Package 'BulkSignalR'

December 1, 2025

Type Package

Title Infer Ligand-Receptor Interactions from bulk expression (transcriptomics/proteomics) data, or spatial transcriptomics

Version 1.2.1

Description Inference of ligand-receptor (LR) interactions from bulk expression (transcriptomics/proteomics) data, or spatial transcriptomics. BulkSignalR bases its inferences on the LRdb database included in our other package, SingleCellSignalR available from Bioconductor. It relies on a statistical model that is specific to bulk data sets. Different visualization and data summary functions are proposed to help navigating prediction results.

URL https://github.com/ZheFrench/BulkSignalR

BugReports https://github.com/ZheFrench/BulkSignalR/issues

License CeCILL | file LICENSE

Encoding UTF-8

LazyData FALSE

Depends R (>= 4.5)

biocViews Network, RNASeq, Software, Proteomics, Transcriptomics, NetworkInference, Spatial

Imports BiocFileCache, httr2, RCurl, cli, curl, rlang, jsonlite, matrixStats, methods, doParallel, glmnet, ggalluvial, ggplot2, gridExtra, grid, Rtsne, ggrepel, foreach, multtest, igraph, orthogene, stabledist, circlize (>= 0.4.14), ComplexHeatmap (>= 2.0.0), stats, scales, RANN, SpatialExperiment, SummarizedExperiment, tools

Suggests knitr, markdown, rmarkdown, STexampleData, testthat (>= 3.0.0), codetools, Matrix, lattice, cluster, survival, MASS, nlme

Config/testthat/edition 3

RoxygenNote 7.3.3

VignetteBuilder knitr

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

git_branch RELEASE_3_22

git_last_commit 68e85a5

git_last_commit_date 2025-11-08

Repository Bioconductor 3.22

Date/Publication 2025-12-01

Author Jacques Colinge [aut] (ORCID: <https://orcid.org/0000-0003-2466-4824>), Jean-Philippe Villemin [cre] (ORCID: <https://orcid.org/0000-0002-1838-5880>)

Maintainer Jean-Philippe Villemin <jpvillemin@gmail.com>

Contents

`BulkSignalR-package` *BulkSignalR: A package to infer ligand-receptor interactions from bulk transcriptomics or proteomics*

Description

Inference of ligand-receptor interactions from bulk (transcriptomic or proteomic) data. BulkSignalR bases its inferences on the LRdb database. It relies on a statistical model that is specific to bulk data sets. Different visualization and data summary functions are proposed to help navigating results.

Author(s)

Maintainer: Jean-Philippe Villemin <jpvillemin@gmail.com> (ORCID)

Authors:

- Jacques Colinge <jacques.colinge@inserm.fr> (ORCID)

See Also

Useful links:

- https://github.com/ZheFrench/BulkSignalR
- Report bugs at https://github.com/ZheFrench/BulkSignalR/issues

---

`.buildPermutatedCountMatrix` *Internal function to generate a randomized expression matrix*

Description

Internal function to generate a randomized expression matrix

Usage

```
.buildPermutatedCountMatrix(ncounts, pind)
```

Arguments

ncounts	A matrix of normalized read counts.
pind	Permutation indices such as returned by .buildPermutationIndices.

Value

ncount with shuffled row names (gene symbols). Shuffling is performed within rows of comparable average expression.

.buildPermutationIndices *Internal function to generate expression matrix permutation indices*

Description

Internal function to generate expression matrix permutation indices

Usage

```
.buildPermutationIndices(ncounts, n.bins = 20)
```

Arguments

ncounts	A matrix of normalized read counts.
n.bins	Number of bins.

Value

A list containing a vectors of row indices. The vector at index i in the list contains the row indices of rows with mean normalized read count in bin i.

.cacheAdd *Add cache for resources.*

Description

Add cache for resources (pathways, lrdb, or network) downloaded from the web or local database. This part is handled with BiocFileCache.

Usage

```
.cacheAdd(fpath, cacheDir, resourceName, verbose = FALSE, download = TRUE)
```

Arguments

fpath	Path to file on the web or local system.
cacheDir	Absolute path to cache directory.
resourceName	Ressource name.
verbose	Default FALSE
download	Logical(TRUE/FALSE) Default TRUE for download.

Value

Returns ‘NULL‘, invisibly.

.cacheCheckIn *Check existence of a record in the cache.*

Description

Check whether the cache record exists or not by passing to the function an associated keyword related to the resource we are looking for.

Usage

```
.cacheCheckIn(bfc, resourceName)
```

Arguments

bfc	Object of class BiocFileCache, created by a call to BiocFileCache::BiocFileCache()
resourceName	keyword associated to a specific resource name.

Value

logical This function returns TRUE if a record with the requested keyword already exists in the file cache, otherwise it returns FALSE.

.cdfAlphaStable *Internal function to compute a censored alpha-) stable CDF*

Description

Internal function to compute a censored alpha-) stable CDF

Usage

```
.cdfAlphaStable(x, par)
```

Arguments

x	A vector of observed values.
par	A list containing the censored stable model parameters.

Value

A vector of probabilities P(X<x|par).

.cdfEmpirical *Internal function to compute an empirical CDF*

Description

Internal function to compute an empirical CDF

Usage

```
.cdfEmpirical(x, par)
```

Arguments

x	A vector of observed values.
par	A list containing the step function implementing the CDF.

Value

A vector of probabilities P(X<x|par).

.cdfGaussian *Internal function to compute a censored Gaussian CDF*

Description

Internal function to compute a censored Gaussian CDF

Usage

```
.cdfGaussian(x, par)
```

Arguments

x	A vector of observed values.
par	A list containing the censored Gaussian model parameters.

Value

A vector of probabilities P(X<x|par).

.cdfKernelEmpirical *Internal function to compute a Gaussian kernel-based empirical CDF*

Description

Internal function to compute a Gaussian kernel-based empirical CDF

Usage

```
.cdfKernelEmpirical(x, par)
```

Arguments

x	A vector of observed values.
par	A list containing the step function implementing the CDF.

Value

A vector of probabilities P(X<x|par).

.cdfMixedGaussian *Internal function to compute a censored mixed-Gaussian CDF*

Description

Internal function to compute a censored mixed-Gaussian CDF

Usage

```
.cdfMixedGaussian(x, par)
```

Arguments

x	A vector of observed values.
par	A list containing the censored mixed-Gaussian model parameters.

Value

A vector of probabilities P(X<x|par).

`.checkInteroperabilityForCounts` *Internal function to check and extract a count matrix if a more complex object than a simple matrix or data frame is given as parameter. Main usage is to link with Bioconductor objects.*

Description

Internal function to check and extract a count matrix if a more complex object than a simple matrix or data frame is given as parameter. Main usage is to link with Bioconductor objects.

Usage

```
.checkInteroperabilityForCounts(
  counts,
  symbol.col,
  x.col,
  y.col,
  barcodeID.col
)
```

Arguments

`counts`	A table or matrix of read counts (or protein abundance). It can also be a SummarizedExperiment or SpatialExperiment object from which the count matrix should be extracted. See `BSRDataModel`.
`symbol.col`	The index of the column containing the gene symbols in case those are not the row names of `counts` already. In a SpatialExperiment object, the index in the data frame returned by rowData().
`x.col`	In a SpatialExperiment object, the index of the column containing the x coordinates in the dafaframe returned by rowData(), usually named array_row.
`y.col`	In a SpatialExperiment object, the index of the column containing the y coordinates in the dafaframe returned by rowData(), usually named array_col.
`barcodeID.col`	In a SpatialExperiment object, the index of the column containing the barcodeID in the dafaframe returned by colData(), usually named barcode_id.

Value

A matrix of count (or abundance) values

`.checkRDSFromCache` *Check for valid RDS cache file.*

Description

This function checks whether a cache entry is a valid RDS file. Returns TRUE if the cache entry is valid, FALSE otherwise. In the case of an invalid file, the cache entry and file are deleted.

Usage

```
.checkRDSFromCache(bfc, resourceName)
```

Arguments

bfc	Object of class BiocFileCache, created by a call to BiocFileCache::BiocFileCache()
resourceName	keyword associated to a specific resource name.

Value

logical TRUE/FALSE

.checkReceptorSignaling *Internal function to check receptor signaling downstream*

Description

Assess the existence of correlations between a receptor, part of a ligand-receptor pair, and genes coding for proteins forming a complex with the receptor or genes regulated by the receptor downstream signaling.

Usage

```
.checkReceptorSignaling(
  ds,
  lr,
  reference = c("REACTOME-GOBP", "REACTOME", "GOBP"),
  max.pw.size = 400,
  min.pw.size = 5,
  min.positive = 4,
  use.full.network = FALSE,
  restrict.pw = NULL,
  with.complex = TRUE
)
```

Arguments

ds	A BSRDataModel object.
lr	A table as returned by .getCorrelatedLR().
reference	Which pathway reference should be used ("REACTOME" for Reactome, "GOBP" for GO Biological Process, or "REACTOME-GOBP" for both).
max.pw.size	Maximum pathway size to consider from the pathway reference.
min.pw.size	Minimum pathway size to consider from the pathway reference.
min.positive	Minimum number of target genes to be found in a given pathway.
use.full.network	A logical to avoid limiting the reference network to the detected genes and use the whole reference network.
restrict.pw	A list of pathway IDs to restrict the application of the function.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.

Value

A data frame extending `lr` content with the pathways found to contain the receptors and data about target gene correlations with those receptors. Strings in semi-colon-separated format are used to report target genes and their Spearman correlations with the receptor in the data frame. The target genes are sorted according to the correlation coefficient.

In a pathway of the reference, i.e., a Reactome pathway or the genes of a GOBP term, the target genes are the genes coding for proteins forming a complex with the receptor and the genes in the pathway downstream the receptor, which are given as regulated by the pathway. If `with.complex` is set to `FALSE`, then only the regulated genes are considered. Participation to a complex and being regulated as well as the pathway directed topologies are defined by Reactome and KEGG pathways as provided by PathwayCommons.

The maximum pathway size is used to limit the redundancy inherent to GOBP and Reactome. The minimum pathway size is used to avoid overspecific, noninformative results.

`.checkRegulatedReceptorSignaling` *Internal function to check receptor signaling downstream*

Description

Assess the existence of concomitant regulations between a receptor, part of a ligand-receptor pair, and genes coding for proteins forming a complex with the receptor or genes regulated by the receptor downstream signaling.

Usage

```
.checkRegulatedReceptorSignaling(
  ds,
  cc,
  lr,
  reference = c("REACTOME-GOBP", "REACTOME", "GOBP"),
  pos.targets = FALSE,
  neg.targets = FALSE,
  min.t.logFC = 0.5,
  use.full.network = FALSE,
  max.pw.size = 400,
  min.pw.size = 5,
  min.positive = 2,
  restrict.pw = NULL,
  with.complex = TRUE
)
```

Arguments

`ds`	An optional BSRDataModel object.
`cc`	A BSRClusterComp object.
`lr`	A table as returned by `.getRegulatedLR()`.
`reference`	Which pathway reference should be used ("REACTOME" for Reactome, "GOBP" for GO Biological Process, or "REACTOME-GOBP" for both).

pos.targets	A logical imposing that all the network targets must display positive logFC, i.e. logFC >= min.t.logFC.
neg.targets	A logical imposing that all the network targets must display negative logFC, i.e. logFC <= - min.t.logFC.
min.t.logFC	The minimum log2 fold-change allowed for targets in case pos.targets or neg.targets are used.
use.full.network	A logical to avoid limiting the reference network to the detected genes and use the whole reference network.
max.pw.size	Maximum pathway size to consider from the pathway reference.
min.pw.size	Minimum pathway size to consider from the pathway reference.
min.positive	Minimum number of target genes to be found in a given pathway.
restrict.pw	A list of pathway IDs to restrict the application of the function.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.

Value

A data frame extending lr content with the pathways found to contain the receptors and data about receptor target gene regulations Strings in semi-colon-separated format are used to report target genes and their regulation P-values in the data frame. The target genes are sorted according to the P-values in decreasing order.

In case ds is set, then correlations between the receptor and target genes will be computed for documentation or additional use. The row names of differentialStats(cc) and ncounts(ds) must match exactly (not necessarily in the same order). The same is true for differentialStats(scc) in case scc is provided.

In a pathway of the reference, i.e., a Reactome pathway or the genes of a GOBP term, the target genes are the genes coding for proteins forming a complex with the receptor and the genes in the pathway downstream the receptor, which are given as regulated by the pathway. If with.complex is set to FALSE, then only the regulated genes are considered. Participation to a complex and being regulated as well as the pathway directed topologies are defined by Reactome and KEGG pathways as provided by PathwayCommons.

The maximum pathway size is used to limit the redundancy inherent to GOBP and Reactome. The minimum pathway size is used to avoid overspecific, noninformative results.

.customheatmap *Heatmap function for gene signature expression*

Description

Generate one heatmap used by signatureHeatmaps.

Usage

```
.customheatmap(
  counts,
  name,
  fontsize = 6,
  col.fontsize = 6,
  legend.fontsize = 8,
  annot.fontsize = 8,
  h.height = 3,
  scoring = NULL,
  cols.scoring = NULL,
  hcl.palette = "Blues 3",
  show_column_names = FALSE
)
```

Arguments

counts	Matrix of counts.
name	Name of the heatmap to generate.
fontsize	Font size for row (gene) names.
col.fontsize	Font size for column (sample) names.
legend.fontsize	Font size for the legends.
annot.fontsize	Font size for column annotation names.
h.height	Heatmap height in cm.
scoring	Vector of sample scores for a chosen pathway. If NULL, then no column annotation is produced.
cols.scoring	Fixed colorRamp2 object.
hcl.palette	Palette from HCL colormaps supported by ComplexHeatmap.

Value

A ComplexHeatmap object.

This is a convenience function that relies on the ComplexHeatmap package.

.cutExtremeValues *Internal function to cut extreme values from a matrix*

Description

Internal function to cut extreme values from a matrix

Usage

```
.cutExtremeValues(m, p)
```

Arguments

m	A matrix.
p	Proportion of top and bottom values for thresholding.

Value

A matrix with values beyond top and bottom thresholds repaced by the latter thresholds.

.downstreamRegulatedSignaling *Internal function to check receptor signaling downstream*

Description

Internal function to check receptor signaling downstream

Usage

```
.downstreamRegulatedSignaling(
  lr,
  pw,
  pw.size,
  rncounts,
  stats,
  id.col,
  gene.col,
  pw.col,
  min.positive,
  pos.targets = FALSE,
  neg.targets = FALSE,
  min.t.logFC = 0.5,
  with.complex = TRUE
)
```

Arguments

lr	A data frame as returned by `.getRegulatedLR()`.
pw	A table defining the reference pathways.
pw.size	A named vector with pathway sizes (names are pathway IDs).
rncounts	A matrix of normalized read counts with at least all the ligands, receptors, and genes in the reference pathways.
stats	A data.frame with a column 'pval' and `rownames(stats)` assigned to gene symbols. Rows must at least include all the ligands, receptors, and genes in the reference pathways.
id.col	Column index or name in pw for the pathway IDs.
gene.col	Column index or name in pw for the gene symbols.
pw.col	Column index or name in pw for the pathway names.
min.positive	Minimum number of target genes to be found in a given pathway.

pos.targets	A logical imposing that all the network targets must display positive logFC, i.e. logFC >= min.t.logFC.
neg.targets	A logical imposing that all the network targets must display negative logFC, i.e. logFC <= - min.t.logFC.
min.t.logFC	The minimum log2 fold-change allowed for targets in case pos.targets or neg.targets are used.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.

Value

A table reporting all the ligand-receptor pairs provided in `lr` along with the pathways found and data about target gene regulation P-values. Note that correlations are currently set to 1 to avoid lengthy computations with scRNA-seq data and multiple cell populations.

.downstreamSignaling *Internal function to check receptor signaling downstream*

Description

Internal function to check receptor signaling downstream

Usage

```
.downstreamSignaling(
  lr,
  pw,
  pw.size,
  rncounts,
  id.col,
  gene.col,
  pw.col,
  min.positive,
  with.complex = TRUE
)
```

Arguments

lr	A data frame as returned by `.getCorrelatedLR()`.
pw	A table defining the reference pathways.
pw.size	A named vector with pathway sizes (names are pathway IDs).
rncounts	A matrix of normalized read counts with at least all the ligands, receptors, and genes in the reference pathways.
id.col	Column index or name in `pw` for the pathway IDs.
gene.col	Column index or name in `pw` for the gene symbols.
pw.col	Column index or name in `pw` for the pathway names.
min.positive	Minimum number of target genes to be found in a given pathway.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.

Value

A table reporting all the ligand-receptor pairs provided in `lr` along with the pathways found and data about target gene correlations with the receptor.

