Package 'cTRAP'

November 15, 2019

Title Identification of candidate causal perturbations from differential gene expression data

Version 1.4.0

Description Compare differential gene expression results with those from known cellular perturbations (such as gene knock-down, overexpression or small molecules) derived from the Connectivity Map. Such analyses allow not only to infer the molecular causes of the observed difference in gene expression but also to identify small molecules that could drive or revert specific transcriptomic alterations.

Depends R (>= 3.6.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

biocViews DifferentialExpression, GeneExpression, RNASeq, Transcriptomics, Pathways, ImmunoOncology, GeneSetEnrichment

URL https://github.com/nuno-agostinho/cTRAP

BugReports https://github.com/nuno-agostinho/cTRAP/issues

Suggests testthat, knitr, covr, rmarkdown

RoxygenNote 6.1.1

Imports biomaRt, cowplot, data.table, dplyr, fgsea, ggplot2, ggrepel, graphics, httr, limma, methods, pbapply, R.utils, readxl, reshape2, rhdf5, stats, tools, utils

VignetteBuilder knitr

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

git_branch RELEASE_3_10

git_last_commit 17ddee4

git_last_commit_date 2019-10-29

Date/Publication 2019-11-14

Author Bernardo P. de Almeida [aut], Nuno Saraiva-Agostinho [aut, cre], Nuno L. Barbosa-Morais [aut, led]

Maintainer Nuno Saraiva-Agostinho <nunodanielagostinho@gmail.com>

R topics documented:

`analyseDrugSetEnrichment` *Analyse drug set enrichment*

Description

Analyse drug set enrichment

Usage

```
analyseDrugSetEnrichment(sets, stats, col = NULL, nperm = 10000,
  maxSize = 500, ...)
```

Arguments

sets	Named list of characters: named sets containing compound identifiers (obtain drug sets by running `prepareDrugSets()`)
stats	Named numeric vector or either a `similarPerturbations` or a `targetingDrugs` object (obtained after running `rankSimilarPerturbations` or `predictTargetingDrugs`, respectively)
col	Character: name of the column to use for statistics (only required if class of `stats` is either `similarPerturbations` or `targetingDrugs`)
nperm	Number of permutations to do. Minimial possible nominal p-value is about 1/nperm
maxSize	Maximal size of a gene set to test. All pathways above the threshold are excluded.
...	Arguments passed on to `fgsea::fgsea`

minSize Minimal size of a gene set to test. All pathways below the threshold are excluded.

nproc If not equal to zero sets BPPARAM to use nproc workers (default = 0).

gseaParam GSEA parameter value, all gene-level statis are raised to the power of ‘gseaParam‘ before calculation of GSEA enrichment scores.

BPPARAM Parallelization parameter used in bplapply. Can be used to specify cluster to run. If not initialized explicitly or by setting ‘nproc‘ default value ‘bpparam()‘ is used.

Value

Enrichment analysis based on GSEA

See Also

Other functions for drug set enrichment analysis: `loadDrugDescriptors`, `plotDrugSetEnrichment`, `prepareDrugSets`

Examples

```
descriptors <- loadDrugDescriptors()
drugSets <- prepareDrugSets(descriptors)

# Analyse drug set enrichment in ranked targeting drugs for a differential
# expression profile
data("diffExprStat")
gdsc      <- loadExpressionDrugSensitivityAssociation("GDSC")
predicted <- predictTargetingDrugs(diffExprStat, gdsc)

analyseDrugSetEnrichment(drugSets, predicted)
```

as.table.similarPerturbations

Cross Tabulation and Table Creation

Description

Cross Tabulation and Table Creation

Usage

```
## S3 method for class 'similarPerturbations'
as.table(x, ..., clean = TRUE)
```

Arguments

x	similarPerturbations object
...	Extra parameters passed to table
clean	Boolean: only show certain columns (to avoid redundancy)?

