Package ‘gespeR’

May 29, 2024

**Imports** Matrix, glmnet, cellHTS2, Biobase, biomaRt, doParallel, parallel, foreach, reshape2, dplyr

**Depends** methods, graphics, ggplot2, R(>= 2.10)

**Suggests** knitr

**biocViews** ImmunoOncology, CellBasedAssays, Preprocessing, GeneTarget, Regression, Visualization

**VignetteBuilder** knitr

**Type** Package

**Lazyload** yes

**Title** Gene-Specific Phenotype EstimatoR

**Version** 1.36.0

**Date** 2015-07-22

**Author** Fabian Schmich

**Maintainer** Fabian Schmich <fabian.schmich@bsse.ethz.ch>

**Description** Estimates gene-specific phenotypes from off-target confounded RNAi screens. The phenotype of each siRNA is modeled based on on-targeted and off-targeted genes, using a regularized linear regression model.

**License** GPL-3

**URL** http://www.cbg.ethz.ch/software/gespeR

**Collate** 'Phenotypes-class.R' 'TargetRelations-class.R'

'gespeR-class.R' 'gespeR-concordance.R' 'gespeR-functions.R'

'gespeR-generics.R' 'gespeR-methods.R' 'gespeR-package.R'

**git_url** https://git.bioconductor.org/packages/gespeR

**git_branch** RELEASE_3_19

**git_last_commit** a1e0cc3

**git_last_commit_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-29
Package: Gene-Specific Phenotype EstimatoR

Description

This package provides a model to deconvolute off-target confounded RNAi knockdown phenotypes, and methods to investigate concordance between ranked lists of (estimated) phenotypes. The regularized linear regression model can be fitted using two different strategies. (a) Cross-validation over regularization parameters optimising the mean-squared-error of the model and (b) stability selection of covariates (genes) based on a method by Nicolai Meinshausen et al.

Author(s)

Fabian Schmich | Computational Biology Group, ETH ZURICH | <fabian.schmich@bsse.ethz.ch>
References

Fabian Schmich et. al, Deconvoluting Off-Target Confounded RNA Interference Screens (2014).

See Also
gespeR

Examples

```r
# Read phenotypes
phenos <- lapply(LETTERS[1:4], function(x) {
  sprintf("Phenotypes_screen_%s.txt", x)
})
phenos <- lapply(phenos, function(x) {
  Phenotypes(system.file("extdata", x, package="gespeR"),
  type = "SSP",
  col.id = 1,
  col.score = 2)
})
phenos
plot(phenos[[1]])

# Read target relations
tr <- lapply(LETTERS[1:4], function(x) {
  sprintf("TR_screen_%s.rds", x)
})
tr <- lapply(tr, function(x) {
  TargetRelations(system.file("extdata", x, package="gespeR"))
})
tr[[1]]
tempfile <- paste(tempfile(pattern = "file", tmpdir = tempdir()), ".rds", sep="")
tr[[1]] <- unloadValues(tr[[1]], writeValues = TRUE, path = tempfile)
tr[[1]]
tr[[1]] <- loadValues(tr[[1]])
tr[[1]]

# Fit gespeR models with cross validation
res.cv <- lapply(1:length(phenos), function(i) {
  gespeR(phenotypes = phenos[[i]],
  target.relations = tr[[i]],
  mode = "cv",
  alpha = 0.5,
  ncores = 1)
})
summary(res.cv[[1]])
res.cv[[1]]
plot(res.cv[[1]])

# Extract scores
ssp(res.cv[[1]])
gsp(res.cv[[1]])
head(scores(res.cv[[1]]))
```
# Fit gespeR models with stability selection
res.stab <- lapply(1:length(phenos), function(i) {
  gespeR(phenotypes = phenos[[i]],
          target.relations = tr[[i]],
          mode = "stability",
          nbootstrap = 100,
          fraction = 0.67,
          threshold = 0.75,
          EV = 1,
          weakness = 0.8,
          ncores = 1)
})
summary(res.stab[[1]])
res.stab[[1]]
plot(res.stab[[1]])

