Package ‘BioCor’

May 29, 2024

**Title**  Functional similarities

**Version**  1.28.0

**Description**  Calculates functional similarities based on the pathways
described on KEGG and REACTOME or in gene sets. These similarities can
be calculated for pathways or gene sets, genes, or clusters and
combined with other similarities. They can be used to improve
networks, gene selection, testing relationships...

**License**  MIT + file LICENSE

**URL**  https://bioconductor.org/packages/BioCor,
         https://llrs.github.io/BioCor/

**BugReports**  https://github.com/llrs/BioCor/issues

**Depends**  R (>= 3.4.0)

**Imports**  BiocParallel, GSEABase, Matrix, methods

**Suggests**  airway, BiocStyle, boot, DESeq2, ggplot2 (>= 3.4.1),
              GOSemSim, Hmisc, knitr (>= 1.35), org.Hs.eg.db, reactome.db,
              markdown, spelling, targetsan.Hs.eg.db, testthat (>= 3.0.0),
              WGCNA

**VignetteBuilder**  knitr

**biocViews**  StatisticalMethod, Clustering, GeneExpression, Network,
                Pathways, NetworkEnrichment, SystemsBiology

**Config/testthat/edition**  3

**Encoding**  UTF-8

**Language**  en-US

**Roxygen**  list(markdown = TRUE)

**RoxygenNote**  7.3.1

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

**git_branch**  RELEASE_3_19

**git_last_commit**  0edd7da

**git_last_commit_date**  2024-04-30
BioCor-package

Description

Calculates a functional similarity measure between gene identifiers based on the pathways described on KEGG and REACTOME.
addSimilarities

Important functions

- **pathSim()**: Calculates the similarity between two pathways.
- **geneSim()**: Calculates the similarity (based on pathSim) between two genes.
- **clusterSim()**: Calculates the similarity between two clusters of genes by joining pathways of each gene.
- **clusterGeneSim()**: Calculates the similarity between two clusters of genes by comparing the similarity between the genes of a cluster.
- **similarities()**: Allows to combine the value of matrices of similarities.
- **conversions()**: Two functions to convert similarity measures.
- **weighted()**: Functions provided to combine similarities.

Author(s)

**Maintainer**: Lluís Revilla Sancho <lluis.revilla@gmail.com> (ORCID)
Other contributors:
- Pau Sancho-Bru (ORCID) [thesis advisor]
- Juan José Salvatella Lozano (ORCID) [thesis advisor]

See Also

Useful links:
- [https://bioconductor.org/packages/BioCor](https://bioconductor.org/packages/BioCor)
- [https://llrs.github.io/BioCor/](https://llrs.github.io/BioCor/)
- Report bugs at [https://github.com/llrs/BioCor/issues](https://github.com/llrs/BioCor/issues)

---

addSimilarities  
Additive integration of similarities

Description

Function that use the previously calculated similarities into a single similarity matrix.

Usage

```r
addSimilarities(x, bio_mat, weights = c(0.5, 0.18, 0.1, 0.22))
```

Arguments

- **x**: A matrix with the similarity of expression
- **bio_mat**: A list of matrices of the same dimension as x.
- **weights**: A numeric vector of weight to multiply each similarity
AintoB

Details

The total weight can’t be higher than 1 to prevent values above 1 but can be below 1. It uses weighted.sum with abs = TRUE internally.

Value

A square matrix of the same dimensions as the input matrices.

Author(s)

Lluís Revilla

See Also

similarities(), weighted().

Examples

```r
set.seed(100)
a <- seq2mat(LETTERS[1:5], rnorm(10))
b <- seq2mat(LETTERS[1:5], seq(from = 0.1, to = 1, by = 0.1))
sim <- list(b)
addSimilarities(a, sim, c(0.5, 0.5))
```

Description

Insert values from a matrix into another matrix based on the rownames and colnames replacing the values.

Usage

AintoB(A, B)

Arguments

A
A matrix to be inserted.

B
A matrix to insert in.

Details

If all the genes with pathway information are already calculated but you would like to use more genes when performing analysis. insert the once you have calculated on the matrix of genes.

Value

A matrix with the values of A in the matrix B.
clusterGeneSim

Author(s)
Lluís Revilla

Examples

```r
B <- matrix(
  ncol = 10, nrow = 10,
  dimnames = list(letters[1:10], letters[1:10])
)
A <- matrix(c(1:15),
  byrow = TRUE, nrow = 5,
  dimnames = list(letters[1:5], letters[1:3])
)
AintoB(A, B)

# Mixed orders
colnames(A) <- c("c", "h", "e")
rownames(A) <- c("b", "a", "f", "c", "j")
AintoB(A, B)

# Missing columns or rows
colnames(A) <- c("d", "f", "k")
AintoB(A, B)
```

clusterGeneSim

Similarity score between clusters of genes based on genes similarity

Description

Looks for the similarity between genes of a group and then between each group’s genes.

