrix(mLZA, normalize = TRUE, clusterize = TRUE,
             centralize = NA, hc.method = NULL, dist.method = "euclidean",
             scale = FALSE, ...)
```

Arguments

mLZA	an object of class multiLocalZScore or a numerical matrix.
normalize	logical, if TRUE the z-score values in the matrix will be normalized. (default = FALSE)
clusterize	logical, if TRUE the matrix will be clustered using the method specified by hc.method (default = TRUE)
centralize	numeric, only z-score values in a number of steps (defined by centralize) around the center of the local association will be used for clustering. If NA, all the values in the matrix will be used for clustering. (default = NA)
hc.method	character, select the hclust() method to use for clustering the matrix. If NULL, the clustering method will be automatically selected by the function chooseHclustMet() . (default = NULL)
dist.method	character, the distance measure to be used from those available in dist() . (default = "euclidean")
scale	logical, if TRUE the matrix will be scaled. (default = FALSE)
...	further arguments to be passed to other methods.

Value

A object of class `multiLocalZScore` containing three slots, with a populated `matrix` slot.

- `@parameters`
- `@multiLocalZscores`
- `@matrix`

See Also

[localZScore](#)

Examples

```
data("cw_Alien")
```

mLZ_regA_ReG

mLZ_regA_ReG

Description

Alien Genome multiLocalZScore calculated for regA regionset from AlienRSList_narrow using `regioneR::resampleGenome()` function as permutation s strategy.

Usage

```
data(cw_Alien)
```

Format

An objects of class `multiLocalZScore`; see `makeLZMatrix()`.

mLZ_regA_ReG_br *mLZ_regA_ReG_br*

Description

Alien Genome multiLocalZScore calculated for regA regionset from AlienRSList_broad using [regioneR::resampleGenome\(\)](#) function as permutation s strategy.

Usage

`data(cw_Alien)`

Format

An object of class [multiLocalZScore](#)

mLZ_regD_ReG *mLZ_regD_ReG*

Description

Alien Genome multiLocalZScore calculated for regD regionset from AlienRSList_narrow using [regioneR::resampleGenome\(\)](#) function as permutation s strategy.

Usage

`data(cw_Alien)`

Format

An object of class [multiLocalZScore](#)

multiLocalZscore	<i>multiLocalZscore</i>
------------------	-------------------------

Description

Perform multiple permutation tests between a region set and each element in a list of region sets using shifted positions to calculate a local z-score.

Usage

```
multiLocalZscore(A, Blist = NULL, sampling = FALSE, fraction = 0.15,
  min_sampling = 5000, ranFUN = "randomizeRegions", evFUN = "numOverlaps",
  ntimes = 100, adj_pv_method = "BH", genome = "hg19", universe = NULL,
  window = 1000, step = 100, ...)
```

Arguments

A	query region set for which to estimate local z-score values.
Blist	GRangesList or list of region sets in any accepted formats by regioneR package (GRanges , data.frame etc.).
sampling	logical, if TRUE the function will use only a sample of each element of Alist to perform the test as specified in fraction. (default = FALSE)
fraction	logical, if sampling=TRUE, defines the fraction of the region sets used to perform the test. (default = 0.15)
min_sampling	numeric, minimum number of regions accepted after sampling is performed with the specified fraction. If the number of sampled regions is less than min_sampling, the number specified by min_sampling will be used as number of regions sampled instead. (default = 5000)
ranFUN	character, the randomization strategy used for the test, see regioneR . (default = "randomizeRegions")
evFUN	character, the evaluation strategy used for the test, see regioneR . (default = "numOverlaps")
ntimes	numeric, number of permutations used in the test. (default = 100)
adj_pv_method	character, the method used for the calculation of the adjusted p-value, to choose between the options of p.adjust() . (default = "BH")
genome	character or GRanges , genome used to compute the randomization. (default = "hg19")
universe	region set to use as universe, used only when regioneR::resampleRegions() function is selected. (default = NULL)
window	numeric, window (number of base pairs) in which the local z-score will be calculated. (default = 1000)
step	numeric, step (number of base pairs) by which will be estimated the local Z-score. (default = 100)
...	further arguments to be passed to other methods.

Details

This function performs multiple permutation tests between a single region set and each element in a list of region sets. For every pairwise combination, the evaluation step is repeated each time shifting the position of all the regions in the query region set by a fixed step inside a defined window (using `regioneR::localZScore()`). This produces a "local z-score" profile that can be indicative of the nature of the association between region sets. For example, an association can occur "centrally" if the z-score value drops sharply when sifting the region set. On the other hand, two region sets may have a peak of local z-score away from the central position if they happen to occur often at a regular distance, showing a "lateral" association.

Value

A object of class `multiLocalZScore` containing three slots

- @parameters
- @multiLocalZscores
- @matrix

See Also

`regioneR::localZScore()`

Examples