# Extract scores
ssp(res.stab[[1]])
gsp(res.stab[[1]])
head(scores(res.stab[[1]]))

# Compare concordance between stability selected GSPs and SSPs
conc.gsp <- concordance(lapply(res.stab, gsp))
conc.ssp <- concordance(lapply(res.stab, ssp))
pl.gsp <- plot(conc.gsp) + ggtitle("GSPs\n")
pl.ssp <- plot(conc.ssp) + ggtitle("SSPs\n")

if (require(grid)) {
  grid.newpage()
  pushViewport(viewport(layout = grid.layout(1, 2) ) )
  print(pl.gsp, vp = viewport(layout.pos.row = 1, layout.pos.col = 1))
  print(pl.ssp, vp = viewport(layout.pos.row = 1, layout.pos.col = 2))
} else {
  plot(pl.gsp)
  plot(pl.ssp)
}

---

**Description**

Query Biomart HGNC symbols for the entrez identifiers of estimated GSPs. Currently, only implemented for species "hsapiens".

**Usage**

```r
## S4 method for signature 'Phenotypes'
```
annotate.gsp(object, organism = "hsapiens")

## S4 method for signature 'gespeR'
annotate.gsp(object, organism = "hsapiens")

**Arguments**

- **object**: A `gespeR` or `Phenotypes` object
- **organism**: String indicating the `biomaRt` organism

**Value**

data.frame containing gene identifier, gene symbol and phenotypic score

**Author(s)**

Fabian Schmich

**See Also**

- `gsp`
- `ssp`
- `scores`

**Examples**

data(stabilityfits)
gspA <- gsp(stabilityfits$A)
## Not run:
annotate.gsp(gspA)
## End(Not run)

---

**as.data.frame,Phenotypes-method**

_Convert Phenotypes object to a data.frame_

**Description**

Convert Phenotypes object to a data.frame

**Usage**

## S4 method for signature 'Phenotypes'
as.data.frame(x)
Arguments

x  A Phenotypes object

Value

A data.frame

Author(s)

Fabian Schmich

Examples

phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
                     type = "SSP",
                     col.id = 1,
                     col.score = 2)
as.data.frame(phenos)

as.data.frame.concordance

Description

Coerce method

Usage

## S3 method for class 'concordance'
as.data.frame(x, ...)

Arguments

x  concordance object

...  additional arguments

Value

data.frame

Author(s)

Fabian Schmich
c,Phenotypes-method

Concatenate Phenotypes objects

Description

Concatenate Phenotypes objects

Usage

## S4 method for signature 'Phenotypes'
c(x, ..., recursive = FALSE)

Arguments

x                     A Phenotypes object
...                   additional Phenotypes objects
recursive              recursive

Value

A concatenated Phenotypes object

Author(s)

Fabian Schmich

Examples

phenos.a <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)
phenos.b <- Phenotypes(system.file("extdata", "Phenotypes_screen_B.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)
c(phenos.a, phenos.b)
**concordance**

*Evaluate the concordance between Phenotype objects*

**Description**

Measures include the correlation (rho) between pairs of phenotypes for the same gene, the rank biased overlap (rbo) of the top and bottom of ranked lists, and the Jaccard index (J) of selected genes.

**Usage**

`concordance(..., min.overlap = 10, cor.method = "spearman", rbo.p = 0.98, rbo.k = NULL, rbo.mid = 0, uneven.lengths = TRUE)`

**Arguments**

- `...`: The phenotypes to be evaluated for concordance
- `min.overlap`: The minimum number of overlapping genes required
- `cor.method`: A character string indicating which correlation coefficient is to be computed
- `rbo.p`: The weighting parameter for rank biased overlap (rbo) in \([0, 1]\). High p implies strong emphasis on top ranked elements
- `rbo.k`: The evaluation depth for rank biased overlap extrapolation
- `rbo.mid`: The mid point to split a ranked list, e.g. in order to split positive and negative scores choose mid=0
- `uneven.lengths`: Indicator if lists have uneven lengths

**Value**

A `concordance` object with the following elements:

- `pair.test`: Indicator of compared phenotypes
- `cor`: The correlation between pairs of phenotypes for the same gene
- `rbo.top`: The rank biased overlap of genes evaluated at the top of the ranked list
- `rbo.bottom`: The rank biased overlap of genes evaluated at the bottom of the ranked list
- `jaccard`: The Jaccard index of selected genes

**Author(s)**

Fabian Schmich

**See Also**

- `Phenotypes`
- `plot.concordance`
- `rbo`
Examples

```r
data(stabilityfits)
conc <- concordance(gsp(stabilityfits$A), gsp(stabilityfits$B),
gsp(stabilityfits$C), gsp(stabilityfits$D))
plot(conc)
```

