Package ‘CausalR’

May 29, 2024

Type Package

Title Causal network analysis methods

Version 1.36.0

Date 2016-11-14

Author Glyn Bradley, Steven Barrett, Chirag Mistry, Mark Pipe, David Wille, David Riley, Bhushan Bonde, Peter Woollard

Maintainer Glyn Bradley <glyn.x.bradley@gsk.com>, Steven Barrett <steven.j.barrett@gsk.com>

Description Causal network analysis methods for regulator prediction and network reconstruction from genome scale data.

Depends R (>= 3.2.0)

Imports igraph

Suggests knitr, RUnit, BiocGenerics

VignetteBuilder knitr

biocViews ImmunoOncology, SystemsBiology, Network, GraphAndNetwork, Network Inference, Transcriptomics, Proteomics, DifferentialExpression, RNASeq, Microarray

License GPL (>= 2)

NeedsCompilation no

RoxygenNote 5.0.1

git_url https://git.bioconductor.org/packages/CausalR
git_branch RELEASE_3_19
git_last_commit bd8a70d
git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-29
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CausalR-package</td>
<td>3</td>
</tr>
<tr>
<td>AddIDsToVertices</td>
<td>4</td>
</tr>
<tr>
<td>AddWeightsToEdges</td>
<td>4</td>
</tr>
<tr>
<td>AnalyseExperimentalData</td>
<td>5</td>
</tr>
<tr>
<td>AnalysePredictionsList</td>
<td>5</td>
</tr>
<tr>
<td>CalculateEnrichmentPValue</td>
<td>6</td>
</tr>
<tr>
<td>CalculateSignificance</td>
<td>7</td>
</tr>
<tr>
<td>CalculateSignificanceUsingCubicAlgorithm</td>
<td>8</td>
</tr>
<tr>
<td>CalculateSignificanceUsingCubicAlgorithm1b</td>
<td>9</td>
</tr>
<tr>
<td>CalculateSignificanceUsingQuarticAlgorithm</td>
<td>10</td>
</tr>
<tr>
<td>CalculateTotalWeightForAllContingencyTables</td>
<td>11</td>
</tr>
<tr>
<td>CalculateWeightGivenValuesInThreeByThreeContingencyTable</td>
<td>11</td>
</tr>
<tr>
<td>CheckPossibleValuesAreValid</td>
<td>12</td>
</tr>
<tr>
<td>CheckRowAndColumnSumValuesAreValid</td>
<td>12</td>
</tr>
<tr>
<td>CompareHypothesis</td>
<td>13</td>
</tr>
<tr>
<td>ComputeFinalDistribution</td>
<td>14</td>
</tr>
<tr>
<td>ComputePValueFromDistributionTable</td>
<td>14</td>
</tr>
<tr>
<td>CreateCCG</td>
<td>15</td>
</tr>
<tr>
<td>CreateCG</td>
<td>16</td>
</tr>
<tr>
<td>CreateNetworkFromTable</td>
<td>16</td>
</tr>
<tr>
<td>DetermineInteractionTypeOfPath</td>
<td>17</td>
</tr>
<tr>
<td>FindApproximateValuesThatWillMaximiseDValue</td>
<td>17</td>
</tr>
<tr>
<td>FindIdsOfConnectedNodesInSubgraph</td>
<td>18</td>
</tr>
<tr>
<td>FindMaximumDValue</td>
<td>19</td>
</tr>
<tr>
<td>GetAllPossibleRoundingCombinations</td>
<td>19</td>
</tr>
<tr>
<td>GetApproximateMaximumDValueFromThreeByTwoContingencyTable</td>
<td>20</td>
</tr>
<tr>
<td>GetApproximateMaximumDValueFromTwoByTwoContingencyTable</td>
<td>21</td>
</tr>
<tr>
<td>GetCombinationsOfCorrectandIncorrectPredictions</td>
<td>22</td>
</tr>
<tr>
<td>GetExplainedNodesOfCCG</td>
<td>22</td>
</tr>
<tr>
<td>GetInteractionInformation</td>
<td>23</td>
</tr>
<tr>
<td>GetMatrixOfCausalRelationships</td>
<td>24</td>
</tr>
<tr>
<td>GetMaxDValueForAFamily</td>
<td>24</td>
</tr>
<tr>
<td>GetMaxDValueForAThreeByTwoFamily</td>
<td>25</td>
</tr>
<tr>
<td>GetMaximumDValueFromTwoByTwoContingencyTable</td>
<td>26</td>
</tr>
<tr>
<td>GetNodeID</td>
<td>27</td>
</tr>
<tr>
<td>GetNodeName</td>
<td>28</td>
</tr>
<tr>
<td>GetNumberOfPositiveAndNegativeEntries</td>
<td>28</td>
</tr>
<tr>
<td>GetPathsInSifFormat</td>
<td>29</td>
</tr>
<tr>
<td>GetRegulatedNodes</td>
<td>29</td>
</tr>
<tr>
<td>GetRowAndColumnSumValues</td>
<td>30</td>
</tr>
<tr>
<td>GetScoreForNumbersOfCorrectandIncorrectPredictions</td>
<td>31</td>
</tr>
<tr>
<td>GetScoresForSingleNode</td>
<td>31</td>
</tr>
<tr>
<td>GetScoresWeightsMatrix</td>
<td>32</td>
</tr>
<tr>
<td>GetScoresWeightsMatrixByCubicAlg</td>
<td>33</td>
</tr>
<tr>
<td>GetSetOfDifferentiallyExpressedGenes</td>
<td>34</td>
</tr>
<tr>
<td>GetSetOfSignificantPredictions</td>
<td>34</td>
</tr>
</tbody>
</table>
The CausalR package

Description

Causal network analysis methods for regulator prediction and network reconstruction from genome scale data.

Details

The most important functions are:

- **CreateCCG**: read a computational causal graph from a .sif file
- **ReadExperimentalData**: read a experimental data from a .txt file
- **MakePredictions**: make causal reasoning predictions from a CCG
- **ScoreHypothesis**: score causal reasoning predictions
- **CalculateSignificance**: calculate statistical significance of a result
- **RankTheHypotheses**: compare different possible regulatory hypotheses on a single CCG
- **runSCANR**: reduce false positives by selecting common hypotheses across pathlengths
- **WriteExplainedNodesToSifFile**: reconstruct hypothesis specific regulatory network
AddWeightsToEdges

Author(s)
Glyn Bradley, Steven J. Barrett, Chirag Mistry, Mark Pipe, David Riley, David Wille, Bhushan Bonde, Peter Woollard

References
- "CausalR - extracting mechanistic sense from genome scale data", Bradley, G. and Barrett, S.J., Application note, Bioinformatics (submitted)

AddIDsToVertices add IDs to vertices

Description
Adds the IDs as a vertex property to the vertices in the network. Used when creating sub-networks where the new nodes will retain the IDs from their original network

Usage
AddIDsToVertices(network)

Arguments
network the network to which the IDs are to be added

Value
network with IDs added

AddWeightsToEdges add weights to edges

Description
Adds weight information to the edges of given network (1 for activation and -1 for inhibition)

Usage
AddWeightsToEdges(network, tableOfInteractions)
AnalyseExperimentalData

Arguments

- network: an igraph constructed from the original .sif file
- tableOfInteractions: a column of the corresponding .sif file indicating the direction of activation/interaction

Value

- an augmented network

Description

Returns the number of up- and down-regulated genes in the experimental data

Usage

AnalyseExperimentalData(experimentalData)

Arguments

- experimentalData: a dataframe containing a list of genes with corresponding direction of change (1 or -1)

Value

- up and down regulation statistics for the experimental data

AnalysePredictionsList

analyse predictions list

Description

Taking the list of predictions from a particular hypothesis, counts the number of positive and negative predictions in the list and the number of 0’s (from numPredictions).

Usage

AnalysePredictionsList(predictionsList, numPredictions)
CalculateEnrichmentPValue

Arguments

predictionsList
  list of predictions
numPredictions
  number of predictions

Value

prediction statistics

Examples

```
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
ccg <- CreateCCG(network)
predictions <- MakePredictions('NodeA', +1, ccg, 2)
AnalysePredictionsList(predictions,8)
```

---

CalculateEnrichmentPValue

*calculates an enrichment p-value*

Description

Calculate a enrichment p-value for a given hypothesis by comparing the corresponding predicted and observed gene changes

Usage

CalculateEnrichmentPValue(predictions, results)

Arguments

predictions
  predictions of changes from the CCG for a particular hypothesis
results
  gene changes observed in the experimental data

Value

an enrichment p-value

Examples

```
predictions <- matrix(c(1,2,3,1,1,-1), ncol = 2)
results<- matrix(c(1,2,3,4,1,1,-1,1), ncol = 2)
CalculateEnrichmentPValue(predictions, results)
```
CalculateSignificance  calculate overall significance p-value

Description

Calculates the p-value of a score given the hypothesis score and the distribution table, using either the quartic or the (faster) cubic algorithm.

