Package 'singscore'

April 16, 2019

Type Package

Title Rank-based single-sample gene set scoring method

Version 1.2.2

Description A simple single-sample gene signature scoring method that uses rank-based statistics to analyze the sample's gene expression profile. It scores the expression activities of gene sets at a single-sample level.

biocViews Software, GeneExpression, GeneSetEnrichment,GO,KEGG,GraphAndNetwork

Depends R (>= 3.5),GSEABase

Imports methods, stats, graphics, ggplot2, ggsci, grDevices, ggrepel, plotly, tidyr, plyr, magrittr, reshape, edgeR, RColorBrewer, Biobase, BiocParallel, SummarizedExperiment, matrixStats

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Roxygen list(markdown = TRUE)

Collate 'singscore.R' 'rankAndScoring.R' 'permuTest.R' 'generatNullGeneric.R' 'plot.R' 'plotRankDensityGeneric.R' 'rankGenesGeneric.R' 'simpleScoreGeneric.R'

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

URL https://github.com/DavisLaboratory/singscore

BugReports https://github.com/DavisLaboratory/singscore/issues

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

git_branch RELEASE_3_8

git_last_commit ecfb21e

git_last_commit_date 2018-11-11

Date/Publication 2019-04-15

Author Momeneh Foroutan [aut, ctb], Dharmesh Bhuva [aut, ctb], Ruqian Lyu [aut, cre]

Maintainer Ruqian Lyu <ruqianl@student.unimelb.edu.au>

R topics documented:

`generateNull` *Permutation test for the derived scores of each sample*

Description

This function generates a number of random gene sets that have the same number of genes as the scored gene set. It scores each random gene set and returns a matrix of scores for all samples. The empirical scores are used to calculate the empirical p-values and plot the null distribution. The implementation uses `BiocParallel::bplapply()` for easy access to parallel backends. Note that one should pass the same values to the `upSet`, `downSet`, `centerScore` and `bidirectional` arguments as what they provide for the `simpleScore()` function to generate a proper null distribution.

Usage

```
generateNull(upSet, downSet = NULL, rankData, centerScore = TRUE,
  knownDirection = TRUE, B = 1000, ncores = 1,
  seed = sample.int(1e+06, 1), useBPPARAM = NULL)

## S4 method for signature 'vector,missing'
generateNull(upSet, downSet = NULL, rankData,
  centerScore = TRUE, knownDirection = TRUE, B = 1000, ncores = 1,
  seed = sample.int(1e+06, 1), useBPPARAM = NULL)

## S4 method for signature 'GeneSet,missing'
generateNull(upSet, downSet = NULL, rankData,
  centerScore = TRUE, knownDirection = TRUE, B = 1000, ncores = 1,
  seed = sample.int(1e+06, 1), useBPPARAM = NULL)
```

```
## S4 method for signature 'vector,vector'
generateNull(upSet, downSet = NULL, rankData,
  centerScore = TRUE, knownDirection = TRUE, B = 1000, ncores = 1,
  seed = sample.int(1e+06, 1), useBPPARAM = NULL)

## S4 method for signature 'GeneSet,GeneSet'
generateNull(upSet, downSet = NULL, rankData,
  centerScore = TRUE, knownDirection = TRUE, B = 1000, ncores = 1,
  seed = sample.int(1e+06, 1), useBPPARAM = NULL)
```

Arguments

upSet	GeneSet object or a vector of gene Ids, up-regulated gene set
downSet	GeneSet object or a vector of gene Ids, down-regulated gene
rankData	matrix, outcome of function rankGenes()
centerScore	A Boolean, specifying whether scores should be centered around 0, default as TRUE
knownDirection	A boolean flag, it deterimines whether the scoring method should derive the scores in a directional mannar when the gene signature only contains one set of gene set (passing the gene set via upSet). It is default as TRUE but one can set the argument to be FALSE to derive the score for a single gene set in a undirectional way. This parameter becomes irrelevant when both upSet and downSet are provided.
B	integer, the number of permutation repeats or the number of random gene sets to be generated, default as 1000
ncores,	integer, the number of CPU cores the function can use
seed	integer, set the seed for randomisation
useBPPARAM,	the backend the function uses, if NULL is provided, the function uses the default parallel backend which is the first on the list returned by BiocParallel::registered() i.e BiocParallel::registered()[[1]] for your machine. It can be changed explicitly by passing a BPPARAM