---

**dim,Phenotypes-method**  
*Dimension of a Phenotypes object*

**Description**

Dimension of a Phenotypes object

**Usage**

```r
## S4 method for signature 'Phenotypes'
dim(x)
```

**Arguments**

- `x`  
  Phenotypes object

**Value**

Dimension of the Phenotypes object

**Author(s)**

Fabian Schmich

---

**gespeR-class**  
*gespeR*

**Description**

Class that represents a gespeR model. It contains a SSP Phenotypes and TargetRelations representing a siRNA knockdown experiment. When the model is fitted, it additionaly contains estimated GSP Phenotypes.
Usage

gespeR(phenotypes, target.relations, ...)

## S4 method for signature 'Phenotypes,TargetRelations'
gespeR(phenotypes, target.relations, 
    mode = c("cv", "stability"), alpha = 0.5, nbootstrap = 100, 
    fraction = 0.67, threshold = 0.9, EV = 1, weakness = 0.8, 
    ncores = 1, ...)

## S4 method for signature 'numeric,Matrix'
gespeR(phenotypes, target.relations, ...)

Arguments

phenotypes The siRNA-specific phenotypes. Single object for univariate phenotypes and list 
of Phenotypes objects for multivariate phenotypes.
target.relations The siRNA-to-gene target relations
...
mode The mode of covariate selectino ("cv" or "stability")
alpha The glmnet mixing parameter
nbootstrap The number of bootstrap samples
fraction The fraction for each bootstrap sample
threshold The selection threshold
EV The expected value of wrongly selected elements
weakness The weakness parameter for randomised lasso
ncores The number of cores for parallel computation

Value

A gespeR object

Slots

SSP The observed siRNA-specific phenotypes
GSP The deconvoluted gene-specific phenotypes
target.relations The siRNA-to-gene target relations, e.g. predicted by TargetScan
is.fitted An indicator whether the gespeR model was fitted
model The fitted regularized linear regression model

Author(s)

Fabian Schmich
gsp 11

See Also

gespeR-package
plot.gespeR
gsp
ssp
scores
stability
target.relations

Examples

phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)
trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))
res <- gespeR(phenotypes = phenos,
target.relations = trels,
mode = "stability",
nbootstrap = 100,
fraction = 0.67,
threshold = 0.75,
EV = 1,
weakness = 0.8,
ncores = 1)
gsp(res)

---

**gsp**

*Retrieve GSPs and SSPs from gespeR objects*

Description

Retrieve GSPs and SSPs from gespeR objects

Usage

gsp(object)

## S4 method for signature 'gespeR'
gsp(object)

ssp(object)

## S4 method for signature 'gespeR'
ssp(object)
Arguments

- object: A `gespeR` object

Value

A `Phenotypes` object of GSPs and SSPs, respectively

Author(s)

Fabian Schmich

See Also

- `annotate.gsp`
- `scores`

Examples

```r
data(stabilityfits)
gsp(stabilityfits$A)
ssp(stabilityfits$B)
```

---

Description

Join a `TargetRelations` object and a `Phenotype` object

Usage

```r
join(targets, phenotypes)
```

```
## S4 method for signature 'TargetRelations,Phenotypes'
join(targets, phenotypes)
```

Arguments

- targets: A `TargetRelations` object.
- phenotypes: A `Phenotypes` object.

Value

List containing the matched targets and phenotypes

Author(s)

Fabian Schmich
lasso.rand

Examples

```r
phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
  type = "SSP",
  col.id = 1,
  col.score = 2)
trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))
phenos <- phenos[1:17]
stripped_down <- join(targets = trels, phenotypes = phenos)
```

---

lasso.rand  *Randomized Lasso*

Description

Based on Meinshausen and Buehlmann (2009)

Usage

```r
lasso.rand(x, y, weakness = 1, subsample = 1:nrow(x), dfmax = (ncol(x) + 1), lambda = NULL, standardize = FALSE, intercept = FALSE, ...)
```

Arguments

- **x**: The design matrix
- **y**: The response vector
- **weakness**: The weakness parameter
- **subsample**: The data subsample (default: none)
- **dfmax**: The maximum number of degrees of freedom
- **lambda**: The regularisation parameter
- **standardize**: Indicator, whether to standardize the design matrix
- **intercept**: Indicator, whether to fit an intercept
- **...**: Additional arguments to `glmnet`