Value

Complete table with metadata based on a similarPerturbations object

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions, getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison, plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations, rankSimilarPerturbations

compareAgainstReference

Compare multiple methods and rank reference accordingly

Description

Compare multiple methods and rank reference accordingly

Usage

```
compareAgainstReference(diffExprGenes, reference, method = c("spearman",
  "pearson", "gsea"), geneSize = 150, cellLines = NULL,
  cellLineMean = "auto", rankByAscending = TRUE,
  rankPerCellLine = FALSE)
```

Arguments

diffExprGenes	Numeric: named vector of differentially expressed genes whose names are gene identifiers and respective values are a statistic that represents significance and magnitude of differentially expressed genes (e.g. t-statistics)
reference	Data matrix or perturbationChanges object (CMap perturbations; read prepareCMapPerturbation
method	Character: comparison methods to run (spearman, pearson or gsea); multiple methods can be selected
geneSize	Number: top and bottom number of differentially expressed genes for gene set enrichment (only used if method = gsea)
cellLines	Integer: number of unique cell lines
cellLineMean	Boolean: add a column with the mean score across cell lines? If cellLineMean = "auto" (default) the mean score will be added if more than one cell line is available
rankByAscending	Boolean: rank values based on their ascending (TRUE) or descending (FALSE) order?
rankPerCellLine	Boolean: when ranking results, also rank them based on individual cell lines instead of only focusing on the mean score across cell lines; if cellLineMean = FALSE, individual cell line conditions are always ranked

Value

List of data frame containing the results per methods of comparison

---

convertENSEMBLtoGeneSymbols

Convert ENSEMBL gene identifiers to gene symbols

Description

Convert ENSEMBL gene identifiers to gene symbols

Usage

```
convertENSEMBLtoGeneSymbols(genes, dataset = "hsapiens_gene_ensembl",
  mart = "ensembl")
```

Arguments

genes	Character: ENSEMBL gene identifiers
dataset	Character: biomaRt dataset name
mart	Character: biomaRt database name

Value

Named character vector where names are the input ENSEMBL gene identifiers and the values are the matching gene symbols

cTRAP *cTRAP package*

Description

Compare differential gene expression results with those from big datasets (e.g. CMap), allowing to infer which types of perturbations may explain the observed difference in gene expression.

Details

Input: To use this package, a named vector of differentially expressed gene metric is needed, where its values represent the significance and magnitude of the differentially expressed genes (e.g. t-statistic) and its names are gene symbols.

Workflow: The differentially expressed genes will be compared against selected perturbation conditions by:

- Spearman or Pearson correlation with z-scores of differentially expressed genes after perturbations from CMap. Use function `rankSimilarPerturbations` with `method = "spearman"` or `method = "pearson"`
- Gene set enrichment analysis (GSEA) using the (around) 12 000 genes from CMap. Use function `rankSimilarPerturbations` with `method = gsea`.

Available perturbation conditions for CMap include:

- Cell line(s).
- Perturbation type (gene knockdown, gene upregulation or drug intake).
- Drug concentration.
- Time points.

Values for each perturbation type can be listed with `getCMapPerturbationTypes()`

Output: The output includes a data frame of ranked perturbations based on the associated statistical values and respective p-values.

`dim.perturbationChanges` *Dimensions of a* `perturbationChanges` *object*

Description

Dimensions of a `perturbationChanges` object

Usage

```
## S3 method for class 'perturbationChanges'
dim(x)
```

Arguments

`x`	`perturbationChanges` object

Value

Dimensions of a perturbationChanges object

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions, getCMapPerturbationTypes,
loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

---

dimnames.perturbationChanges

Dimnames of a perturbationChanges *object*

Description

Dimnames of a perturbationChanges object

Usage

```
## S3 method for class 'perturbationChanges'
dimnames(x)
```

Arguments

x	perturbationChanges object

Value

Retrieve dimnames of a perturbationChanges object

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, filterCMapMetadata, getCMapConditions, getCMapPerturbationTypes,
loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

downloadENCODEknockdownMetadata *Download metadata for ENCODE knockdown experiments*

Description

Download metadata for ENCODE knockdown experiments

Usage

```
downloadENCODEknockdownMetadata(cellLine = NULL, gene = NULL)
```

Arguments

cellLine	Character: cell line
gene	Character: target gene

Value

Data frame containing ENCODE knockdown experiment metadata

See Also

Other functions related with using ENCODE expression data: loadENCODEsamples, performDifferentialExpression
prepareENCODEgeneExpression

Examples

```
downloadENCODEknockdownMetadata("HepG2", "EIF4G1")
```

filterCMapMetadata *Filter CMap metadata*

Description

Filter CMap metadata

Usage

```
filterCMapMetadata(metadata, cellLine = NULL, timepoint = NULL,
  dosage = NULL, perturbationType = NULL)
```

Arguments

metadata	Data frame (CMap metadata) or character (respective filepath)
cellLine	Character: cell line (if NULL, all values are loaded)
timepoint	Character: timepoint (if NULL, all values are loaded)
dosage	Character: dosage (if NULL, all values are loaded)
perturbationType	Character: type of perturbation (if NULL, all perturbation types are loaded)