Usage

CalculateSignificance(hypothesisScore, predictionListStats, experimentalResultStats, epsilon = 1e-05, useCubicAlgorithm = TRUE, use1bAlgorithm = TRUE)

Arguments

- hypothesisScore: score for a particular hypothesis
- predictionListStats: numbers of predicted up-regulated, predicted down-regulated and ambiguous predictions predicted by the algorithm
- experimentalResultStats: numbers of up-regulated, down-regulated and not significantly changed transcripts in the experimental data
- epsilon: threshold that is used when calculating the p-value using the cubic algorithm
- useCubicAlgorithm: use the cubic algorithm, defaults to TRUE
- use1bAlgorithm: use the 1b version of the algorithm, defaults to TRUE used to calculate the p-value

Value

the resulting p-value

Examples

CalculateSignificance(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=TRUE)
CalculateSignificanceUsingQuarticAlgorithm(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=FALSE)
CalculateSignificance(5, c(7,4,19), c(6,6,18), 1e-5)
CalculateSignificance(5, c(7,4,19), c(6,6,18), epsilon=1e-5, useCubicAlgorithm=TRUE)
CalculateSignificanceUsingCubicAlgorithm(5, c(7,4,19), c(6,6,18), 1e-5)
CalculateSignificanceUsingCubicAlgorithm

*calculate significance using the cubic algorithm*

**Description**

Calculates the p-value of a score given the hypothesis score and the distribution table (calculated using the cubic algorithm)

**Usage**

```r
CalculateSignificanceUsingCubicAlgorithm(hypothesisScore, predictionListStats, experimentalDataStats, epsilon)
```

**Arguments**

- `hypothesisScore` the score whose p-value we want to find.
- `predictionListStats` numbers of predicted up-regulated, predicted down-regulated and ambiguous predictions.
- `experimentalDataStats` numbers of up-regulated, down-regulated and not significantly changed transcripts in the experimental data.
- `epsilon` an epsilon threshold that is used when calculating the p-value using the cubic algorithm. Defaults to 1e-5.

**Value**

p-value

**References**


**Examples**

```r
CalculateSignificance(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=TRUE)
CalculateSignificanceUsingQuarticAlgorithm(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=FALSE)
CalculateSignificance(5, c(7,4,19), c(6,6,18), 1e-5)
CalculateSignificance(5, c(7,4,19), c(6,6,18), epsilon=1e-5, useCubicAlgorithm=TRUE)
CalculateSignificanceUsingCubicAlgorithm(5, c(7,4,19), c(6,6,18), 1e-5)
```
Description

Calculate the p-value of a score given the hypothesis score and the distribution table (calculated using the cubic algorithm 1b in Assessing statistical significance in causal graphs - Chindelevitch et al)

Usage

`CalculateSignificanceUsingCubicAlgorithm1b(hypothesisScore, predictionListStats, experimentalDataStats, epsilon)`

Arguments

- `hypothesisScore`:
  - The score whose p-value we want to find.

- `predictionListStats`:
  - Number of predicted up-regulated, predicted down-regulated and ambiguous predictions.

- `experimentalDataStats`:
  - Number of up-regulated, down-regulated and not significantly changed transcripts in the experimental data.

- `epsilon`:
  - The threshold that is used when calculating the p-value using the cubic algorithm. (Defaults to 1e-5, only used for the cubic algorithm, ignored if useCubicAlgorithm is FALSE.)

Value

p value

Examples

`CalculateSignificance(5, c(7,4,19), c(6,6,18))`
`CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=TRUE)`
`CalculateSignificanceUsingQuarticAlgorithm(5, c(7,4,19), c(6,6,18))`
`CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=FALSE)`
`CalculateSignificance(5, c(7,4,19), c(6,6,18), 1e-5)`
`CalculateSignificance(5, c(7,4,19), c(6,6,18), epsilon=1e-5, useCubicAlgorithm=TRUE)`
`CalculateSignificanceUsingCubicAlgorithm1b(5, c(7,4,19), c(6,6,18), 1e-5)`
CalculateSignificanceUsingQuarticAlgorithm

calculate significance using the quartic algorithm

Description

Computes the significance of a given hypothesis. For a detailed description of the algorithm see Causal reasoning on biological networks: interpreting transcriptional changes - Chindelevitch et al., section 2. from which the methods and notation is taken.

Usage

CalculateSignificanceUsingQuarticAlgorithm(hypothesisScore, predictionListStats, experimentalDataStats)

Arguments

hypothesisScore
the score for which a p-value is required

predictionListStats
a vector containing the values q+, q- and q0 (the number of positive/negative/non-significant or contradictory) predictions

experimentalDataStats
a vector containing the values n+, n- and n0 (the number of positive/negative/non-significant (or contradictory) transcripts in the results) (or contradictory) transcripts in the results

Value

the corresponding p-value

References


Examples

CalculateSignificance(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=TRUE)
CalculateSignificanceUsingQuarticAlgorithm(5, c(7,4,19), c(6,6,18))
CalculateSignificance(5, c(7,4,19), c(6,6,18), useCubicAlgorithm=FALSE)
CalculateSignificance(5, c(7,4,19), c(6,6,18), 1e-5)
CalculateSignificance(5, c(7,4,19), c(6,6,18), epsilon=1e-5, useCubicAlgorithm=TRUE)
CalculateSignificanceUsingCubicAlgorithm(5, c(7,4,19), c(6,6,18), 1e-5)
CalculateTotalWeightForAllContingencyTables

*calculate total weight for all contingency tables*

**Description**
Calculates the total weights or D-values for all possible contingency tables. This value can be used to calculate the p-value.

**Usage**

```
CalculateTotalWeightForAllContingencyTables(experimentalDataStats, returnlog = FALSE)
```

**Arguments**

- `experimentalDataStats`  
a vector containing the values n+, n- and n0, the number of positive/negative/non-significant (or contradictory) transcripts in the results
- `returnlog`  
whether the result should be returned as a log. Default is FALSE.

**Value**
a D-value or weight

---

CalculateWeightGivenValuesInThreeByThreeContingencyTable

*calculate weight given values in three-by-three contingency table*

**Description**

Given the values in the three by three contingency table and the values of the number of positive/negative/non-significant predictions (q+, q-, q0) this function calculates the D-value (or weight).

**Usage**

```
CalculateWeightGivenValuesInThreeByThreeContingencyTable(threeByThreeContingencyTable, logOfFactorialOfPredictionListStats, returnlog = FALSE)
```

**Arguments**

- `threeByThreeContingencyTable`  
a 3x3 contingency table
- `logOfFactorialOfPredictionListStats`  
log of Factorial of prediction statistics
- `returnlog`  
should the result be returned as a log value. Default is FALSE.
CheckRowAndColumnSumValuesAreValid

Value
a D-value (or weight)

CheckPossibleValuesAreValid
check possible values are valid

Description
Checks if the a given set of possible values for n++, n+, n-+ and n– are agree with the predicted and experimental data

Usage
CheckPossibleValuesAreValid(predictionDataStats, experimentalDataStats, possibleValues)

Arguments
predictionDataStats
a vector of predicted results
experimentalDataStats
a vector of observed experimental results
possibleValues a vector of possible values n++, n+, n-+ and n–

Value
TRUE if and only if the given vector of possible values is valid

CheckRowAndColumnSumValuesAreValid
check row and column sum values are valid

Description
Checks to see if the values of r+, r-, c+ and c- which are stored in rowAndColumnSumValues define a valid contingency table

Usage
CheckRowAndColumnSumValuesAreValid(rowAndColumnSumValues, predictionListStats, experimentalResultStats)
**Arguments**

- `rowAndColumnSumValues`:
  A 4x1 vector containing the row and column sum values \((r+, r-, c+, c-)\) for a 2x2 contingency table.

- `predictionListStats`:
  A vector containing the values \(q+, q-, q0\).

- `experimentalResultStats`:
  A vector containing the values \(n+, n-, n0\).