Usage

```r
clusterGeneSim(cluster1, cluster2, info, method = c("max", "rcmax.avg"), ...)

## S4 method for signature 'character,character,GeneSetCollection'
clusterGeneSim(cluster1, cluster2, info, method = c("max", "rcmax.avg"), ...)
```

Arguments

- `cluster1,cluster2`
  A vector with genes.
- `info`
  A GeneSetCollection or a list of genes and the pathways they are involved.
- `method`
  A vector with two or one argument to be passed to combineScores the first one is used to summarize the similarities of genes, the second one for clusters.
- `...`
  Other arguments passed to combineScores
Details

Differs with clusterSim that first each combination between genes is calculated, and with this values then the comparison between the two clusters is done. Thus applying combineScores twice, one at gene level and another one at cluster level.

Value

Returns a similarity score between the genes of the two clusters.

Methods (by class)

- `clusterGeneSim(cluster1 = character, cluster2 = character, info = GeneSetCollection)`: Calculates the gene similarities in a GeneSetCollection and combine them using `combineScoresPar()`

Author(s)

Lluís Revilla

See Also

`mclusterGeneSim()`, `combineScores()` and `clusterSim()`

Examples

```r
if (require("org.Hs.eg.db")) {
  # Extract the paths of all genes of org.Hs.eg.db from KEGG (last update in
  # data of June 31st 2011)
  genes.kegg <- as.list(org.Hs.egPATH)
  clusterGeneSim(c("18", "81", "10"), c("100", "10", "1"), genes.kegg)
  clusterGeneSim(  
    c("18", "81", "10"), c("100", "10", "1"), genes.kegg,  
    c("avg", "avg")
  )
  clusterGeneSim(  
    c("18", "81", "10"), c("100", "10", "1"), genes.kegg,  
    c("avg", "rcmax.avg")
  )
  (clus <- clusterGeneSim(  
    c("18", "81", "10"), c("100", "10", "1"),  
    genes.kegg, "avg"
  ))
  combineScores(clus, "rcmax.avg")
} else {
  warning("You need org.Hs.eg.db package for this example")
}
```
clusterSim

**clusterSim**

*Similarity score between clusters of genes based on pathways similarity*

---

**Description**

Looks for the similarity between genes in groups

**Usage**

```r
clusterSim(cluster1, cluster2, info, method = "max", ...)
```

```r
# S4 method for signature 'character,character,GeneSetCollection'
clusterSim(cluster1, cluster2, info, method = "max", ...)
```

**Arguments**

- `cluster1,cluster2`
  A vector with genes.
- `info`
  A GeneSetCollection or a list of genes and the pathways they are involved.
- `method`
  one of c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"), see Details.
- `...`
  Other arguments passed to `combineScores`

**Details**

Once the pathways for each cluster are found they are combined using `combineScores()`.

**Value**

`clusterSim` returns a similarity score of the two clusters

**Methods (by class)**

- `clusterSim(cluster1 = character, cluster2 = character, info = GeneSetCollection)
  Calculates all the similarities of the GeneSetCollection and combine them using `combineScoresPar()`

**Author(s)**

Lluís Revilla

**See Also**

For a different approach see `clusterGeneSim()`, `combineScores()` and `conversions()`
Examples

```r
if (require("org.Hs.eg.db")) {
    # Extract the paths of all genes of org.Hs.eg.db from KEGG (last update in
    # data of June 31st 2011)
    genes.kegg <- as.list(org.Hs.egPATH)
    clusterSim(c("9", "15", "10"), c("33", "19", "20"), genes.kegg)
    clusterSim(c("9", "15", "10"), c("33", "19", "20"), genes.kegg, NULL)
    clusterSim(c("9", "15", "10"), c("33", "19", "20"), genes.kegg, "avg")
  } else {
    warning("You need org.Hs.eg.db package for this example")
  }
```

---

**combinadic**

\(i\)-th combination of \(n\) elements taken from \(r\)

---

**Description**

Function similar to `combn` but for larger vectors. To avoid allocating a big vector with all the combinations each one can be computed with this function.