Value

A `glmnet` object

Author(s)

Fabian Schmich

Examples

```r
y <- rnorm(50)
x <- matrix(runif(50 * 20), ncol = 20)
lasso.rand(x = x, y = y)
```
loadValues  

Methods for values of `TargetRelations` objects

Description

Load, unload or write to file the values of a `TargetRelations` object

Usage

```r
loadValues(object)
## S4 method for signature 'TargetRelations'
loadValues(object)

## S4 method for signature 'gespeR'
loadValues(object)

unloadValues(object, ...)
## S4 method for signature 'TargetRelations'
unloadValues(object, writeValues = TRUE,
  overwrite = FALSE, path = NULL)

## S4 method for signature 'gespeR'
unloadValues(object, writeValues = TRUE,
  overwrite = FALSE, path = NULL)

writeValues(object, ...)
## S4 method for signature 'TargetRelations'
writeValues(object, overwrite = FALSE)
```

Arguments

- `object` A `TargetRelations` object or `gespeR` object
- `...` Additional arguments
- `writeValues` Indicator, whether to write values
- `overwrite` Indicator, wheter to overwrite values if file exists at path
- `path` The path to write out values

Value

A `TargetRelations` object or `gespeR` object

Author(s)

Fabian Schmich
Remove NA/Inf values from phenotype vectors

Description
Remove NA/Inf values from phenotype vectors

Usage
na.rem(object)

## S4 method for signature 'Phenotypes'
na.rem(object)

Arguments
object A Phenotypes object

Value
A Phenotypes object without NA scores values

Author(s)
Fabian Schmich

Examples
phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)
nna.rem(phenos)
Phenotypes-class

---

**path<-'**

The path of a `TargetRelations` object can be set.

### Usage

```r
path(object) <- value
```

### Arguments

- `object`: A `TargetRelations` object
- `value`: A string defining the path

### Value

A `TargetRelations` object with the set path.

### Author(s)

Fabian Schmich

### Examples

```r
trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))
path(trels) <- "/dev/null"
```

---

Phenotypes-class

**Phenotypes**

### Description

Class used to represent various types of phenotypes, e.g. from siRNA-specific (SSP) or estimated gene-specific phenotypes (GSP).
Usage

Phenotypes(phenotypes, ...)

## S4 method for signature 'character'
Phenotypes(phenotypes, type = c("SSP", "GSP"),
          sep = "\t", col.id = 1, col.score = 2)

## S4 method for signature 'cellHTS'
Phenotypes(phenotypes, channel, sample)

## S4 method for signature 'Matrix'
Phenotypes(phenotypes, ids = NULL, pnames = NULL,
          type = c("SSP", "GSP"))

Arguments

phenotypes  The phenotypes as numeric vector, path to a .txt file with two columns (1: identifiers, 2: values), or a cellHTS object
          ...
          Additional arguments
type        The type of phenotype (GSP, SSP)
sep         The separator string
col.id      Column number for the siRNA identifiers
col.score   Column number(s) for the phenotype score
channel     The cellHTS channel identifier
sample      The cellHTS sample index
ids         The siRNA/gene identifiers
pnames      The phenotype identifiers

Value

A Phenotypes object

Slots

type  The type of represented phenotypes (i.e., "SSP" or "GSP")
ids   The entity identifiers (i.e., siRNA or gene ids)
pnames The phenotype identifiers
values The phenotypic values

Author(s)

Fabian Schmich
See Also

plot.Phenotypes
join
gsp
ssp
scores
concordance

Examples

phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)

plot.concordance

Description

Plots boxplots of concordance evaluated between multiple Phenotype objects. Measures include the correlation (rho) between pairs of phenotypes for the same gene, the rank biased overlap (rbo) of the top and bottom of ranked lists, and the Jaccard index (J) of selected genes.