`.edgesLRIntracell` *Internal function to generate a ligand-receptor-downstream signaling network*

Description

Internal function to generate a ligand-receptor-downstream signaling network

Usage

```
.edgesLRIntracell(
  pairs,
  pw,
  tg.genes,
  tg.corr,
  id.col,
  gene.col,
  min.cor = 0.25,
  pos.targets = FALSE,
  neg.targets = FALSE,
  tg.pval = NULL,
  max.pval = NULL,
  tg.logFC = NULL,
  min.logFC = 0
)
```

Arguments

`pairs`	A ligand-receptor table such as output by `LRinter(BSRInference)`.
`pw`	A table defining the reference pathways.
`tg.genes`	Target gene list such as output by `tgGenes(BSRInference)`.
`tg.corr`	Target gene correlation list (with the receptor) such as output by `tgCorr(BSRInference)`.
`id.col`	Column index or name in `pw` for the pathway IDs.
`gene.col`	Column index or name in `pw` for the gene symbols.
`min.cor`	Minimum correlation required for the target genes.
`pos.targets`	A logical imposing that all the network targets must display positive correlation or logFC a BSRInferenceComp object is used to generate the network.
`neg.targets`	A logical imposing that all the network targets must display negative correlation or logFC a BSRInferenceComp object is used to generate the network. Correlations must be <= -min.cor or logFC <= - min.logFC with this option activated.
`tg.pval`	Target gene P-value list such as returned by `tgPval(BSRInferenceComp)`.
`max.pval`	Maximum (regulation) P-value for target genes in case a BSRInferenceComp object is used to generate the network.
`min.logFC`	Minimum logFC required for the target genes in case a BSRInferenceComp object is used to generate the network.

Value

An `igraph` object featuring the ligand-receptor-downstream signaling network. Default colors and node sizes are assigned. In case the `max.pval` parameter is set, it is assumed that `tg.pval` is set as well and downstream signaling genes are selected by their P-values in the comparison of clusters of samples.

`.formatPathwaysFromGmt`

Transform gmt file to data frame

Description

We note discrepancies between the formats available from internet sources. Here, we consider a valid gmt file format defined on each lines as follows: First is Pathway name, then comes the ID, finally you will find genes symbols part of the pathway defined on the line.

Usage

```
.formatPathwaysFromGmt(file, resourceName = NULL)
```

Arguments

`file`	Path to GMT file
`resourceName`	Two options "GO-BP" or "REACTOME"

Details

You can find an example here. - For Reactome. (Directly from their website) https://reactome.org/download/current/ReactomePathways.gmt.zip Note that you need to unzip the file to read the content. The code is inspired from read.gmt function from the gsa R package.

Value

Dataframe with pathwayID, geneName and pathwayName

`.formatPathwaysFromJson`

Format a data frame according to json input

Description

Format a data frame according to json input

Usage

```
.formatPathwaysFromJson(file, resourceName = NULL)
```

Arguments

file	Path to file.
resourceName	Two options "GO-BP" or "REACTOME".

Value

Data frame with pathwayID, geneName and pathwayName

.formatPathwaysFromTxt *Read dataframe from txt file*

Description

Read dataframe from txt file

Usage

```
.formatPathwaysFromTxt(file, resourceName = NULL)
```

Arguments

file	Path to a tabular file.
resourceName	Two options "GO-BP" "REACTOME".

Value

Dataframe with pathwayID, geneName and pathwayName

.geneNameConversion *Convert gene symbols to another organism*

Description

Convert gene symbols to another organism based on a dictionary with human and orthologs in the other species.

Usage

```
.geneNameConversion(genes, conversion.dict)
```

Arguments

genes	The genes you want to convert.
conversion.dict	A data frame containing gene names for the source species and Homo sapiens.

Value

Depend on the type of input genes LRinter return a vector of genes tgGenes receptors ligands: return list of list of genes

.getAlphaStableParam *Internal function to fit a censored (alpha-) stable distribution*

Description

Maximum-likelihood estimators are used.

Usage

```
.getAlphaStableParam(d, title, verbose = FALSE, file.name = NULL)
```

Arguments

d	A vector of values to fit.
title	A plot title.
verbose	Provide details on computations.
file.name	The file name of a PDF file.

Value

A list with the four stable distribution parameters (S0 representation).

If `file.name` is provided, a control plot is generated in a PDF with a data histogram and the fitted Gaussian. `title` is used to give this plot a main title.

.getCorrelatedLR *Get correlated ligand-receptor pairs*

Description

Internal function to compute the Spearman correlations of all the ligand-receptor pairs in LRdb and return those above a minimum value.

Usage

```
.getCorrelatedLR(ds, min.cor = 0.25, restrict.genes = NULL)
```

Arguments

ds	A BSRDataModel object.
min.cor	The minimum correlation required.
restrict.genes	A list of gene symbols that restricts ligands and receptors.

Details

The `restrict.genes` parameter is used for special cases where LRdb must be further restricted to a subset. The putative ligand-receptor pairs has 3 columns : R, L and corr.

Value

A data frame containing putative ligand-receptor pairs along with their correlations above `min.cor`. This table is the first step of a ligand-receptor analysis.

.getEmpiricalNull *Sampling of correlations downstream the receptors null distribution*

Description

Perform receptor downstream analysis with `.checkReceptorSignaling` based on randomized expression data and ligand-receptor pairs selected from the same randomized data.

Usage

```
.getEmpiricalNull(obj)
```

Arguments

`obj`	A BSRDatamodel without learned paramaters.

Details

A large number of correlations (ligand-receptor and receptor-downstream target genes) is reported in each randomized matrix. Therefore, `n.rand` should be given a modest value to avoid unnecessarily long computations.

See `.checkReceptorSignaling` for more details about the parameters.

Value

A list of `n.rand` tables such as output by `.checkReceptorSignaling`. Each table is computed from a randomized expression matrix (randomized `ncounts`).

.getEmpiricalNullCorrLR *Sampling of ligand-receptor correlation null distribution*

Description

Perform a ligand-receptor Spearman correlation analysis based on randomized expression data.

Usage

```
.getEmpiricalNullCorrLR(obj)
```

Arguments

`obj`	A BSRDatamodel without learned paramaters.

Details

A large number of correlations is reported in each randomized matrix. Therefore, `n.rand.LR` should be given a modest value to avoid unnecessarily long computations.

See `.getCorrelatedLR` for more details about the parameters.

Value

A list of `n.rand.LR` tables such as output by `.getCorrelatedLR`. Each table is computed from a randomized expression matrix (randomized `ncounts`).

`.getEmpiricalParam` *Internal function to fit an empirical distribution*

Description

Based on stats::ecdf.

Usage

```
.getEmpiricalParam(d, title, verbose = FALSE, file.name = NULL)
```

Arguments

`d`	A vector of values to fit.
`title`	A plot title.
`verbose`	Provide details on computations.
`file.name`	The file name of a PDF file.

Value

A list with the step function implementing the CDF of the empirical distribution (`empirCDF`).

If `file.name` is provided, a control plot is generated in a PDF with a data histogram and the fitted Gaussian. `title` is used to give this plot a main title.

`.getGaussianParam` *Internal function to fit a censored Gaussian distribution*

Description

Maximum-likelihood estimators are used.

Usage

```
.getGaussianParam(d, title, verbose = FALSE, file.name = NULL)
```

Arguments

`d`	A vector of values to fit.
`title`	A plot title.
`verbose`	Provide details on computations.
`file.name`	The file name of a PDF file.

Value

A list with the mean (`mu`) and standard deviation (`sigma`) estimates.

If `file.name` is provided, a control plot is generated in a PDF with a data histogram and the fitted Gaussian. `title` is used to give this plot a main title.

.getKernelEmpiricalParam

Internal function to fit a Gaussian kernel-based empirical distribution

Description

Based on stats::density.

Usage

```
.getKernelEmpiricalParam(d, title, verbose = FALSE, file.name = NULL, n = 512)
```

Arguments

d	A vector of values to fit.
title	A plot title.
verbose	Provide details on computations.
file.name	The file name of a PDF file.
n	The number of grid points for density FFT

Value

A list with the step function implementing the CDF of the empirical distribution (kernelCDF).

If file.name is provided, a control plot is generated in a PDF with a data histogram and the fitted Gaussian. title is used to give this plot a main title.

.getMixedGaussianParam

Internal function to fit a censored mixed-Gaussian distribution

Description

Maximum-likelihood estimators are used.

Usage

```
.getMixedGaussianParam(d, title, verbose = FALSE, file.name = NULL)
```

Arguments

d	A vector of values to fit.
title	A plot title.
verbose	Provide details on computations.
file.name	The file name of a PDF file.

Value

A list with the mean (`mu`) and standard deviation (`sigma`) estimates of each distribution along with the weight alpha applied to the first distribution.

If `file.name` is provided, a control plot is generated in a PDF with a data histogram and the fitted Gaussian. `title` is used to give this plot a main title.

`.getRegulatedLR` *Get regulated ligand-receptor pairs.*

Description

Internal function to return all the pairs of ligands and receptors having both a P-value below a given threshold.

Usage

```
.getRegulatedLR(
  ds,
  cc,
  scc = NULL,
  max.pval = 0.01,
  min.logFC = 1,
  neg.receptors = FALSE,
  restrict.genes = NULL
)
```

Arguments

`ds`	A BSRDataModel object
`cc`	A BSRClusterComp object.
`scc`	An optional BSRClusterComp object in case ligand regulation was assessed in a separate cluster comparison.
`max.pval`	The maximum P-value imposed to both the ligand and the receptor.
`min.logFC`	The maximum log2 fold-change allowed for both the receptor and the ligand.
`neg.receptors`	A logical indicating whether receptors are only allowed to be upregulated (FALSE), or up- and downregulated (TRUE).
`restrict.genes`	A list of gene symbols that restricts ligands and receptors.

Details

The `restrict.genes` parameter is used for special cases where LRdb must be further restricted to a subset. The putative ligand-receptor pairs has 6 columns : R, L, LR.pval, corr, L.logFC, and R.logFC. Note that correlations are currently set to 1 to avoid lengthy computations with scRNA-seq data and multiple cell populations.

Value

A data frame containing putative ligand-receptor pairs along with the product of their respective P-values. This table is the first step of a ligand-receptor analysis.

.pValuesLR *Internal function to assign P-values to LR interactions*

Description

Estimate the P-value of each ligand-receptor pair based on the data frame output by `.checkReceptorSignaling`.

Usage

```
.pValuesLR(
  pairs,
  param,
  rank.p = 0.75,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

`pairs`	A data frame output by `checkReceptorSignaling`.
`param`	A list containing the statistical model parameters.
`rank.p`	A number between 0 and 1 defining the rank of the last considered target genes.
`fdr.proc`	The procedure for adjusting P-values according to `mt.rawp2adjp`.

Value

A data.frame with the data in `pairs` complemented with P-values and adjusted P-values.

.pValuesRegulatedLR *Internal function to assign P-values to LR interactions*

Description

Estimate the P-value of each ligand-receptor pair based on the data frame output by `.checkRegulatedReceptorSignali`

Usage

```
.pValuesRegulatedLR(
  pairs,
  param,
  rank.p = 0.75,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

pairs	A data frame output by checkRegulatedReceptorSignaling.
param	A list containing the statistical model parameters.
rank.p	A number between 0 and 1 defining the rank of the last considered target genes.
fdr.proc	The procedure for adjusting P-values according to mt.rawp2adjp.

Value

A data.frame with the data in pairs complemented with P-values and adjusted P-values.

.readRDSFromCache *Read RDS from the cache.*

Description

Access resources (pathways or network stored in the cache.

Usage

```
.readRDSFromCache(bfc, resourceName)
```

Arguments

bfc	Object of class BiocFileCache, created by a call to BiocFileCache::BiocFileCache()
resourceName	keyword associated to a specific resourceName

Value

Returns BiocFileCache::bfcquery object or FALSE

.shufflePermutationIndices *Internal function to shuffle permutation indices*

Description

Internal function to shuffle permutation indices

Usage

```
.shufflePermutationIndices(pind)
```

Arguments

pind	Permutation indices such as returned by .buildPermutationIndices.

Value

A list with same structure as pind with shuffled indices within each bin.

.testCacheFiles *Check there is a well formatted cache*

Description

Check there is a well formatted cache

Usage

```
.testCacheFiles()
```

Value

Returns ‘NULL‘, invisibly.

.testRemoteServer *Check whether the reference DB server is up*

Description

Check whether the reference DB server is up

Usage

```
.testRemoteServer()
```

Value

Returns ‘NULL‘, invisibly.

addClusterComp *Add a comparison between two clusters of samples*

Description

Add a comparison to a BSRDataModelComp object.

Usage

```
## S4 method for signature 'BSRDataModelComp'
addClusterComp(obj, cmp, cmp.name)
```

Arguments

obj	A BSRDataModelComp object output by setAs.
cmp	A BSRClusterComp object to add.
cmp.name	The name of the comparison to add.

Details

Add cmp to the list of comparisons contained in obj.

Value

A BSRDataModelComp object.

Examples

```
# prepare data
data(sdc, package = "BulkSignalR")
normal <- grep("^N", names(sdc))
bsrdm <- BSRDataModel(sdc[, -normal])

# define the comparison
bsrdm.comp <- as(bsrdm, "BSRDataModelComp")
colA <- as.integer(1:3)
colB <- as.integer(12:15)
n <- nrow(ncounts(bsrdm.comp))
stats <- data.frame(
    pval = runif(n), logFC = rnorm(n, 0, 2),
    expr = runif(n, 0, 10)
)
rownames(stats) <- rownames(ncounts(bsrdm.comp))
bsrcc <- BSRClusterComp(bsrdm.comp, colA, colB, stats)

bsrdm.comp <- addClusterComp(bsrdm.comp, bsrcc, "random.example")
```

---

alluvialPlot *Alluvial plot*

Description

Representation of the links between ligands, receptors, and pathways.

Usage

```
alluvialPlot(bsrinf, keywords, type = c("L", "R", "pw.id"), qval.thres = 0.01)
```

Arguments

bsrinf	A BSRInference object.
keywords	vector of keywoprds to filter pathways.
type	filter on Ligand, Receptor or pathway id.
qval.thres	threshold over Q-value.

Value

NULL

This is a convenience function that relies on the ggalluvial package to propose a simple way of representing ligands, receptors,

Examples

```
data(bsrinf, package = "BulkSignalR")
alluvialPlot(bsrinf,
    keywords = c("LAMC1"),
    type = "L",
    qval.thres = 0.01)
```

`annotation.spa` *A skinny data frame used in the spatial workflow*

Description

Data frame subset describing the spatial spots

Usage

```
data(annotation.spa)
```

Format

Data frame that contains the following columns: barcode_id,sample_id, in_tissue,array_row array_col, ground_truth, reference, cell_count, idSpatial

barcode_id is the id of the spot idSpatial is the spatial id of the spot(array_rowXarray_col) ground_truth is the label (Layer1/2 were only kept)

They are the mandatory data necessary to generate plots for the spatial workflow.

Source

http://spatial.libd.org/spatialLIBD/

`assignCellTypesToInteractions` *Assign cell types to L-R interactions*

Description

Generate a data.frame linking interactions to cell types.

Usage

```
assignCellTypesToInteractions(
  bsrdm,
  bsrinf,
  ct.scores,
  normalize.scores = TRUE,
  min.weight = 0.1,
  min.r2 = 0.25,
  min.r2.after = 0.35,
  lasso = TRUE,
  qval.thres = 0.001
)
```

Arguments

bsrdm	A BSRDataModel object.
bsrinf	A BSRInference object.
ct.scores	A matrix of cell type signature scores.
normalize.scores	A logical indicating whether scores should be normalized before assigning cell types.
min.weight	Minimum weight to keep in the linear model (cell types with lower weights will be discarded) if lasso==TRUE. Otherwise, minimum correlation coefficient of each individual cell type.
min.r2	Minimum r2 between a candidate cell type and a L-R gene signature score.
min.r2.after	Minimum r2 between the proposed linear model and a L-R gene signature score to retain the model.
lasso	Logical indicating that the LASSO (or linear regression if only one cell type satisfies the min.r2 criterion) should be used. Otherwise, Spearman linear correlation is used.
qval.thres	Maximum Q-value of the L-R pairs to be considered.

Value

A data.frame containing the cell type assignments for each L-R interaction. Unique interactions are considered only (thanks to "reduceToBestPathway" that is applied internally). An interaction can be associated with several cell types or none. In case it is associated with a single cell type, it is labelled autocrine (indicative only).