**Usage**

`combinadic(n, r, i)`

**Arguments**

- \(n\) : Elements to extract the combination from
- \(r\) : Number of elements per combination
- \(i\) : \(i\)th combination

**Value**

The combination \(i\)th of the elements

**Author(s)**

Joshua Ulrich

**References**

- StackOverflow answer 4494469/2886003

**See Also**

`combn()`
**combineScores**

**Examples**

```r
# Output of all combinations
combn(LETTERS[1:5], 2)
# Output of the second combination
combindic(LETTERS[1:5], 2, 2)
```

**Description**

Combine several similarities into one using several methods.

**Usage**

```r
combineScores(
    scores,
    method = c("max", "avg", "rcmax", "rcmax.avg", "BMA", "reciprocal"),
    round = FALSE,
    t = 0
)
```

```r
combineScoresPar(scores, method, subSets = NULL, BPPARAM = NULL, ...)
```

**Arguments**

- **scores**: Matrix of scores to be combined
- **method**: one of c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"), see Details.
- **round**: Should the resulting value be rounded to the third digit?
- **t**: Numeric value to filter scores below this value. Only used in the reciprocal method.
- **subSets**: List of combinations as info in other functions.
- **BPPARAM**: BiocParallel back-end parameters. By default (NULL) a for loop is used.
- **...**: Other arguments passed to combineScores

**Details**

The input matrix can be a base matrix or a matrix from package Matrix. The methods return:

- **avg**: The average or mean value.
- **max**: The max value.
- **rcmax**: The max of the column means or row means.
- **rcmax.avg**: The sum of the max values by rows and columns divided by the number of columns and rows.
combineScores

- **BMA**: The same as \texttt{rcmax.avg}.
- **reciprocal**: The double of the sum of the reciprocal maximal similarities (above a threshold) divided by the number of elements. See equation 3 of the Tao \textit{et al} 2007 article.

**Value**

A numeric value as described in details.

**Note**

\texttt{combineScores} is a version of the function of the same name in package GO\texttt{SemSim} (\texttt{GOSemSim::combineScores()}) with optional rounding and some internal differences.

**Author(s)**

Lluís Revilla based on Guangchuang Yu.

**References**

Ying Tao, Lee Sam, Jianrong Li, Carol Friedman, Yves A. Lussier; Information theory applied to the sparse gene ontology annotation network to predict novel gene function. Bioinformatics 2007; 23 (13): i529-i538. doi: 10.1093/bioinformatics/btm195

**See Also**

\texttt{register} in BiocParallel about the arguments accepted by \texttt{BPPARAM}.

**Examples**

```r
(d <- structure(c(  0.4, 0.6, 0.222222222222222, 0.4, 0.4, 0, 0.25, 0.5, 0.285714285714286
    ),
   .Dim = c(3L, 3L),
   .Dimnames = list(c("a", "b", "c"), c("d", "e", "f"))
))
e <- d
sapply(c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"),
   combineScores,
   scores = d)
d[1, 2] <- NA
sapply(c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"),
   combineScores,
   scores = d)
colnames(e) <- rownames(e)
combineScoresPar(e, list(a = c("a", "b"), b = c("b", "c")),
    method = "max"
)```

**Description**

Given several sources of pathways with the same id of the genes it merge them.

**Usage**

`combineSources(...)`

**Arguments**

... Lists of genes and their pathways.

**Details**

It assumes that the identifier of the genes are the same for both sources but if many aren't equal it issues a warning. Only unique pathways identifiers are returned.

**Value**

A single list with the pathways of each source on the same gene.

**Examples**