Usage

## S3 method for class 'concordance'
plot(x, ...)

Arguments

x The data of class concordance

... Additional parameters for plot

Value

Boxplots of concordance measures

Author(s)

Fabian Schmich
**plot.gespeR**

*Plot method for gespeR objects*

---

**Description**

Plot method for gespeR objects

**Usage**

```r
## S3 method for class 'gespeR'
plot(x, ...)
```

**Arguments**

- `x` A gespeR object
- `...` Additional parameters for plot

**Value**

Histogram of SSPs or GSPs

**Author(s)**

Fabian Schmich

---

**plot.Phenotypes**

*Plot method for Phenotype objects*

---

**Description**

Plot method for Phenotype objects

**Usage**

```r
## S3 method for class 'Phenotypes'
plot(x, ...)
```

**Arguments**

- `x` A Phenotypes object
- `...` Additional arguments for plot

**Value**

Histogram of scores

```r
phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"), type = "SSP", col.id = 1, col.score = 2)
plot(phenos)
```
**rbo**

**Author(s)**
Fabian Schmich

**Rank biased overlap (Webber et al., 2010)**

**Description**
Evaluates the rank biased overlap (rbo) of two ranked lists based on formula based on (32) from 
"A Similarity Measure for Indefinite Rankings" (Webber et al.). Two ranked lists with high rbo are 
very similar, whereas low rbo indicates dissimilar lists. rbo ranges between 0 and 1. In this method 
the extrapolated version of rbo is implemented.

**Usage**

```r
rbo(s, t, p, k = floor(max(length(s), length(t))/2), side = c("top", "bottom"), mid = NULL, uneven.lengths = TRUE)
```

**Arguments**
- `s`: List 1
- `t`: List 2
- `p`: Weighting parameter in [0, 1]. High p implies strong emphasis on top ranked elements
- `k`: Evaluation depth for extrapolation
- `side`: Evaluate similarity between the top or the bottom of the ranked lists
- `mid`: Set the mid point to for example only consider positive or negative scores
- `uneven.lengths`: Indicator if lists have uneven lengths

**Value**
rank biased overlap (rbo)

**Author(s)**
Fabian Schmich

**See Also**
concordance

**Examples**
```r
a <- rnorm(26)
b <- rnorm(26)
names(a) <- names(b) <- LETTERS
rbo(a, b, p = 0.95)
```
Description

Return a named vector of phenotype scores

Usage

```r
## S4 method for signature 'Phenotypes'
scores(object)

## S4 method for signature 'gespeR'
scores(object, type = c("GSP", "SSP"))
```

Arguments

- `object`: A `gespeR` or `Phenotypes` object
- `type`: The type of phenotype scores (GSP, SSP)

Value

A named vector of scores for each phenotype identifier

Author(s)

Fabian Schmich

See Also

gespeR
Phenotypes

Examples

data(stabilityfits)
scores(stabilityfits$A)
### simData

**Example data: Simulated phenotypes and target relations for 4 screens**

\[(A, B, C, D)\]

**Description**

The data set contains simulated data for four screens. Each screen consists of a phenotype vector and target relations between siRNAs and genes, i.e. which siRNA binds which genes (on- and off-targets). The size of each simulated screen is \(N = 1000\) siRNAs \(\times p = 1500\) genes. SSPs are generated by first defining GSPs and multiplying the true GSPs with the sampled target relation matrices. For sampling the GSPs, we set the number of effect genes to 5 from Normal(0, 3). Target relation matrices are simulated by sampling the number of off-targets per siRNA from Normal(3e-2 * \(N\), 3e-3 * \(N\)) and the strength of off-targets is sampled from Beta(2, 5). On-target components are set to 0.75.

**Details**

The code used to simulate the data can be found in `system.file("example", "data_simulation.R", package="gespeR")`

**Examples**

```r
pheno.a <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package="gespeR"), type = "SSP", col.id = 1, col.score = 2)
targets.a <- TargetRelations(system.file("extdata", "TR_screen_A.rds", package="gespeR"))
```

### stability

Retrieve a `Phenotypes` object with stability values from a `gespeR` object.