Cell type signature scores must be provided. They can be computed with BulkSignalR utility function "scoreSignatures", but also any other external tool such as CIBERSORT or BisqueRNA. In case such a tool would score cell types in a nonlinear fashion, we recommend to transform the score matrix to restore a linear relationship cell type abundance/score. By default, cell type (and L-R gene signature) scores are normalized between 0 and 1 to make the weights of each cel type in the linear models as comparable as possible.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")
data(tme.signatures, package = "BulkSignalR")

immune.signatures <- immune.signatures[immune.signatures$signature %in%
    c("T cells"), ]

signatures <- rbind(immune.signatures, tme.signatures[
    tme.signatures$signature %in% c("Fibroblasts"),
])

tme.scores <- scoreSignatures(bsrdm, signatures)

# assignment
lr2ct <- assignCellTypesToInteractions(bsrdm, bsrinf, tme.scores)
```

bodyMap.mouse *Mouse transcriptoms across tissues*

Description

A data set containing RPKM values of brain and liver.

Usage

```
data(bodyMap.mouse)
```

Format

A data frame with 24543 rows and 8 variables.

Source

Bin Li & al., Scientific Reports, 2017;

BSRClusterComp *Definition of the comparison between two clusters of samples*

Description

Define the columns of the expression matrix that belong to each cluster, and store the result of the cluster differences statistical analysis obtained by an external tool such as edgeR or DESeq2 in a dedicated data frame.

Usage

```
BSRClusterComp(obj, col.clusterA, col.clusterB, differential.stats)
```

Arguments

`obj`	A BSRDataModelComp object.
`col.clusterA`	Cluster A column indices.
`col.clusterB`	Cluster B column indices.
`differential.stats`	A data.frame containing statistics about the differential analysis cluster A versus B. `differentialStats` must contain at least the columns 'pval' (for P-values), 'logFC' for log-fold-changes A/B, and 'expr' for the expression of the genes in cluster A.

Details

Create a BSRClusterComp object describing a comparison of two clusters of columns taken from the expression matrix in the BSRDataModelComp object `obj`. Such a cluster comparison description is the basis for inferring LRIs from differential expression P-values instead of correlation analysis.

The rows of `differentialStats` must be in the same order as those of the count matrix in `obj`. Alternatively, `differentialStats` rows can be named and a 1-1 correspondence must exist between these names and those of the count matrix.

Value

A BSRClusterComp object.

Examples

```
# prepare data
data(sdc, package = "BulkSignalR")
normal <- grep("^N", names(sdc))
bsrdm <- BSRDataModel(sdc[, -normal])

# define the comparison
bsrdm.comp <- as(bsrdm, "BSRDataModelComp")
colA <- as.integer(1:3)
colB <- as.integer(12:15)
n <- nrow(ncounts(bsrdm.comp))
stats <- data.frame(
    pval = runif(n), logFC = rnorm(n, 0, 2),
    expr = runif(n, 0, 10)
)
rownames(stats) <- rownames(ncounts(bsrdm.comp))
bsrcc <- BSRClusterComp(bsrdm.comp, colA, colB, stats)
```

BSRClusterComp-class *BulkSignalR Cluster Comparison Object*

Description

An S4 class to represent the comparison of two clusters of samples to infer LR interactions based on the resulting P-values, log-fold-changes (logFC), and expression values.

Slots

`col.clusterA` Column indices for the samples in cluster A.

`col.clusterB` Column indices for the samples in cluster B.

`differential.stats` Comparison statistics A versus B as a data.frame and containing at least 3 columns named 'pval', 'logFC', and 'expr'.

Examples

```
new("BSRClusterComp")
```

`BSRDataModel` *Constructor of the BSRDataModel class*

Description

Take a matrix or data frame containing RNA sequencing, microarray, or expression proteomics data as input parameter and return a BSRDataModel object ready for subsequent training.

Usage

```
BSRDataModel(
  counts,
  normalize = TRUE,
  symbol.col = NULL,
  min.count = 10,
  prop = 0.1,
  method = c("UQ", "TC"),
  log.transformed = FALSE,
  min.LR.found = 80,
  species = "hsapiens",
  conversion.dict = NULL,
  UQ.pc = 0.75,
  x.col = NULL,
  y.col = NULL,
  barcodeID.col = NULL
)
```

Arguments

`counts`	A table or matrix of read counts.
`normalize`	A logical indicating whether `counts` should be normalized according to `method` or if it was normalized beforehand.
`symbol.col`	The index of the column containing the gene symbols in case those are not the row names of `counts` already.
`min.count`	The minimum read count of a gene to be considered expressed in a sample.
`prop`	The minimum proportion of samples where a gene must be expressed higher than `min.count` to keep that gene.
`method`	The normalization method ('UQ' for upper quartile or 'TC' for total count). If `normalize==FALSE`, then method must be used to document the name of the normalization method applied by the user.
`log.transformed`	A logical indicating whether expression data were already log2-transformed, e.g., some microarray data.
`min.LR.found`	The minimum number of ligands or receptors found in `count` row names after eliminating the rows containing too many zeros according to `min.count` and `prop`.
`species`	Data were obtained for this organism.

`conversion.dict`
: Correspondence table of HUGO gene symbols human/nonhuman. Not used unless the organism is different from human.

`UQ.pc`
: Percentile for upper-quartile normalization, number between 0 and 1 (in case the default 0.75 - hence the name - is not appropriate).

`x.col`
: In a SpatialExperiment object, the index of the column containing the x coordinates in the dafaframe returned by rowData(), usually named array_row.

`y.col`
: In a SpatialExperiment object, the index of the column containing the y coordinates in the dafaframe returned by rowData(), usually named array_col.

`barcodeID.col`
: In a SpatialExperiment object, the index of the column containing the barcodeID in the dafaframe returned by colData(), usually named barcode_id.

Details

Note that this constructor replaces the function prepareDataset that was part of the previous version of BulkSignalR library.

The `counts` matrix or table should be provided with expression levels of protein coding genes in each samples (column) and `rownames(counts)` set to HUGO official gene symbols. For commodity, it is also possible to provide `counts` with the gene symbols stored in one of its columns. This column must be specified with `symbol.col`. In such a case, `BSRDataModel` will extract this column and use it to set the row names. Because row names must be unique, `BSRDataModel` will eliminate rows with duplicated gene symbols by keeping the rows with maximum average expression. Gene symbol duplication may occur in protein coding genes after genome alignment due to errors in genome feature annotation files (GTF/GFF), where a handful of deprecated gene annotations might remain, or some genes are not given their fully specific symbols. If your read count extraction pipeline does not take care of this phenomenon, the maximum mean expression selection strategy implemented here should solve this difficulty for the sake of inferring ligand-receptor interactions.

If `normalize` is `TRUE` then normalization is performed according to `method`. If those two simple methods are not satisfying, then it is possible to provide a pre-normalized matrix setting `normalize` to `FALSE`. In such a case, the parameter `method` must be used to document the name of the normalization algorithm used.

In case proteomic or microarray data are provided, `min.count` must be understood as its equivalent with respect to those data types.

Value

A BSRModelData object with empty model parameters.

Examples

```
data(sdc, package = "BulkSignalR")
idx <- sample(nrow(sdc), 4000)
bsrdm <- BSRDataModel(sdc[idx, c("N22","SDC17")],
normalize = FALSE,method="UQ")
```

BSRDataModel-class *BulkSignalR Data Model Object*

Description

An S4 class to represent the expression data used for inferring ligand-receptor interactions.

Slots

`ncounts` Normalized read count matrix. Row names must be set to HUGO official gene symbols.

`log.transformed` Logical indicating whether values in `ncounts` were log2-transformed.

`normalization` Name of the normalization method.

`param` List containing the statistical model parameters.

`initial.organism` Organism for which the data were obtained.

`initial.orthologs` List of genes for which human orthologs exist.

Examples

```
BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
```

BSRDataModelComp *Constructor of the BSRDataModelComp class*

Description

Take a matrix or data frame containing RNA sequencing, microarray, or expression proteomics data as input parameter and return a BSRDataModelComp object ready for subsequent use.

Usage

```
BSRDataModelComp(
  counts,
  normalize = TRUE,
  symbol.col = NULL,
  min.count = 10,
  prop = 0.1,
  method = c("UQ", "TC"),
  log.transformed = FALSE,
  min.LR.found = 80,
```

```
  species = "hsapiens",
  conversion.dict = NULL,
  UQ.pc = 0.75,
  x.col = NULL,
  y.col = NULL,
  barcodeID.col = NULL
)
```

Arguments

counts	A table or matrix of read counts.
normalize	A logical indicating whether counts should be normalized according to method or if it was normalized beforehand.
symbol.col	The index of the column containing the gene symbols in case those are not the row names of counts already.
min.count	The minimum read count of a gene to be considered expressed in a sample.
prop	The minimum proportion of samples where a gene must be expressed higher than min.count to keep that gene.
method	The normalization method ('UQ' for upper quartile or 'TC' for total count). If normalize==FALSE, then method must be used to document the name of the normalization method applied by the user.
log.transformed	A logical indicating whether expression data were already log2-transformed, e.g., some microarray data.
min.LR.found	The minimum number of ligands or receptors found in count row names after eliminating the rows containing too many zeros according to min.count and prop.
species	Data were obtained for this organism.
conversion.dict	Correspondence table of HUGO gene symbols human/nonhuman. Not used unless the organism is different from human.
UQ.pc	Percentile for upper-quartile normalization, number between 0 and 1 (in case the default 0.75 - hence the name - is not appropriate).
x.col	In a SpatialExperiment object, the index of the column containing the x coordinates in the dafaframe returned by rowData(), usually named array_row.
y.col	In a SpatialExperiment object, the index of the column containing the y coordinates in the dafaframe returned by rowData(), usually named array_col.
barcodeID.col	In a SpatialExperiment object, the index of the column containing the barcodeID in the dafaframe returned by colData(), usually named barcode_id.

Details

The counts matrix or table should be provided with expression levels of protein coding genes in each samples (column) and rownames(counts) set to HUGO official gene symbols. For commodity, it is also possible to provide counts with the gene symbols stored in one of its columns. This column must be specified with symbol.col. In such a case, BSRDataModel will extract this column and use it to set the row names. Because row names must be unique, BSRDataModel will eliminate rows with duplicated gene symbols by keeping the rows with maximum average expression. Gene symbol duplication may occur in protein coding genes after genome alignment due to errors in

genome feature annotation files (GTF/GFF), where a handful of deprecated gene annotations might remain, or some genes are not given their fully specific symbols. If your read count extraction pipeline does not take care of this phenomenon, the maximum mean expression selection strategy implemented here should solve this difficulty for the sake of inferring ligand-receptor interactions.

If `normalize` is `TRUE` then normalization is performed according to `method`. If those two simple methods are not satisfying, then it is possible to provide a pre-normalized matrix setting `normalize` to `FALSE`. In such a case, the parameter `method` must be used to document the name of the normalization algorithm used.

In case proteomic or microarray data are provided, `min.count` must be understood as its equivalent with respect to those data types.

Value

A BSRModelDataComp object with empty model parameters.

Examples

```
data(sdc, package = "BulkSignalR")
idx <- sample(nrow(sdc), 4000)
bsrdm.comp <- BSRDataModelComp(sdc[idx, c("N22","SDC17")],
                               normalize=FALSE, method="UQ")
```

BSRDataModelComp-class

BulkSignalR Data Model Compare Object

Description

An S4 class to represent the expression data used for inferring ligand-receptor interactions based on sample cluster comparisons.

Slots

`comp` A named list of BSRClusterComp objects, one per comparison.

`mu` A number representing the average value in the normalized and lop1p-transformed gene expression matrix. This value is used to compute the LR-score (cf. SingleCellSignalR paper, Cabello-Aguilar, et al., Nucleic Acids Res, 2020)

Examples

```
new("BSRDataModelComp")
```

bsrdm *A skinny BSRDataModel object related to sdc.*

Description

Output from the 'learnParameters' function to get BulkSignalR statistical model parameters.

Usage

```
data(bsrdm)
```

Format

An example of an object created by 'BSRDataModel' applied to an sdc subset (Patients N20,N22,SDC17,SDC25) and 10 000 genes sampled (seed set to 123) 'learnParameters' was also called to get statistical model parameters.

bsrdm.comp *A skinny BSRDataModelComp object related to sdc.*

Description

See Vignette BulkSignalR-Differential.

Usage

```
data(bsrdm.comp)
```

Format

An example of an BSRDataModelComp object

bsrdm.spa *A skinny BSRDataModel object related to a spatial data set*

Description

Obtained from STexampleData::Visium_humanDLPFC. A single sample (sample 151673) of human brain dorsolateral prefrontal cortex (DLPFC) in the human brain, measured using the 10x Genomics Visium platform. This is a subset of the full dataset published by Maynard and Collado-Torres et al. (2021). The subset is reproduced in the vignette. name.idx <- c("10x32","3x47","4x50", "17x111","5x59","0x20","8x100", "8x108","14x30","11x39")

Usage

```
data(bsrdm.spa)
```

Format

An example of an object created by 'BSRDataModel' applied to a subset of a spatial data set. 'learnParameters' was also called to get statistical model parameters.

Details

Output from the 'learnParameters' function to get BulkSignalR statistical model parameters for a subset of a spatial data set.

Source

http://spatial.libd.org/spatialLIBD/

---

bsrinf *A skinny BSRInference object related to sdc.*

Description

From the previous object 'bsrdm', you can generate inferences by calling its constructor 'BSRInference'. The resulting BSRInference object is 'bsrinf', It contains all the inferred L-R interactions with their associated pathways and corrected p-values.

Usage

```
data(bsrinf)
```

Format

An example of an object created by inference function

---

bsrinf.comp *A skinny BSRInferenceComp object related to sdc.*

Description

See Vignette BulkSignalR-Differential.

Usage

```
data(bsrinf.comp)
```

Format

An example of an BSRInferenceComp object

bsrinf.mouse *A skinny BSRInference object related to bodyMap.mouse*

Description

see related workflow for non human organism in the vignette

Usage

```
data(bsrinf.mouse)
```

Format

An example of an object created by inference function

bsrinf.spa *A skinny BSR-inference object related to spatial data set*

Description

Output from the ‘learnParameters‘ function to get BulkSignalR statistical model parameters.

Usage

```
data(bsrinf.spa)
```

Format

From the previous object ‘bsrdm.spa‘, you can generate inferences by calling its method ‘BSRInference‘. The resulting BSRInference object is ‘bsrinf.spa‘, It contains all the inferred L-R interactions with their associated pathways and corrected p-values. ‘learnParameters‘ was also called to train the statistical model parameters.

Source

http://spatial.libd.org/spatialLIBD/

BSRInference *Inference of ligand-receptor interactions*

Description

Computes putative LR interactions along with their statistical confidence. In this initial inference, all the relevant pathways are reported, see reduction functions to reduce this list.

Usage

```
BSRInference(
  obj,
  rank.p = 0.55,
  min.cor = 0.25,
  restrict.genes = NULL,
  reference = c("REACTOME-GOBP", "REACTOME", "GOBP"),
  max.pw.size = NULL,
  min.pw.size = NULL,
  min.positive = NULL,
  use.full.network = FALSE,
  restrict.pw = NULL,
  with.complex = NULL,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

obj	A BSRDataModel output by BSRDataModel with statistical model parameters trained by the "learnParameters" method.
rank.p	A number between 0 and 1 defining the rank of the last considered target genes.
min.cor	The minimum Spearman correlation required between the ligand and the receptor.
restrict.genes	A list of gene symbols that restricts ligands and receptors.
reference	Which pathway reference should be used ("REACTOME" for Reactome, "GOBP" for GO Biological Process, or "REACTOME-GOBP" for both).
max.pw.size	Maximum pathway size to consider from the pathway reference.
min.pw.size	Minimum pathway size to consider from the pathway reference.
min.positive	Minimum number of target genes to be found in a given pathway.
use.full.network	A logical to avoid limiting the reference network to the detected genes and use the whole reference network.
restrict.pw	A list of pathway IDs to restrict the application of the function.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.
fdr.proc	The procedure for adjusting P-values according to mt.rawp2adjp.

Details

Perform the initial ligand-receptor inference. Initial means that no reduction is applied. All the (ligand, receptor, downstream pathway) triples are reported, i.e., a given LR pair may appear multiple times with different pathways downstream the receptor. Specific reduction functions are available from the package to operate subsequent simplifications based on the BSRInference object created by the initial inference.

Parameters defining minimum/maximum pathway sizes, etc. are set to NULL by default, meaning that their values will be taken from what was set during the training of the statistical model with "learnParameters"

To use different values at the time of inference sounds like a bad idea, although this could be used to explore without retraining the underlying model. Retraining of the model with adjusted parameters is advised following such an exploration.

Value

A BSRInference object with initial inferences set.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")

# We use a subset of the reference to speed up
# inference in the context of the example.

reactSubset <- getResource(resourceName = "Reactome",
cache = FALSE)

subset <- c("REACTOME_BASIGIN_INTERACTIONS",
"REACTOME_SYNDECAN_INTERACTIONS",
"REACTOME_ECM_PROTEOGLYCANS",
"REACTOME_CELL_JUNCTION_ORGANIZATION")

reactSubset <- reactSubset[
reactSubset$`Reactome name` %in% subset,]

resetPathways(dataframe = reactSubset,
resourceName = "Reactome")

bsrinf <- BSRInference(bsrdm,
    min.cor = 0.2,restrict.genes=immune.signatures$gene,
    reference="REACTOME")
```

BSRInference-class *BulkSignalR Inference Object*

Description

An S4 class to represent inferred ligand-receptor interactions.

Details

This class contains inferred LR interactions along with their statistical significance. Data representation supports subsequent reductions to pathways, etc. See reduction functions "reduceToBestPathway", "reduceToLigand", "reduceToReceptor" and "reduceToPathway".

Slots

`LRinter` A data frame describing the (ligand,receptor,pathway) triples with P- and Q-values.

`ligands` A list of ligands, one entry per LR interaction.

`receptors` A list of receptors, one entry per LR interaction.

`tg.genes` A list of target genes, one entry per LR interaction.

`tg.corr` A list of target gene correlations to the receptor, one entry per interaction

`inf.param` The parameters used for the inference.

Examples

```
new("BSRInference")
```

`BSRInferenceComp` *Inference of ligand-receptor interactions based on regulation*

Description

This method supports two configurations that we refer to as paracrine and autocrine.

Usage

```
BSRInferenceComp(
  obj,
  cmp.name,
  src.cmp.name = NULL,
  rank.p = 0.55,
  max.pval = 0.01,
  min.logFC = 1,
  neg.receptors = FALSE,
  pos.targets = FALSE,
  neg.targets = FALSE,
  min.t.logFC = 0.5,
  restrict.genes = NULL,
  use.full.network = FALSE,
  reference = c("REACTOME-GOBP", "REACTOME", "GOBP"),
  max.pw.size = 400,
  min.pw.size = 5,
  min.positive = 2,
  restrict.pw = NULL,
  with.complex = TRUE,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

obj	A BSRDataModelComp object.
cmp.name	The name of the cluster comparison that should be used for the inference. Autocrine interactions if only this comparison name is provided, paracrine if a source comparison name is provided as well.
src.cmp.name	The name of the source cluster comparison that should be used for paracrine interaction inferences.
rank.p	A number between 0 and 1 defining the rank of the last considered target genes.
max.pval	The maximum P-value imposed to both the ligand and the receptor.
min.logFC	The minimum log2 fold-change allowed for both the receptor and the ligand.
neg.receptors	A logical indicating whether receptors are only allowed to be upregulated (FALSE), or up- and downregulated (TRUE).
pos.targets	A logical imposing that all the network targets must display positive logFC, i.e. logFC >= min.t.logFC.
neg.targets	A logical imposing that all the network targets must display negative logFC, i.e. logFC <= - min.t.logFC.
min.t.logFC	The minimum log2 fold-change allowed for targets in case pos.targets or neg.targets are used.
restrict.genes	A list of gene symbols that restricts ligands and receptors.
use.full.network	A logical to avoid limiting the reference network to the detected genes and use the whole reference network.
reference	Which pathway reference should be used ("REACTOME" for Reactome, "GOBP" for GO Biological Process, or "REACTOME-GOBP" for both).
max.pw.size	Maximum pathway size to consider from the pathway reference.
min.pw.size	Minimum pathway size to consider from the pathway reference.
min.positive	Minimum number of target genes to be found in a given pathway.
restrict.pw	A list of pathway IDs to restrict the application of the function.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.
fdr.proc	The procedure for adjusting P-values according to mt.rawp2adjp.