Value

A matrix of empirical scores for all samples

Author(s)

Ruqian Lyu

See Also

Post about BiocParallel browseVignettes("BiocParallel")

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)

# find out what backends can be registered on your machine
```

```
BiocParallel::registered()
# the first one is the default backend
# ncores = ncores <- parallel::detectCores() - 2
permuteResult = generateNull(upSet = toy_gs_up, downSet = toy_gs_dn, ranked,
centerScore = TRUE, B =10, seed = 1, ncores = 1 )
```

`getPvals` *Estimate the empirical p-values*

Description

With null distributions estimated using the `generateNull()` function, p-values are estimated using a one-tailed test. A minimum p-value of 1/B can be achieved with B permutations.

Usage

```
getPvals(permuteResult, scoredf)
```

Arguments

`permuteResult`	A matrix, null distributions for each sample generated using the `generateNull()` function
`scoredf`	A dataframe, the scored results of samples under test generated using the `simpleScore()` function

Value

Estimated p-values for enrichment of the signature in each sample. A p-value of 1/B indicates that the estimated p-value is less than or equal to 1/B.

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
# find out what backends can be registered on your machine
BiocParallel::registered()
# the first one is the default backend, and it can be changed explicitly.
# See vignette for more details
permuteResult = generateNull(upSet = toy_gs_up, downSet = toy_gs_dn, ranked,
B =10, seed = 1, useBPPARAM = NULL)

# call the permutation function to generate the empirical scores
# for B times.
pvals <- getPvals(permuteResult,scoredf)
```

`plotDispersion` *Plot the score v.s. despersion for all samples*

Description

This function takes the output from the simpleScore() function and generates scatter plots of score vs. dispersion for the total score, the up score and the down score of samples. If you wish to use the plotting function but with some customized inputs (instead of outputs from `simpleScore` function), you need to make sure the formats are the same. To be specific, you need to have columns names "TotalScore" "TotalDispersion" "UpScore" "UpDispersion" "DownScore" "DownDispersion" and rows names as samples.

Usage

```
plotDispersion(scoredf, annot = NULL, alpha = 1, size = 1,
  textSize = 1.5, isInteractive = FALSE)
```

Arguments

`scoredf`	data.frame, generated using the `simpleScore()` function
`annot`	annot any numeric or factor annotation provided by the user that needs to be plot. Annotations must be ordered in the same way as the scores
`alpha`	numeric, set the transparency of points
`size`	numeric, set the size of each point
`textSize`	numeric, relative text sizes for title, labels, and axis values
`isInteractive`	Boolean, determine whether the plot is interactive

Value

A ggplot object

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
plotDispersion(scoredf)
plotDispersion(scoredf, isInteractive = TRUE)
```

`plotNull` *Plot the empirically estimated null distribution and associated p-values*

Description

This function takes the results from function `generateNull()` and plots the density curves of permuted scores for the provided samples via `sampleNames` parameter. It can plot null distribution(s) for a single sample or multiple samples.

Usage

```
plotNull(permuteResult, scoredf, pvals, sampleNames = NULL,
  cutoff = 0.01, textSize = 2, labelSize = 5)
```

Arguments

permuteResult	A matrix, null distributions for each sample generated using the generateNull() function
scoredf	A dataframe, singscores generated using the simpleScore() function
pvals	A vector, estimated p-values using the getPvals() function permuteResult,scoredf and pvals are the results for the same samples.
sampleNames	A character vector, sample IDs for which null distributions will be plotted
cutoff	numeric, the cutoff value for determining significance
textSize	numeric, size of axes labels, axes values and title
labelSize	numeric, size of label texts

Value

a ggplot object

Author(s)

Ruqian Lyu

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
# find out what backends can be registered on your machine
BiocParallel::registered()
# the first one is the default backend, and it can be changed explicitly.
permuteResult = generateNull(upSet = toy_gs_up, downSet = toy_gs_dn, ranked,
B =10, seed = 1,useBPPARAM = NULL)
# call the permutation function to generate the empirical scores
#for B times.
pvals <- getPvals(permuteResult,scoredf)
# plot for all samples
plotNull(permuteResult,scoredf,pvals,sampleNames = names(pvals))
#plot for the first sample
plotNull(permuteResult,scoredf,pvals,sampleNames = names(pvals)[1])
```