Value

Filtered CMap metadata

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, getCMapConditions, getCMapPerturbationTypes,
loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

Examples

```
## Not run:
cmapMetadata <- loadCMapData("cmapMetadata.txt", "metadata")

## End(Not run)
filterCMapMetadata(cmapMetadata, cellLine="HEPG2", timepoint="2 h",
                   dosage="25 ng/mL")
```

---

getCMapConditions *List available conditions in CMap datasets*

Description

Downloads metadata if not available

Usage

```
getCMapConditions(metadata, cellLine = NULL, timepoint = NULL,
  dosage = NULL, perturbationType = NULL, control = FALSE)
```

Arguments

metadata	Data frame (CMap metadata) or character (respective filepath)
cellLine	Character: cell line (if NULL, all values are loaded)
timepoint	Character: timepoint (if NULL, all values are loaded)
dosage	Character: dosage (if NULL, all values are loaded)
perturbationType	Character: type of perturbation (if NULL, all perturbation types are loaded)
control	Boolean: show controls for perturbation types?

Value

List of conditions in CMap datasets

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapPerturbationTypes
loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

Examples

```
## Not run:
cmapMetadata <- loadCMapData("cmapMetadata.txt", "metadata")

## End(Not run)
getCMapConditions(cmapMetadata)
```

---

getCMapPerturbationTypes

Get perturbation types

Description

Get perturbation types

Usage

```
getCMapPerturbationTypes(control = FALSE)
```

Arguments

control	Boolean: return perturbation types used as control?

Value

Perturbation types and respective codes as used by CMap datasets

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

Examples

```
getCMapPerturbationTypes()
```

loadCMapData *Load CMap data*

Description

Load CMap data (if not found, `file` will be automatically downloaded)

Usage

```
loadCMapData(file, type = c("metadata", "geneInfo", "zscores",
  "compoundInfo"), zscoresID = NULL)
```

Arguments

`file`	Character: path to file
`type`	Character: type of data to load (`metadata`, `geneInfo`, `zscores` or `compoundInfo`)
`zscoresID`	Character: identifiers to partially load z-scores file (for performance reasons; if `NULL`, all identifiers will be loaded)

Value

Metadata as a data table

Note

If `type = "compoundInfo"`, two files from **The Drug Repurposing Hub** will be downloaded containing information about drugs and perturbations. The files will be named `file` with `_drugs` and `_samples` before their extension, respectively.

See Also

Other functions related with the ranking of CMap perturbations: `[.perturbationChanges`, `as.table.similarPerturb`
`dim.perturbationChanges`, `dimnames.perturbationChanges`, `filterCMapMetadata`, `getCMapConditions`,
`getCMapPerturbationTypes`, `loadCMapZscores`, `parseCMapID`, `plot.perturbationChanges`, `plot.referenceComp`
`plotTargetingDrugsVSsimilarPerturbations`, `prepareCMapPerturbations`, `print.similarPerturbations`,
`rankSimilarPerturbations`

Examples

```
# Load CMap metadata (data is automatically downloaded if not available)
cmapMetadata <- loadCMapData("cmapMetadata.txt", "metadata")

# Load CMap gene info
loadCMapData("cmapGeneInfo.txt", "geneInfo")

# Load CMap zscores based on filtered metadata
cmapMetadataKnockdown <- filterCMapMetadata(
  cmapMetadata, cellLine="HepG2",
  perturbationType="Consensus signature from shRNAs targeting the same gene")
loadCMapData("cmapZscores.gctx.gz", "zscores", cmapMetadataKnockdown$sig_id)
```

loadCMapZscores *Load matrix of CMap perturbation's differential expression z-scores*

Description

Load matrix of CMap perturbation's differential expression z-scores

Usage

```
loadCMapZscores(data, perturbationChanges = FALSE, verbose = TRUE)
```

Arguments

data	perturbationChanges object
perturbationChanges	Boolean: convert to perturbationChanges object?
verbose	Boolean: print messages?