Details

In the autocrine case, a single cluster comparison name is provided. In the corresponding cluster comparison, a group of samples A was compared to a group of samples B to determine fold-changes and associated P-values. The inferred ligand-receptor interactions take place in the samples of group A. A typical single-cell example would be a population of macrophages (group A) compared to all the other populations (group B) to represent specific increased or decreased expression in macrophages. The resulting ligand-receptor interactions will be autocrine interactions that are exacerbated (or reduced depending on the chosen parameters) in macrophages.

In the paracrine case, two cluster comparison names must be provided. For instance, a first comparison could involve macrophages versus all the other cell populations as above. The second comparison could be B-cells against all the other populations. Now, calling BSRInferenceComp() with comparison macrophages *versus* the rest and, as source comparison, B-cells *versus* the rest, will result in inferring interactions between B-cells (ligands) and macrophages (receptors and

downstream pathways). To obtain macrophages to B-cells paracrine interactions, it is necessary to call the method a second time with permuted cluster comparison names. Another example in spatial transcriptomics could be two thin bands at the boundary of two tissue regions, one emitting the ligand and the other one expressing the receptor.

In this initial inference, all the receptor-containing pathways are reported, see reduction functions to reduce this list.

Perform the initial ligand-receptor inference. Initial means that no reduction is applied. All the (ligand, receptor, downstream pathway) triples are reported, i.e., a given LR pair may appear multiple times with different pathways downstream the receptor. Specific reduction functions are available from the package to operate subsequent simplifications based on the BSRInferenceComp object created by this method.

Here, ligand-receptor interactions are inferred based on gene or protein regulation-associated P-values when comparing two clusters of samples. Since a BSRDataModelComp object can contain several such comparisons, the name of the comparison to use must be specified (parameter `cmp.name`).

Note that since the introduction of the use.full.network parameter (April 29, 2024), the pathway sizes are always computed before potential intersection with the observed data (use.full.network set to FALSE) for consistency. Accordingly, the minimum and maximum pathway default values have been raised from 5 & 200 to 5 & 400 respectively. By default, use.full.network is set to FALSE.

In addition to statistical significance estimated according to BulkSignalR statistical model, we compute SingleCellSignalR original LR-score, based on L and R cluster average expression. In the paracrine case, L average expression is taken from the source cluster.

Value

A BSRInferenceComp object with initial inferences set.

Examples

```
data(bsrdm.comp, package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")

# infer ligand-receptor interactions from the comparison
bsrinf.comp <- BSRInferenceComp(bsrdm.comp, max.pval = 1,
reference="REACTOME",
"random.example")
```

BSRInferenceComp-class

BulkSignalR cluster comparison-based inference object

Description

An S4 class to represent ligand-receptor interactions inferred from a comparison between two clusters of samples. This class inherits from BSRInference.

Details

This class is contains inferred LR interactions along with their statistical significance. Data representation supports subsequent reductions to pathways, etc. See reduction functions `"reduceToBestPathway"`, `"reduceToLigand"`, `"reduceToReceptor"` and `"reduceToPathway"`.

Slots

`cmp.name` The name of the BSRClusterComp object in a BSRDataModelComp object comp list.

`src.cmp.name` The name of an optional BSRClusterComp object in a BSRDataModelComp object comp list in case paracrine inferences were performed.

`tg.pval` A list of target gene P-values, one entry per interaction

`tg.logFC` A list of target gene logFC, one entry per interaction

`tg.expr` A list of target gene expression, one entry per interaction

Examples

```
new("BSRInferenceComp")
```

BSRSignature *Extract gene signatures of LR pair activity*

Description

Obtain gene signatures reflecting ligand-receptor as well as receptor downstream activity to score ligand-receptor pairs across samples subsequently with "scoreLRGeneSignatures"

Usage

```
BSRSignature(obj, pval.thres = NULL, qval.thres = NULL, with.pw.id = FALSE)
```

Arguments

`obj`	BSRinference object.
`pval.thres`	P-value threshold.
`qval.thres`	Q-value threshold.
`with.pw.id`	A logical indicating whether the ID of a pathway should be concatenated to its name.

Value

A BSRSignature object containing a gene signature for each triple ligand-receptor pair. A reduction to the best pathway for each pair is automatically performed and the gene signature is comprised of the ligand, the receptor, and all the target genes with rank equal or superior to `pairs$rank`.

Examples

```
data(bsrinf, package = "BulkSignalR")

bsrinf.redP <- reduceToPathway(bsrinf)
bsrsig.redP <- BSRSignature(bsrinf, qval.thres = 0.001)
```

BSRSignature-class *BulkSignalR ligand-receptor signature Object*

Description

S4 class to represent gene signatures of inferred ligand-receptor interactions, including their reduced versions.

Slots

`ligands` A list of ligands, one entry per LR interaction.

`receptors` A list of receptors, one entry per LR interaction.

`tg.genes` A list of target genes, one entry per LR interaction.

`pathways` An atomic vector of pathway names, one per interaction.

`tg.corr` A list of target genes correlation.

Examples

```
new("BSRSignature")
```

BSRSignatureComp *Extract gene signatures of LR pair activity*

Description

Obtains gene signatures reflecting ligand-receptor as well as receptor downstream activity to score ligand-receptor pairs across samples subsequently with "scoreLRGeneSignatures"

Usage

```
BSRSignatureComp(obj, pval.thres = NULL, qval.thres = NULL, with.pw.id = FALSE)
```

Arguments

`obj`	BSRInferenceComp object.
`pval.thres`	P-value threshold.
`qval.thres`	Q-value threshold.
`with.pw.id`	A logical indicating whether the ID of a pathway should be concatenated to its name.

Value

A BSRSignatureComp object containing a gene signature for each triple ligand-receptor pair. A reduction to the best pathway for each pair is automatically performed and the gene signature is comprised of the ligand, the receptor, and all the target genes with rank equal or superior to `pairs$rank`.

Examples

```
data(bsrinf.comp, package = "BulkSignalR")

bsrinf.redP <- reduceToPathway(bsrinf.comp)
bsrsig.redP <- BSRSignatureComp(bsrinf.redP, qval.thres = 0.001)
```

BSRSignatureComp-class *BulkSignalR ligand-receptor signature object for cluster comparisons*

Description

S4 class to represent gene signatures associated with ligand-receptor interactions that were inferred from the comparison of two clusters of samples. This class inherits from BSRSignature.

Slots

`cmp.name` The name of the comparison.

`tg.pval` A list of target genes P-values.

`tg.logFC` A list of target genes logFC.

`tg.expr` A list of target genes expression

Examples

```
new("BSRSignatureComp")
```

---

bubblePlotPathwaysLR *Bubble Plot to explore LR & Pathways*

Description

Quick check to observe LR - Pathways association with their respective correlation and Q-values.

Usage

```
bubblePlotPathwaysLR(
  bsrinf,
  pathways,
  qval.thres = 1,
  filter.L = NULL,
  filter.R = NULL,
  color = "#16a647",
  pointsize = 6
)
```

Arguments

bsrinf	BulkSignalR inference object.
pathways	Vector of pathway names to keep.
qval.thres	Maximum Q-value.
filter.L	Vector of ligands to keep.
filter.R	Vector of receptors to keep.
color	Main color used for the gradient.
pointsize	Global point size.

Value

A bubble plot displayed in the current viewport.

This is a convenience function to propose a simple way of representing LR - Pathways association with their respective correlation and Q-values.

Examples

```
data(bsrinf, package = "BulkSignalR")
pathways <- LRinter(bsrinf)[1,c("pw.name")]
bubblePlotPathwaysLR(bsrinf,
pathways = pathways,
qval.thres = 0.1,
color = "red",
pointsize = 8
)
```

---

cacheClear *Delete cache content.*

Description

Delete the content of cache directory.

Usage

```
cacheClear(dir = c("resources"))
```

Arguments

dir	Directory to remove. Can be only 'resources'.

Value

Returns 'NULL', invisibly.

Examples

```
cacheClear(dir="resources")
# need to recreate database in order to run examples well
createResources(verbose=TRUE)
```

cacheInfo *Get cache content information.*

Description

Get cache content information for a specific cache directory.

Usage

```
cacheInfo(dir = c("resources"))
```

Arguments

dir	Directory to remove in order to clean the cache. Can be only 'resources'

Value

Returns 'NULL', invisibly.

Examples

```
cacheInfo()
```

cacheVersion *Check whether remote resource files have been changed.*

Description

Check to see whether some resource has been updated.

Usage

```
cacheVersion(dir = c("resources"))
```

Arguments

dir	Directory for which you want to check Version. Can be only 'resources'.

Value

Returns 'NULL', invisibly.

Examples

```
cacheVersion()
```

`cellTypeFrequency` *Cell type frequencies in relations to gene sets*

Description

Count how many times and with which weights cell types were involved in the (L,R,pathway) triples that targeted genes in a gene set.

Usage

```
cellTypeFrequency(rel, lr, min.n.genes = 1)
```

Arguments

`rel`	The data.frame output by "`relateToGeneSet`".
`lr`	The data.frame output by "`assignCellTypesToInteractions`".
`min.n.genes`	Minimum number of genes in the gene set for one (L,R,pathway) triple.

Value

A list of two slots: t for counting how many times each cell type is involved; s for summing the weights of each involved cell type.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")
data(tme.signatures, package = "BulkSignalR")
data(p.EMT, package = "BulkSignalR")

immune.signatures <- immune.signatures[immune.signatures$signature %in%
    c("T cells"), ]

signatures <- rbind(immune.signatures, tme.signatures[
    tme.signatures$signature %in% c("Fibroblasts"),
])

tme.scores <- scoreSignatures(bsrdm, signatures)

# assignment
lr2ct <- assignCellTypesToInteractions(bsrdm, bsrinf, tme.scores)

# relate to p-EMT (should be done in HNSCC normally, not in SDC)
p.EMT <- p.EMT$gene
triggers <- relateToGeneSet(bsrinf, p.EMT)

# counts
cf <- cellTypeFrequency(triggers, lr2ct)
```

cellularNetwork *Build a cellular network*

Description

Generate a igraph object including all the links between cell types.

Usage

```
cellularNetwork(tab)
```

Arguments

tab	The data.frame output by "cellularNetworkTable".

Value

A igraph object containing all the links in the cellular network.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
data("tme.signatures", package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")

immune.signatures <- immune.signatures[immune.signatures$signature %in%
    c("T cells"), ]

signatures <- rbind(immune.signatures, tme.signatures[
    tme.signatures$signature %in% c("Fibroblasts"),
])

tme.scores <- scoreSignatures(bsrdm, signatures)

# assignment
lr2ct <- assignCellTypesToInteractions(bsrdm, bsrinf, tme.scores)

# cellular network
g.table <- cellularNetworkTable(lr2ct)
gCN <- cellularNetwork(g.table)

#plot(gCN, edge.width=5*E(gCN)$score)
```

cellularNetworkTable *Build a table describing a cellular network*

Description

Generate a data.frame including all the links between cell types mediated by L-R interactions with their respective weights.

Usage

```
cellularNetworkTable(lr, autocrine = FALSE)
```

Arguments

lr	The data.frame output by "assignCellTypesToInteractions".
autocrine	A logical indicating whether autocrine interactions should be included.

Value

A data.frame containing all the links in the cellular network. A link is created between two cell types as soon as there was a L-R interaction that was associated with both cell types. The link is given a score equal to the geometric mean of each cell type assignment r2.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")
data(tme.signatures, package = "BulkSignalR")

immune.signatures <- immune.signatures[immune.signatures$signature %in%
    c("T cells"), ]

signatures <- rbind(immune.signatures, tme.signatures[
    tme.signatures$signature %in% c("Fibroblasts"),
])

tme.scores <- scoreSignatures(bsrdm, signatures)

# assignment
lr2ct <- assignCellTypesToInteractions(bsrdm, bsrinf, tme.scores)

# cellular network
g.table <- cellularNetworkTable(lr2ct)
```

---

chordDiagramLR *Chord Diagram of LR interactions with correlations*

Description

Chord diagram.

Usage

```
chordDiagramLR(
  bsrinf,
  pw.id.filter = NULL,
  qval.thres = 1,
  ligand = NULL,
  receptor = NULL,
  limit = 20
)
```

Arguments

bsrinf	A BSRInference object
pw.id.filter	One Pathway ID accepted only to
qval.thres	Threshold over Q-values.
ligand	Ligand of the LR pair that you want to highlight in the chord diagram.
receptor	Receptor of the LR pair that you want to highlight in the chord diagram.
limit	Number of interactions you can visualize.

Value

Circos Plot on the screen or a file

Examples

```
data(bsrinf, package = "BulkSignalR")
chordDiagramLR(bsrinf,
pw.id.filter = "R-HSA-3000178",
limit = 20,
ligand="ADAM15",
receptor="ITGAV"
)
```

---

coerce *Convert BSRDataModel to BSRDataModelComp*

Description

Enable the promotion of a BSRDataModel object into a BSRDataModelComp object by coercion.

Arguments

from	BSRDataModel object

Value

A BSRDataModelComp object

Examples

```
bsrdm <- new("BSRDataModel")
bsrdm.comp <- as(bsrdm, "BSRDataModelComp")
```

colClusterA *Cluster A columns accessor*

Description

Cluster A columns accessor

Usage

```
## S4 method for signature 'BSRClusterComp'
colClusterA(x)
```

Arguments

x	object BSRClusterComp

Value

col.clusterA

Examples

```
bsrcc <- new("BSRClusterComp")
colClusterA(bsrcc)
```

colClusterA<-,BSRClusterComp-method *Cluster A columns setter (internal use only)*

Description

Cluster A columns setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRClusterComp'
colClusterA(x) <- value
```

Arguments

x	object BSRClusterComp
value	value to be set for BSRClusterComp

Value

returns NULL

colClusterB *Cluster B columns accessor*

Description

Cluster B columns accessor

Usage

```
## S4 method for signature 'BSRClusterComp'
colClusterB(x)
```

Arguments

x	object BSRClusterComp

Value

col.clusterB

Examples

```
bsrcc <- new("BSRClusterComp")
colClusterB(bsrcc)
```

colClusterB<-,BSRClusterComp-method *Cluster B columns setter (internal use only)*

Description

Cluster B columns setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRClusterComp'
colClusterB(x) <- value
```

Arguments

x	object BSRClusterComp
value	value to be set for BSRClusterComp

Value

returns NULL

comparison *Comparisons list accessor*

Description

Comparisons list accessor

Usage

```
## S4 method for signature 'BSRDataModelComp'
comparison(x)
```

Arguments

x	object BSRDataModelComp

Value

comp

Examples

```
bsrdm.comp <- new("BSRDataModelComp")
comparison(bsrdm.comp)
```

comparison<-,BSRDataModelComp-method *Comparisons list setter (internal use only, use addComparison() otherwise)*

Description

Comparisons list setter (internal use only, use addComparison() otherwise)

Usage

```
## S4 replacement method for signature 'BSRDataModelComp'
comparison(x) <- value
```

Arguments

x	object BSRDataModelComp
value	value to be set for BSRDataModelComp

Value

returns NULL

comparisonName *Comparison name accessor*

Description

Comparison name accessor

Comparison name accessor

Usage

```
## S4 method for signature 'BSRInferenceComp'
comparisonName(x)

## S4 method for signature 'BSRSignatureComp'
comparisonName(x)
```

Arguments

x	BSRSignatureComp object

Value

cmp.name

cmp.name

Examples

```
bsrinf <- new("BSRInferenceComp")
comparisonName(bsrinf)
```

comparisonName<-,BSRInferenceComp-method *Comparison name setter (internal use only)*

Description

Comparison name setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInferenceComp'
comparisonName(x) <- value
```

Arguments

x	BSRInferenceComp object
value	value to be set for bsrinf

Value

returns NULL

convertToHuman *Transpose to Human Gene Names*

Description

By default, BulkSignalR is designed to work with Homo sapiens. In order to work with other organisms, gene names need to be first converted to human by orthology.