```
fakeGenome<- regioneR::toGRanges("chrF",1,1000)
regA <- regioneR::createRandomRegions(nregions = 10, length.mean = 10,
length.sd = 2,genome = fakeGenome)
regB <- regioneR::createRandomRegions(nregions = 10,length.mean = 10,
length.sd = 2,genome = fakeGenome)
regAs <-similarRegionSet(GR = regA,genome = fakeGenome, name = "A",
vectorPerc = seq(0.1,0.3,by =0.1))
regBs <-similarRegionSet(GR = regB,genome = fakeGenome, name = "B",
vectorPerc = seq(0.1,0.3,by =0.1))
ABList <- c(regAs,regBs)

mlz_ptAB <- multiLocalZscore(A = regA, Blist = ABList,
genome = fakeGenome, ntimes = 10)
summary(mlz_ptAB)
```

multiLocalZScore-class

multiLocalZScore Class

Description

An S4 class for "multiLocalZScore" object.

Slots

parameters List of parameters used to create the object

multiLocalZscores Results of multiple pairwise permutation tests on shifted region sets generated with `multiLocalZscore()`.

matrix List of numerical matrices containing local z-scores and correlation values generated with `makeLZMatrix()`.

Examples

```
data("cw_Alien")

AlienRSList_narrow_small <- AlienRSList_narrow[c("regA", "regB", "regC")]

mlz_test <- multiLocalZscore(A = AlienRSList_narrow_small$regA, Blist = AlienRSList_narrow_small,
                             sampling = FALSE, genome = AlienGenome, per.chromosome = TRUE,
                             ranFUN = "resampleGenome", evFUN = "numOverlaps",
                             ntimes = 10, mc.cores = 2)

class(mlz_test)
```

plotCrosswiseDimRed *plotCrosswiseDimRed*

Description

Plot a visualization of a [genoMatriXeR](#) object (or matrix) using different dimensional reduction algorithms (PCA, tSNE and UMAP).

Usage