```r
DB1 <- list(g1 = letters[6:8], g2 = letters[1:5], g3 = letters[4:7])
DB2 <- list(
  g1 = c("one", "two"),
  g2 = c("three", "four"),
  g3 = c("another", "two")
)
combineSources(DB1, DB2)
combineSources(DB1, DB1)
DB3 <- list(
  g1 = c("one", "two"),
  g2 = c("three", "four"),
  g4 = c("five", "six", "seven"),
  g5 = c("another", "two")
)
combineSources(DB1, DB3) # A warning is expected
```
Conversions

Convert the similarities formats

Description

Functions to convert the similarity coefficients between Jaccard and Dice. D2J is the opposite of J2D.

Usage

D2J(D)

J2D(J)

Arguments

D

Dice coefficient, as returned by diceSim(), geneSim(), clusterSim() and clusterGeneSim()

J

Jaccard coefficient

Value

A numeric value.

Author(s)

Lluís Revilla

Examples

D2J(0.5)

J2D(0.5)

D2J(J2D(0.5))

diceSim

Compare pathways

Description

Function to estimate how much two list of genes overlap by looking how much of the nodes are shared. Calculates the Dice similarity

Usage

diceSim(g1, g2)
duplicateIndices

Arguments

  g1, g2  
  A character list with the names of the proteins in each pathway.

Details

  It requires a vector of characters otherwise will return an NA.

Value

  A score between 0 and 1 calculated as the double of the proteins shared by g1 and g2 divided by the
  number of genes in both groups.

Author(s)

  Lluís Revilla

See Also

  Used for geneSim(), see conversions() help page to transform Dice score to Jaccard score.

Examples

    genes.id2 <- c("52", "11342", "80895", "57654", "548953", "11586", "45985")
    genes.id1 <- c(
      "52", "11342", "80895", "57654", "58493", "1164", "1163",
      "4150", "2130", "159"
    )
    diceSim(genes.id1, genes.id2)
    diceSim(genes.id2, genes.id2)

---

duplicateIndices

Finds the indices of the duplicated events of a vector

Description

  Finds the indices of duplicated elements in the vector given.

Usage

    duplicateIndices(vec)

Arguments

  vec  
  Vector of identifiers presumably duplicated

Details

  For each duplication it can return a list or if all the duplication events are of the same length it
  returns a matrix, where each column is duplicated.
**Value**

The format is determined by the `simplify2array`

**Author(s)**

Lluís Revilla

**See Also**

`removeDup()`

**Examples**

duplicateIndices(c("52", "52", "53", "55")) # One repeated element
duplicateIndices(c("52", "52", "53", "55", "55")) # Repeated elements
duplicateIndices(c("52", "55", "53", "55", "52")) # Mixed repeated elements

geneSim

<table>
<thead>
<tr>
<th>geneSim</th>
<th>Similarity score genes based on pathways similarity</th>
</tr>
</thead>
</table>

**Description**

Given two genes, calculates the Dice similarity between each pathway which is combined to obtain a similarity between the genes.

**Usage**

geneSim(gene1, gene2, info, method = "max", ...)

## S4 method for signature 'character,character,GeneSetCollection'
geneSim(gene1, gene2, info, method = "max", ...)

**Arguments**

gene1, gene2  
Ids of the genes to calculate the similarity, to be found in genes.

info  
A `GeneSetCollection` or a list of genes and the pathways they are involved.

method  
one of c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"). see Details.

...  
Other arguments passed to `combineScores`

**Details**

Given the information about the genes and their pathways, uses the ids of the genes to find the Dice similarity score for each pathway comparison between the genes. Later this similarities are combined using `combineScoresPar()`.
Value

The highest Dice score of all the combinations of pathways between the two ids compared if a method to combine scores is provided or NA if there isn’t information for one gene. If an NA is returned this means that there isn’t information available for any pathways for one of the genes. Otherwise a number between 0 and 1 (both included) is returned. Note that there isn’t a negative value of similarity.

Methods (by class)

- geneSim(gene1 = character, gene2 = character, info = GeneSetCollection): Calculates all the similarities of the GeneSetCollection and combine them using combineScoresPar()

Author(s)

Lluís Revilla

See Also

mgeneSim(), conversions() help page to transform Dice score to Jaccard score. For the method to combine the scores see combineScoresPar().

Examples

```r
if (require("org.Hs.eg.db") & require("reactome.db")) {
  # Extract the paths of all genes of org.Hs.eg.db from KEGG
  # (last update in data of June 31st 2011)
  genes.kegg <- as.list(org.Hs.egPATH)
  # Extracts the paths of all genes of org.Hs.eg.db from reactome
  genes.react <- as.list(reactomeEXTID2PATHID)
  geneSim("81", "18", genes.react)
  geneSim("81", "18", genes.kegg)
  geneSim("81", "18", genes.react, NULL)
  geneSim("81", "18", genes.kegg, NULL)
} else {
  warning("You need reactome.db and org.Hs.eg.db package for this example")
}
```

---

**incidence, list-method**  
*Creates the incidence matrix*

**Description**

Given a list of pathways and its genes creates an incidence matrix.

**Usage**

```r
## S4 method for signature 'list'
incidence(x)
```
**Arguments**

- **x**: A list

**Value**

A matrix with pathways as rows and genes in columns.

**Note**

Designed to be easier to work with list and GeneSetCollection

**Author(s)**

Lluís Revilla

---

### inverseList

**Invert a list**

**Description**

Calculate the pathways per gene of list

**Usage**