**Value**

TRUE if the table is valid; otherwise FALSE.

**Description**

Compare the predictions from a hypothesis with the experimental data returning an matrix with columns for node ID, predictions, experimental results and the corresponding scores.

**Usage**

```r
CompareHypothesis(matrixOfPredictions, matrixOfExperimentalData, ccg = NULL, sourceNode = NULL)
```

**Arguments**

- `matrixOfPredictions`:
  A matrix of predictions.

- `matrixOfExperimentalData`:
  A matrix of experimental data.

- `ccg`:
  A CCG network (default=NULL).

- `sourceNode`:
  A starting node (default=NULL).

**Value**

A matrix containing predictions, observations and scores.

**Examples**

```r
predictions <- matrix(c(1,2,3,+1,0,-1),ncol=2)
experimentalData <- matrix(c(1,2,4,+1,+1,-1),ncol=2)
ScoreHypothesis(predictions,experimentalData)
CompareHypothesis(predictions,experimentalData)
```
ComputeFinalDistribution

describe final distribution

Description
Computes a final reference distribution of the score used to compute the final p-value.

Usage
ComputeFinalDistribution(resultsMatrix)

Arguments
- resultsMatrix: a matrix containing the scores and weights from which the distribution is to be calculated

Value
distributionMatrix: a matrix containing the reference distribution for the score

ComputePValueFromDistributionTable

describe a p-value from the distribution table

Description
Computes the p-value of the score of a hypothesis, based on a distribution table.

Usage
ComputePValueFromDistributionTable(scoreOfHypothesis, distributionMatrix, totalWeights)

Arguments
- scoreOfHypothesis: a score of hypothesis
- distributionMatrix: a distribution table presented as a matrix
- totalWeights: a matrix of total weights

Value
a p-value
**CreateCCG**

create a Computational Causal Graph (CCG)

**Description**

Creates a computational causal graph from a network file.

**Usage**

```r
CreateCCG(filename, nodeInclusionFile = NULL, excludeNodesInFile = TRUE)
```

**Arguments**

- `filename`: file name of the network file (in .sif file format)
- `nodeInclusionFile`: optional path to a text file listing nodes to exclude in the CCG (or include - see argument `excludeNodesInFile`).
- `excludeNodesInFile`: flag to determine if nodes in inclusion file should be taken as nodes to include or nodes to exclude. Default is TRUE to exclude.

**Value**

an igraph object containing the CCG.

**Note**

CreateCG and CreateCCG create causal and computational causal graphs respectively.

**References**


**Examples**

```r
# get path to example .sif file
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
# create ccg
ccg = CreateCCG(network)
```
**CreateCG**

Create a Computational Graph (CG)

**Description**

Creates a CG network from a .sif file. Takes in a .sif file output from Cytoscape, and creates an 'igraph' representing the network. The edges will be annotated with the type of interaction and a weight (1 for activation and -1 for inhibition).

**Usage**

CreateCG(sifFile)

**Arguments**

- **sifFile**
  
  the path of the .sif file that contains all the information about the network

**Value**

- a CG network

**Examples**

```r
# get path to example .sif file
doctable <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
#create cg
cg = CreateCG(doctable)
```

---

**CreateNetworkFromTable**

Create network from table

**Description**

Creates a network from an internal data table created from a .sif file: this function converts the data read in from the .sif file into an igraph in R.

**Usage**

CreateNetworkFromTable(dataTable)

**Arguments**

- **dataTable**
  
  the data table containing the information read in from the .sif file representing the network.
**DetermineInteractionTypeOfPath**

Determines the sign of a given path. Given a path and through the network, this function will determine if the path results in activation or inhibition. Activation is indicated by 1, inhibition by -1.

### Usage

```r
determineInteractionTypeOfPath(network, nodesInPath)
```

### Arguments

- **network**: an igraph representing the network
- **nodesInPath**: an ordered list of the nodes visited on the path - note that these contain numbers which use R’s internal reference to the edges

### Value

a signed integer representing the paths sign

---

**FindApproximateValuesThatWillMaximiseDValue**

find approximate values that will maximise D value

### Description

Finds an approximate table values to maximise D. Given the values of q+, q-, q0, n+, n- and n0, this function will produce the approximate values of n++, n+-, n-+ and n– that will maximise the D value. See Assessing statistical significance of causal graphs, page 6. The values are approximate since they need to be rounded, although the direction of rounding is not clear at this stage.

### Usage

```r
findApproximateValuesThatWillMaximiseDValue(predictionListStats, experimentalDataStats)
```
Arguments

- predictionListStats
  a vector containing the values q+, q- and q0: numbers of positive, negative and non-significant/contradictory predictions

- experimentalDataStats
  a vector containing the values n+, n- and n0: numbers of positive, negative and non-significant/contradictory predictions

Value

a 2x2 contingency table which approximately maximises D

References


Description

Adds the IDs of the connected nodes in a subgraph to an existing list. Given the IDs of connected nodes in the full network, this function will find the corresponding IDs in the subgraph

Usage

FindIdsOfConnectedNodesInSubgraph(idsOfConnectedNodes, subgraphOfConnectedNodes)

Arguments

- idsOfConnectedNodes
  a list of connected nodes in the full graph

- subgraphOfConnectedNodes
  a subgraph

Value

a list of connected nodes in the subgraph
FindMaximumDValue  

find maximum D value

Description

computes the maximum possible D-value for given values q+, q-, q0 and n+, n-, n0.

Usage

FindMaximumDValue(predictionListStats, experimentalDataStats, 
logOfFactorialOfPredictionListStats, returnlog = FALSE)

Arguments

predictionListStats  
a vector containing the predicted values q+, q- and q0: numbers of positive, negative and non-significant/contradictory predictions

experimentalDataStats  
A vector containing the observed values n+, n- and n0: numbers of positive, negative and non-significant/contradictory observations

logOfFactorialOfPredictionListStats  
a vector containing the log of the factorial value for each entry in predictionListStats

returnlog  
should the result be returned as a log; default FALSE

Value

the maximum possible D value

GetAllPossibleRoundingCombinations

get score for numbers of correct and incorrect predictions

Description

Returns all possible rounding combinations of a 2x2 table. Given the values of n++, n+-, n-+ and n– (stored in twoByTwoContingencyTable) this function will compute all possibilities of rounding each value up or down.

Usage

GetAllPossibleRoundingCombinations(twoByTwoContingencyTable)
GetApproximateMaximumDValueFromThreeByTwoContingencyTable

Arguments

twoByTwoContingencyTable
  approximate values of n++, n+-, n-+ and n--, these values are calculated to optimise the D-value (see page 6 of Assessing statistical significance of causal graphs)

Value

a matrix of rounding combinations

GetApproximateMaximumDValueFromThreeByTwoContingencyTable

returns approximate maximum D value or weight for a 3x2 superfamily

Description

Computes an approximate maximum D value (or weight) for a superfamily (3x2 table). The result is only approximate as only the first valid D value that is return. This has been done to speed up the overall algorithm.

Usage

GetApproximateMaximumDValueFromThreeByTwoContingencyTable(threeByTwoContingencyTable, predictionListStats, logOfFactorialOfPredictionListStats, returnlog = FALSE)

Arguments

threeByTwoContingencyTable
  approximate values of n++, n+-, n-+, n-- and n0+ and n0-, these values are calculated to optimise the D-value (see page 6 of Assessing statistical significance of causal graphs)

predictionListStats
  a vector containing the values q+, q- and q0 (the number of positive/negative/non-significant (or contradictory) predictions)

logOfFactorialOfPredictionListStats
  a vector containing the log of the factorial value for each entry in predictionListStats

returnlog
  return the result as a log, default is FALSE

Value

an approximate maximum D value or weight
GetApproximateMaximumDValueFromTwoByTwoContingencyTable

computes an approximate maximum D value or weight

Description

Computes an approximate maximum D value (or weight). The calculation is approximate since only the first valid D value that is round. This has been done to speed up the overall algorithm - to get the exact answer use GetMaximumDValueFromTwoByTwoContingencyTable.