**Usage**

```r
stability(object)
```

#### stability(object)

**Arguments**

- `object` A `gespeR` object

**Value**

A `Phenotypes` object of SSPs
stability.selection

Author(s)

Fabian Schmich

Examples

```r
phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
          type = "SSP",
          col.id = 1,
          col.score = 2)
trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))
res <- gespeR(phenotypes = phenos,
               target.relations = trels,
               mode = "stability",
               nbootstrap = 100,
               fraction = 0.67,
               threshold = 0.75,
               EV = 1,
               weakness = 0.8,
               ncores = 1)
stab <- stability(res)
ans <- merge(as.data.frame(gsp(res)), as.data.frame(stability(res)), by = "ID")
colnames(ans)[2:3] <- c("Phenotype", "Stability")
ans[order(ans$Stability, decreasing = TRUE),]
```

stability.selection  Stability Selection

Description

Based on Meinshausen and Buehlmann (2009)

Usage

```r
stability.selection(x, y, fraction = 0.5, threshold = 0.75, EV = 1,
                     nbootstrap = 100, weakness = 1, intercept = FALSE, ncores = 1, ...)
```

Arguments

- `x` The design matrix
- `y` The response vector
- `fraction` The fraction for each bootstrap sample
- `threshold` The selection threshold
- `EV` The expected value of wrongly selected elements
- `nbootstrap` The number of bootstrap samples
- `weakness` The weakness parameter for randomised lasso
- `intercept` Indicator, whether to fit an intercept
- `ncores` The number of cores for parallel computation
- `...` Additional arguments to `lasso.rand`
target.relations

Value
A list containing selected covariates with frequencies, and the fitted model

Author(s)
Fabian Schmich

stabilityfits
Example fits for phenotypes from simulated screening data A, B, C and D

Description
The data set contains four fitted gespeR models using stability selection from the four simulated screens.

Examples
data(stabilityfits)

target.relations
target.relations

Description
Retrieve siRNA-to-gene target relations from a gespeR object.

Usage
target.relations(object)

## S4 method for signature 'gespeR'
target.relations(object)

Arguments
object A gespeR object

Value
A TargetRelations object

Author(s)
Fabian Schmich
Examples

data(stabilityfits)
target.relations(stabilityfits$A)

Description

Class used to represent siRNA-to-gene on- and off-target relations for a knockdown library and a set of genes.

Usage

TargetRelations(targets)

## S4 method for signature 'character'
TargetRelations(targets)

## S4 method for signature 'Matrix'
TargetRelations(targets)

Arguments

targets Path to a .rds target relations matrix file or Matrix object

Value

A TargetRelations object

Slots

siRNAs The siRNA identifiers
genes The gene identifiers (Entrez)
path The path to and .rds TargetRelations file
is.loaded An indicator if target relations values are loaded
values The quantitative target relation values between siRNAs and genes

Author(s)

Fabian Schmich
See Also

join
loadValues
unloadValues
writeValues
values
path<-

Examples

trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))

Description

Retrieve the numeric values from a TargetRelations or Phenotypes object

Usage

values(object)

## S4 method for signature 'TargetRelations'
values(object)

## S4 method for signature 'Phenotypes'
values(object)

Arguments

object A TargetRelations or Phenotypes object

Value

A Matrix object

Author(s)

Fabian Schmich
Examples

trels <- TargetRelations(readRDS(system.file("extdata", "TR_screen_A.rds", package = "gespeR")))
values(trels)[1:5, 1:5]

phenos <- Phenotypes(system.file("extdata", "Phenotypes_screen_A.txt", package = "gespeR"),
type = "SSP",
col.id = 1,
col.score = 2)
values(phenos)

Description

Subsetting for Phenotype objects.

Usage

## S4 method for signature 'Phenotypes,ANY,ANY,ANY'

x[i, j, ..., drop = TRUE]

Arguments

x A Phenotypes object
i The subsetting indices for siRNAs
j Subsetting indices for multivariate phenotypes
... Additional parameters
drop Drop Redundant Extent Information

Value

A Phenotypes object

Author(s)

Fabian Schmich
Subsetting for TargetRelations objects.