---

plotRankDensity *Plot the densities of ranks for one sample*

Description

This function takes a single-column data frame, which is a single-column subset of the ranked matrix data generated using rankGenes() function, and the gene sets of interest as inputs. It plots the density of ranks for genes in the gene set and overlays a barcode plot of these ranks. Ranks are normalized by dividing them by the maximum rank. Densities are estimated using KDE.

Usage

```
plotRankDensity(rankData, upSet, downSet = NULL, isInteractive = FALSE,
  textSize = 1.5)

## S4 method for signature 'ANY,vector,missing'
plotRankDensity(rankData, upSet,
  downSet = NULL, isInteractive = FALSE, textSize = 1.5)

## S4 method for signature 'ANY,GeneSet,missing'
plotRankDensity(rankData, upSet,
  downSet = NULL, isInteractive = FALSE, textSize = 1.5)

## S4 method for signature 'ANY,vector,vector'
plotRankDensity(rankData, upSet,
  downSet = NULL, isInteractive = FALSE, textSize = 1.5)

## S4 method for signature 'ANY,GeneSet,GeneSet'
plotRankDensity(rankData, upSet,
  downSet = NULL, isInteractive = FALSE, textSize = 1.5)
```

Arguments

rankData	one column of the ranked gene expression matrix obtained from the rankGenes() function, use drop = FALSE when subsetting the ranked gene expression matrix, see examples.
upSet	GeneSet object or a vector of gene Ids, up-regulated gene set
downSet	GeneSet object or a vector of gene Ids, down-regulated gene set
isInteractive	Boolean, determine whether the returned plot is interactive
textSize	numberic, set the size of text on the plot

Value

A ggplot object (or a plotly object) with a rank density plot overlayed with a barcode plot

Examples

```
ranked <- rankGenes(toy_expr_se)
plotRankDensity(ranked[,2,drop = FALSE], upSet = toy_gs_up)
```

---

plotScoreLandscape *Plot landscape of two gene signatures scores*

Description

This function takes two data frames which are outputs from the simpleScore() function and plots the relationship between the two gene set scores for samples in the gene expression matrix.Scoredf1 and Scoredf2 are two scoring results of the same set of samples against two different gene signatures. If you wish to use the plotting function but with some customized inputs (instead of outputs from the simpleScore function), you need to make sure the formats are the same To be specific, you need to have column names "TotalScore" "TotalDispersion" "UpScore" "UpDispersion" "DownScore" "DownDispersion" and rows names as samples.

Usage

```
plotScoreLandscape(scoredf1, scoredf2, scorenames = c(),
  textSize = 1.5, isInteractive = FALSE, hexMin = 100)
```

Arguments

scoredf1	data.frame, result of the simpleScore() function which scores the gene expression matrix against a gene set of interest
scoredf2	data.frame, result of the simpleScore() function which scores the gene expression matrix against another gene set of interest
scorenames	character vector of length 2, names for the two scored gene set/signatures stored in scoredf1 and scoredf2
textSize	numeric, set the text size for the plot, default as 1.5
isInteractive	boolean, whether the plot is interactive default as FALSE
hexMin	integer, the threshold which decides whether hex bin plot or scatter plot is displayed, default as 100

Value

A ggplot object, a scatter plot, demonstrating the relationship between scores from two signatures on the same set of samples.