Value

Matrix containing CMap perturbation z-scores (genes as rows, perturbations as columns)

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
getCMapPerturbationTypes, loadCMapData, parseCMapID, plot.perturbationChanges, plot.referenceCompari
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

Examples

```
metadata <- loadCMapData("cmapMetadata.txt", "metadata")
metadata <- filterCMapMetadata(metadata, cellLine="HepG2")
perts <- prepareCMapPerturbations(metadata, "cmapZscores.gctx",
                                  "cmapGeneInfo.txt")
zscores <- loadCMapZscores(perts[ , 1:10])
```

loadDrugDescriptors *Load table with drug descriptors*

Description

Load table with drug descriptors

Usage

```
loadDrugDescriptors(source = c("NCI60", "CMap"), type = c("2D", "3D"),
  file = NULL)
```

Arguments

source	Character: molecular descriptors for compounds in NCI60 or CMap
type	Character: load 2D or 3D molecular descriptors
file	Character: filepath to drug descriptors (automatically downloaded if file does not exist)

Value

Data table with drug descriptors

See Also

Other functions for drug set enrichment analysis: analyseDrugSetEnrichment, plotDrugSetEnrichment, prepareDrugSets

Examples

```
loadDrugDescriptors()
```

---

loadENCODEsamples *Load ENCODE samples*

Description

Samples are automatically downloaded if they are not found in the current working directory.

Usage

```
loadENCODEsamples(metadata)
```

Arguments

metadata	Character: ENCODE metadata

Value

List of loaded ENCODE samples

See Also

Other functions related with using ENCODE expression data: downloadENCODEknockdownMetadata, performDifferentialExpression, prepareENCODEgeneExpression

Examples

```
if (interactive()) {
  # Load ENCODE metadata for a specific cell line and gene
  cellLine <- "HepG2"
  gene <- c("EIF4G1", "U2AF2")
  ENCODEmetadata <- downloadENCODEknockdownMetadata(cellLine, gene)

  # Load samples based on filtered ENCODE metadata
  loadENCODEsamples(ENCODEmetadata)
}
```

loadExpressionDrugSensitivityAssociation *Load gene expression and drug sensitivity correlation matrix*

Description

Load gene expression and drug sensitivity correlation matrix

Usage

```
loadExpressionDrugSensitivityAssociation(source, file = NULL)
```

Arguments

source	Character: source (CTRP 2.1, GDSC 7 or NCI60)
file	Character: filepath to gene expression and drug sensitivity association dataset (automatically downloaded if file does not exist)

Value

Correlation matrix between gene expression (rows) and drug sensitivity (columns)

See Also

Other functions related with the prediction of targeting drugs: plot.referenceComparison, plotTargetingDrugsVSsi
predictTargetingDrugs

Examples

```
loadExpressionDrugSensitivityAssociation("GDSC 7")
```

parseCMapID *Parse CMap identifier*

Description

Parse CMap identifier

Usage

```
parseCMapID(id, cellLine = FALSE)
```

Arguments

id	Character: CMap identifier
cellLine	Boolean: if TRUE, return cell line information from CMap identifier; else, return the CMap identifier without the cell line

Value

Character vector with information from CMap identifiers

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
getCMapPerturbationTypes, loadCMapData, loadCMapZscores, plot.perturbationChanges,
plot.referenceComparison, plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations,
print.similarPerturbations, rankSimilarPerturbations

Examples

```
id <- c("CVD001_HEPG2_24H:BRD-K94818765-001-01-0:4.8",
        "CVD001_HEPG2_24H:BRD-K96188950-001-04-5:4.3967",
        "CVD001_HUH7_24H:BRD-A14014306-001-01-1:4.1")
parseCMapID(id, cellLine=TRUE)
parseCMapID(id, cellLine=FALSE)
```

---

performDifferentialExpression

Perform differential gene expression based on ENCODE data

Description

Perform differential gene expression based on ENCODE data

Usage

```
performDifferentialExpression(counts)
```

Arguments

counts	Data frame: gene expression

Value

Data frame with differential gene expression results between knockdown and control

See Also

Other functions related with using ENCODE expression data: downloadENCODEknockdownMetadata,
loadENCODEsamples, prepareENCODEgeneExpression

Examples

```
if (interactive()) {
  # Download ENCODE metadata for a specific cell line and gene
  cellLine <- "HepG2"
  gene <- "EIF4G1"
  ENCODEmetadata <- downloadENCODEknockdownMetadata(cellLine, gene)

  # Download samples based on filtered ENCODE metadata
```

```
  ENCODEsamples <- loadENCODEsamples(ENCODEmetadata)[[1]]

  counts <- prepareENCODEgeneExpression(ENCODEsamples)

  # Remove low coverage (at least 10 counts shared across two samples)
  minReads   <- 10
  minSamples <- 2
  filter <- rowSums(counts[ , -c(1, 2)] >= minReads) >= minSamples
  counts <- counts[filter, ]

  # Convert ENSEMBL identifier to gene symbol
  counts$gene_id <- convertENSEMBLtoGeneSymbols(counts$gene_id)

  # Perform differential gene expression analysis
  diffExpr <- performDifferentialExpression(counts)
}
```