```r
inverseList(x)
```

**Arguments**

- **x**: A list with genes as names and names of pathways as values of the list

**Value**

The number of pathways each gene has.

**Author(s)**

Lluís Revilla
mclusterGeneSim

**Similarity score between clusters of genes based on genes similarity**

**Description**

Looks for the similarity between genes of a group and then between each group’s genes.

**Usage**

```r
mclusterGeneSim(clusters, info, method = c("max", "rcmax.avg"), ...)
```

```r
## S4 method for signature 'list,GeneSetCollection'

mclusterGeneSim(clusters, info, method = c("max", "rcmax.avg"), ...)
```

**Arguments**

- **clusters**
  A list of clusters of genes to be found in id.
- **info**
  A GeneSetCollection or a list of genes and the pathways they are involved.
- **method**
  A vector with two or one argument to be passed to combineScores the first one is used to summarize the similarities of genes, the second one for clusters.
- ... Other arguments passed to combineScores

**Value**

Returns a matrix with the similarity scores for each cluster comparison.

**Methods (by class)**

- mclusterGeneSim(clusters = list, info = GeneSetCollection): Calculates all the similarities of the GeneSetCollection and combine them using `combineScoresPar()`

**Author(s)**

Lluís Revilla

**See Also**

`clusterGeneSim()`, `clusterSim()` and `combineScores()`

**Examples**

```r
if (require("org.Hs.eg.db")) {
  genes.kegg <- as.list(org.Hs.egPATH)
  clusters <- list(
    cluster1 = c("18", "81", "10"),
    cluster2 = c("100", "594", "836"),
    cluster3 = c("18", "10", "83")
  )
}
```
mclusterGeneSim(clusters, genes.kegg)
mclusterGeneSim(clusters, genes.kegg, c("max", "avg"))
mclusterGeneSim(clusters, genes.kegg, c("max", "BMA"))
} else {
  warning("You need org.Hs.eg.db package for this example")
}

mclusterSim

Similarity score between clusters of genes based on pathways similarity

Description

Looks for the similarity between genes in groups. Once the pathways for each cluster are found they are combined using codecombineScores.

Usage

mclusterSim(clusters, info, method = "max", ...)

## S4 method for signature 'list,GeneSetCollection'
mclusterSim(clusters, info, method = "max", ...)

Arguments

clusters A list of clusters of genes to be found in id.
info A GeneSetCollection or a list of genes and the pathways they are involved.
method one of c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"), see Details.
... Other arguments passed to combineScores

Value

mclusterSim returns a matrix with the similarity scores for each cluster comparison.

Methods (by class)

* mclusterSim(clusters = list, info = GeneSetCollection): Calculates all the similarities of the GeneSetCollection and combine them using combineScoresPar()

Author(s)

Lluís Revilla

See Also

For a different approach see clusterGeneSim(), combineScores() and conversions()
Examples

```r
if (require("org.Hs.eg.db")) {
  # Extract the paths of all genes of org.Hs.eg.db from KEGG (last update in
  # data of June 31st 2011)
  genes.kegg <- as.list(org.Hs.egPATH)

  clusters <- list(
    cluster1 = c("18", "81", "10"),
    cluster2 = c("100", "10", "1"),
    cluster3 = c("18", "10", "83")
  )

  mclusterSim(clusters, genes.kegg)
  mclusterSim(clusters, genes.kegg, "avg")
} else {
  warning("You need org.Hs.eg.db package for this example")
}
```

---

**mgeneSim**

*Similarity score genes based on pathways similarity*

**Description**

Given two genes, calculates the Dice similarity between each pathway which is combined to obtain a similarity between the genes.

**Usage**

```r
mgeneSim(genes, info, method = "max", ...)
```

## S4 method for signature 'character,GeneSetCollection'

```r
mgeneSim(genes, info, method = "max", ...)
```

## S4 method for signature 'missing,GeneSetCollection'