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
scoredf2 <- simpleScore(ranked, upSet = toy_gs_up)
plotScoreLandscape(scoredf, scoredf2)
```

---

projectScoreLandscape *Project data on the landscape plot obtained from* plotScoreLandscape()

Description

This function takes the output (ggplot object) of the function plotScoreLandscape() and a new dataset. It projects the new data points onto the landscape plot and returns a new ggplot object with projected data points.

Usage

```
projectScoreLandscape(plotObj = NULL, scoredf1, scoredf2,
  subSamples = NULL, sampleLabels = NULL, annot = NULL,
  isInteractive = FALSE)
```

Arguments

plotObj	a dataframe, resulted from plotScoreLandscape()
scoredf1	data.frame, result of the simpleScore() function which scores the gene expression matrix against a gene set of interest
scoredf2	data.frame, result of the simpleScore() function which scores the gene expression matrix against another gene set of interest. Scores in scoredf1 and scoredf2 consist of the new data points that will be projected on the plotObj landscape plot.
subSamples	vector of character or indices for subsetting the scoredfs, default as NULL and all samples in scoredfs will be plotted. The subsetted samples are projected onto the landscape plot of plotObj.
sampleLabels	vector of character, sample names to display, ordered in the same way as samples are ordered in the 'scoredfs' data matrix, default as NULL which means the projected points are not labelled.
annot	vector of characters, annotations used to colour the data and should have the same number of samples as in scoredfs
isInteractive	boolean, whether the plot is interactive default as FALSE

Value

New data points on the already plotted ggplot object from plotScoreLanscape()

See Also

plotScoreLandscape()

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf1 <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
scoredf2 <- simpleScore(ranked, upSet = toy_gs_up)
psl <- plotScoreLandscape(scoredf1, scoredf2)
projectScoreLandscape(psl,scoredf1, scoredf2)
```

---

rankGenes *Rank genes by the gene expression intensities*

Description

The rankGenes function is a generic function that can deal with mutilple types of inputs. Given a matrix of gene expression that has samples in columns, genes in rows, and values being gene expression intensity,rankGenes ranks gene expression intensities in each sample.

It can also work with S4 objects that have gene expression matrix as a component (i.e ExpressionSet, DGEList,SummarizedExperiment). It calls the rank function in the base package which ranks the gene expression matrix by its absolute expression level. If the input is S4 object of DGEList, ExpressionSet, or SummarizedExperiment, it will extract the gene expression matrix from the object and rank the genes. The default 'tiesMethod' is set to 'min'.

Usage

```
rankGenes(expreMatrix, tiesMethod = "min")

## S4 method for signature 'matrix'
rankGenes(expreMatrix, tiesMethod = "min")

## S4 method for signature 'data.frame'
rankGenes(expreMatrix, tiesMethod = "min")

## S4 method for signature 'DGEList'
rankGenes(expreMatrix, tiesMethod = "min")

## S4 method for signature 'ExpressionSet'
rankGenes(expreMatrix, tiesMethod = "min")

## S4 method for signature 'SummarizedExperiment'
rankGenes(expreMatrix,
  tiesMethod = "min")
```

Arguments

expreMatrix	A gene expression matrix (matrix,data.frame) or S4 object (ExpressionSet,DGEList, SummarizedExperiment)
tiesMethod	A character indicating what method to use when dealing with ties

Value

The ranked gene expression matrix that has samples in columns and genes in rows

See Also

rank ExpressionSet SummarizedExperiment DGEList

Examples

```
rankGenes(toy_expr_se) # toy_expr_se is a gene expression dataset
tiesMethod = 'min'
# get counts from toy_expr_se
counts <- SummarizedExperiment::assay(toy_expr_se)
# or it can be a ExpressionSet object

e <- Biobase::ExpressionSet(assayData = as.matrix(counts))
rankGenes(e)
```

---

scoredf_ccle_epi *Pre-computed scores of the CCLE dataset against an epithelial gene signature*

Description

This data.frame stores pre-computed scores of the CCLE dataset Barretina et al calculated using the `simpleScore()` function against the epithelial gene signature from Tan, Tuan Zea et al. The data.frame has scores for 55 samples. Please refer to the vignettes for instructions on how to obtain the full datasets.