`plot.perturbationChanges` *Plot perturbation comparison against a differential expression profile*

Description

Plot perturbation comparison against a differential expression profile

Usage

```
## S3 method for class 'perturbationChanges'
plot(x, perturbation, diffExprGenes,
  method = c("spearman", "pearson", "gsea"), geneSize = 150,
  genes = c("both", "top", "bottom"), ...)
```

Arguments

`x`	`perturbationChanges` object
`perturbation`	Character (perturbation identifier) or a `similarPerturbations` table (from which the respective perturbation identifiers are retrieved)
`diffExprGenes`	Numeric: named vector of differentially expressed genes whose names are gene identifiers and respective values are a statistic that represents significance and magnitude of differentially expressed genes (e.g. t-statistics)
`method`	Character: method to plot results (`spearman`, `pearson` or `gsea`)
`geneSize`	Number: top and bottom number of differentially expressed genes for gene set enrichment (only used if `method = gsea`)
`genes`	Character: when plotting gene set enrichment analysis (GSEA), plot top genes (`genes = "top"`), bottom genes (`genes = "bottom"`) or both (`genes = "both"`); only used if `method = "gsea"`
`...`	Extra arguments (currently undocumented)

Value

CMap data comparison plots

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.referenceComparison,
plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations,
rankSimilarPerturbations

Examples

```
data("diffExprStat")
data("cmapPerturbationsKD")

compareKD <- rankSimilarPerturbations(diffExprStat, cmapPerturbationsKD)
EIF4G1knockdown <- grep("EIF4G1", compareKD[[1]], value=TRUE)
plot(cmapPerturbationsKD, EIF4G1knockdown, diffExprStat, method="spearman")
plot(cmapPerturbationsKD, EIF4G1knockdown, diffExprStat, method="pearson")
plot(cmapPerturbationsKD, EIF4G1knockdown, diffExprStat, method="gsea")

data("cmapPerturbationsCompounds")
pert <- "CVD001_HEPG2_24H:BRD-A14014306-001-01-1:4.1"
plot(cmapPerturbationsCompounds, pert, diffExprStat, method="spearman")
plot(cmapPerturbationsCompounds, pert, diffExprStat, method="pearson")
plot(cmapPerturbationsCompounds, pert, diffExprStat, method="gsea")

# Multiple cell line perturbations
pert <- "CVD001_24H:BRD-A14014306-001-01-1:4.1"
plot(cmapPerturbationsCompounds, pert, diffExprStat, method="spearman")
plot(cmapPerturbationsCompounds, pert, diffExprStat, method="pearson")

# Currently unsupported!
# plot(cmapPerturbationsCompounds, pert, diffExprStat, method="gsea")
```

---

plot.referenceComparison

Plot data comparison

Description

Plot data comparison

Usage

```
## S3 method for class 'referenceComparison'
plot(x, method = c("spearman", "pearson",
  "gsea", "rankProduct"), n = c(3, 3), showMetadata = TRUE,
  plotNonRankedPerturbations = FALSE, alpha = 0.3, ...)
```

Arguments

x	referenceComparison object: obtained after running rankSimilarPerturbations or predictTargetingDrugs
method	Character: method to plot results (spearman, pearson, gsea or rankProduct)

n	Numeric: number of top and bottom genes to label (if a vector of two numbers is given, the first and second numbers will be used as the number of top and bottom genes to label, respectively)
showMetadata	Boolean: show available metadata information instead of identifiers (if available)?
plotNonRankedPerturbations	Boolean: plot non-ranked data in grey?
alpha	Numeric: transparency
...	Extra arguments currently not used

Value

Plot illustrating the reference comparison

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions, getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations, print.similarPerturbations, rankSimilarPerturbations

Other functions related with the prediction of targeting drugs: loadExpressionDrugSensitivityAssociation, plotTargetingDrugsVSsimilarPerturbations, predictTargetingDrugs