Usage

GetApproximateMaximumDValueFromTwoByTwoContingencyTable(n_pp, n_pm, n_mp, n_mm, predictionListStats, experimentalDataStats, logOfFactorialOfPredictionListStats, returnlog = FALSE)

Arguments

n_pp the count n++ from the prediction-observation contingency matrix
n_pm the count n+- from the prediction-observation contingency matrix
n_mp the count n-+ from the prediction-observation contingency matrix
n_mm the count n– from the prediction-observation contingency matrix
predictionListStats a vector containing the values q+, q- and q0: the number of positive, negative, non-significant/contradictory predictions
experimentalDataStats a vector containing the values n+, n- and n0: the number of positive, negative, non-significant/contradictory observations
logOfFactorialOfPredictionListStats a vector containing the log of the factorial value for each entry in predictionListStats
returnlog return the result as a log, default is FALSE

Value

the maximum D value or weight
GetCombinationsOfCorrectandIncorrectPredictions
returns table of correct and incorrect predictions

Description
Returns the numbers of correct and incorrect positive and negative predictions

Usage
GetCombinationsOfCorrectandIncorrectPredictions(predictionDataStats,
experimentalDataStats)

Arguments
predictionDataStats
prediction data statistics table
experimentalDataStats
Experimental data statistics table

Value
a matrix the numbers of correct and incorrect positive and negative prediction

GetExplainedNodesOfCCG
Get explained nodes of CCG

Description
Returns a table of node names and values for explained nodes, i.e. nodes that appear in both
network and data with the same sign. The table contain the name in column 1 and the value (1 or
-1) in column 2

Usage
GetExplainedNodesOfCCG(hypothesisnode, signOfHypothesis, network,
experimentalData, delta)
**GetInteractionInformation**

**Arguments**

- **hypothesisnode**: a hypothesis node
- **signOfHypothesis**: the direction of change of hypothesis node
- **network**: a computational causal graph
- **experimentalData**: The experimental data read in using `ReadExperimentalData`. The results is an \( n \times 2 \) matrix; where the first column contains the node ids of the nodes in the network that the results refer to. The second column contains values indicating the direction of regulation in the results - (+)1 for up, -1 for down and 0 for insignificant amounts of regulation. The name of the first column is the filename the data was read from.
- **delta**: the number of edges across which the hypothesis should be followed

**Value**

- a vector of explained nodes

**Description**

Gets the interaction information from the input data

**Usage**

`GetInteractionInformation(dataTable)`

**Arguments**

- **dataTable**: a data table containing the information read in from the .sif file representing the network.

**Value**

- a vector of interaction information
GetMatrixOfCausalRelationships

**Description**
Get a matrix of causal relationships from the network and the IDs of connected nodes

**Usage**

GetMatrixOfCausalRelationships(hypothesis, network, idsOfConnectedNodesFromSubgraph)

**Arguments**

- **hypothesis**: a hypothesis node
- **network**: a CCG network
- **idsOfConnectedNodesFromSubgraph**: a list of connected nodes in the subgraph of interest

**Value**
causal relationships matrix

GetMaxDValueForAFamily

**Description**
Computes the maximum D value for a particular family - denoted as D_fam on page 6 of Assessing Statistical Significance of Causal Graphs

**Usage**

GetMaxDValueForAFamily(r_p, r_m, c_p, predictionListStats, experimentalDataStats, logOfFactorialOfPredictionListStats, returnlog = FALSE)
GetMaxDValueForAThreeByTwoFamily

Arguments

\( r_p \)
row sum \( r^+ \)

\( r_m \)
row sum \( r^- \)

\( c_p \)
column sum \( c^+ \)

\( \text{predictionListStats} \)
approximate values of \( n^+, n^-, n^+ \) and \( n^- \)

\( \text{experimentalDataStats} \)
a vector containing the values \( q^+, q^- \) and \( q^0 \): number of positive, negative, non-significant/contradictory predictions

\( \text{logOfFactorialOfPredictionListStats} \)
a vector containing the values \( n^+, n^- \) and \( n^0 \): number of positive, negative, non-significant/contradictory observations

\( \text{returnlog} \)
return result as log, default value is \text{FALSE}

Value

the maximum DFam Value

References


---

GetMaxDValueForAThreeByTwoFamily

get maximum \( D \) value for three-by-two a family

Description

Returns the maximum \( D \) value for a particular family as described as \( D_{fam} \) on pages 6 and 7 of Assessing Statistical Significance of Causal Graphs in Assessing Statistical Significance of Causal Graphs

Usage

\[
\text{GetMaxDValueForAThreeByTwoFamily}(r_p, r_m, r_z, n_p, n_m, \text{predictionListStats}, \text{logOfFactorialOfPredictionListStats}, \text{returnlog} = \text{FALSE})
\]

Arguments

\( r_p \)
a \( r^+ \) row sum from the prediction-observation matrix

\( r_m \)
a \( r^- \) row sum from the prediction-observation matrix

\( r_z \)
a \( r^0 \) row sum from the prediction-observation matrix

\( n_p \)
a number of predicted increases from the prediction-observation matrix
GetMaximumDValueFromTwoByTwoContingencyTable

Get Maximum D_fam Value

References

Description
Computes the maximum D value (or weight) given approximate values of n++, n+, n- and n--. These values are approximate and in general are non-integer values; they are found by using an approximation detailed in the paper Assessing statistical significance in causal graphs on page 6 i.e. n_ab is approximately equal to q_a*n_b/t where a and b are either +, - or 0. The value is an approximation since the direction in which the number should be rounded is not clear and hence this function runs through all possible combinations of rounding before concluding the maximum D-value.

Usage
GetMaximumDValueFromTwoByTwoContingencyTable(twoByTwoContingencyTable, predictionListStats, experimentalDataStats, logOfFactorialOfPredictionListStats, returnlog = FALSE)

Arguments

twoByTwoContingencyTable
  approximate values of n++, n+, n- and n--, these values are calculated to optimise the D-value

predictionListStats
  a vector containing the values q+, q- and q0 the number of positive/negative/non-significant (or contradictory) predictions
experimentalDataStats
   a vector containing the values n+, n- and n0 (the number of positive/negative/non-significant (or contradictory) transcripts in the results)

logOfFactorialOfPredictionListStats
   a vector containing the log of the factorial value for each entry in predictionListStats

returnlog
   whether or not the value should be returned as a log (TRUE) or not (FALSE)

Value
   the maximal D-value

References

---

### GetNodeID

**get CCG node ID**

**Description**

Returns the CCG node ID from a node name or a vector of node names and a given direction of regulation.

**Usage**

GetNodeID(network, nodename, direction = 1)

**Arguments**

- network: a CCG object
- nodename: the node name, or names, for which the ID is required
- direction: the direction of regulation of the required node or nodes. Maybe +1 (default) or -1.

**Value**

a scalar or vector containing the node ID or IDs requested
GetNodeName

get node name

Description

Returns the node name from one or more node IDs, or substitute node names for node IDs, given in
first column of a matrix typically of predictions or experimental data

Usage

GetNodeName(network, nodeID, signed = FALSE)

Arguments

network
Built from igraph
nodeID
a node ID or a matrix containing node IDs in its first column
signed
whether or not the node name should be signed. Setting this value to TRUE
gives a signed name indicating whether the gene is up or down regulated in the
network

Value

a node name or a vector of node names depending if the input is an matrix.

Examples

network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
ccg = CreateCCG(network)
nodeID <- 10
GetNodeName(ccg, nodeID)

GetNumberOfPositiveAndNegativeEntries

counts the number of positive and negative entries

Description

Counts the number of entries in the in the second column of an input table that are +1 or -1.

Usage

GetNumberOfPositiveAndNegativeEntries(dataList)

Arguments

dataList
an array or dataframe in which the second column is numeric
GetPathsInSifFormat

Value

a vector of two components, the first of which giving the number of +1 entries, the second the
two + number of -1's.

Examples

expData<read.table(system.file(package='CausalR', 'extdata', 'testData.txt'))
GetNumberOfPositiveAndNegativeEntries(expData)

GetRegulatedNodes

get regulated nodes

Description

This function will compute the nodes regulated by the given hypothesis gene and write the results
to a file.