```r
mgeneSim(genes, info, method = "max", ...)
```

**Arguments**

- **genes**: A vector of genes.
- **info**: A GeneSetCollection or a list of genes and the pathways they are involved.
- **method**: one of c("avg", "max", "rcmax", "rcmax.avg", "BMA", "reciprocal"), see Details.
- **...**: Other arguments passed to combineScores

**Details**

Given the information about the genes and their pathways, uses the ids of the genes to find the Dice similarity score for each pathway comparison between the genes. Later this similarities are combined using `combineScoresPar()`.
Value

\texttt{mgeneSim} returns the matrix of similarities between the genes in the vector.

Methods (by class)

- \texttt{mgeneSim(genes = character, info = GeneSetCollection)}: Calculates all the similarities of the list and combine them using \texttt{combineScoresPar()}

- \texttt{mgeneSim(genes = missing, info = GeneSetCollection)}: Calculates all the similarities of the list and combine them using \texttt{combineScoresPar()}

Note

genes accept named characters and the output will use the names of the genes.

See Also

geneSim(), conversions() help page to transform Dice score to Jaccard score. For the method to combine the scores see combineScoresPar().

Examples

if (require("org.Hs.eg.db") & require("reactome.db")) {
  # Extract the paths of all genes of org.Hs.eg.db from KEGG
  # (last update in data of June 31st 2011)
  genes.kegg <- as.list(org.Hs.egPATH)
  # Extracts the paths of all genes of org.Hs.eg.db from reactome
  genes.react <- as.list(reactomeEXTID2PATHID)
  mgeneSim(c("81", "18", "10"), genes.react)
  mgeneSim(c("81", "18", "10"), genes.react, "avg")
  named_genes <- structure(c("81", "18", "10"),
  .Names = c("ACTN4", "ABAT", "NAT2")
  )
  mgeneSim(named_genes, genes.react, "max")
} else {
  warning("You need reactome.db and org.Hs.eg.db package for this example")
}

\begin{verbatim}

mpathSim

Calculates the Dice similarity between pathways

Description

Calculates the similarity between several pathways using dice similarity score. If one needs the matrix of similarities between pathways set the argument methods to NULL.
\end{verbatim}
**Usage**

```r
mpathSim(pathways, info, method = NULL, ...)
```

## S4 method for signature 'character,GeneSetCollection,ANY'

```r
mpathSim(pathways, info, method = NULL, ...)
```

## S4 method for signature 'missing,GeneSetCollection,ANY'

```r
mpathSim(pathways, info, method = NULL, ...)
```

## S4 method for signature 'missing,list,ANY'

```r
mpathSim(pathways, info, method = NULL, ...)
```

## S4 method for signature 'missing,list,missing'

```r
mpathSim(pathways, info, method = NULL, ...)
```

**Arguments**

- **pathways**
  - Pathways to calculate the similarity for
- **info**
  - A list of genes and the pathways they are involved or a GeneSetCollection object
- **method**
  - To combine the scores of each pathway, one of c("avg", "max", "rcmax", "rcmax.avg", "BMA"), if NULL returns the matrix of similarities.
- **...**
  - Other arguments passed to `combineScoresPar()`

**Value**

The similarity between those pathways or all the similarities between each comparison.

**Methods (by class)**

- `mpathSim(pathways = character, info = GeneSetCollection, method = ANY)`: Calculates the similarity between the provided pathways of the GeneSetCollection using `combineScoresPar`
- `mpathSim(pathways = missing, info = GeneSetCollection, method = ANY)`: Calculates all the similarities of the GeneSetCollection and combine them using `combineScoresPar`
- `mpathSim(pathways = missing, info = list, method = ANY)`: Calculates all the similarities of the list and combine them using `combineScoresPar`
- `mpathSim(pathways = missing, info = list, method = missing)`: Calculates all the similarities of the list

**Note**

Pathways accept named characters, and then the output will have the names

**See Also**

- `pathSim()` For single pairwise comparison.
- `conversions()` To convert the Dice similarity to Jaccard similarity
pathSim

Calculates the Dice similarity between pathways

Description

Calculates the similarity between pathways using dice similarity score. diceSim is used to calculate similarities between the two pathways.

Usage

pathSim(pathway1, pathway2, info)

## S4 method for signature 'character,character,GeneSetCollection'
pathSim(pathway1, pathway2, info)

Arguments

- **pathway1**, **pathway2**
  A single pathway to calculate the similarity

- **info**
  A GeneSetCollection or a list of genes and the pathways they are involved.

Value

The similarity between those pathways or all the similarities between each comparison.

Methods (by class)

- pathSim(pathway1 = character, pathway2 = character, info = GeneSetCollection): Calculates all the similarities of a GeneSetCollection and combine them using combineScoresPar
plot_data

Author(s)
Lluís Revilla

See Also
conversions() help page to transform Dice score to Jaccard score. mpathSim() for multiple pairwise comparison of pathways.

Examples
if (require("reactome.db")) {
  # Extracts the paths of all genes of org.Hs.eg.db from reactome
  genes.react <- as.list(reactomeEXTID2PATHID)
  (paths <- sample(unique(unlist(genes.react)), 2))
  pathSim(paths[1], paths[2], genes.react)
} else {
  warning("You need reactome.db package for this example")
}

plot_data
The position of the nodes is based on the similarity between them.

Description
The position of the nodes is based on the similarity between them.
Plot how similar are the data

Usage
plot_data(x, top)
plot_similarity(pd)

Arguments

x
Matrix with the similarities.

top
a number between 0 and 1 to select the edges relating the elements of the matrix.

pd
The plot data from plot_data() function.

Value
A list with two elements:

- nodes: The position and name of the nodes
- edges: The information about the selected edges

A ggplot object
Examples