Usage

```r
## S4 method for signature 'TargetRelations,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

- `x`: A `TargetRelations` object
- `i`: The row subsetting indices (siRNAs)
- `j`: The column subsetting indeces (genes)
- `...`: Additional parameters
- `drop`: Drop Redundant Extent Information

Value

A `TargetRelations` object

Author(s)

Fabian Schmich
# Index

<table>
<thead>
<tr>
<th>Package</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>gespeR-package</td>
<td>2</td>
</tr>
<tr>
<td>annotate.gsp</td>
<td>4, 12</td>
</tr>
<tr>
<td>annotate.gsp, gespeR-method</td>
<td>4</td>
</tr>
<tr>
<td>annotate.gsp, Phenotypes-method</td>
<td>4</td>
</tr>
<tr>
<td>as.data.frame, Phenotypes-method</td>
<td>5</td>
</tr>
<tr>
<td>c, Phenotypes-method</td>
<td>7</td>
</tr>
<tr>
<td>concordance</td>
<td>8, 8, 18, 20</td>
</tr>
<tr>
<td>dim, Phenotypes-method</td>
<td>9</td>
</tr>
<tr>
<td>gespeR</td>
<td>3, 5, 10–12, 14, 19, 21, 22, 24</td>
</tr>
<tr>
<td>gespeR, numeric, Matrix-method</td>
<td>9</td>
</tr>
<tr>
<td>gespeR, Phenotypes, TargetRelations-method</td>
<td>9</td>
</tr>
<tr>
<td>gespeR-class</td>
<td>9</td>
</tr>
<tr>
<td>gespeR-package</td>
<td>2</td>
</tr>
<tr>
<td>glmnet</td>
<td>10, 13</td>
</tr>
<tr>
<td>gsp, 5, 11, 18</td>
<td></td>
</tr>
<tr>
<td>gsp, gespeR-method (gsp)</td>
<td>11</td>
</tr>
<tr>
<td>join, 12, 18, 26</td>
<td></td>
</tr>
<tr>
<td>loadValues</td>
<td>14, 26</td>
</tr>
<tr>
<td>loadValues, gespeR-method (loadValues)</td>
<td>14</td>
</tr>
<tr>
<td>loadValues, TargetRelations-method (loadValues)</td>
<td>14</td>
</tr>
<tr>
<td>Matrix</td>
<td>25, 26</td>
</tr>
<tr>
<td>na.rem</td>
<td>15</td>
</tr>
<tr>
<td>na.rem, Phenotypes-method (na.rem)</td>
<td>15</td>
</tr>
<tr>
<td>path&lt;-</td>
<td>16</td>
</tr>
<tr>
<td>path&lt;-, TargetRelations, character-method (path&lt;-)</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, 5–10, 12, 15, 17, 19, 21, 22, 26, 27</td>
<td></td>
</tr>
<tr>
<td>Phenotypes (Phenotypes-class)</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, cellHTS-method</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, character-method</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, Matrix-method</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, Phenotypes-class</td>
<td>16</td>
</tr>
<tr>
<td>Phenotypes, plot.concordance</td>
<td>8, 18</td>
</tr>
<tr>
<td>Phenotypes, plot.gespeR</td>
<td>11, 19</td>
</tr>
<tr>
<td>Phenotypes, plot.Phenotypes</td>
<td>18, 19</td>
</tr>
<tr>
<td>rbo</td>
<td>8, 20</td>
</tr>
<tr>
<td>scores</td>
<td>5, 11, 12, 18, 21</td>
</tr>
<tr>
<td>scores, gespeR-method (scores)</td>
<td>21</td>
</tr>
<tr>
<td>scores, Phenotypes-method (scores)</td>
<td>21</td>
</tr>
<tr>
<td>simData</td>
<td>22</td>
</tr>
<tr>
<td>ssp, 5, 11, 18</td>
<td></td>
</tr>
<tr>
<td>ssp (gsp)</td>
<td>11</td>
</tr>
<tr>
<td>ssp, gespeR-method (gsp)</td>
<td>11</td>
</tr>
<tr>
<td>stability</td>
<td>11, 22</td>
</tr>
<tr>
<td>stability, gespeR-method (stability)</td>
<td>22</td>
</tr>
<tr>
<td>stability.selection</td>
<td>23</td>
</tr>
<tr>
<td>stabilityfits</td>
<td>24</td>
</tr>
<tr>
<td>target.relations</td>
<td>11, 24</td>
</tr>
</tbody>
</table>
target.relations, gespeR-method (target.relations), 24
TargetRelations, 9, 12, 14, 16, 24–26, 28
TargetRelations (TargetRelations-class), 25
TargetRelations, character-method (TargetRelations-class), 25
TargetRelations, Matrix-method (TargetRelations-class), 25
TargetRelations-class, 25

unloadValues, 26
unloadValues (loadValues), 14
unloadValues, gespeR-method (loadValues), 14
unloadValues, TargetRelations-method (loadValues), 14

values, 26, 26
values, Phenotypes-method (values), 26
values, TargetRelations-method (values), 26

writeValues, 26
writeValues (loadValues), 14
writeValues, TargetRelations-method (loadValues), 14