Usage

GetRegulatedNodes(PPInet, Expressiondata, delta, hypothesisGene = "",
signOfHypothesis = 1, outputFile = "")
Arguments

PPInet a protein-protein interaction network
Expressiondata a table of observed gene expression data
delta the number of edges to follow along the network. This should typically be between 1 and 5 dependent on network size/topology
hypothesisGene the name of the hypothesis gene
signOfHypothesis the sign of action expected from the hypothesis, +1 for up regulation, -1 for down
outputfile the file to which the results should be written

Value

Nodes regulated by hypothesis gene

Description

Returns the possible values of $r_+$, $r_-$, $c_+$ and $c_-$ (the column and row sum values) following page 6 of Assessing statistical significance in causal graphs (Chindelevitch et. al)

Usage

getRowAndColumnSumValues(predictionListStats, experimentalResultStats)

Arguments

predictionListStats a vector containing the number of positive, negative, or non-significant/contradictory predictions ($q_+$, $q_-$ and $q_0$)
experimentalResultStats a vector containing the number of positive, negative, or non-significant/contradictory observations ($n_+$, $n_-$ and $n_0$)

Value

a matrix of row and column values $r_+$, $r_-$, $c_+$ and $c_-$

References

GetScoreForNumbersOfCorrectandIncorrectPredictions

returns the score for a given number of correct and incorrect predictions

Description

Returns the score based on the values of n++, n+-, n-+ and n–

Usage

GetScoreForNumbersOfCorrectandIncorrectPredictions(matrixRow)

Arguments

matrixRow a row of a matrix of correct and incorrect prediction scores

Value

the corresponding score for the given row

GetScoresForSingleNode

Get scores for single node

Description

A helper function for RankTheHypotheses to calculate a line of the scoresMatrix table

Usage

GetScoresForSingleNode(iNode, timeToRunSoFar, nodesToBeTested, network, delta, processedExperimentalData, numPredictions, epsilon, useCubicAlgorithm, use1bAlgorithm, symmetricCCG, correctPredictionsThreshold, experimentalDataStats, quiet)

Arguments

iNode this node
timeToRunSoFar the time to run so far
nodesToBeTested List of all nodes to be tested
network Computational Causal Graph, as an igraph.
delta Distance to search within the causal graph.
processedExperimentalData
   The processed experimental data

numPredictions
   The number of predictions

epsilon
   The threshold that is used when calculating the p-value using the cubic algorithm
   (see 'Assessing statistical significance in causal graphs').

useCubicAlgorithm
   An indicator specifying which algorithm will be used to calculate the p-value. The
default is set as useCubicAlgorithm = TRUE which uses the cubic algorithm. If this value is set as FALSE, the algorithm will use the much slower quartic algorithm which does compute the exact answer, as opposed to using approximations like the cubic algorithm.

use1bAlgorithm
   An indicator specifying whether the 1a or 1b (default, faster) variant of the cubic algorithm described in Chindelevitch’s paper will be used to calculate the p-value.

symmetricCCG
   This flag specifies whether the CCG is assumed to be symmetric. The value is set as TRUE as a default. If this is the case the running time of the algorithm is reduced since the negative node values can be calculated using symmetry and the results of calculations performed for the positive node.

correctPredictionsThreshold
   A threshold on the number of correct predictions for a given hypothesis. If a hypothesis produces fewer correct predictions than predictionsThreshold then the algorithm will not calculate the two p-values. Instead 'NA' will be displayed in the final two columns of the corresponding row of the results table. As a default correctPredictionsThreshold is set as -Inf, so that the p-values are calculated for all specified hypotheses. Note: Set to Inf to turn off p-value calculations entirely.

experimentalDataStats
   Stats from the experimental data

quiet
   a flag to suppress progress output

Value

If symmetricCCG is false, this returns a single line of the scoreMatrix for the 'iNode'th node in nodesToBeTested. If symmetricCCG is true this returns two lines. The first of which corresponds to the positive node and the second the negative node.

---

GetScoresWeightsMatrix

get scores weight matrix

---

Description

Computes the score and weight for a network/set of experimental data based on the table containing possible values of n++, n+, n-+ and n--.
GetScoresWeightsMatrixByCubicAlg

Usage

GetScoresWeightsMatrix(matrixOfPossibleValues, predictionDataStats, experimentalDataStats, logOfFactorialOfPredictionListStats)

Arguments

matrixOfPossibleValues
values of n++, n+-, n-+ and n– that need to be assessed
predictionDataStats
a table of predictions
experimentalDataStats
a table of observed experimental data
logOfFactorialOfPredictionListStats
a vector containing the log of the factorial value for each entry in predictionListStats

Value

a matrix containing scores and logs of the weights

Description

Implements the cubic algorithm as described on pages 6 and 7 of Assessing statistical significance in causal graphs, Chindelevitch et al. 2012

Usage

GetScoresWeightsMatrixByCubicAlg(predictionListStats, experimentalDataStats, epsilon)

Arguments

predictionListStats
a vector containing the values q+, q- and q0
experimentalDataStats
a vector containing the values n+, n- and n0
epsilon
the algorithms tolerance epsilon

Value

a matrix containing the ternary dot product distribution
**GetSetOfSignificantPredictions**

**Description**

Gets the set of positive and negative predictions, the combination of the sets Sh+ and Sh- in Causal reasoning on biological networks: Interpreting transcriptional changes, L Chindelevitch et al.

**Usage**

```r
GetSetOfSignificantPredictions(predictions)
```

**Arguments**

- `predictions` a table of predictions

**References**


---

**GetSetOfDifferentiallyExpressedGenes**

* get set of differentially expressed genes

**Description**

Gets the set of differentially expressed genes in the results, G+ as defined by in Causal reasoning on biological networks: Interpreting transcriptional changes, L Chindelevitch et al.

**Usage**

```r
GetSetOfDifferentiallyExpressedGenes(results)
```

**Arguments**

- `results` a table of results

**Value**

- a matrix of differentially expressed genes

**References**


---

**References**

GetShortestPathsFromCCG

Value

a matrix of positive and negative predictions

References


Description

Gets the node names in the shortest path from one node in a CCG to another

Usage

GetShortestPathsFromCCG(network, hypothesisnode, targetnode, showbothdirs = FALSE, quiet = FALSE)

Arguments

- `network`: built from iGraph
- `hypothesisnode`: hypothesis node ID
- `targetnode`: target node ID
- `showbothdirs`: where multiple paths from a positive and negative node, FALSE returns only the shortest. Otherwise both are returned.
- `quiet`: a flag to suppress output to console. FALSE by default.

Value

a list of vectors containing the nodes of individual paths

Examples

```r
network <- system.file(package='CausalR', extdata, 'testNetwork.sif'
ccg = CreateCCG(network)
hypothesisnode = 1
targetnode = 10
GetShortestPathsFromCCG (ccg, hypothesisnode, targetnode)
```
GetWeightForNumbersOfCorrectandIncorrectPredictions

get weight for numbers of correct and incorrect predictions

Description

Gets the weight based on the values of \( n^{++}, n^{-+}, n^{-+} \) and \( n^{--} \).

Usage

GetWeightForNumbersOfCorrectandIncorrectPredictions(n_{pp}, n_{pm}, n_{mp}, n_{mm},
predictionDataStats, experimentalDataStats,
logOfFactorialOfPredictionListStats, returnlog = FALSE)

Arguments

- \( n_{pp} \): the contingency table entry \( n^{++} \)
- \( n_{pm} \): the contingency table entry \( n^{-+} \)
- \( n_{mp} \): the contingency table entry \( n^{-+} \)
- \( n_{mm} \): the contingency table entry \( n^{--} \)
- predictionDataStats: prediction data statistics
- experimentalDataStats: experimental data statistics
- logOfFactorialOfPredictionListStats: log of factorial of prediction list stats
- returnlog: true if the result should be returned as a log

Value

none

GetWeightsAboveHypothesisScoreAndTotalWeights

get weights above hypothesis score and total weights

Description

Gets the score based on the values of \( n^{++}, n^{+-}, n^{-+} \) and \( n^{--} \). Used as part of a p-value calculation.

Usage

GetWeightsAboveHypothesisScoreAndTotalWeights(r_{p}, r_{m}, c_{p},
predictionListStats, experimentalDataStats,
logOfFactorialOfPredictionListStats, hypothesisScore, logepsDMax, logDMax)
**GetWeightsAboveHypothesisScoreForAThreeByTwoTable**

**Arguments**

- `r_p` the row sum r+
- `r_m` the row sum r-
- `c_p` the column sum c+
- `predictionListStats` statistics for the prediction list
- `experimentalDataStats` statistics for the experimental data
- `logOfFactorialOfPredictionListStats` log of factorial of prediction list stats
- `hypothesisScore` the hypothesis score to be considered
- `logepsDMax` Exponential of logD Maximum value
- `logDMax` A logD Maximum value

**Value**

score data

---

**Description**

Finds the D-Values (weights) from any 3x2 contingency tables that have a score above and including the hypothesis score. It also calculates the total weight, and returns a 2x1 vector of the two values. The ratio of these values is the p-value.