Usage

```
scoredf_ccle_epi
```

Format

An object of class `data.frame` with 55 rows and 2 columns.

References

Barretina, Jordi, Giordano Caponigro, Nicolas Stransky, Kavitha Venkatesan, Adam A Margolin, Sungjoon Kim, Christopher J Wilson, et al. 2012. "The Cancer Cell Line Encyclopedia Enables Predictive Modelling of Anticancer Drug Sensitivity." Nature 483 (7391): 603–7.

Tan, Tuan Zea, Qing Hao Miow, Yoshio Miki, Tetsuo Noda, Seiichi Mori, Ruby Yun-Ju Huang, and Jean Paul Thiery. 2014–10AD. "Epithelial-Mesenchymal Transition Spectrum Quantification and Its Efficacy in Deciphering Survival and Drug Responses of Cancer Patients." EMBO Molecular Medicine 6 (10). Oxford, UK: BlackWell Publishing Ltd: 1279–93. doi:10.15252/emmm.201404208.

See Also

scoredf_ccle_mes

`scoredf_ccle_mes` *Pre-computed scores of the CCLE dataset against a mesenchymal gene signature*

Description

This data.frame stores pre-computed scores of the CCLE dataset Barretina et al calculated using the `simpleScore()` function against the mesenchymal gene signature from Tan, Tuan Zea et al. The data.frame has scores for 55 samples. Please refer to the vignettes for instructions on how to obtain the full datasets.

Usage

```
scoredf_ccle_mes
```

Format

An object of class `data.frame` with 55 rows and 2 columns.

References

Barretina, Jordi, Giordano Caponigro, Nicolas Stransky, Kavitha Venkatesan, Adam A Margolin, Sungjoon Kim, Christopher J Wilson, et al. 2012. "The Cancer Cell Line Encyclopedia Enables Predictive Modelling of Anticancer Drug Sensitivity." Nature 483 (7391): 603–7.

Tan, Tuan Zea, Qing Hao Miow, Yoshio Miki, Tetsuo Noda, Seiichi Mori, Ruby Yun-Ju Huang, and Jean Paul Thiery. 2014–10AD. "Epithelial-Mesenchymal Transition Spectrum Quantification and Its Efficacy in Deciphering Survival and Drug Responses of Cancer Patients." EMBO Molecular Medicine 6 (10). Oxford, UK: BlackWell Publishing Ltd: 1279–93 doi:10.15252/emmm.201404208.

See Also

scoredf_ccle_epi

`scoredf_tcga_epi` *Pre-computed scores of the TCGA breast cancer gene expression matrix against an epithelial signature*

Description

This data.frame stores pre-computed scores of the TCGA dataset calculated using the `simpleScore()` function against the epithelial gene signature from Tan, Tuan Zea et al. Please refer to the vignettes for instructions on how to obtain the full datasets.

Usage

```
scoredf_tcga_epi
```

Format

An object of class `data.frame` with 1119 rows and 2 columns.

References

Tan, Tuan Zea, Qing Hao Miow, Yoshio Miki, Tetsuo Noda, Seiichi Mori, Ruby Yun-Ju Huang, and Jean Paul Thiery. 2014–10AD. "Epithelial-Mesenchymal Transition Spectrum Quantification and Its Efficacy in Deciphering Survival and Drug Responses of Cancer Patients." EMBO Molecular Medicine 6 (10). Oxford, UK: BlackWell Publishing Ltd: 1279–93 doi:10.15252/emmm.201404208.

See Also

scoredf_tcga_mes

scoredf_tcga_mes *Pre-computed scores of the TCGA breast cancer gene expression matrix against a mesenchymal signature*

Description

This data.frame stores pre-computed scores of the TCGA dataset calculated using the `simpleScore()` function against the mesenchymal gene signature from Tan, Tuan Zea et al. Please refer to the vignettes for instructions on how to obtain the full datasets.

Usage

```
scoredf_tcga_mes
```

Format

An object of class `data.frame` with 1119 rows and 2 columns.