Examples

```
# Example of a differential expression profile
data("diffExprStat")

## Not run:
# Download and load CMap perturbations to compare with
cellLine <- "HepG2"
cmapMetadataKD <- filterCMapMetadata(
  "cmapMetadata.txt", cellLine=cellLine,
  perturbationType="Consensus signature from shRNAs targeting the same gene")

cmapPerturbationsKD <- prepareCMapPerturbations(
  cmapMetadataKD, "cmapZscores.gctx", "cmapGeneInfo.txt", loadZscores=TRUE)

## End(Not run)

# Rank similar CMap perturbations
compareKD <- rankSimilarPerturbations(diffExprStat, cmapPerturbationsKD)

plot(compareKD, "spearman", c(7, 3))
plot(compareKD, "pearson")
plot(compareKD, "gsea")
```

`plotDrugSetEnrichment` *Plot drug set enrichment*

Description

Plot drug set enrichment

Usage

```
plotDrugSetEnrichment(sets, stats, col = "rankProduct_rank",
  selectedSets = NULL)
```

Arguments

`sets`	Named list of characters: named sets containing compound identifiers (obtain drug sets by running `prepareDrugSets()`)
`stats`	Named numeric vector or either a `similarPerturbations` or a `targetingDrugs` object (obtained after running `rankSimilarPerturbations` or `predictTargetingDrugs`, respectively)
`col`	Character: name of the column to use for statistics (only required if class of `stats` is either `similarPerturbations` or `targetingDrugs`)
`selectedSets`	Character: drug sets to plot (if `NULL`, plot all)

Value

List of GSEA plots per drug set

See Also

Other functions for drug set enrichment analysis: `analyseDrugSetEnrichment`, `loadDrugDescriptors`, `prepareDrugSets`

Examples

```
descriptors <- loadDrugDescriptors()
drugSets <- prepareDrugSets(descriptors)

# Analyse drug set enrichment in ranked targeting drugs for a differential
# expression profile
data("diffExprStat")
gdsc      <- loadExpressionDrugSensitivityAssociation("GDSC")
predicted <- predictTargetingDrugs(diffExprStat, gdsc)

plotDrugSetEnrichment(drugSets, predicted)
```

plotTargetingDrugsVSsimilarPerturbations
Plot similar perturbations against predicted targeting drugs

Description

Plot similar perturbations against predicted targeting drugs

Usage

```
plotTargetingDrugsVSsimilarPerturbations(targetingDrugs,
  similarPerturbations, column, labelBy = "pert_iname",
  quantileThreshold = 0.25, showAllScores = FALSE)
```

Arguments

targetingDrugs	targetingDrugs object
similarPerturbations	similarPerturbations object
column	Character: column to plot (must be available in both databases)
labelBy	Character: column in similarPerturbations, its metadata or compound information to be used for labelling
quantileThreshold	Numeric: quantile to use for highlight values within [0, 1]
showAllScores	Boolean: showl all scores? If FALSE, only the best score per compound will be plotted

Value

ggplot2 plot

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges,
plot.referenceComparison, prepareCMapPerturbations, print.similarPerturbations, rankSimilarPerturba

Other functions related with the prediction of targeting drugs: loadExpressionDrugSensitivityAssociation,
plot.referenceComparison, predictTargetingDrugs

predictTargetingDrugs *Predict targeting drugs*

Description

Identify compounds that may target the phenotype associated with a user-provided differential expression profile by comparing such against a correlation matrix of gene expression and drug sensitivity.

Usage

```
predictTargetingDrugs(diffExprGenes, expressionDrugSensitivityCor,
  method = c("spearman", "pearson", "gsea"), geneSize = 150,
  isDrugActivityDirectlyProportionalToSensitivity = NULL)
```

Arguments

`diffExprGenes`	Numeric: named vector of differentially expressed genes whose names are gene identifiers and respective values are a statistic that represents significance and magnitude of differentially expressed genes (e.g. t-statistics)
`expressionDrugSensitivityCor`	Matrix: correlation matrix of gene expression (rows) and drug sensitivity (columns) across cell lines. Pre-prepared gene expression and drug sensitivity associations are available to download using `loadExpressionDrugSensitivityAssociation`.
`method`	Character: comparison methods to run (`spearman`, `pearson` or `gsea`); multiple methods can be selected
`geneSize`	Number: top and bottom number of differentially expressed genes for gene set enrichment (only used if `method = gsea`)
`isDrugActivityDirectlyProportionalToSensitivity`	Boolean: are the values used for drug activity directly proportional to drug sensitivity? See details.