**Usage**

```
GetWeightsAboveHypothesisScoreForAThreeByTwoTable(weights, r_p, r_m, r_z, n_p, n_m, predictionListStats, experimentalDataStats, logOfFactorialOfPredictionListStats, hypothesisScore, logepsDMax, logDMax)
```

**Arguments**

- `weights` Weights
- `r_p` the row sum r+
- `r_m` the row sum r-
- `r_z` the row sum r0
- `n_p` the column sum n+
GetWeightsFromInteractionInformation

Returns a matrix of weights (-1,0,+1) indicating the direction of regulation from the interaction information.

Usage

GetWeightsFromInteractionInformation(interactionInfo)

Arguments

interactionInfo
a central column of the .sif file, giving the type of edge interaction

Value

a matrix of weights corresponding the the direction of regulation
**MakePredictions**

**Description**

Creates a matrix of predictions for a particular hypothesis. The output is an array containing the relationship between each node and the hypothesis. The hypothesis provided will be the vertex id of one of the nodes in the network (as an integer node ID or name, including + or - for up/down regulation in the case of a CCG). The signOfHypothesis variable should be a 1 or -1, indicating up/down regulation.

**Usage**

```r
MakePredictions(hypothesisnode, signOfHypothesis, network, delta, 
    nodesInExperimentalData = NULL)
```

**Arguments**

- `hypothesisnode` the node in the causal graph from which predictions should be made. Can be either a (numerical) node ID or a (string) node name.
- `signOfHypothesis` whether the hypothesis node is up- or down-regulated. Should be +1 or -1.
- `network` a (Computational) Causal Graph, as an igraph.
- `delta` the distance to search within the causal graph.
- `nodesInExperimentalData` optional. Nodes to include in the output. Should be a list of node IDs.

**Value**

a matrix of predictions for the given particular hypothesis

**Examples**

```r
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
ccg <- CreateCCG(network)
predictions <- MakePredictions('NodeA', +1, ccg, 2)
```
MakePredictionsFromCCG

make predictions from CCG

**Description**

Create a matrix of predictions for a particular hypothesis starting from a network with separate nodes for up- and down-regulation (+ve and -ve). The output is an array containing the relationship between each node and the hypothesis. The hypothesis provided will be the vertex id of one of the nodes in the network (as an integer or name including + or - for up/down regulation). The signOfHypothesis variable should be a 1 or -1, indicating up/down regulation. (It generally shouldn’t be necessary to reverse the sign of a node when working from a CCG, but this facility is included for consistency with MakePredictionsFromCG)

**Usage**

MakePredictionsFromCCG(hypothesisnode, signOfHypothesis, network, delta, nodesInExperimentalData = NULL)

**Arguments**

- **hypothesisnode** a hypothesis node
- **signOfHypothesis** the direction of change of hypothesis node
- **network** a computational causal graph
- **delta** the number of edges across which the hypothesis should be followed
- **nodesInExperimentalData** the number of nodes in experimental data

**Value**

an matrix containing the relationship between each node and the hypothesis

**Examples**

```r
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
ccg <- CreateCCG(network)
MakePredictionsFromCCG('NodeA', +1, ccg, 2)
```
MakePredictionsFromCG  make predictions from CG

Description
Create a matrix of predictions for a particular hypothesis - the output is a matrix containing the relationship between each node and the hypothesis. The hypothesis provided will be the vertex id of one of the nodes in the network (as an integer). The signOfHypothesis variable should be a 1 or -1, indicating up/down regulation.

Usage
MakePredictionsFromCG(hypothesisnode, signOfHypothesis, network, delta, nodesInExperimentalData = NULL)

Arguments
hypothesisnode  a hypothesis node
signOfHypothesis  the direction of change of hypothesis node
network  a computational causal graph
delta  the number of edges across which the hypothesis should be followed
nodesInExperimentalData  the number of nodes in experimental data

Value
an matrix containing the relationship between each node and the hypothesis

Examples
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
cg <- CreateCG(network)
MakePredictionsFromCG('NodeA', +1, cg, 2)

OrderHypotheses  order hypotheses

Description
Ranks the hypotheses. Takes a matrix containing the scores for each node of the network, and ranks them placing the hypothesis with the most correct predictions is at the top.

Usage
OrderHypotheses(scoresMatrix)
Arguments

scoresMatrix  a matrix containing the scores for each node of the network

Value

a ranked table of hypotheses

Description

Plots an igraph with the node names. Plots a igraph to the screen displaying the names of the nodes input rather than R’s internal numbering.

Usage

PlotGraphWithNodeNames(igraph)

Arguments

igraph  internal an igraph representation of an interaction network

Value

network visualisation

Examples

network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
ccg <- CreateCCG(network)
PlotGraphWithNodeNames(ccg)

PopulateTheThreeByThreeContingencyTable

populate the three-by-three contingency table

Description

Populates 3x3 signed contingency table of expected versus observed changes. Given the values of n++, n+, n- and n–, calculates n0+, n0-, n+0, n-0 and n00. Notation from Chindelevitch et al. Causal reasoning on biological networks Bioinformatics (2012) paper.
Usage

`PopulateTheThreeByThreeContingencyTable(n_pp, n_pm, n_mp, n_mm, predictionDataStats, experimentalDataStats)`

Arguments

- `n_pp`: n++ contingency table entry
- `n_pm`: n+- contingency table entry
- `n_mp`: n-+ contingency table entry
- `n_mm`: n– contingency table entry
- `predictionDataStats`: a prediction data table.
- `experimentalDataStats`: an experimental data table

Value

Vector of calculated values for n0+, n0-, n+0, n-0 and n00 - See: Chindelevitch et al. Bioinformatics (2012).

Description

Calculates a 2x2 contingency table. Given the value of n++ and the row and column sums (r+, r-, c+, c-), Calculates the remaining values in the 2x2 contingency table i.e. n+, n-+, and n–. See Chindelevich et al. BMC Bioinformatics (2012) paper ‘Assessing Statistical significance of causal graphs’ for clarification on notation.

Usage

`PopulateTwoByTwoContingencyTable(rowAndColumnSumValues, n_pp)`

Arguments

- `rowAndColumnSumValues`: the row and column sums (r+, r-, c+, c-).
- `n_pp`: the value of n++.

Value

the completed 2x2 contingency table: n++, n+-, n-+, n–
References


ProcessExperimentalData

\textit{process experimental data}

Description

Processes experimental data to get it into the correct form for scoring. The node names that are read in as strings acquire an internal id when the network is created. This function will replace the node name with its id.

Usage

\texttt{ProcessExperimentalData(experimentalData, network)}

Arguments

\begin{itemize}
  \item \texttt{experimentalData} \hspace{1cm} \textit{input experimental data.}
  \item \texttt{network} \hspace{1cm} \textit{an input interaction network.}
\end{itemize}

Value

processed experimental data formatted ready for scoring

RankTheHypotheses

\textit{rank the hypotheses}

Description

Rank the hypotheses in the causal network. This function can be run with parallelisation using the doParallel flag.

Usage

\texttt{RankTheHypotheses(network, experimentalData, delta, epsilon = 1e-05, useCubicAlgorithm = TRUE, use1bAlgorithm = TRUE, symmetricCCG = TRUE, listOfNodes = NULL, correctPredictionsThreshold = -Inf, quiet = FALSE, doParallel = FALSE, numCores = NULL, writeFile = TRUE, outputDir = getwd())}
RankTheHypotheses