References

Tan, Tuan Zea, Qing Hao Miow, Yoshio Miki, Tetsuo Noda, Seiichi Mori, Ruby Yun-Ju Huang, and Jean Paul Thiery. 2014–10AD. "Epithelial-Mesenchymal Transition Spectrum Quantification and Its Efficacy in Deciphering Survival and Drug Responses of Cancer Patients." EMBO Molecular Medicine 6 (10). Oxford, UK: BlackWell Publishing Ltd: 1279–93 doi:10.15252/emmm.201404208.

See Also

scoredf_tcga_epi

simpleScore *single-sample gene-set scoring method*

Description

This function computes 'singscores' using a ranked gene expression matrix obtained from the `rankGenes()` function and a gene set or a pair of up-regulated and down-regulated gene sets. It returns a data.frame of scores and dispersions for each sample. The gene sets can be in vector format or as GeneSet objects (from GSEABase packages). If samples need to be scored against a single gene set, the `upSet` argument should be used to pass the gene set while the `downSet` argument is set to `NULL`. This setting is ideal for gene sets representing gene ontologies where the nature of the genes is unknown (up- or down-regulated).

Usage

```
simpleScore(rankData, upSet, downSet = NULL, subSamples = NULL,
  centerScore = TRUE, dispersionFun = mad, knownDirection = TRUE)

## S4 method for signature 'ANY,vector,missing'
simpleScore(rankData, upSet,
  downSet = NULL, subSamples = NULL, centerScore = TRUE,
```

```
  dispersionFun = mad, knownDirection = TRUE)

## S4 method for signature 'ANY,GeneSet,missing'
simpleScore(rankData, upSet,
  downSet = NULL, subSamples = NULL, centerScore = TRUE,
  dispersionFun = mad, knownDirection = TRUE)

## S4 method for signature 'ANY,vector,vector'
simpleScore(rankData, upSet,
  downSet = NULL, subSamples = NULL, centerScore = TRUE,
  dispersionFun = mad, knownDirection = TRUE)

## S4 method for signature 'ANY,GeneSet,GeneSet'
simpleScore(rankData, upSet,
  downSet = NULL, subSamples = NULL, centerScore = TRUE,
  dispersionFun = mad, knownDirection = TRUE)
```

Arguments

rankData	A matrix object, ranked gene expression matrix data generated using the rankGenes() function
upSet	A GeneSet object or character vector of gene IDs of up-regulated gene set or a gene set where the nature of genes is not known
downSet	A GeneSet object or character vector of gene IDs of down-regulated gene set or NULL where only a single gene set is provided
subSamples	A vector of sample labels/indices that will be used to subset the rankData matrix. All samples will be scored if not provided
centerScore	A Boolean, specifying whether scores should be centered around 0, default as TRUE
dispersionFun	A function, dispersion function with default being mad
knownDirection	A boolean flag, it deterimines whether the scoring method should derive the scores in a directional mannar when the gene signature only contains one set of gene set (passing the gene set via upSet). It is default as TRUE but one can set the argument to be FALSE to derive the score for a single gene set in a undirectional way. This parameter becomes irrelevant when both upSet and downSet are provided.

Value

A data.frame consists of singscores and dispersions for all samples

See Also

rank "GeneSet"

Examples

```
ranked <- rankGenes(toy_expr_se)
scoredf <- simpleScore(ranked, upSet = toy_gs_up, downSet = toy_gs_dn)
# toy_gs_up is a GeneSet object, alternatively a vector of gene ids may also
# be supplied.
```

`singscore` *singscore: A package for deriving gene-set scores at a single sample level*

Description

The package provides functions for calculating gene-set enrichment scores at a single-sample level using gene expression data. It includes functions to perform hypothesis testing and provides visualisations to enable diagnosis of scores and gene sets along with visualisations to enable exploration of results.

`tgfb_expr_10_se` *An example gene expression dataset*

Description

A microarray gene expression dataset that was originally obtained from the integrated TGFb-EMT data published by (Foroutan et al, 2017). (ComBat corrected values). `tgfb_expr_10` is a subset of the integrated TGFb-EMT data consisting of 10 samples (4 TGFb treated and 6 controls) each with expression values for 11900 genes.