Details

If `isDrugActivityDirectlyProportionalToSensitivity` is set to `NULL` (as by default), the attribute `isDrugMetricDirectlyProportionalToSensitivity` on the object passed as argument `expressionDrugSensitivityCor` is used (objects obtained via `loadExpressionDrugSensitivityAssociation` have the mentioned attribute set).

Value

Data table with correlation or GSEA results comparing differential expression values against gene expression and drug sensitivity associations

GSEA score

Weighted connectivity scores (WTCS) are calculated when `method = "gsea"` (https://clue.io/connectopedia/cmap_algorithms).

See Also

Other functions related with the prediction of targeting drugs: loadExpressionDrugSensitivityAssociation, plot.referenceComparison, plotTargetingDrugsVSsimilarPerturbations

Examples

```
# Example of a differential expression profile
data("diffExprStat")

# Load expression and drug sensitivity association derived from GDSC data
gdsc <- loadExpressionDrugSensitivityAssociation("GDSC 7")

# Predict targeting drugs on a differential expression profile
predictTargetingDrugs(diffExprStat, gdsc)
```

---

prepareCMapPerturbations

Prepare CMap perturbation data

Description

Prepare CMap perturbation data

Usage

```
prepareCMapPerturbations(metadata, zscores, geneInfo,
  compoundInfo = NULL, loadZscores = FALSE)
```

Arguments

metadata	Data frame (CMap metadata) or character (respective filepath to load data from file)
zscores	Data frame (GCTX z-scores) or character (respective filepath to load data from file)
geneInfo	Data frame (CMap gene info) or character (respective filepath to load data from file)
compoundInfo	Data frame (CMap compound info) or character (respective filepath to load data from file)
loadZscores	Boolean: load perturbation z-scores? Not recommended in memory-constrained systems

Value

CMap perturbation data attributes and filename

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions, getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges, plot.referenceComparison, plotTargetingDrugsVSsimilarPerturbations, print.similarPerturbations, rankSimilarPerturbations

Examples

```
## Not run:
metadata <- loadCMapData("cmapMetadata.txt", "metadata")
metadata <- filterCMapMetadata(metadata, cellLine="HepG2")
prepareCMapPerturbations(metadata, "cmapZscores.gctx", "cmapGeneInfo.txt")

## End(Not run)
```

---

prepareDrugSets *Prepare drug sets from a table with compound descriptors*

Description

Prepare drug sets from a table with compound descriptors

Usage

```
prepareDrugSets(table, id = 1, maxUniqueElems = 15)
```

Arguments

table	Data frame: drug descriptors
id	Integer or character: index or name of the column containing identifiers
maxUniqueElems	Numeric: maximum number of unique elements in a descriptor to consider when creating discrete drug sets

Value

Named list of characters: named drug sets with respective compound identifiers as list elements

See Also

Other functions for drug set enrichment analysis: analyseDrugSetEnrichment, loadDrugDescriptors, plotDrugSetEnrichment

Examples

```
descriptors <- loadDrugDescriptors("NCI60")
prepareDrugSets(descriptors)
```

prepareENCODEgeneExpression *Load an ENCODE gene expression data*

Description

Load an ENCODE gene expression data

Usage

```
prepareENCODEgeneExpression(samples)
```

Arguments

samples	List of loaded ENCODE samples

Value

Data frame containing gene read counts

See Also

convertENSEMBLtoGeneSymbols

Other functions related with using ENCODE expression data: downloadENCODEknockdownMetadata, loadENCODEsamples, performDifferentialExpression

Examples

```
if (interactive()) {
  # Load ENCODE metadata for a specific cell line and gene
  cellLine <- "HepG2"
  gene <- "EIF4G1"
  ENCODEmetadata <- loadENCODEknockdownMetadata(cellLine, gene)

  # Load samples based on filtered ENCODE metadata
  ENCODEsamples <- loadENCODEsamples(ENCODEmetadata)[[1]]

  prepareENCODEgeneExpression(ENCODEsamples)
}
```

print.similarPerturbations *Print a* similarPerturbations *object*

Description

Print a similarPerturbations object

Usage

```
## S3 method for class 'similarPerturbations'
print(x, perturbation = NULL, ...)
```

Arguments

x	similarPerturbations object
perturbation	Character (perturbation identifier) or numeric (perturbation index)
...	Extra parameters passed to print

Value

Information on perturbationChanges object or on specific perturbations (if perturbation is set)

See Also

Other functions related with the ranking of CMap perturbations: [.perturbationChanges, as.table.similarPerturb
dim.perturbationChanges, dimnames.perturbationChanges, filterCMapMetadata, getCMapConditions,
getCMapPerturbationTypes, loadCMapData, loadCMapZscores, parseCMapID, plot.perturbationChanges,
plot.referenceComparison, plotTargetingDrugsVSsimilarPerturbations, prepareCMapPerturbations,
rankSimilarPerturbations

---

rankSimilarPerturbations

Rank CMap perturbations' similarity to a differential expression profile

---

Description

Compare differential expression results against CMap perturbations.