Arguments

- **network**: Computational Causal Graph, as an igraph.
- **experimentalData**: The experimental data read in using `ReadExperimentalData`. The results is an n x 2 matrix; where the first column contains the node ids of the nodes in the network that the results refer to. The second column contains values indicating the direction of regulation in the results - (+)1 for up, -1 for down and 0 for insignificant amounts of regulation. The name of the first column is the filename the data was read from.
- **delta**: Distance to search within the causal graph.
- **epsilon**: The threshold that is used when calculating the p-value using the cubic algorithm (see 'Assessing statistical significance in causal graphs').
- **useCubicAlgorithm**: An indicator specifying which algorithm will be used to calculate the p-value. The default is set as `useCubicAlgorithm = TRUE` which uses the cubic algorithm. If this value is set as FALSE, the algorithm will use the much slower quartic algorithm which does compute the exact answer, as opposed to using approximations like the cubic algorithm.
- **use1bAlgorithm**: An indicator specifying whether the 1a or 1b (default, faster) variant of the cubic algorithm described in Chindelevitch’s paper will be used to calculate the p-value.
- **symmetricCCG**: This flag specifies whether the CCG is assumed to be symmetric. The value is set as TRUE as a default. If this is the case the running time of the algorithm is reduced since the bottom half of the table can be filled in using the results of calculations performed earlier.
- **listOfNodes**: A list of nodes specified by the user. The algorithm will only calculate and store the results for the nodes in the specified list. The default value is NULL; here the algorithm will calculate and store results for all the nodes in the network.
- **correctPredictionsThreshold**: A threshold on the number of correct predictions for a given hypothesis. If a hypothesis produces fewer correct predictions than predictionsThreshold then the algorithm will not calculate the two p-values. Instead 'NA' will be displayed in the final two columns of the corresponding row of the results table. As a default `correctPredictionsThreshold` is set as -Inf, so that the p-values are calculated for all specified hypotheses.
- **quiet**: a flag to suppress output to console. FALSE by default.
- **doParallel**: A flag for running `RankTheHypothesis` in parallel mode.
- **numCores**: Number of cores to use if using parallel mode. If the default value of NULL is used, it will attempt to detect the number of cores available and use all of them bar one.
- **writeFile**: A flag for determining if the output should be written to a file in the working directory. Default is TRUE.
- **outputDir**: the directory to output the files to. Default is the working directory.
A data frame containing the results of the algorithm.

References


Examples

# get path to example network file
networkFile <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')
# create ccg
network <- CreateCCG(networkFile)
# get path to example experimental data
experimentalDataFile <- system.file(package='CausalR', 'extdata', 'testData.txt')
# read in experimental data
experimentalData <- ReadExperimentalData(experimentalDataFile, network)
# run in single threaded mode
RankTheHypotheses(network, experimentalData, 2)
# run in parallel mode
RankTheHypotheses(network, experimentalData, 2, doParallel=TRUE, numCores=2)

Description

Reads experimental data for the causal reasoning algorithm from a text file.

Usage

ReadExperimentalData(fileName, network, removeDuplicates)

Arguments

fileName   a file containing the experimental data (text file format)
network    a (Computational) Causal Graph, as an igraph.
removeDuplicates
            Optional, defaults to true. Remove duplicated nodes the experimental file (i.e. where the result for a node is repeated, use the first value given only; the alternative is to return a result which contains multiple rows for this node).

Value

(n x 2) matrix of nodes and direction of regulation. The first column of the matrix contains the node IDs from the network, and the second contains the experimental values.
Examples

```r
# get path to example network file
network <- system.file(package='CausalR', 'extdata', 'testNetwork.sif')

# create ccg
ccg <- CreateCCG(network)

# get path to example experimental data
fileName <- system.file(package='CausalR', 'extdata', 'testData.txt')
ReadExperimentalData(fileName, ccg)
```

---

**ReadSifFileToTable**

*read .sif to Table*

**Description**

Reads a .sif file into a table in R.

**Usage**

```r
ReadSifFileToTable(sifFile)
```

**Arguments**

- `sifFile` the .sif file to be read in

**Value**

a R table containing the data from the .sif file.

---

**RemoveIDsNotInExperimentalData**

*remove IDs not in experimental data*

**Description**

Takes in a list of connected nodes and removes those not in the experimental data.

**Usage**

```r
RemoveIDsNotInExperimentalData(connectedNodes, nodesInExperimentalData)
```

**Arguments**

- `connectedNodes` a list of connected nodes
- `nodesInExperimentalData` a list of nodes in the experimental data

**Value**

`connectedNodesInExperimentalData` a list of connected nodes with the redundant nodes removed
runRankHypothesis

run rank the hypothesis

Description
A top level function that used to run CausalR

Usage
runRankHypothesis(PPInet, Expressiondata, delta, correctPredictionsThreshold)

Arguments
PPInet  PPInet is the PPI interaction file
Expressiondata observed gene expression data
delta  the number of links to follow from any hypothesis. Depending on network size/topology, this value typically ranges between 1 and 5
correctPredictionsThreshold Minimal score for p-values calculation. Hypotheses with scores below this value will get NAs for p-value and enrichment p-value. The usual default is -inf within the RankTheHypotheses function, where it is employed.

Value
rankedHypothesis table of results produced by the algorithm

runSCANR

run ScanR

Description
This function will return nodes regulated by the given hypothesisGene

Usage
runSCANR(network, experimentalData, numberOfDeltaToScan = 5, topNumGenes = 150, correctPredictionsThreshold = Inf, writeResultFiles = TRUE, writeNetworkFiles = "all", doParallel = FALSE, numCores = NULL, quiet = FALSE, outputDir = getwd())
Arguments

**network**
- Computational Causal Graph, as an igraph.

**experimentalData**
- The experimental data read in using `ReadExperimentalData`. The results is an n x 2 matrix; where the first column contains the node ids of the nodes in the network that the results refer to. The second column contains values indicating the direction of regulation in the results - (+)1 for up, -1 for down and 0 for insignificant amounts of regulation.

**numberOfDeltaToScan**
- Iteratively scan for 1 to `numberOfDeltaToScan` delta values

**topNumGenes**
- A value to select top genes to report (typically top 100 genes)

**correctPredictionsThreshold**
- Minimal score for p-values calculation. Value is passed to `RankTheHypothesis` - scores below this value will get NAs for p-value and enrichment p-value. The default is Inf, so that no p-values are calculated.

**writeResultFiles**
- If set to TRUE the results of the scan will be written to two text files in the working directory. Default is TRUE.

**writeNetworkFiles**
- If set to "all" .sif files and corresponding _anno.txt files will be generated for the top correctly explained, incorrectly explained and ambiguously explained nodes. If set to "correct" they will only be calculated for correctly explained nodes. If set to "none", no networks will be generated. Default is "all".

**doParallel**
- A flag for running `RankTheHypothesis` in parallel mode. Default is FALSE.

**numCores**
- Number of cores to use if using parallel mode. If the default value of NULL is used, it will attempt to detect the number of cores available and use all of them bar one.

**quiet**
- A flag to suppress output to console. FALSE by default.

**outputDir**
- the directory to output the files to. Default is the working directory

Value

returns list of genes from each delta scan run

Examples

```r
numberOfDeltaToScan <- 2
topNumGenes <- 4
#get path to example network file
networkFile <- system.file(package = 'CausalR', 'extdata', 'testNetwork.sif')
#create ccg
network <- CreateCCG(networkFile)
#get path to example experimental data
experimentalData <- system.file(package = 'CausalR', 'extdata', 'testData.txt')
#read in experimetal data
experimentalData <- ReadExperimentalData(experimentalDataFile, network)
#run in single threaded mode
```
runSCANR(network, experimentalData, numberOfDeltaToScan, topNumGenes)
#run in parallel mode
runSCANR(network, experimentalData, numberOfDeltaToScan, topNumGenes,
         doParallel = TRUE, numCores = 2)

ScoreHypothesis  score hypothesis

Description
Score a single hypothesis, using the predictions from the network and the experimental data returning a vector of scoring statistics.

Usage
ScoreHypothesis(matrixOfPredictions, matrixOfExperimentalData)

Arguments
matrixOfPredictions
  a matrix of predictions
matrixOfExperimentalData
  a matrix of experimental data

Value
scoreBreakdown a vector giving, in order, the overall score, and the numbers of correct, incorrect and ambiguous predictions

Examples
predictions <- matrix(c(1,2,3,+1,0,-1),ncol=2)
experimentalData <- matrix(c(1,2,4,+1,+1,-1),ncol=2)
ScoreHypothesis(predictions,experimentalData)
CompareHypothesis(predictions,experimentalData)

ValidateFormatOfDataTable
validate format of the experimental data table

Description
Checks the format of the experimental data. This is expected to be two columns, the first containing the gene name and the second the direction of regulation, -1, 0 or 1. The function checks the number of columns and the values of the second column.
ValidateFormatOf dataTable

Usage

ValidateFormatOf dataTable (dataTable)

Arguments

dataTable the data table to be tested

Value

true if the data table is valid

WriteAllExplainedNodesToSifFile

Write all explained nodes to .sif file

Description

Outputs networks of all explained nodes in .sif file format, named by node name with sign of regulation, each with a corresponding annotation file for producing visualisations using Cytoscape.