```
plotCrosswiseDimRed(mPT, type = "PCA", GM_clust = NA, clust_met =
  "hclust", nc = 5, listRS = NULL, main = "", labSize = 2, emphasize = FALSE,
  labAll = FALSE, labMaxOverlap = 100, ellipse = TRUE, colPal = NULL,
  perplexity = 10, theta = 0.1, return_table = FALSE, return_plot = TRUE, ...)
```

Arguments

mPT	an object of class <code>genoMatriXeR</code> or a numerical matrix.
type	character, dimensional reduction algorithm to use ("PCA", "tSNE", "UMAP"). (default = "PCA")
GM_clust	numeric, vector of assigned clusters used to cluster the matrix. If NA, the matrix will be clustered using the method defined by <code>clust_met</code> . (default = NA)
clust_met	character, unsupervised cluster strategy used (hclust , kmeans or pam). (default = "hclust")

nc	numeric, number of clusters to define if using the default "kmeans" method. (default = 5)
listRS	list, a list of names of region sets of interest to be highlighted in the graph. (default = NULL)
main	character, title for the plot. (default = "")
labSize	numeric, size for point labels in the plot. If 0, no labels will be plotted. (default = 2)
emphasize	logical, if TRUE, only the cluster in which the elements of listRS are present will be highlighted. (default = FALSE)
labAll	logical, if TRUE all data points are labelled, even if not in listRS when emphasize = TRUE. (default = FALSE)
labMaxOverlap	numeric, max.overlaps for geom_text_repel . (default = 100)
ellipse	logical, if TRUE ellipses will be drawn around the clusters. (default = FALSE)
colPal	character, colors to use as palette for the plot. If NULL, default colors will be used. (default = NULL)
perplexity, theta	numeric, if type = "tSNE" values of perplexity and theta for the function Rtsne() . (default = 10)
return_table	logical, if TRUE a table with the cluster assigned to each region is returned. (default = FALSE)
return_plot	logical, if TRUE a plot is returned. (default = TRUE)
...	further arguments to be passed on to other methods

Details

This function generates a plot with a two-dimensional representation of the association data stored in a [genoMatriXeR](#) object by using either PCA, tSNE or UMAP transformations of the data. This function incorporates a clustering step and allows to highlight specific region sets of interest and the clusters they belong to. In addition to generating a plot, a table with the cluster assignments can be retrieved.

Value

A ggplot object or a table with cluster assignments is returned.

See Also

[crosswisePermTest\(\)](#)

Examples

```
data("cw_Alien")

cw_Alien_ReG <- makeCrosswiseMatrix(cw_Alien_ReG)

plotCrosswiseDimRed(cw_Alien_ReG, type = "PCA")
```

```
CDR_clust <- plotCrosswiseDimRed(cw_Alien_ReG, type = "UMAP", return_table = TRUE)

print(CDR_clust)
```

plotCrosswiseMatrix *plotCrosswiseMatrix*

Description

Plot matrix of associations/correlations stored in a [genoMatriXeR](#) object.

Usage

```
plotCrosswiseMatrix(mPT, lineColor = NA, interpolate = FALSE, colMatrix =
"default", matrix_type = "association", cor = "row",
maxVal = NA, main = "", ord_mat = NULL)
```

Arguments

mPT	an object of class genoMatriXeR or a numerical matrix.
lineColor	logical, color for the line grid delineating the tiles of the matrix plot. If NA, no line will be drawn. (default = NA)
interpolate	logical, if TRUE the image will be interpolated using the function geom_raster() . (default = FALSE)
colMatrix	character or vector of colors, if "default" will be used a default selection see..
matrix_type	character, type of matrix to be plotted, either "association" or "correlation". (default = "association")
cor	character, if matrix_type is "correlation", choose if the function cor() will be executed on each "row" or "col" of the matrix. (default = "row")
maxVal	numeric, maximum absolute value displayed by the plot. If "max", the maximum values in the matrix are used. If NA, the 0.95 quantile of all absolute values is used. (default = NA)
main	character, title of the plot. (default = "")
ord_mat	numeric, list with two numeric vectors that represent the ordering of rows and column of the matrix to be used in the plot. If NULL, the order of the matrix is preserved as is. (default = NULL)

Details

This functions creates a graphical representation of the matrix of associations stored in a [genoMatriXeR](#) object. The values plotted and clustering options can be controlled when creating the matrix with the function [makeCrosswiseMatrix](#).

Value

Returns a ggplot object.

See Also

[crosswisePermTest](#) [makeCrosswiseMatrix](#)

Examples

```
data("cw_Alien")

cw_Alien_ReG <- makeCrosswiseMatrix( cw_Alien_ReG)

plotCrosswiseMatrix(cw_Alien_ReG, matrix_type = "association")

plotCrosswiseMatrix(cw_Alien_ReG, matrix_type = "correlation")
```

plotLocalZScoreMatrix *Plot Local Z-Score Matrix*

Description

Plot Local Z-Score Matrix of associations/correlations stored in a [multiLocalZScore](#) object.

Usage