```r
if (require("org.Hs.eg.db") & require("reactome.db")) {
  # Extract the paths of all genes of org.Hs.eg.db from KEGG
  # (last update in data of June 31st 2011)
  genes.kegg <- as.list(org.Hs.egPATH)
  # Extracts the paths of all genes of org.Hs.eg.db from reactome
  genes.react <- as.list(reactomeEXTID2PATHID)

  sim <- mgeneSim(c("81", "18", "10"), genes.react)
  pd <- plot_data(sim, top = 0.25)
  if (requireNamespace("ggplot2", quietly = TRUE)){
    plot_similarity(pd)
  }
}
```

removeDup  

---

**removeDup**  

*Remove duplicated rows and columns*

Description

Given the indices of the duplicated entries remove the columns and rows until just one is left, it keeps the duplicated with the highest absolute mean value.

Usage

```r
removeDup(cor_mat, dupli)
```

Arguments

- `cor_mat`: List of matrices
- `dupli`: List of indices with duplicated entries

Value

A matrix with only one of the columns and rows duplicated

Author(s)

Lluís Revilla

See Also

duplicateIndices() to obtain the list of indices with duplicated entries.
`seq2mat`  

**Examples**

```r
a <- seq2mat(c("52", "52", "53", "55"), runif(choose(4, 2)))
b <- seq2mat(c("52", "52", "53", "55"), runif(choose(4, 2)))
mat <- list("kegg" = a, "react" = b)
mat
dupli <- duplicateIndices(rownames(a))
remat <- removeDup(mat, dupli)
remat
```

---

**seq2mat**

*Transforms a vector to a symmetric matrix*

**Description**

Fills a matrix of \( n_{col} = \text{length}(x) \) and \( n_{row} = \text{length}(x) \) with the values in \( \text{dat} \) and setting the diagonal to 1.

**Usage**

```r
seq2mat(x, dat)
```

**Arguments**

- **x**: names of columns and rows, used to define the size of the matrix.
- **dat**: Data to fill with the matrix with except the diagonal.

**Details**

\( \text{dat} \) should be at least \( \text{choose(length(x), 2)} \) of length. It assumes that the data provided comes from using the row and column id to obtain it.

**Value**

A square matrix with the diagonal set to 1 and \( \text{dat} \) on the upper and lower triangle with the columns ids and row ids from \( x \).

**Author(s)**

Lluís Revilla

**See Also**

`upper.tri()` and `lower.tri()`

**Examples**

```r
seq2mat(LETTERS[1:5], 1:10)
seq2mat(LETTERS[1:5], seq(from = 0.1, to = 1, by = 0.1))
```
similarities

Apply a function to a list of similarities

Description

Function to join list of similarities by a function provided by the user.

Usage

similarities(sim, func, ...)

Arguments

sim list of similarities to be joined. All similarities must have the same dimensions. The genes are assumed to be in the same order for all the matrices.

func function to perform on those similarities: prod, sum... It should accept as many arguments as similarities matrices are provided, and should use numbers.

... Other arguments passed to the function func. Usually na.rm or similar.

Value

A matrix of the size of the similarities

Note

It doesn’t check that the columns and rows of the matrices are in the same order or are the same.

Author(s)

Lluís Revilla

See Also

weighted() for functions that can be used, and addSimilarities() for a wrapper to one of them

Examples

set.seed(100)
a <- seq2mat(LETTERS[1:5], rnorm(10))
b <- seq2mat(LETTERS[1:5], seq(from = 0.1, to = 1, by = 0.1))
sim <- list(b, a)
similarities(sim, weighted.prod, c(0.5, 0.5))
# Note the differences in the sign of some values
similarities(sim, weighted.sum, c(0.5, 0.5))
Weighted operations

Description
Calculates the weighted sum or product of \( x \). Each value should have its weight, otherwise it will throw an error.

Usage

\[
\begin{align*}
\text{weighted.sum}(x, w, \text{abs} = \text{TRUE}) \\
\text{weighted.prod}(x, w)
\end{align*}
\]

Arguments

- \( x \): an object containing the values whose weighted operations is to be computed
- \( w \): a numerical vector of weights the same length as \( x \) giving the weights to use for elements of \( x \).
- \( \text{abs} \): If any \( x \) is negative you want the result negative too?

Details
This functions are thought to be used with similarities. As some similarities might be positive and others negative the argument \( \text{abs} \) is provided for \text{weighted.sum}, assuming that only one similarity will be negative (usually the one coming from expression correlation).

Value

\text{weighted.sum} returns the sum of the product of \( x \times \text{weights} \) removing all NA values. See parameter \( \text{abs} \) if there are any negative values.
\text{weighted.prod} returns the product of product of \( x \times \text{weights} \) removing all NA values.

Author(s)
Lluís Revilla

See Also

\text{weighted.mean()}, \text{similarities()} and \text{addSimilarities()}

Examples