Usage

```
rankSimilarPerturbations(diffExprGenes, perturbations,
  method = c("spearman", "pearson", "gsea"), geneSize = 150,
  cellLineMean = "auto", rankPerCellLine = FALSE)
```

Arguments

diffExprGenes	Numeric: named vector of differentially expressed genes whose names are gene identifiers and respective values are a statistic that represents significance and magnitude of differentially expressed genes (e.g. t-statistics)
perturbations	perturbationChanges object: CMap perturbations (check prepareCMapPerturbations)
method	Character: comparison method (spearman, pearson or gsea; multiple methods may be selected at once)
geneSize	Number: top and bottom number of differentially expressed genes for gene set enrichment (only used if method = gsea)
cellLineMean	Boolean: add a column with the mean score across cell lines? If cellLineMean = "auto" (default) the mean score will be added if more than one cell line is available

rankPerCellLine
: Boolean: when ranking results, also rank them based on individual cell lines instead of only focusing on the mean score across cell lines; if `cellLineMean = FALSE`, individual cell line conditions are always ranked

Value

Data table with correlation or GSEA results comparing differential expression values with those associated with CMap perturbations

GSEA score

Weighted connectivity scores (WTCS) are calculated when `method = "gsea"` (https://clue.io/connectopedia/cmap_algorithms).

See Also

Other functions related with the ranking of CMap perturbations: `[.perturbationChanges`, `as.table.similarPerturb`
`dim.perturbationChanges`, `dimnames.perturbationChanges`, `filterCMapMetadata`, `getCMapConditions`, `getCMapPerturbationTypes`, `loadCMapData`, `loadCMapZscores`, `parseCMapID`, `plot.perturbationChanges`, `plot.referenceComparison`, `plotTargetingDrugsVSsimilarPerturbations`, `prepareCMapPerturbations`, `print.similarPerturbations`

Examples

```
# Example of a differential expression profile
data("diffExprStat")

## Not run:
# Download and load CMap perturbations to compare with
cellLine <- c("HepG2", "HUH7")
cmapMetadataCompounds <- filterCMapMetadata(
    "cmapMetadata.txt", cellLine=cellLine, timepoint="24 h",
    dosage="5 \u00B5M", perturbationType="Compound")

cmapPerturbationsCompounds <- prepareCMapPerturbations(
    cmapMetadataCompounds, "cmapZscores.gctx", "cmapGeneInfo.txt",
    "cmapCompoundInfo_drugs.txt", loadZscores=TRUE)

## End(Not run)
perturbations <- cmapPerturbationsCompounds

# Rank similar CMap perturbations (by default, Spearman's and Pearson's
# correlation are used, as well as GSEA with the top and bottom 150 genes of
# the differential expression profile used as reference)
rankSimilarPerturbations(diffExprStat, perturbations)

# Rank similar CMap perturbations using only Spearman's correlation
rankSimilarPerturbations(diffExprStat, perturbations, method="spearman")
```

`[.perturbationChanges` *Subset a* `perturbationChanges` *object*

Description

Subset a `perturbationChanges` object

Usage

```
## S3 method for class 'perturbationChanges'
x[i, j, drop = FALSE, ...]
```

Arguments

`x`	`perturbationChanges` object
`i, j`	Character or numeric indexes specifying elements to extract
`drop`	Boolean: coerce result to the lowest possible dimension?
`...`	Extra parameters passed to `` `[` ``

Value

`perturbationChanges` object with subset data

See Also

Other functions related with the ranking of CMap perturbations: `as.table.similarPerturbations`, `dim.perturbationChanges`, `dimnames.perturbationChanges`, `filterCMapMetadata`, `getCMapConditions`, `getCMapPerturbationTypes`, `loadCMapData`, `loadCMapZscores`, `parseCMapID`, `plot.perturbationChanges`, `plot.referenceComparison`, `plotTargetingDrugsVSsimilarPerturbations`, `prepareCMapPerturbations`, `print.similarPerturbations`, `rankSimilarPerturbations`

Index