```
plotLocalZScoreMatrix (mLZ, lineColor = NA, colMatrix = "default",
matrix_type = "association", maxVal = "max", main = "", labSize = 6,
revert = FALSE, highlight = NULL, highlight_size = 2.5, highlight_max = FALSE,
smoothing = FALSE, ...)
```

Arguments

mLZ	an object of class multiLocalZScore or a matrix
lineColor	logical, color for the line grid delineating the tiles of the matrix plot. If NA, no line will be drawn. (default = NA)
colMatrix	character or vector of colors, if "default" will be used a default selection see..
matrix_type	character, type of matrix to be plotted, either "association" or "correlation". (default = "association")
maxVal	numeric, maximum absolute value displayed by the plot. If "max", the maximum values in the matrix are used. If NA, the 0.95 quantile of all absolute values is used. (default = NA)
main	character, title of the plot. (default = "")
labSize	numeric, size for the plot labels. (default = 6)

revert	logical, if TRUE reverts the order of the plotted elements. (default = FALSE)
highlight	character, vector indicating the region set names to highlight by adding labels pointing to the 0 shift position (default = NULL)
highlight_size	numeric, size of the highlight labels. (default = 2.5)
highlight_max	logical, if TRUE the highlight labels are placed at the maximum local z-score value instead of the 0 shift position. (default = FALSE)
smoothing	logical, if TRUE the <code>stats::smooth.spline</code> function will be applied to the local z-score profile. (default = FALSE)
...	further arguments to be passed to other methods.

Value

Returns a ggplot object.

See Also

[multiLocalZscore](#) [makeLZMatrix](#) [multiLocalZScore](#)

Examples

```
data("cw_Alien")
```

plotSingleLZ

plotSingleLZ

Description

Plot the result of specific local Z-Score tests from a [multiLocalZScore](#) object in the form of line plot profiles.

Usage

```
plotSingleLZ(mLZ, RS, xlab = "", normZS = TRUE, ylim = NULL, main = NA,
  colPal = NULL, labValues = TRUE, labSize = 2.5, labMax = FALSE, smoothing = FALSE, ...)
```

Arguments

mLZ	an object of class multiLocalZScore .
RS	character, vector of region set names for which to plot the local Z-score results.
xlab	character, label for the x axis. (default = NA)
normZS	logical, indicates whether the normalized Z-score values should be plotted. If FALSE, the raw Z-score is used. (default = TRUE)

<code>yylim</code>	numeric, vector with two elements: minimum and maximum Y values of the plot. If NULL, the plot limits are set by default so all data points can be plotted. (default = NULL)
<code>main</code>	character, title for the plot. If NA, the name of the query region set in the multi-LocalZScore object will be used. (default = NA)
<code>colPal</code>	character, colors to use as palette for the plot. If NULL, default colors will be used. (default = NULL)
<code>labValues</code>	logical, if TRUE each local Z-score profile is labelled at position 0 with the name of the region set and its Z-score value at the central position. (default = TRUE)
<code>labSize</code>	numerical, size of the labels from labValues in the plot. (default = 2.5)
<code>labMax</code>	logical, if TRUE the labels are placed at the maximum value of each local Z-score profile instead of the center. (default = FALSE)
<code>smoothing</code>	logical, if TRUE the <code>smooth.spline()</code> function will be applied to the localZ-score profile. (default = FALSE)
<code>...</code>	further arguments to be passed to other methods.

Details

This function generates a line plot with the local Z-score profiles of selected region sets from a [multiLocalZScore](#) object. This type of plot complements the local Z-score matrix (generated by [plotLocalZScoreMatrix](#), since it allows to visualize in detail the local Z-score profile of just the region sets of interest.

This plot is well suited for a single or a few region sets, but will get busy if attempting to plot many different profiles. For the latter, the full matrix generated by [plotLocalZScoreMatrix](#) is usually a better visualization option.

Value

Returns a ggplot object.

See Also

[multiLocalZscore\(\)](#), [makeLZMatrix\(\)](#)

Examples