Usage

```
convertToHuman(counts, dictionary)
```

Arguments

counts	A table or matrix of read counts.
dictionary	A data frame where the first column is made of gene symbols for the actual organism and row names are the ortholog human gene symbols.

Value

Return a counts matrix transposed for Human.

Examples

```
data(bodyMap.mouse)

idx <- sample(nrow(bodyMap.mouse), 500)
bodyMap.mouse <- bodyMap.mouse[idx,]

ortholog.dict <- findOrthoGenes(
    from_organism = "mmusculus",
    from_values = rownames(bodyMap.mouse)
)

matrix.expression.human <- convertToHuman(
    counts = bodyMap.mouse,
    dictionary = ortholog.dict
)
```

createResources *Create all resources*

Description

Create a cache for all resources (pathways, lrdb & network) downloaded from the web when the library is first loaded. This functionality is handled with BiocFileCache.

Usage

```
createResources(onRequest = TRUE, verbose = FALSE)
```

Arguments

onRequest	logical TRUE if you want to force downloading again. This will overwrite the pre-existing local database. Default is TRUE.
verbose	Default is FALSE

Value

Returns 'NULL', invisibly.

Examples

```
createResources(onRequest=FALSE)
```

---

differentialStats *Cluster comparison statistics accessor*

Description

Cluster comparison statistics accessor

Usage

```
## S4 method for signature 'BSRClusterComp'
differentialStats(x)
```

Arguments

x	BSRClusterComp object

Value

diffferential.stats

Examples

```
bsrcc <- new("BSRClusterComp")
differentialStats(bsrcc)
```

differentialStats<-,BSRClusterComp-method *Cluster comparison statistics setter (internal use only)*

Description

Cluster comparison statistics setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRClusterComp'
differentialStats(x) <- value
```

Arguments

x	object BSRClusterComp
value	value to be set for BSRClusterComp

Value

returns NULL

findOrthoGenes *Orthologs Gene Names*

Description

By default, BulkSignalR is designed to work with Homo sapiens. In order to work with other organisms, gene names need to be first converted to human following an orthology mapping process.

Usage

```
findOrthoGenes(
  from_organism,
  from_values,
  method = c("gprofiler", "homologene", "babelgene")
)
```

Arguments

from_organism	An organism defined as in Ensembl: drerio, mmusculus, celegans, dmelanogaster, etc. This is the source organism from which you want to convert the gene names to Homo sapiens.
from_values	A vector of gene names from the current species studied.
method	Ortholog mapping method.

Value

Return a data frame with 2 columns containing the gene names for the two species. First column is the gene name from the source organism and the second column corresponds to the homologous gene name in Homo sapiens.

Examples

```
data(bodyMap.mouse)

idx <- sample(nrow(bodyMap.mouse), 20)
bodyMap.mouse <- bodyMap.mouse[idx,]

ortholog.dict <- findOrthoGenes(
    from_organism = "mmusculus",
    from_values = rownames(bodyMap.mouse)
)
```

---

generateSpatialPlots *Generate L-R interaction score spatial plots in a folder*

Description

Generate a series of individual spatial score plots in a folder. Not limited to BulkSignalR gene signature scores.

Usage

```
generateSpatialPlots(
  scores,
  areas,
  plot.folder,
  width = 5,
  height = 3,
  pointsize = 8,
  rev.y = TRUE,
  ref.plot = TRUE,
  image.raster = NULL,
  x.col = "array_col",
  y.col = "array_row",
  label.col = "label",
  idSpatial.col = "idSpatial",
  cut.p = 0.01,
  low.color = "royalblue3",
  mid.color = "white",
  high.color = "orange",
  title.fs = 12,
  legend.fs = 10,
  axis.fs = 10,
  label.fs = 12,
  dot.size = 0.5,
  ref.colors = NULL
)
```

Arguments

scores	A matrix of scores, one L-R interaction per row and spatial locations in the columns. This matrix is typically obtained from BulkSignalR functions scoreLRGeneSignatures or scScoring.
areas	A data.frame containing at least the x and y coordinates of the locations as well as the unique IDs of spatial locations. In case ref.plot is set to TRUE, a label column is required additionally.
plot.folder	The folder name in which the plot files will be written.
width	The width of each individual plot.
height	The height of each individual plot.
pointsize	PDF font point size.
rev.y	A Boolean indicating whether low y coordinates should be at the top of the plot.
ref.plot	A Boolean indicating whether a reference map of the tissue with area labels should be plot aside.
image.raster	Raster object image to plot raw tissue image as reference.
x.col	Column name in areas containing x coordinates.
y.col	Column name in areas containing y coordinates.
label.col	Column name in areas containing area labels.
idSpatial.col	Column name in areas containing the unique IDs of spatial locations.
cut.p	Proportion of top and bottom values for thresholding.
low.color	Color for low score values.
mid.color	Color for score = 0.
high.color	Color for high score values.
title.fs	Title font size.
legend.fs	Legend items font size.
axis.fs	Axis ticks font size.
label.fs	Legend titles and axis names font size.
dot.size	Dot size.
ref.colors	A vector of colors to bypass those automatically chosen by ggplot2 for the tissue areas in the reference plot.

Details

A set of PDF files are created in the provided folder.

Value

Create PDF file and returns ‘NULL‘, invisibly.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
```

```
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

generateSpatialPlots(scores.red[1:2,],
annotation.spa, ".", label.col = "ground_truth")
```

`getLRIntracellNetwork` *Generate a ligand-receptor-downstream signaling network*

Description

Generate a ligand-receptor network from a BSRInference object and add the shortest paths from the receptors to correlated target genes following Reactome and KEGG pathways.

Usage

```
getLRIntracellNetwork(
  bsrinf,
  pval.thres = NULL,
  qval.thres = NULL,
  min.cor = 0.25,
  max.pval = NULL,
  min.logFC = NULL,
  pos.targets = FALSE,
  neg.targets = FALSE,
  restrict.pw = NULL,
  node.size = 5
)
```

Arguments

`bsrinf`	A BSRInference or BSRInferenceComp object.
`pval.thres`	P-value LR interaction threshold.
`qval.thres`	Q-value LR interaction threshold.
`min.cor`	Minimum correlation required for the target genes.
`max.pval`	Maximum P-value required for the target genes in case a BSRInferenceComp object is provided.
`min.logFC`	Minimum logFC required for the target genes in case a BSRInferenceComp object is provided.
`pos.targets`	A logical imposing that all the network targets must display positive correlation or logFC in case of a BSRInferenceComp object.
`neg.targets`	A logical imposing that all the network targets must display negative correlation or logFC in case of a BSRInferenceComp object. Correlations must be <= - min.cor or logFC <= - min.logFC with this option activated.
`restrict.pw`	A vector of pathway IDs to which receptor downstream signaling is restricted.
`node.size`	Default node size in the network.

Value

An igraph object featuring the ligand-receptor-downstream signaling network. Default colors and node sizes are assigned, which can be changed afterwards if necessary.

The target genes to which the min.cor correlation is imposed are those listed in tgGenes(bsrinf), correlations are in tgCorr(bsrinf). The construction of shortest paths from the receptors to those selected targets adds other genes, which were either some targets with too low correlation or genes along the shortest paths to reach the selected targets.

Examples

```
data(bsrinf, package = "BulkSignalR")

bsrinf.redBP <- reduceToBestPathway(bsrinf)

pairs <- LRinter(bsrinf.redBP)
top <- unique(pairs[pairs$pval < 1e-20, c("pw.id", "pw.name")])

gLRintra.res <- getLRIntracellNetwork(bsrinf.redBP,
qval.thres = 0.01,
restrict.pw = top[1,]$pw.id
)

# write.graph(gLRintra, file="SDC-LR-intracellular-network.reduced.graphml",
# format="graphml")
```

---

getLRNetwork *Generate a ligand-receptor network*

Description

Generate a ligand-receptor network from a ligand-receptor table.

Usage

```
getLRNetwork(
  bsrinf,
  pval.thres = NULL,
  qval.thres = NULL,
  node.size = 5,
  red.pairs = NULL
)
```

Arguments

bsrinf	A BSRInference object.
pval.thres	P-value threshold.
qval.thres	Q-value threshold.
node.size	Default node size in the network.
red.pairs	A data frame with columns L (ligands) and R (receptors) that restrict LR pairs to those listed.

Value

An igraph object featuring the ligand-receptor network. Default colors and node sizes are assigned, which can be changed afterwards if necessary.

Examples

```
data(bsrinf, package = "BulkSignalR")

gLR <- getLRNetwork(bsrinf, qval.thres = 1e-4)
# plot(gLR)
# write.graph(gLR, file="SDC-LR-network.graphml", format="graphml")
```

---

getPathwayStats *Basic statistics about hit pathways*

Description

Basic statistics about hit pathways

Usage

```
## S4 method for signature 'BSRInference'
getPathwayStats(obj, pval.thres = NULL, qval.thres = NULL)
```

Arguments

obj	BSRinf object.
pval.thres	P-value threshold.
qval.thres	Q-value threshold.

Value

A table with the pathways selected after the chosen threshold was applied to rows in LRinter(obj). Each pathway is reported along with various statistics: the number of selected receptors in this pathway, the total number of receptors described this pathway, the number of selected ligand-receptor pairs hitting this pathway, and the total number of ligand-receptor pairs described that could hit this pathway.

Obviously, one could imagine computing enrichment in receptors or ligand-receptor pairs based on such statistics, but the actual meaning of such an analysis would be ambiguous since the pathways were already selected as significantly regulated by the receptor. We thus did not implement this (hypergeometric test) computation.

Examples

```
data(bsrinf, package = "BulkSignalR")

pw.stat <- getPathwayStats(bsrinf)
```

getResource *Get resources from the cache*

Description

Get resources (pathways, lrdb or network stored in the cache.

Usage

```
getResource(resourceName = NULL, cache = FALSE)
```

Arguments

resourceName	Resource name.
cache	Logical. Default is FALSE If TRUE, you will use environment variables.

Value

Returns a data frame containing the requested resource.

Examples

```
reactome <- getResource(resourceName = "Reactome",cache=TRUE)
```

immune.signatures *Immune cell gene signatures*

Description

A data set containing gene signatures for general immune cell populations.

Usage

```
data(immune.signatures)
```

Format

A data frame with 1541 rows and 2 variables:

gene HUGO gene symbol

signature cell population name

Source

PanglaoDB (Franzén et al., Database, 2019).

inferenceParameters *Inference parameters accessor*

Description

Inference parameters accessor

Usage

```
## S4 method for signature 'BSRInference'
inferenceParameters(x)
```

Arguments

x	BRSInference object.

Value

inf.param

Examples

```
bsrinf <- new ("BSRInference")
inferenceParameters(bsrinf)
```

inferenceParameters<-,BSRInference-method *Inference parameters setter (internal use only)*

Description

Inference parameters setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
inferenceParameters(x) <- value
```

Arguments

x	BRSInference object.
value	value to be set.

Value

returns NULL

initialOrganism *organism accessor*

Description

organism accessor

Usage

```
## S4 method for signature 'BSRDataModel'
initialOrganism(x)
```

Arguments

x	Object BSRDataModel

Value

initialOrganism

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
initialOrganism(bsrdm)
```

initialOrthologs *Model parameter accessor*

Description

Model parameter accessor

Usage

```
## S4 method for signature 'BSRDataModel'
initialOrthologs(x)
```

Arguments

x	Object BSRDataModel

Value

initialOrthologs

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
initialOrthologs(bsrdm)
```

`learnParameters` *Training of BulkSignalR model parameters*

Description

Unique entry point for training the parameters behind BulkSignalR statistical models.

Usage

```
## S4 method for signature 'BSRDataModel'
learnParameters(
  obj,
  plot.folder = NULL,
  verbose = FALSE,
  n.rand.LR = 5L,
  n.rand.RT = 2L,
  with.complex = TRUE,
  max.pw.size = 400,
  min.pw.size = 5,
  min.positive = 4,
  quick = FALSE,
 null.model = c("automatic", "mixedNormal", "normal", "kernelEmpirical", "empirical",
    "stable"),
  filename = NULL,
  min.corr.LR = -1
)
```

Arguments

`obj`	A BSRDatamodel without learned paramaters.
`plot.folder`	A folder name for generating control plots.
`verbose`	A logical activating progress messages for the user.
`n.rand.LR`	The number of random expression matrices to use for learning the ligand-receptor correlation distribution.

n.rand.RT	The number of random expression matrices to use for learning the receptor-target genes correlation distribution.
with.complex	A logical indicating whether receptor co-complex members should be included in the target genes.
max.pw.size	Maximum pathway size to consider from the pathway reference.
min.pw.size	Minimum pathway size to consider from the pathway reference.
min.positive	Minimum number of target genes to be found in a given pathway.
quick	A logical indicating whether approximate parameters for the receptor-target correlations should be used.
null.model	The null model to use for Spearman correlation null distributions.
filename	Name of the output plot.
min.corr.LR	The minimum ligand-receptor correlation required.

Details

Estimates the model parameters that are stored in the slot `param`.

In a reference pathway, i.e., a Reactome pathway or the genes of a GOBP term, the target genes are the genes coding for proteins forming a complex with the receptor and the genes in the pathway downstream the receptor, which are given as regulated by the pathway. If `with.complex` is set to `FALSE`, then only the regulated genes are considered. Participation to a complex, being regulated, and pathway directed topologies are defined by Reactome and KEGG pathways as provided by PathwayCommons.

The `min.pw.size`, `max.pw.size`, and `min.positive` parameters should be identical to the values intended when searching for ligand-receptor pairs with `.getCorrelatedLR`) and `.checkReceptorSignaling`) Although the statistical distributions are rather robust, it is not advisable to use different parameters that could introduce unanticipated biases, but for saving compute time and exploring.

The maximum pathway size is used to limit the redundancy inherent to GOBP and Reactome. The minimum pathway size is used to avoid overspecific, noninformative results.

BulkSignalR approach relies on modeling (Spearman) correlations and different models of null distributions are available for this purpose (parameter `null.model`). By default, the "automatic" option is selected meaning that censored normal and mixed normal as well as an empirical model based on Gaussian kernels (R `density()` function) are compared to pick the one closest to the data. Preference is given to normal and then mixture of normal over the empirical version for comparable quality of fit. It is also to bypass the automatic selection. Fitting of an alpha-stable distribution is quite time consuming as the computation of its PDF is compute-intensive. Finally, in the automaic selection mode, the choice of the actual model will be done based on the L-R null assuming a similar shape for the R-T null (with different parameters though, unless `quick` was set to `TRUE`).

Note that since the introduction of the use.full.network parameter (April 29, 2024) in the BSRInference method parameters, the pathway sizes are always computed before potential intersection with the observed data (use.full.network set to FALSE) for consistency. Accordingly, the minimum and maximum pathway default values have been raised from 5 & 200 to 5 & 400 respectively. By default, use.full.network is set to TRUE, meaning no intersection and hence larger pathways.

Value

A BSRDataModel object with trained model parameters

Examples

```
data(sdc, package = "BulkSignalR")
idx <- sample(nrow(sdc), 4000)
bsrdm <- BSRDataModel(sdc[idx, c("N22","SDC17")],min.LR.found = 20)
bsrdm <- learnParameters(bsrdm, n.rand.LR = 1L,
verbose=FALSE,quick=TRUE)
```

`ligands` *ligands accessor*

Description

ligands accessor

ligands accessor

Usage

```
## S4 method for signature 'BSRInference'
ligands(x)

## S4 method for signature 'BSRSignature'
ligands(x)
```

Arguments

`x`	BSRSignature

Value

ligands

ligands

Examples

```
bsr.sig <- new("BSRSignature")
ligands(bsr.sig)
```

`ligands<-,BSRInference-method` *ligands setter (internal use only)*

Description

ligands setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
ligands(x) <- value
```

Arguments

x	BRSInference object
value	Value to be set for bsrinf

Value

returns NULL

---

logTransformed *log.transformed accessor*

Description

log.transformed accessor

Usage

```
## S4 method for signature 'BSRDataModel'
logTransformed(x)
```

Arguments

x	Object BRSDataModel

Value

logTransformed

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                  ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
logTransformed(bsrdm)
```

LRinter *LRinter accessor*

Description

LRinter accessor

Usage

```
## S4 method for signature 'BSRInference'
LRinter(x)
```

Arguments

x	BSRInference object

Value

LRinter

Examples

```
bsrinf <- new ("BSRInference")
LRinter(bsrinf)
```

LRinter<-,BSRInference-method *LRinter setter (internal use only)*

Description

LRinter setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
LRinter(x) <- value
```

Arguments

x	BSRInference object
value	value to be set to BSRInference

Value

returns NULL

LRinterScore *Simplified LRinter accessor with focus on the LR-score*

Description

Simplified LRinter accessor with focus on the LR-score

Usage

```
## S4 method for signature 'BSRInferenceComp'
LRinterScore(x)
```

Arguments

x	BSRInferenceComp object

Value

LRinterScore

Examples

```
data(bsrinf.comp, package = "BulkSignalR")
LRinterScore(bsrinf.comp)[5,]
```

LRinterShort *Simplified LRinter accessor reporting the essential columns*

Description

Simplified LRinter accessor reporting the essential columns

Simplified LRinter accessor reporting the essential columns

Usage

```
## S4 method for signature 'BSRInference'
LRinterShort(x)

## S4 method for signature 'BSRInferenceComp'
LRinterShort(x)
```

Arguments

x	BSRInferenceComp object

Value

LRinterShort

LRinterShort

Examples

```
data(bsrinf.comp, package = "BulkSignalR")
LRinterShort(bsrinf.comp)[5,]
```

`maxLigandSpatialCounts` *Get maximal ligand expression at nearby locations*

Description

Get maximal ligand expression at nearby locations

Usage

```
maxLigandSpatialCounts(
  bsrdm,
  areas,
  nnn = 4,
  radius = NULL,
  x.col = "array_col",
  y.col = "array_row"
)
```

Arguments

`bsrdm`	A BSRDataModel object containing the expression data to smooth.
`areas`	A data.frame containing at least the x and y coordinates of the locations.
`nnn`	Number of nearest-neighbor locations to use for smoothing each location. In case `radius` is set, then it is the maximum number of nearest neighbors within the radius.
`radius`	A maximal distance to include neighbors in the smoothing.
`x.col`	Column name in `areas` containing x coordinates.
`y.col`	Column name in `areas` containing y coordinates.