Usage

WriteAllExplainedNodesToSifFile(scanResults, network, experimentalData, delta, correctlyExplainedOnly = TRUE, quiet = TRUE)
WriteExplainedNodesToSifFile

Arguments

- scanResults: a results object produced by ScanR
- network: a computational causal graph
- experimentalData: The experimental data read in using `ReadExperimentalData`.
- delta: the number of edges across which the hypothesis should be followed, the setting should be that used to generate the input ScanR object.
- correctlyExplainedOnly: if TRUE network files will only be produced for correctly explained nodes. If FALSE network files will be produced for each of correctly explained, incorrectly explained and ambiguously explained nodes.
- quiet: a flag to suppress output to console. FALSE by default.

Value

files containing paths from hypothesis node to explained nodes in sif format and corresponding annotation (_anno.txt) files

Examples

```r
networkFile <- system.file(package='CausalR', 'extdata', 'testNetwork1.sif')
network <- CreateCCG(networkFile)
experimentalDataFile <- system.file(package='CausalR', 'extdata', 'testData1.txt')
experimentalData <- ReadExperimentalData(experimentalDataFile, network)
delta <- 2
scanResults <- runSCANR(network, experimentalData, numberOfDeltaToScan = delta, topNumGenes = 2, writeResultFiles = FALSE, writeNetworkFiles = "none", quiet = FALSE, doParallel = TRUE, numCores = 2)
WriteAllExplainedNodesToSifFile(scanResults, network, experimentalData, delta, correctlyExplainedOnly = TRUE, quiet = TRUE)
```

WriteExplainedNodesToSifFile

Write explained nodes to Sif file

Description

Outputs networks of explained nodes in .sif file format for producing visualisations using Cytoscape. Output will be to a directory beginning with a timestamp.

Usage

```r
WriteExplainedNodesToSifFile(hypothesisnode, signOfHypothesis, network, experimentalData, delta, outputDir = getwd(), outputFileNames = "", correctlyExplainedOnly = FALSE, quiet = FALSE)
```
WriteExplainedNodesToSifFile

Arguments

- **hypothesisNode**  a hypothesis node
- **signOfHypothesis** the direction of change of hypothesis node
- **network** a computational causal graph
- **experimentalData** The experimental data read in using ReadExperimentalData. The results is an n x 2 matrix; where the first column contains the node ids of the nodes in the network that the results refer to. The second column contains values indicating the direction of regulation in the results - (+)1 for up, -1 for down and 0 for insignificant amounts of regulation. The name of the first column is the filename the data was read from.
- **delta** the number of edges across which the hypothesis should be followed
- **outputDir** the directory to output the files to. Default is the working directory
- **outputFilesName** a character string to use for the name of the output files. Default value is "", which results in files using the default naming convention of "network file name-data file name-delta value-node name". Set to NA if not writing to file.
- **correctlyExplainedOnly** if TRUE network files will only be produced for correctly explained nodes. If FALSE network files will be produced for each of correctly explained, incorrectly explained and ambiguously explained nodes.
- **quiet** a flag to suppress output to console. FALSE by default.

Value

files containing paths from hypothesis node to explained nodes in sif format and corresponding annotation (_anno.txt) files

Examples

```r
hypothesisNode <- "Node0"
signOfHypothesis <- +1
networkFile <- system.file(package='CausalR', 'extdata', 'testNetwork1.sif')
network <- CreateCCG(networkFile)
experimentalDataFile <- system.file(package='CausalR', 'extdata', 'testData1.txt')
experimentalData <- ReadExperimentalData(experimentalDataFile, network)
delta <- 2
WriteExplainedNodesToSifFile(hypothesisNode, signOfHypothesis, network, experimentalData, delta, outputFilesName=NA)
```
Index

* CausalR
  AnalysePredictionsList, 5
  CalculateEnrichmentPValue, 6
  CalculateSignificance, 7
  CalculateSignificanceUsingCubicAlgorithm, 8
  CalculateSignificanceUsingCubicAlgorithm1b, 9
  CalculateSignificanceUsingQuarticAlgorithm, 10
  CausalR-package, 3
  CompareHypothesis, 13
  CreateCCG, 15
  CreateCG, 16
  GetNodeName, 28
  GetNumberOfPositiveAndNegativeEntries, 28
  GetShortestPathsFromCCG, 35
  MakePredictions, 39
  MakePredictionsFromCCG, 40
  MakePredictionsFromCG, 41
  PlotGraphWithNodeNames, 42
  RankTheHypotheses, 44
  ReadExperimentalData, 46
  runSCANR, 48
  ScoreHypothesis, 50
  WriteAllExplainedNodesToSifFile, 51
  WriteExplainedNodesToSifFile, 52

* AddIDsToVertices, 4
  AddWeightsToEdges, 4
  AnalyseExperimentalData, 5
  AnalysePredictionsList, 5

* CalculateEnrichmentPValue, 6
  CalculateSignificance, 7
  CalculateSignificanceUsingCubicAlgorithm, 8
  CalculateSignificanceUsingCubicAlgorithm1b, 9
  CalculateSignificanceUsingQuarticAlgorithm, 10
  CalculateTotalWeightForAllContingencyTables, 11
  CalculateWeightGivenValuesInThreeByThreeContingencyTable, 11
  CausalR (CausalR-package), 3
  CheckPossibleValuesAreValid, 12
  CausalR-package, 3
  CompareHypothesis, 13
  CreateCCG, 15
  CreateCG, 16
  GetNodeName, 28
  GetNumberOfPositiveAndNegativeEntries, 28
  GetShortestPathsFromCCG, 35
  MakePredictions, 39
  MakePredictionsFromCCG, 40
  MakePredictionsFromCG, 41
  PlotGraphWithNodeNames, 42
  RankTheHypotheses, 44
  ReadExperimentalData, 46
  runSCANR, 48
  ScoreHypothesis, 50
  WriteAllExplainedNodesToSifFile, 51
  WriteExplainedNodesToSifFile, 52
INDEX

CheckRowAndColumnSumValuesAreValid, 12
CompareHypothesis, 13
ComputeFinalDistribution, 14
ComputePValueFromDistributionTable, 14
CreateCCG, 3, 15
CreateCG, 16
CreateNetworkFromTable, 16
DetermineInteractionTypeOfPath, 17
FindApproximateValuesThatWillMaximiseDValue, 17
FindIdsOfConnectedNodesInSubgraph, 18
FindMaximumDValue, 19
GetAllPossibleRoundingCombinations, 19
GetApproximateMaximumDValueFromThreeByTwoContingencyTable, 20
GetApproximateMaximumDValueFromTwoByTwoContingencyTable, 21
GetCombinationsOfCorrectandIncorrectPredictions, 22
GetExplainedNodesOfCCG, 22
GetInteractionInformation, 23
GetMatrixOfCausalRelationships, 24
GetMaxDValueForAFamily, 24
GetMaxDValueForAThreeByTwoFamily, 25
GetMaximumDValueFromTwoByTwoContingencyTable, 26
GetNodeID, 27
GetNodeName, 28
GetNumberOfPositiveAndNegativeEntries, 28
GetPathsInSifFormat, 29
GetRegulatedNodes, 29
GetRowAndColumnSumValues, 30
GetScoreForNumbersOfCorrectandIncorrectPredictions, 31
GetScoresForSingleNode, 31
GetScoresWeightsMatrix, 32
GetScoresWeightsMatrixByCubicAlg, 33
GetSetOfDifferentiallyExpressedGenes, 34
GetSetOfSignificantPredictions, 34
GetShortestPathsFromCCG, 35
GetWeightForNumbersOfCorrectandIncorrectPredictions, 36
GetWeightsAboveHypothesisScoreForAThreeByTwoTable, 37
GetWeightsFromInteractionInformation, 38
MakePredictions, 3, 39
MakePredictionsFromCCG, 40
MakePredictionsFromCG, 41
OrderHypotheses, 41
PlotGraphWithNodeNames, 42
PopulateTheThreeByThreeContingencyTable, 42
PopulateTwoByTwoContingencyTable, 43
ProcessExperimentalData, 44
RankTheHypotheses, 3, 44
ReadExperimentalData, 3, 23, 45, 46, 49, 52
ReadSifFileToTable, 47
RemoveIDsNotInExperimentalData, 47
runRankHypothesis, 48
runSCANR, 3, 48
ScoreHypothesis, 3, 50
ValidateFormatOfDataTable, 50
ValidateFormatOfTable, 51
WriteAllExplainedNodesToSifFile, 51
WriteExplainedNodesToSifFile, 3, 52