```
data("cw_Alien")

plotSingleLZ(mLZ_regA_ReG, RS = c("regD", "regD_02", "regA", "regAB_04"),
labMax = TRUE, smoothing = TRUE)
```

`plotSinglePT`*plotSinglePT*

Description

Plot the result of a single pairwise permutation test from a `genoMatriXeR` object.

Usage

```
plotSinglePT(mPT, RS1, RS2, xlab = NA, main = NA)
```

Arguments

<code>mPT</code>	an object of class genoMatriXeR .
<code>RS1</code> , <code>RS2</code>	character, names of region sets in a <code>genoMatriXeR</code> object for which to represent the pairwise permutation test results.
<code>xlab</code>	character, label for x axis. (default = NA)
<code>main</code>	title for the plot, if NA the name of the <code>genoMatriXeR</code> object is used (default = NA)

Details

This function generates a plot representing the result of a single permutation test stored in a [genoMatriXeR](#) object. This includes a plot of the density distribution of the randomized evaluations and a vertical line showing the observed evaluation in the original region set. The values of the mean randomized evaluations and the value of the observed evaluation are shown, in addition to the calculated Z-score, normalized Z-score and adjusted p-value.

Value

Returns a ggplot object.

See Also

[crosswisePermTest](#) [makeCrosswiseMatrix](#)

Examples

```
data("cw_Alien")
plotSinglePT(cw_Alien_ReG, RS1 = "regA", RS2 = "regA_05")
plotSinglePT(cw_Alien_ReG, RS1 = "regA", RS2 = "regC")
```

randomizeRegionsPerc *randomizeRegionsPerc*

Description

Create a random region set similar to a reference region set.

Usage

```
randomizeRegionsPerc(GR, genome = "hg19", frac = 0.2, ...)
```

Arguments

GR	a GRanges object with the input region set.
genome	genome of reference to generate the similar region sets. (default = "hg19")
frac	fraction of the original region set to randomize. (default = 0.2)
...	further arguments to be passed to other methods.

Details

This function takes an input region set and generates a region set where a fraction of the regions has been randomized.

Value

a [GRanges](#) object

See Also

[similarRegionSet\(\)](#)

Examples

```
data("cw_Alien")

nreg <- 100

regA <-
  createRandomRegions(
    nregions = nreg,
    length.mean = 100,
    length.sd = 10,
    non.overlapping = TRUE,
    genome = AlienGenome
  )

regA_02 <- randomizeRegionsPerc(GR = regA, genome = AlienGenome, frac = 0.2)
```

similarRegionSet *similar RegionSets*

Description

Create a list of similar region sets to a reference region set.

Usage

```
similarRegionSet(GR, name, genome, vectorPerc)
```

Arguments

GR	a GRanges object with the input region set.
name	character, name for the output region sets. The names will be generated by adding an underscore and the fraction of similarity after the name of each region set generated. (default = "A")
genome	genome of reference to generate the similar region sets. (default = "hg19")
vectorPerc	numeric, vector of desired randomized fractions. (default = seq(.1,.9,.1))

Details

This function takes a region set as an input and a vector of desired randomized fractions. For each fraction value, a new region set will be generated where that fraction of the original regions in the input region set has been randomized. In effect, this creates region sets that are "similar" to a controlled degree to the original region set. This tool can be useful for validation purposes and its use in the demonstration of the usage of this package can be seen in the `RegionReloaded` vignette.

Value

A list of [GRanges](#) objects.

See Also

[GRanges](#)

Examples

```
data("cw_Alien")