Details

Ligand expression data contained in a BSRDataModel object are modified to consider the possibility that the ligand of a L-R interaction might be expressed at nearby locations. This is achieved replacing each ligand expression by its maximum over the central location and its neighbors. Since ligands and receptors are never used as gene targets in computing the receptor downstream signal correlations, this substitution is compatible with our statistical model. Moreover, the reciprocal configuration where the ligand is expressed at the central location and hits a receptors at a neighbor location is covered when the same ligand maximization scheme is applied to the neighbor. L-R localization and gene signature scoring is defined by the location at which the receptor is expressed after applying this function.

Two strategies are available to identify the neighbors. It is possible to simply set the number of nearest-neighbors (parameter `nnn`). An alternative consists in providing a distance radius (`radius`) along with a a maximum number of nearest-neighbors within the radius (`nnn.radius`). To properly define the radius, the user must know the location coordinates. The strategy with the radius enables

having corner locations with two neighbors only and border locations with three neighbors only, whereas to simply set a maximum of four neighbors for instance would retrieve the four closest neighbors in every case.

Value

A BSRDataModel object containing the maximized ligand expressions.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

max.bsrdm <- maxLigandSpatialCounts(bsrdm.spa, annotation.spa,
radius = 1.2, nnn = 4)
```

---

mu *Mu accessor*

Description

Mu accessor

Usage

```
## S4 method for signature 'BSRDataModelComp'
mu(x)
```

Arguments

x	object BSRDataModelComp

Value

mu

Examples

```
bsrdm.comp <- new("BSRDataModelComp")
mu(bsrdm.comp)
```

mu<-,BSRDataModelComp-method *Mu setter (internal use only)*

Description

Mu setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRDataModelComp'
mu(x) <- value
```

Arguments

x	object BSRDataModelComp
value	value to be set for BSRDataModelComp

Value

returns NULL

ncounts *Normalized count matrix accessor*

Description

Normalized count matrix accessor

Usage

```
## S4 method for signature 'BSRDataModel'
ncounts(x)
```

Arguments

x	object BSRDataModel

Value

ncounts

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
ncounts(bsrdm)
```

---

ncounts<-,BSRDataModel-method *Normalized count matrix setter (internal use only)*

Description

Normalized count matrix setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRDataModel'
ncounts(x) <- value
```

Arguments

x	object BSRDataModel
value	value to be set for BSRDataModel

Value

returns NULL

---

normalization *Normalization accessor*

Description

Normalization accessor

Usage

```
## S4 method for signature 'BSRDataModel'
normalization(x)
```

Arguments

x	object BSRDatamModel

Value

normalization

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
normalization(bsrdm)
```

---

ortholog.dict *A skinny data frame used in the mouse workflow*

Description

Synthetic object used during the call to the function ‘resetToInitialOrganism“

Usage

```
data(ortholog.dict)
```

Format

An example of a data frame created by findOrthoGenes

---

p.EMT *Partial EMT gene signature*

Description

A data set containing a partial EMT gene signature.

Usage

```
data(p.EMT)
```

Format

A data frame with 100 rows and 1 variables:

gene HUGO gene symbol

Source

Puram, SV & al., Cell, 2017.

parameters *Model parameter accessor*

Description

Model parameter accessor

Usage

```
## S4 method for signature 'BSRDataModel'
parameters(x)
```

Arguments

x	BSRDataModel oject

Value

param

Examples

```
bsrdm <- BSRDataModel(
    counts = matrix(1.5,
                nrow = 2, ncol = 2,
                dimnames = list(c("A", "B"), c("C", "D"))
                   ),
    method = "TC",
    log.transformed = TRUE,
    normalize = FALSE,
    min.LR.found = 0
)
parameters(bsrdm)
```

parameters<-,BSRDataModel-method *Parameters dataModel setter (internal use only)*

Description

Parameters dataModel setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRDataModel'
parameters(x) <- value
```

Arguments

x	object BSRDataModel
value	value to be set for BSRDataModel

Value

returns NULL

---

pathways *pathways accessor*

Description

pathways accessor

Usage

```
## S4 method for signature 'BSRSignature'
pathways(x)
```

Arguments

x	BSRSignature

Value

pathways

Examples

```
bsr.sig <- new("BSRSignature")
pathways(bsr.sig)
```

---

receptors *receptors accessor*

Description

receptors accessor

receptors accessor

Usage

```
## S4 method for signature 'BSRInference'
receptors(x)

## S4 method for signature 'BSRSignature'
receptors(x)
```

Arguments

x	BSRSignature

Value

receptors

receptors

Examples

```
bsr.sig <- new("BSRSignature")
ligands(bsr.sig)
```

---

receptors<-,BSRInference-method *receptors setter (internal use only)*

Description

receptors setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
receptors(x) <- value
```

Arguments

x	BRSInference object
value	value to be set for BRSInference

Value

returns NULL

---

reduceToBestPathway *Keep one pathway per ligand-receptor pair*

Description

Keep one pathway per ligand-receptor pair

Keep one pathway per ligand-receptor pair

Usage

```
## S4 method for signature 'BSRInference'
reduceToBestPathway(obj)

## S4 method for signature 'BSRInferenceComp'
reduceToBestPathway(obj)
```

Arguments

obj	BSRInferenceComp object

Details

Ligand-receptor pairs are evaluated in relation with pathways that allow checking receptor downstream correlations. It is thus possible that several pathways are reported for a same LR pair.

Ligand-receptor pairs are evaluated in relation with pathways that allow checking receptor downstream correlations. It is thus possible that several pathways are reported for a same LR pair.

Value

A BSRInference object reduced to only report one pathway per ligand-receptor pair. The pathway with the smallest P-value is selected.

A BSRInferenceComp object reduced to only report one pathway per ligand-receptor pair. The pathway with the smallest P-value is selected.

Examples

```
data(bsrinf, package = "BulkSignalR")
bsrinf.redBP <- reduceToBestPathway(bsrinf)

data(bsrinf.comp, package = "BulkSignalR")


reduceToBestPathway(bsrinf.comp)
```

---

reduceToLigand *Aggregate the receptors of a same ligand*

Description

Simplifies a ligand-receptor table to focus on the ligands.

Simplifies a ligand-receptor table to focus on the ligands.

Usage

```
## S4 method for signature 'BSRInference'
reduceToLigand(obj)

## S4 method for signature 'BSRInferenceComp'
reduceToLigand(obj)
```

Arguments

obj	BSRInferenceComp object

Value

A BSRInference object but reduced to one row per ligand. All the receptors are combined in a semi-colon-separated list surrounded by curly brackets in the tabular slot `LRinter`, and in vectors in the `ligands` (list) slot.

The reported P-value and target genes are those from the pathway with the smallest P-value.

A BSRInferenceComp object but reduced to one row per ligand. All the receptors are combined in a semi-colon-separated list surrounded by curly brackets in the tabular slot `LRinter`, and in vectors in the `ligands` (list) slot.

The reported P-value and target genes are those from the pathway with the smallest P-value. The same logic applies to the LR-score, and the receptor expression.

Examples

```
data(bsrinf, package = "BulkSignalR")

bsrinf.redL <- reduceToLigand(bsrinf)

data(bsrinf.comp, package = "BulkSignalR")

bsrinf.redL <- reduceToLigand(bsrinf.comp)
```

`reduceToPathway` *Aggregate ligands and receptors at the pathway level*

Description

Simplifies a ligand-receptor inference object to focus on the pathways.

Simplifies a ligand-receptor inference object to focus on the pathways.

Usage

```
## S4 method for signature 'BSRInference'
reduceToPathway(obj)

## S4 method for signature 'BSRInferenceComp'
reduceToPathway(obj)
```

Arguments

`obj`	BSRInferenceComp object

Value

A BSRInference object reduced to only report one row per pathway. The information of which ligand interacted with which receptor is lost as all the ligands and all the receptors forming pairs related to a certain pathway are combined. For a given pathway, the reported P-values and target genes are those of the best ligand-receptor pair that was in this pathway. Receptors and ligands are combined in two semi-colon-separated lists surrounded by curly brackets in the tabular slot `LRinter`, while the list representation slots (`ligands` and `receptors`) are update accordingly.

A BSRInferenceComp object reduced to only report one row per pathway. The information of which ligand interacted with which receptor is lost as all the ligands and all the receptors forming pairs related to a certain pathway are combined. For a given pathway, the reported P-values and target genes are those of the best ligand-receptor pair that was in this pathway. The same logic applies to the LR-score, and the ligand and receptor expression. Receptors and ligands are combined in two semi-colon-separated lists surrounded by curly brackets in the tabular slot `LRinter`, while the list representation slots (`ligands` and `receptors`) are update accordingly.

Examples

```
data(bsrinf, package = "BulkSignalR")

bsrinf.redP <- reduceToPathway(bsrinf)
data(bsrinf.comp, package = "BulkSignalR")

bsrinf.redP <- reduceToPathway(bsrinf.comp)
```

---

reduceToReceptor *Aggregate the ligands of a same receptor*

Description

Simplifies a ligand-receptor table to focus on the receptors.

Simplifies a ligand-receptor table to focus on the receptors.

Usage

```
## S4 method for signature 'BSRInference'
reduceToReceptor(obj)

## S4 method for signature 'BSRInferenceComp'
reduceToReceptor(obj)
```

Arguments

`obj`	BRSInferenceComp object

Value

BSRInference object reduced to one row per receptor. All the ligands are combined in a semi-colon-separated list surrounded by curly brackets in the tabular slot `LRinter`, and in vectors in the `ligands` (list) slot.

The reported P-value and target genes are those from the line with the pathway featuring the smallest P-value.

BSRInferenceComp object reduced to one row per receptor. All the ligands are combined in a semi-colon-separated list surrounded by curly brackets in the tabular slot `LRinter`, and in vectors in the `ligands` (list) slot.

The reported P-value and target genes are those from the line with the pathway featuring the smallest P-value. The same logic applies to the LR-score, and the ligand expression.

Examples

```
data(bsrinf, package = "BulkSignalR")

bsrinf.redR <- reduceToReceptor(bsrinf)

data(bsrinf.comp, package = "BulkSignalR")
# reduction
bsrinf.redR <- reduceToReceptor(bsrinf.comp)
```

`relateToGeneSet` *Relate ligands to a gene set*

Description

Finds ligands related to a gene set by following receptor, and receptor downstream pathway targets.

Usage

```
relateToGeneSet(bsrinf, gs, min.cor = 0.25, qval.thres = 0.001)
```

Arguments

`bsrinf`	BSRInference object.
`gs`	The gene set.
`min.cor`	Minimum Spearman correlation between the receptor of a triple (L,R,pw) and a gene of the gene set.
`qval.thres`	Maximum Q-value imposed to the (L,R,pw) triples to be considered.

Value

A data.frame listing all the (L,R,pathway) triples that lead to at least one gene in the gene set. The number of genes found by each triple is indicated in the column `n.genes`.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")

data(p.EMT, package = "BulkSignalR")
p.EMT <- p.EMT$gene
triggers <- relateToGeneSet(bsrinf, p.EMT)
```

`removeClusterComp` *Remove a comparison from a BSRDataModelComp object.*

Description

Remove a comparison from a BSRDataModelComp object.

Usage

```
## S4 method for signature 'BSRDataModelComp'
removeClusterComp(obj, cmp.name)
```

Arguments

`obj`	A BSRDataModelComp object output by `setAs`.
`cmp.name`	The name of the comparison to remove.

Details

Remove the comparison with `cmp.name` from the list of comparisons contained in `obj`.

Value

A BSRDataModelComp object.

Examples

```
# prepare data
data(sdc, package = "BulkSignalR")
normal <- grep("^N", names(sdc))
bsrdm <- BSRDataModel(sdc[, -normal])

# define the comparison
bsrdm.comp <- as(bsrdm, "BSRDataModelComp")
colA <- as.integer(1:3)
colB <- as.integer(12:15)
n <- nrow(ncounts(bsrdm.comp))
stats <- data.frame(
    pval = runif(n), logFC = rnorm(n, 0, 2),
    expr = runif(n, 0, 10)
)
rownames(stats) <- rownames(ncounts(bsrdm.comp))
bsrcc <- BSRClusterComp(bsrdm.comp, colA, colB, stats)

bsrdm.comp <- addClusterComp(bsrdm.comp, bsrcc, "random.example")
bsrdm.comp <- removeClusterComp(bsrdm.comp, "random.example")
```

rescoreInference *Inference re-scoring*

Description

A method to re-score an existing BSRInference object (P- and Q-value estimations).

A method to re-score an existing BSRInferenceComp object (P- and Q-value estimations).

Usage

```
## S4 method for signature 'BSRInference'
rescoreInference(
  obj,
  param,
  rank.p = 0.55,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)

## S4 method for signature 'BSRInferenceComp'
rescoreInference(
  obj,
  param = NULL,
  rank.p = 0.55,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

obj	BSRInferecenceComp object.
param	NULL by default
rank.p	A number between 0 and 1 defining the rank of the last considered target genes.
fdr.proc	The procedure for adjusting P-values according to mt.rawp2adjp.

Details

A BSRInference object should be created by calling "BSRInference"

Parameters controlling the estimation of the statistical significance of the ligand/receptor/pathway triples (param) are provided at the time of calling the latter method.

Nonetheless, it might be useful to change the initially-provided parameters, in which case this method should not be called.

A BSRInferenceComp object should be created by calling "BSRInferenceComp"

Value

A BSRInference object.

A BSRInferenceComp object.

Examples

```
data(bsrinf, package = "BulkSignalR")
data(bsrdm, package = "BulkSignalR")

bsrinf.new <- rescoreInference(bsrinf,
                        param = parameters(bsrdm)
              )
data(bsrinf.comp, package = "BulkSignalR")

bsrinf.less <- rescoreInference(bsrinf.comp,
rank.p = 0.75)
```

`resetLRdb` *Modify LRdb database*

Description

Users can provide a data frame with 2 columns named ligand and receptor. This can be used to extend or replace the existing LRdb.

Users can provide a data frame with 2 columns named ligand and receptor. This can be used to extend or replace the existing LRdb.

Usage

```
resetLRdb(db, switch = FALSE)

resetLRdb(db, switch = FALSE)
```

Arguments

db	A data frame with 2 columns named ligand and receptor.
switch	A logical indicating whether LRdb should be extended only (FALSE, default) or completely replaced (TRUE).

Value

Returns 'NULL', invisibly.

Returns 'NULL', invisibly.

Examples

```
resetLRdb(db = data.frame(ligand = "A2M", receptor = "LRP1"), switch = FALSE)
resetLRdb(db = data.frame(ligand = "A2M", receptor = "LRP1"), switch = FALSE)
```

resetNetwork *Import a refernce network of your own*

Description

Network is a data frame that defines interactions between genes. It's composed of 3 columns named as follows:

Usage

```
resetNetwork(network)
```

Arguments

network	Network data frame made of 3 columns a.gn, b.gn & type. 'a.gn' & 'b.gn' should be gene symbols of gene interactions. 'type' should be set as 'controls-expression-of' when a user provides his own file.

Details

a.gn: Gene Symbol 1 type: controls-expression-of b.gn: Gene Symbol 2

When the user provides his own network, 'type' should be set to 'controls-expression-of'.

Value

Returns 'NULL', invisibly.

Examples

```
 BulkSignalR_Network <- getResource(resourceName = "Network",
 cache = FALSE)
resetNetwork(BulkSignalR_Network)
```

resetPathways *Import pathways from a file or data frame*

Description

resetPathways is a function we provide to users who want to refresh REACTOME and GO-BP content included in BulkSignalR.

Usage

```
resetPathways(
  dataframe = NULL,
  file = NULL,
  fileType = c("json", "gmt", "txt"),
  resourceName = NULL
)
```

Arguments

dataframe	Data frame formated as follows. When resourceName is set to "Reactome", dataframe colnames must be defined as "Reactome ID", "Gene name", and "Reactome name" When resourceName is set to "GO-BP", #' dataframe colnames must be defined as "GO ID", "Gene name", and "GO name"
file	Path to file.
fileType	Default is Json. Other options are gmt or txt files.
resourceName	Two options "GO-BP" or "Reactome".

Details

Pathways in 'BulkSignalR' (as sets of genes/proteins) are defined after Reactome and GOBP databases. Those can be updated using json files from the Human Molecular Signatures Database (MSigDB) at https://www.gsea-msigdb.org/ Gmt file format also can be imported. A data frame can be used directly also.

Value

Returns 'NULL', invisibly.

Examples