```r
expr <- c(-0.2, 0.3, 0.5, 0.8, 0.1)
weighted.sum(expr, c(0.5, 0.2, 0.1, 0.1, 0.1))
weighted.sum(expr, c(0.5, 0.2, 0.1, 0.2, 0.1), FALSE)
weighted.sum(expr, c(0.4, 0.2, 0.1, 0.2, 0.1))
weighted.sum(expr, c(0.4, 0.2, 0.1, 0.2, 0.1), FALSE)
weighted.sum(expr, c(0.4, 0.2, 0, 0.2, 0.1))
weighted.sum(expr, c(0.5, 0.2, 0, 0.2, 0.1))
# Compared to weighted.prod:
weighted.prod(expr, c(0.5, 0.2, 0.1, 0.1, 0.1))
weighted.prod(expr, c(0.4, 0.2, 0.1, 0.2, 0.1))
weighted.prod(expr, c(0.4, 0.2, 0, 0.2, 0.1))
weighted.prod(expr, c(0.5, 0.2, 0, 0.2, 0.1))
```
# Index

* **internal**
  * incidence, list-method, 15

  addSimilarities, 3
  addSimilarities(), 26, 27
  AintoB, 4

  BioCor (BioCor-package), 2
  BioCor-package, 2

  clusterGeneSim, 5
  clusterGeneSim(), 3, 7, 12, 17, 18
  clusterGeneSim, character, character, GeneSetCollection-method
     (clusterGeneSim), 5
  clusterSim, 7
  clusterSim(), 3, 6, 12, 17
  clusterSim, character, character, GeneSetCollection-method
     (clusterSim), 7

  combinadic, 8
  combineScores, 9, 18
  combineScores(), 6, 7, 12, 17, 18
  combineScoresPar (combineScores), 9
  combineScoresPar(), 6, 7, 14, 15, 17–21
  combineSources, 11
  combn(), 8
  conversions, 12
  conversions(), 3, 7, 13, 15, 18, 20, 21, 23

  D2J (conversions), 12
  diceSim, 12
  diceSim(), 12
  duplicateIndices, 13
  duplicateIndices(), 24

  geneSim, 14
  geneSim(), 3, 12, 13, 20
  geneSim, character, character, GeneSetCollection-method
     (geneSim), 14
  GOSemSim::combineScores(), 10

  incidence, list-method, 15

  inverseList, 16

  J2D (conversions), 12

  lower.tri(), 25

  mclusterGeneSim, 17
  mclusterGeneSim(), 6
  mclusterGeneSim, list, GeneSetCollection-method
     (mclusterGeneSim), 17
  mclusterSim, 18
  mclusterSim, list, GeneSetCollection-method
     (mclusterSim), 18
  mgeneSim, 19
  mgeneSim(), 15
  mgeneSim, character, GeneSetCollection-method
      (mgeneSim), 19
  mgeneSim, missing, GeneSetCollection-method
     (mgeneSim), 19
  mpathSim, 20
  mpathSim(), 23
  mpathSim, character, GeneSetCollection, ANY-method
     (mpathSim), 20
  mpathSim, missing, GeneSetCollection, ANY-method
     (mpathSim), 20
  mpathSim, missing, list, ANY-method
     (mpathSim), 20
  mpathSim, missing, list, missing-method
     (mpathSim), 20

  pathSim, 22
  pathSim(), 3, 21
  pathSim, character, character, GeneSetCollection-method
     (pathSim), 22
  plot_data, 23
  plot_similarity (plot_data), 23

  register, 10
  removeDup, 24
  removeDup(), 14
seq2mat, 25  
similarities, 26  
similarities(), 3, 4, 27

upper.tri(), 25

weighted, 27  
weighted(), 3, 4, 26  
weighted.mean(), 27