A<-createRandomRegions(nregions = 20, length.mean = 1000, length.sd = 100,
genome = AlienGenome)

similAList <- similarRegionSet(GR = A, genome = AlienGenome,
vectorPerc = seq(0.1,0.9,0.2), name = "test")

summary (similAList)
```

```
data("cw_Alien")

regA <- createRandomRegions(
  nregions = 100,
  length.mean = 10,
  length.sd = 5,
  genome = AlienGenome
)

listRegA <- similarRegionSet(GR = regA, genome = AlienGenome)
summary(listRegA)
```

Index

* datasets

- AlienGenome, 3
 - AlienRSList_broad, 3
 - AlienRSList_narrow, 4
 - cw_Alien, 7
 - cw_Alien_RaR, 8
 - cw_Alien_ReG, 8
 - cw_Alien_ReG_no_Square, 9
 - cw_Alien_ReR, 9
 - mLZ_regA_ReG, 16
- AlienGenome, 3, 3, 4
- AlienRSList_broad, 3
- AlienRSList_narrow, 4
- chooseHclustMet, 4
- chooseHclustMet(), 14, 15
- cor(), 22
- createRandomRegions(), 4
- createUniverse, 5
- crosswisePermTest, 6, 13, 23, 26
- crosswisePermTest(), 10, 14, 15, 21
- cw_Alien, 7
- cw_Alien_RaR, 8
- cw_Alien_ReG, 8
- cw_Alien_ReG_no_Square, 9
- cw_Alien_ReR, 9
- data.frame, 6, 18
- dist(), 5, 14, 15
- genoMatriXeR, 7–14, 20–22, 26
- genoMatriXeR-class, 10
- geom_raster(), 22
- geom_text_repel, 21
- getHClust, 10
- getMatrix, 11
- getMultiEvaluation, 12
- getParameters, 13
- gMXR (genoMatriXeR-class), 10
- GRanges, 3, 4, 6, 7, 18, 27, 28
- GRangesList, 6, 18
- hclust, 5, 10, 11, 14, 20
- hclust(), 4, 5, 14, 15
- kmeans, 20
- localZScore, 16
- makeCrosswiseMatrix, 11, 13, 22, 23, 26
- makeCrosswiseMatrix(), 8–10
- makeLZMatrix, 11, 15, 24
- makeLZMatrix(), 16, 20, 25
- mLZ_regA_ReG, 16
- mLZ_regA_ReG_br, 17
- mLZ_regD_ReG, 17
- mLZS (multiLocalZScore-class), 19
- multiLocalZScore, 10–13, 15–17, 19, 23–25
- multiLocalZscore, 18, 24
- multiLocalZscore(), 12, 20, 25
- multiLocalZScore-class, 19
- multiPermTest(), 14
- p.adjust(), 6, 18
- pam, 20
- plotCrosswiseDimRed, 20
- plotCrosswiseMatrix, 22
- plotCrosswiseMatrix(), 15
- plotLocalZScoreMatrix, 23, 25
- plotSingleLZ, 24
- plotSinglePT, 26
- randomizeRegionsPerc, 27
- regioneR::circularRandomizeRegions, 7
- regioneR::joinRegions(), 5
- regioneR::localZScore(), 19
- regioneR::overlapPermTest(), 7
- regioneR::permTest(), 7
- regioneR::randomizeRegions, 7
- regioneR::randomizeRegions(), 8

`regioneR::resampleGenome`, [7](#)
`regioneR::resampleGenome()`, [8](#), [9](#), [16](#), [17](#)
`regioneR::resampleRegions`, [7](#)
`regioneR::resampleRegions()`, [5](#), [6](#), [9](#), [18](#)
`Rtsne()`, [21](#)

`similarRegionSet`, [28](#)
`similarRegionSet()`, [4](#), [27](#)
`smooth.spline()`, [25](#)
`stats::smooth.spline`, [24](#)

`t()`, [14](#)