```
reactSubset <- getResource(resourceName = "Reactome",
cache = TRUE)

subset <- c("REACTOME_BASIGIN_INTERACTIONS",
"REACTOME_SYNDECAN_INTERACTIONS",
"REACTOME_ECM_PROTEOGLYCANS",
"REACTOME_CELL_JUNCTION_ORGANIZATION")

reactSubset <- reactSubset[
reactSubset$`Reactome name` %in% subset,]

resetPathways(dataframe = reactSubset,
resourceName = "Reactome")
```

---

resetToInitialOrganism

Reset gene names to initial organism provided in the first instance

---

Description

Reset gene names to initial organism provided in the first instance

Usage

```
## S4 method for signature 'BSRInference'
resetToInitialOrganism(obj, conversion.dict)
```

Arguments

obj	BSRInference object
conversion.dict	A dictionnary

Value

An BSRInference object updated for gene names. The gene names are replaced by the ones from the organism provided in the first instance.

Examples

```
data(bodyMap.mouse, package = "BulkSignalR")
data(bsrinf.mouse, package = "BulkSignalR")
data(ortholog.dict, package = "BulkSignalR")

#idx <- sample(nrow(bodyMap.mouse), 7500)

#bodyMap.mouse <- bodyMap.mouse[idx,1:3]

#ortholog.dict <- findOrthoGenes(
#    from_organism = "mmusculus",
#    from_values = rownames(bodyMap.mouse)
#)

#matrix.expression.human <- convertToHuman(
#    counts = bodyMap.mouse,
#    dictionary = ortholog.dict
#)

#bsrdm <- BSRDataModel(
#    counts = matrix.expression.human,
#    species = "mmusculus",
#    conversion.dict = ortholog.dict
#)

#bsrdm <- learnParameters(bsrdm,
#    quick = TRUE
#)

#reactSubset <- getResource(resourceName = "Reactome",
#cache = TRUE)

#subset <- c("REACTOME_BASIGIN_INTERACTIONS",
#"REACTOME_SYNDECAN_INTERACTIONS",
#"REACTOME_ECM_PROTEOGLYCANS",
#"REACTOME_CELL_JUNCTION_ORGANIZATION")

#reactSubset <- reactSubset[
#reactSubset$`Reactome name` %in% subset,]

#bsrinf.mouse <- BSRInference(bsrdm,reference="REACTOME")

bsrinf <- resetToInitialOrganism(bsrinf.mouse,
conversion.dict = ortholog.dict)
```

`scoreLRGeneSignatures` *Score ligand-receptor gene signatures*

Description

Compute ligand-receptor gene signature scores over a BSRDataModel.

Compute ligand-receptor gene signature scores over a BSRDataModelComp specific comparison.

Usage

```
## S4 method for signature 'BSRDataModel'
scoreLRGeneSignatures(
  obj,
  sig,
  LR.weight = 0.5,
  robust = FALSE,
  name.by.pathway = FALSE,
  abs.z.score = FALSE,
  rownames.LRP = FALSE
)

## S4 method for signature 'BSRDataModelComp'
scoreLRGeneSignatures(
  obj,
  sig,
  LR.weight = 0.5,
  robust = FALSE,
  name.by.pathway = FALSE,
  abs.z.score = FALSE,
  rownames.LRP = FALSE
)
```

Arguments

`obj`	A BSRDataModelComp object.
`sig`	A BSRSignatureComp object.
`LR.weight`	A number between 0 and 1 defining the relative weight of the ligand and the receptor in the signature.
`robust`	A logical indicating that z-scores should be computed with median and MAD instead of mean and standard deviation.
`name.by.pathway`	A logical indicating whether row names of the resulting score matrix should be pathway names.
`abs.z.score`	A logical to use absolute z-scores (useful if the activity of a paythway is reported by a mixture of up- and down-genes whose z-score averages might hide actual activity).
`rownames.LRP`	A logical indicating, in case `name.by.pathway` was set to TRUE, whether ligand and receptor names should be added on top. No role if `name.by.pathway` was set to FALSE.

Value

A matrix containing the scores of each ligand-receptor gene signature in each sample.

A matrix containing the scores of each ligand-receptor gene signature in each sample.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")

bsrinf.redBP <- reduceToBestPathway(bsrinf)
bsrsig.redBP <- BSRSignature(bsrinf.redBP, qval.thres = 0.001)
res <-scoreLRGeneSignatures(bsrdm, bsrsig.redBP,
    name.by.pathway = FALSE
)
# prepare data
data(bsrdm.comp, package = "BulkSignalR")
data(bsrinf.comp, package = "BulkSignalR")

# reduction
bsrinf.red <- reduceToBestPathway(bsrinf.comp)
# signature extraction and scoring
bsrsig.red <- BSRSignatureComp(bsrinf.red, qval.thres = 1e-6)
scores.red <- scoreLRGeneSignatures(bsrdm.comp, bsrsig.red,
    name.by.pathway = TRUE, rownames.LRP = TRUE
)
```

---

scoreSignatures *Generic gene signature scoring*

Description

Scores generic gene signatures over the samples of a BSRDataModel object.

Usage

```
scoreSignatures(ds, ref.signatures, robust = FALSE)
```

Arguments

ds	A BSRDataModel object.
ref.signatures	Gene signatures.
robust	A logical indicating that z-scores should be computed with median and MAD instead of mean and standard deviation.

Details

This function relies on a simple average of gene z-scores over each signature. It is no replacement for mode advanced methods such as CIBERSORT or BisqueRNA. It is provided for convenience.

Value

A matrix containing the scores of each gene signature in each sample. Note that ligand-receptor gene signature scores should be computed with "scoreLRGeneSignatures" instead.

Examples

```
data(sdc, package = "BulkSignalR")
data(bsrdm, package = "BulkSignalR")

data(immune.signatures, package = "BulkSignalR")
imm.scores <- scoreSignatures(bsrdm, immune.signatures)
```

---

sdc *Salivary duct carcinoma transcriptoms*

Description

A data set containing the read counts of salivary duct carcinomas (SDCs) and adjacent normal tissues.

Usage

```
data(sdc)
```

Format

A data frame with 19764 rows and 26 variables.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE138581

---

separatedLRPlot *Generate separated plots for a L-R interaction*

Description

Generate a detailed view related to a chosen interaction made of series of small individual spatial plots: tissue organization (optional), gene signature score, ligand and receptor expression.

Usage

```
separatedLRPlot(
  v,
  L,
  R,
  ncounts,
  areas,
  inter.name = NULL,
  rev.y = TRUE,
  ref.plot = TRUE,
  image.raster = NULL,
  x.col = "array_col",
  y.col = "array_row",
  label.col = "label",
```

```
  idSpatial.col = "idSpatial",
  cut.p = 0.01,
  low.color = "royalblue3",
  mid.color = "white",
  high.color = "orange",
  title.fs = 12,
  legend.fs = 10,
  axis.fs = 10,
  label.fs = 12,
  dot.size = 0.5,
  legend.dot.factor = 10,
  ref.colors = NULL
)
```

Arguments

v	A named vector containing the gene signature scores for the L-R interaction including the contribution of the pathway, names must be the IDs of each location. Alternatively, v can be a gene signature score matrix such as those returned by `scoreLRGeneSignatures` and the row named "L / R" will be used.
L	The name of the ligand.
R	The name of the receptor.
ncounts	The (normalized) expression matrix with column names equal to the IDs of each location.
areas	A data.frame containing at leastcluster_columns the x and y coordinates of the locations as well as the unique IDs of spatial locations. In case `ref.plot` is set to TRUE, a label column is required additionally.
inter.name	Interaction name to display as plot title, equal to "L / R" unless specified.
rev.y	A Boolean indicating whether low y coordinates should be at the top of the plot.
ref.plot	A Boolean indicating whether a reference map of the tissue with area labels should be plot aside.
image.raster	Raster object image to plot raw tissue image as reference.
x.col	Column name in `areas` containing x coordinates.
y.col	Column name in `areas` containing y coordinates.
label.col	Column name in `areas` containing area labels.
idSpatial.col	Column name in `areas` containing the unique IDs of spatial locations.
cut.p	Proportion of top and bottom values for thresholding.
low.color	Color for low score values.
mid.color	Color for score = 0.
high.color	Color for high score values.
title.fs	Title font size.
legend.fs	Legend items font size.
axis.fs	Axis ticks font size.
label.fs	Legend titles and axis names font size.
dot.size	Dot size.
legend.dot.factor	A factor applied to obtain the legend dot size.
ref.colors	A vector of colors to bypass those automatically chosen by ggplot2 for the tissue areas in the reference plot.

Details

A set of spatial plots are generated including an optional reference tissue plot (image or areas represented), the gene signature scores, the ligand expression values, and the receptor expression values.

Value

A set of spatial plots.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

separatedLRPlot(scores.red, "SLIT2", "GPC1",
ncounts(bsrdm.spa),
annotation.spa,
label.col = "ground_truth")
```

---

signatureHeatmaps *Heatmap function to dissect one pathway signature*

Description

Plots a stack of three heatmaps to assess the expression of the target genes or proteins in a chosen pathway, the receptor expressions, and the ligand expressions.

Usage

```
signatureHeatmaps(
  pathway,
  bsrdm,
  bsrsig,
  heights = c(4, 2, 4),
  fontsize = 6,
  legend.fontsize = 8,
  title.fontsize = 8,
  col.fontsize = 6,
  annot.fontsize = 8,
  ht.gap = 3,
  show_column_names = TRUE
)
```

Arguments

pathway	The chosen pathway name.
bsrdm	BulkSignalR data model object.
bsrsig	BulkSignalR signature object.
heights	A vector of 3 heights (in cm) for the 3 heatmaps.
fontsize	Font size for the gene names.
legend.fontsize	Font size for the legends.
title.fontsize	Font size for the pathway name as plot title.
col.fontsize	Font size for column (sample) names.
annot.fontsize	Font size for column annotation names.
ht.gap	Space between heatmaps (in mm).
show_column_names	Add column names in the heatmaps.

Value

A plot is created.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
if(FALSE){
bsrinf.redP <- reduceToPathway(bsrinf)
bsrinf.redPBP <- reduceToBestPathway(bsrinf)
bsrsig.redPBP <- BSRSignature(bsrinf, qval.thres = 1)
pathway1 <- pathways(bsrsig.redPBP)[1]
signatureHeatmaps(
pathway = pathway1,
bsrdm = bsrdm,
bsrsig = bsrsig.redPBP,
show_column_names = TRUE)
}
```

---

simpleHeatmap *Heatmap function for LR scores*

Description

Generate a heatmap representing ligand-receptor gene signature scores.

Usage

```
simpleHeatmap(
  mat.c,
  dend.row = NULL,
  dend.spl = NULL,
  cols = NULL,
  pointsize = 4,
  bottom.annotation = NULL,
  n.col.clust = 0,
  n.row.clust = 0,
  gap.size = 0.5,
  cut.p = 0.01,
  row.names = TRUE,
  column.names = TRUE,
  hcl.palette = NULL,
  reverse = FALSE
)
```

Arguments

mat.c	A matrix with the signature scores such as output by scoreLRGeneSignatures().
dend.row	A precomputed row dendrogram.
dend.spl	A precompute sample (column) dendrogram.
cols	A vector of colors to use for the heatmap.
pointsize	Heatmap font point size
bottom.annotation	ComplexHeatmap package bottom annotations.
n.col.clust	Number of column clusters.
n.row.clust	Number of row clusters.
gap.size	Gap size between clusters.
cut.p	Proportion of top and bottom values for thresholding.
row.names	A logical to turn on/off the display of row names.
column.names	A logical to turn on/off the display of column (sample) names.
hcl.palette	support for HCL colormaps in ComplexHeatmap using color mapping function with circlize::colorRamp2(). palettes are listed in grDevides::hcl.pals(). of row (gene) names.
reverse	A logical to reverse or not colors in hcl.palette.

Value

A heatmap. Since heatmap plotting tend to be slow on the screen, it is advisable to plot in a file instead.

If hcl.palette is set, the colors parameter won't be used.

Extreme values (top and bottom) can be replaced by global quantiles at cut.p and 1-cut.p to avoid color scales shrunk by a few outliers.

This is a convenience function that relies on the ComplexHeatmap package to propose a simple way of representing signature scores. If more advanced features are needed or more graphic parameters should be controlled, users should implement their own function.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")

bsrinf.redBP <- reduceToBestPathway(bsrinf)
bsrsig.redBP <- BSRSignature(bsrinf,
    qval.thres = 0.001
)

scoresLR <- scoreLRGeneSignatures(bsrdm, bsrsig.redBP,
    name.by.pathway = FALSE
)
simpleHeatmap(scoresLR[1:3, ],
    column.names = TRUE,
    hcl.palette = "Cividis")
```

`smoothSpatialCounts` *Smooth spatial expression data*

Description

Smooth spatial expression data

Usage

```
smoothSpatialCounts(
  bsrdm,
  areas,
  nnn = 4,
  radius = NULL,
  weight.ratio = 0.5,
  x.col = "array_col",
  y.col = "array_row"
)
```

Arguments

`bsrdm`	A BSRDataModel object containing the expression data to smooth.
`areas`	A data.frame containing at least the x and y coordinates of the locations.
`nnn`	Number of nearest-neighbor locations to use for smoothing each location. In case `radius` is set, then it is the maximum number of nearest neighbors within the radius.
`radius`	A maximal distance to include neighbors in the smoothing.
`weight.ratio`	The weight given to the central location.
`x.col`	Column name in `areas` containing x coordinates.
`y.col`	Column name in `areas` containing y coordinates.

Details

The expression data contained in a BSRDataModel object are smoothed using a weighted average of nearby locations.

Two strategies are available to identify the neighbors. It is possible to simply set the number of nearest-neighbors (parameter `nnn`). An alternative consists in providing a distance radius (`radius`) along with a a maximum number of nearest-neighbors within the radius (`nnn.radius`). To properly define the radius, the user must know the location coordinates. The strategy with the radius enables having corner locations with two neighbors only and border locations with three neighbors only, whereas to simply set a maximum of four neighbors for instance would retrieve the four closest neighbors in every case.

For each location, its nearest-neighbors are found and a weighted average computed with `weight.ratio` given to the central location itself and a total weight of 1-`weight.ratio` shared within the neighbors based on the inverse of their distances. In case `radius` is set, some locations may have less than `nnn` neighbors (see above). At such locations, the weight given to the central location is augmented according to 1-(1-`weight.ratio`)*(number of neighbors)/`nnn`.

Value

A BSRDataModel object containing the smoothed ncounts.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")
sm.bsrdm <- smoothSpatialCounts(bsrdm.spa, annotation.spa,
radius = 1.2, nnn = 4)
```

---

sourceComparisonName *Source comparison name accessor*

Description

Source comparison name accessor

Usage

```
## S4 method for signature 'BSRInferenceComp'
sourceComparisonName(x)
```

Arguments

x	BSRInferenceComp object

Value

src.comp.name

Examples

```
bsrinf <- new("BSRInferenceComp")
sourceComparisonName(bsrinf)
```

sourceComparisonName<-,BSRInferenceComp-method *Source comparison name setter (internal use only)*

Description

Source comparison name setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInferenceComp'
sourceComparisonName(x) <- value
```

Arguments

x	BSRInferenceComp object
value	value to be set for bsrinf

Value

returns NULL

spatialAssociation *Statistical association of scores with area labels*

Description

Compute the statistical association of L-R interaction score spatial distributions with tissue area labels. Not limited to BulkSignalR gene signature scores.

Usage

```
spatialAssociation(
  scores,
  areas,
  test = c("Kruskal-Wallis", "ANOVA", "Spearman", "r2"),
  label.col = "label",
  idSpatial.col = "idSpatial",
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

scores	A matrix of scores, one L-R interaction per row and spatial locations in the columns. This matrix is typically obtained from BulkSignalR functions scoreLRGeneSignatures or scScoring.
areas	A data.frame containing at least the x and y coordinates of the locations, the unique IDs of spatial locations, and a label column.
test	The chosen statistical test or statistics (see details below).
label.col	Column name in areas containing area labels.
idSpatial.col	Column name in areas containing the unique IDs of spatial locations.
fdr.proc	Multiple hypothesis correction procedure, see multtest.

Details

In case the nonparametric Kruskal-Wallis test is chosen, additional columns are provided testing each label for significantly larger scores (Kruskal-Wallis is global and only says whether one or several labels show a bias). Individual labels are tested with Wilcoxon and two columns are added *per* label, one for the statistics and one for a Bonferroni-corrected P-value over all the labels.

In case an actual statistical test is chosen, a parametric test (ANOVA) and a non-parametric test (Kruskal-Wallis) are available for the global analysis. Individual labels are tested with T-tests or Wilcoxon (Bonferroni-corrected) accordingly.

In case a statistics is preferred, Spearman correlation or explained variance (r2 or coefficient of determination, through linear models) are available. They mesure the relationship between each individual area and scores. For the explained variance, a global value (R2) is also computed from a multi-linear model (the same as what is used for the ANOVA).