Usage

```
tgfb_expr_10_se
```

Format

A SummarizedExperiment object

Source

Foroutan et al,2017

References

Foroutan, Momeneh, Joseph Cursons, Soroor Hediyeh-Zadeh, Erik W Thompson, and Melissa J Davis. 2017. "A Transcriptional Program for Detecting Tgfbeta-Induced Emt in Cancer." Molecular Cancer Research. American Association for Cancer Research. doi:10.1158/1541-7786.MCR-16-0313.

tgfb_gs_dn *Gene set of down-regulated genes for the TGFb-induced EMT gene signature*

Description

A GeneSet object that contains the down-regulated genes of the TGFb-induced EMT gene signature that was derived by (Foroutan et al,2017), using two meta-analysis techniques. The gene signature contains an up-regulated gene set (up-set) and a down-regulated gene set (down-set). Please refer to the vignettes for the steps to acquire the exact data object.

Usage

```
tgfb_gs_dn
```

Format

A GeneSet object

Source

Foroutan et al,2017

References

Foroutan, Momeneh, Joseph Cursons, Soroor Hediyeh-Zadeh, Erik W Thompson, and Melissa J Davis. 2017. "A Transcriptional Program for Detecting Tgfbeta-Induced Emt in Cancer." Molecular Cancer Research. American Association for Cancer Research. doi:10.1158/1541-7786.MCR-16-0313.

See Also

"GeneSet",tgfb_gs_up

tgfb_gs_up *Gene set of up-regulated genes for the TGFb-induced EMT gene signature*

Description

A GeneSet object that contains the up-regulated genes of the TGFb-induced EMT gene signature that was derived by (Foroutan et al.,2017), using two meta-analysis techniques. The gene signature contains an up-regulated gene set (up-set) and a down-regulated gene set (down-set). Please refer to the vignettes for the steps to acquire the exact data object.

Usage

```
tgfb_gs_up
```

Format

A GeneSet object

Source

Foroutan et al,2017

References

Foroutan, Momeneh, Joseph Cursons, Soroor Hediyeh-Zadeh, Erik W Thompson, and Melissa J Davis. 2017. "A Transcriptional Program for Detecting Tgfbeta-Induced Emt in Cancer." Molecular Cancer Research. American Association for Cancer Research. doi:10.1158/1541-7786.MCR-16-0313.

See Also

"GeneSet",tgfb_gs_dn

---

toy_expr_se *A toy gene expression dataset of two samples*

Description

A toy dataset consisting of 2 samples with the expression values of 20 genes. The data was created by sampling 2 samples and 20 genes from the dataset by Foroutan et al, 2017.

Usage

```
toy_expr_se
```

Format

A SummarizedExperiment of 2 samples each with 20 genes

D_Ctrl_R1 a control sample

D_TGFb_R1 a TGFb-treated sample

Source

Foroutan et al.,2017

References

Foroutan, Momeneh, Joseph Cursons, Soroor Hediyeh-Zadeh, Erik W Thompson, and Melissa J Davis. 2017. "A Transcriptional Program for Detecting Tgfbeta-Induced Emt in Cancer." Molecular Cancer Research. American Association for Cancer Research. doi:10.1158/1541-7786.MCR-16-0313.

toy_gs_dn *A gene set object of down-regulated genes for the toy dataset*

Description

A GeneSet object with 5 genes randomly selected from the toy dataset. These genes are independent of those in toy_gs_up

Usage

```
toy_gs_dn
```

Format

A GSEABase::GeneSet object with 5 genes

See Also

"GeneSet",toy_expr_se,toy_gs_up

toy_gs_up *A gene set object of up-regulated genes for the toy dataset*

Description

A GeneSet object with 5 genes randomly selected from the toy dataset. These genes are independent of those in toy_gs_dn

Usage

```
toy_gs_up
```

Format

A GeneSet object with 5 genes

See Also

"GeneSet",toy_expr_se,toy_gs_dn

Index