Value

A data.frame with the names of the interactions, the value of the chosen statistics, and the corresponding Q-value.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")
thres <- 0.01
#bsrinf.red <- reduceToBestPathway(bsrinf.spa)
#s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
#scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

# Run in other examples no need to be run again
# spatialAssociation(scores.red[c(1:2),], areas = annotation.spa,
# label.col = "ground_truth")
```

`spatialAssociationPlot` *Heatmap plot of association of scores with area labels*

Description

Plot a heatmap featuring Q-values or values of statistical association between L-R interaction score spatial distributions and tissue area labels.

Usage

```
spatialAssociationPlot(
  associations,
  qval.thres = 0.01,
  absval.thres = 0,
  colors = NULL
)
```

Arguments

`associations`	A statistical association data.frame generated by the function `spatialAssociation`.
`qval.thres`	The maximum Q-value to consider in the plot (a L-R interaction must associate with one label at least with a Q-value smaller or equal to this threshold).
`absval.thres`	The minimum value to consider in the plot (a L-R interaction must associate with one label at least with an absolute value larger or equal to this threshold).
`colors`	A function returning a color for a given value such as generated by `circlize::colorRamp2`.

Details

Display a heatmap linking L-R interactions to labels.

Value

ComplexHeatmap::Heatmap object

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

# statistical association with tissue areas based on correlations

assoc.bsr.corr <- spatialAssociation(scores.red[c(1:10), ],
areas = annotation.spa, label.col = "ground_truth",test = "Spearman")

spatialAssociationPlot(assoc.bsr.corr)
```

`spatialDiversityPlot` *2D-projection of spatial score distributions*

Description

Use PCA or t-SNE to obtain a 2D-projection of a set of spatial scores or associations. This plot summarizes the diversity of patterns occuring in a spatial dataset. Use the function `spatialIndexPlot` to create a large visual index of many spatial distributions. Not limited to BulkSignalR gene signature scores.

Usage

```
spatialDiversityPlot(
  scores,
  associations,
  proj = c("PCA", "tSNE"),
  score.based = FALSE,
  qval.thres = 0.01,
  val.thres = 0,
  with.names = FALSE,
  text.fs = 2.5,
  legend.fs = 10,
  axis.fs = 10,
  label.fs = 12,
  dot.size = 1,
  perplexity = 10
)
```

Arguments

`scores`	A matrix of scores, one L-R interaction per row and spatial locations in the columns. This matrix is typically obtained from BulkSignalR functions `scoreLRGeneSignatures` or `scScoring`.
`associations`	A statistical association data.frame generated by the function `spatialAssociation`.
`proj`	Projection method : 'PCA' or 'tSNE' are available arguements.
`score.based`	A logical indicating whether the plot should be based on scores or the associations directly.
`qval.thres`	The maximum Q-value to consider in the plot (a L-R interaction must associate with one label at least with a Q-value smaller or equal to this threshold). Relevant for Kruskal-Wallis and ANOVA tests in `spatialAssociation`.
`val.thres`	The minimum value to consider in the plot (a L-R interaction must associate with one label at least with a value larger or equal to this threshold). Relevant for Spearman and r2 associations in `spatialAssociation`.
`with.names`	A logical indicating whether L-R names should be plotted.
`text.fs`	Point label font size in case `with.names` is TRUE.
`legend.fs`	Legend items font size.
`axis.fs`	Axis ticks font size.
`label.fs`	Legend titles and axis names font size.
`dot.size`	Dot size.
`perplexity`	Perplexity parameter for t-SNE.

Details

Display a 2D-projection of the score spatial distributions.

Value

Display a 2D-projection of the score spatial distributions.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

# statistical association with tissue areas based on correlations
# For display purpose, we only use a subset here


assoc.bsr.corr <- spatialAssociation(scores.red[c(1:3), ],
annotation.spa, label.col = "ground_truth",test = "Spearman")

spatialDiversityPlot(scores.red[c(1:3),],assoc.bsr.corr)
```

---

spatialIndexPlot *Generate a visual index of spatial score distributions*

Description

Generate an index made of series of small individual spatial score plots in a PDF. Not limited to BulkSignalR gene signature scores.

Usage

```
spatialIndexPlot(
  scores,
  areas,
  out.file,
  ref.plot = TRUE,
  image.raster = NULL,
  x.col = "array_col",
  y.col = "array_row",
  label.col = "label",
  idSpatial.col = "idSpatial",
  cut.p = 0.01,
  low.color = "royalblue3",
  mid.color = "white",
  high.color = "orange",
```

```
  title.fs = 12,
  legend.fs = 10,
  axis.fs = 10,
  label.fs = 12,
  dot.size = 0.25,
  ratio = 1.25,
  base.v = 2.5,
  base.h = 3,
  ref.colors = NULL
)
```

Arguments

scores	A matrix of scores, one L-R interaction per row and spatial locations in the columns. This matrix is typically obtained from BulkSignalR functions scoreLRGeneSignatures or scScoring.
areas	A data.frame containing at least the x and y coordinates of the locations, the unique IDs of spatial locations, and a tissue label column.
out.file	File name for the output PDF.
ref.plot	A Boolean indicating whether a reference map of the tissue with area labels should be plot first.
image.raster	Raster object image to plot raw tissue image as reference.
x.col	Column name in areas containing x coordinates.
y.col	Column name in areas containing y coordinates.
label.col	Column name in areas containing area labels.
idSpatial.col	Column name in areas containing the unique IDs of spatial locations.
cut.p	Proportion of top and bottom values for thresholding.
low.color	Color for low score values.
mid.color	Color for score = 0.
high.color	Color for high score values.
title.fs	Title font size.
legend.fs	Legend items font size.
axis.fs	Axis ticks font size.
label.fs	Legend titles and axis names font size.
dot.size	Dot size.
ratio	the vertical/horizontal ratio.
base.v	Height of each plot.
base.h	Width of each plot.
ref.colors	A vector of colors to bypass those automatically chosen by ggplot2 for the tissue areas in the reference plot.

Details

A PDF file is created that contains the index.

Value

Create PDF file and returns ‘NULL‘, invisibly.

Examples

```
data(bsrdm.spa, package = "BulkSignalR")
data(bsrinf.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)

# generate visual index on disk in pdf file
spatialIndexPlot(scores.red[1:2,], annotation.spa,
label.col = "ground_truth",
out.file = "spatialIndexPlot")
```

`spatialPlot` *L-R interaction score spatial display*

Description

Generate a plot with scores at the spatial coordinates of the corresponding sample locations. Not limited to BulkSignalR gene signature scores.

Usage

```
spatialPlot(
  v,
  areas,
  inter.name,
  rev.y = TRUE,
  ref.plot = FALSE,
  ref.plot.only = FALSE,
  image.raster = NULL,
  x.col = "array_col",
  y.col = "array_row",
  label.col = "label",
  idSpatial.col = "idSpatial",
  cut.p = 0.01,
  low.color = "royalblue3",
  mid.color = "white",
  high.color = "orange",
  title.fs = 12,
  legend.fs = 10,
  axis.fs = 10,
  label.fs = 12,
  dot.size = 0.5,
  legend.dot.factor = 10,
  ref.colors = NULL
)
```

Arguments

v	A named vector containing the scores, names must be the IDs of each location.
areas	A data.frame containing at least the x and y coordinates of the locations as well as the unique IDs of spatial locations. In case ref.plot is set to TRUE, a label column is required additionally.
inter.name	Interaction name to display as plot title.
rev.y	A Boolean indicating whether low y coordinates should be at the top of the plot.
ref.plot	A Boolean indicating whether a reference map of the tissue with area labels should be plot aside.
ref.plot.only	A Boolean indicating that only the reference plot should be output.
image.raster	Raster object image to plot raw tissue image as reference.
x.col	Column name in areas containing x coordinates.
y.col	Column name in areas containing y coordinates.
label.col	Column name in areas containing area labels.
idSpatial.col	Column name in areas containing the unique IDs of spatial locations.
cut.p	Proportion of top and bottom values for thresholding.
low.color	Color for low score values.
mid.color	Color for score = 0.
high.color	Color for high score values.
title.fs	Title font size.
legend.fs	Legend items font size.
axis.fs	Axis ticks font size.
label.fs	Legend titles and axis names font size.
dot.size	Dot size.
legend.dot.factor	A factor applied to obtain the legend dot size.
ref.colors	A vector of colors to bypass those automatically chosen by ggplot2 for the tissue areas in the reference plot.

Details

A single (scores) or side-by-side (reference tissue & scores) plot is generated.

Value

A spatial plot

Examples

```
data(bsrinf.spa, package = "BulkSignalR")
data(bsrdm.spa, package = "BulkSignalR")
data(annotation.spa, package = "BulkSignalR")

thres <- 0.01
bsrinf.red <- reduceToBestPathway(bsrinf.spa)
s.red  <- BSRSignature(bsrinf.red, qval.thres=thres)
scores.red <- scoreLRGeneSignatures(bsrdm.spa,s.red)
```

```
inter <- "{SLIT2} / {GPC1}"

spatialPlot(scores.red[inter, ], annotation.spa, inter,
   ref.plot = TRUE, ref.plot.only = FALSE,
   image.raster = NULL, dot.size = 1,
   label.col = "ground_truth")
```

---

summarizedCellularNetwork

Build a summary cellular network

Description

Generate a igraph object with one link between each cell type.

Usage

```
summarizedCellularNetwork(tab)
```

Arguments

tab	The data.frame output by "cellularNetworkTable".

Value

A igraph object containing a summary cellular network with edge weights proportional to the sum of individual link scores. Edge weight are normalized to a total of one.

Examples

```
data(bsrdm, package = "BulkSignalR")
data(bsrinf, package = "BulkSignalR")
data("tme.signatures", package = "BulkSignalR")
data(immune.signatures, package = "BulkSignalR")

immune.signatures <- immune.signatures[immune.signatures$signature %in%
    c("T cells"), ]

signatures <- rbind(immune.signatures, tme.signatures[
    tme.signatures$signature %in% c("Fibroblasts"),
])

tme.scores <- scoreSignatures(bsrdm, signatures)

# assignment
lr2ct <- assignCellTypesToInteractions(bsrdm, bsrinf, tme.scores)

# cellular network
g.table <- cellularNetworkTable(lr2ct[c(1:25),])
gSummary <- summarizedCellularNetwork(g.table)
# plot(gSummary, edge.width=1+30*E(gSummary)$score)
```

tgCorr *Target gene correlations accessor*

Description

Target gene correlations accessor

Target gene correlations accessor

Usage

```
## S4 method for signature 'BSRInference'
tgCorr(x)

## S4 method for signature 'BSRSignature'
tgCorr(x)
```

Arguments

x	BSRSignature

Value

tgCorr

Examples

```
bsr.sig <- new("BSRSignature")
tgCorr(bsr.sig)
```

tgCorr<-,BSRInference-method *Target gene correlations setter (internal use only)*

Description

Target gene correlations setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
tgCorr(x) <- value
```

Arguments

x	BSRInference object
value	value to be set for bsrinf

Value

returns NULL

tgExpr *Target gene expression accessor*

Description

Target gene expression accessor

Target gene expression accessor

Usage

```
## S4 method for signature 'BSRInferenceComp'
tgExpr(x)

## S4 method for signature 'BSRSignatureComp'
tgExpr(x)
```

Arguments

x	BSRSignatureComp object

Value

tgExpr

tg.expr

Examples

```
bsrinf <- new("BSRInferenceComp")
tgExpr(bsrinf)
```

tgExpr<-,BSRInferenceComp-method *Target gene expression setter (internal use only)*

Description

Target gene expression setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInferenceComp'
tgExpr(x) <- value
```

Arguments

x	BSRInferenceComp object
value	value to be set for bsrinf

Value

returns NULL

tgGenes *Target genes accessor*

Description

Target genes accessor

Target genes accessor

Usage

```
## S4 method for signature 'BSRInference'
tgGenes(x)

## S4 method for signature 'BSRSignature'
tgGenes(x)
```

Arguments

x	BSRSignature

Value

tgGenes

tgGenes

Examples

```
bsr.sig <- new("BSRSignature")
tgGenes(bsr.sig)
```

tgGenes<-,BSRInference-method *Target genes setter (internal use only)*

Description

Target genes setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInference'
tgGenes(x) <- value
```

Arguments

x	BSRInference object
value	value to be set BSRInference

Value

returns NULL

tgLogFC *Target gene logFC accessor*

Description

Target gene logFC accessor

Target gene logFC accessor

Usage

```
## S4 method for signature 'BSRInferenceComp'
tgLogFC(x)

## S4 method for signature 'BSRSignatureComp'
tgLogFC(x)
```

Arguments

x	BSRSignatureComp object

Value

tgLogFC

tg.logFC

Examples

```
bsrinf <- new("BSRInferenceComp")
tgLogFC(bsrinf)
```

tgLogFC<-,BSRInferenceComp-method *Target gene logFC setter (internal use only)*

Description

Target gene logFC setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInferenceComp'
tgLogFC(x) <- value
```

Arguments

x	BSRInferenceComp object
value	value to be set for bsrinf

Value

returns NULL

tgPval *Target gene P-values accessor*

Description

Target gene P-values accessor

Target gene P-values accessor

Usage

```
## S4 method for signature 'BSRInferenceComp'
tgPval(x)

## S4 method for signature 'BSRSignatureComp'
tgPval(x)
```

Arguments

x	BSRSignatureComp object

Value

tgPval

tg.pval

Examples

```
bsrinf <- new("BSRInferenceComp")
tgPval(bsrinf)
```

tgPval<-,BSRInferenceComp-method *Target gene P-values setter (internal use only)*

Description

Target gene P-values setter (internal use only)

Usage

```
## S4 replacement method for signature 'BSRInferenceComp'
tgPval(x) <- value
```

Arguments

x	BSRInferenceComp object
value	value to be set for bsrinf

Value

returns NULL

tme.signatures *Tumor microenvironment gene signatures*

Description

A data set containing gene signatures for some immune and stromal cell populations that are present in the microenvironment of a tumor.

Usage

```
data(tme.signatures)
```

Format

A data frame with 209 rows and 2 variables:

gene HUGO gene symbol

signature cell population name

Source

Becht & al., Genome Biol, 2016; Angelova et al., Genome Biol, 2015.

updateInference *Inference updating*

Description

A method to update the data underlying statistical significance estimations prior to rescoring for an existing BSRInferenceComp object (P- and Q-value estimations as well as LR-score).

Usage

```
## S4 method for signature 'BSRInferenceComp'
updateInference(
  obj,
  bsrcc,
  ncounts,
  src.bsrcc = NULL,
  rank.p = 0.55,
  max.pval = 0.01,
  min.logFC = 1,
  min.LR.score = 0,
  neg.receptors = FALSE,
  pos.targets = FALSE,
  neg.targets = FALSE,
  min.t.logFC = 0.5,
  min.positive = 2,
 fdr.proc = c("BH", "Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BY", "ABH",
    "TSBH")
)
```

Arguments

obj	BSRInferenceComp object.
bsrcc	BSRClusterComp object relative to target cells.
ncounts	Matrix counts normalized.
src.bsrcc	BSRClusterComp object relative to source cells.
rank.p	A number between 0 and 1 defining the rank of the last considered target genes.
max.pval	The maximum P-value imposed to both the ligand and the receptor.
min.logFC	The minimum log2 fold-change allowed for both the receptor and the ligand.
min.LR.score	The minimum LR-score allowed for the interaction.
neg.receptors	A logical indicating whether receptors are only allowed to be upregulated (FALSE), or up- and downregulated (TRUE).
pos.targets	A logical imposing that all the network targets must display positive logFC, i.e. logFC >= min.t.logFC.
neg.targets	A logical imposing that all the network targets must display negative logFC, i.e. logFC <= - min.t.logFC.
min.t.logFC	The minimum log2 fold-change allowed for targets in case pos.targets or neg.targets are used.
min.positive	Minimum number of target genes to be found in a given pathway.
fdr.proc	The procedure for adjusting P-values according to mt.rawp2adjp.

Details

A BSRInferenceComp object should be created by calling "BSRInferenceComp"

Value

A BSRInferenceComp object. The main application of this method is to take a "universal" inference obtained by assigning each gene to good logFC, P-values and expression levels whose role is to find all the reachable targets per receptor/pathway, and to update it by using actual logFC, P-values, and expression data. The benefit is to save time when multiple sample comparisons are performed, only one network exploration is necessary. Note that if a restrictive logic such as positive.targets=TRUE is used, the result will be correct provided all the targets were in the initial BSRInferenceComp object. If a restriction on the targets was applied, then the update is likely to miss some targets, i.e., the statistical analysis will be wrong.

In case no L-R interaction is above the required thresholds, the value 'NULL' is returned.

Note that correlations are set to 1 to avoid lengthy computations with scRNA-seq data and multiple cell populations.

The main role of this method is to support our SingleCellSignalR Version 2 package.

Examples

```
data(bsrdm.comp, package = "BulkSignalR")
data(bsrinf.comp, package = "BulkSignalR")
colA <- as.integer(1:2)
colB <- as.integer(3:4)

#bsrdm.comp <- as(bsrdm, "BSRDataModelComp")
```

```
n <- nrow(ncounts(bsrdm.comp))
stats <- data.frame(pval = runif(n),
logFC = rnorm(n, 0, 2),
expr = runif(n, 0, 10))
rownames(stats) <- rownames(ncounts(bsrdm.comp))


# update
stats$pval <- stats$pval / 100
stats$logFC <- stats$logFC + 0.5

bsrcc.2 <- BSRClusterComp(bsrdm.comp, colA, colB, stats)
bsrinf.updated <- updateInference(bsrinf.comp, bsrcc.2,
max.pval = 1, min.logFC = 0.1)
```

Index