Package ‘QuaternaryProd’
November 22, 2016

Type Package
Title Computes the Quaternary Dot Product Scoring Statistic for Signed and Unsigned Causal Graphs
Version 1.2.0
Date 2015-10-22
Description QuaternaryProd is an R package that performs causal reasoning on biological networks, including publicly available networks such as String-db. QuaternaryProd is a free alternative to commercial products such as Quiagen and Inginuity pathway analysis. For a given a set of differentially expressed genes, QuaternaryProd computes the significance of upstream regulators in the network by performing causal reasoning using the Quaternary Dot Product Scoring Statistic (Quaternary Statistic), Ternary Dot product Scoring Statistic (Ternary Statistic) and Fisher's exact test. The Quaternary Statistic handles signed, unsigned and ambiguous edges in the network. Ambiguity arises when the direction of causality is unknown, or when the source node (e.g., a protein) has edges with conflicting signs for the same target gene. On the other hand, the Ternary Statistic provides causal reasoning using the signed and unambiguous edges only. The Vignette provides more details on the Quaternary Statistic and illustrates an example of how to perform causal reasoning using String-db.

License GPL (>=3)
biocViews GraphAndNetwork, GeneExpression, Transcription
Depends R (>= 3.2.0), Rcpp (>= 0.11.3)
Suggests readr, org.Hs.eg.db, dplyr, stringr, knitr, fdrtool
LinkingTo Rcpp
LazyData true
VignetteBuilder knitr
RoxygenNote 5.0.1
NeedsCompilation yes
Author Carl Tony Fakhry [cre, aut], Ping Chen [ths], Kourosh Zarringhalam [aut, ths]
Maintainer Carl Tony Fakhry <cfakhry@cs.umb.edu>

R topics documented:
QuaternaryProd-package
QuaternaryProd-package

Computes the Quaternary Dot Product Scoring Statistic for Signed and Unsigned Causal Graphs

Description

QuaternaryProd is an R package that performs causal reasoning on biological networks, including publicly available networks such as String-db. QuaternaryProd is a free alternative to commercial products such as Quiagen and Inginuity pathway analysis. For a given a set of differentially expressed genes, QuaternaryProd computes the significance of upstream regulators in the network by performing causal reasoning using the Quaternary Dot Product Scoring Statistic (Quaternary Statistic), Ternary Dot product Scoring Statistic (Ternary Statistic) and Fisher’s exact test. The Quaternary Statistic handles signed, unsigned and ambiguous edges in the network. Ambiguity arises when the direction of causality is unknown, or when the source node (e.g., a protein) has edges with conflicting signs for the same target gene. On the other hand, the Ternary Statistic provides causal reasoning using the signed and unambiguous edges only. The Vignette provides more details on the Quaternary Statistic and illustrates an example of how to perform causal reasoning using String-db.

Details

Package: QuaternaryProd
Type: Package
Version: 1.0.6
Date: 2015-10-22
License: GPL (>= 2)

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam
Maintainer: Carl Tony Fakhry <cfakhry@cs.umb.edu>

References

BioQCREtoNet


BioQCREtoNet

Compute the Quaternary Dot Product Scoring Statistic for a biological causal network.

Description

This function computes the Quaternary Dot Product Scoring Statistic for all source nodes in a causal network once new gene expression data is presented. This is a Causal Relation Engine (CRE) on the levels of a causal network of predictions and a new set of realizations that arise in real world situations.

Usage

BioQCREtoNet(relations, evidence, entities, method = "Quaternary", fc.thresh = log2(1.3), is.Logfc = TRUE, pval.thresh = 0.05)

Arguments

relations A data frame containing pairs of connected entities in a causal network (e.g Protein-Protein interactions), and the type of causal relation between them. The data frame must have three columns with column names: srcuid, trguid and mode respective of order. srcuid stands for source entity, trguid stands for target entity and mode stands for the type of relation between srcuid and trguid. The relation has to be one of increases, decreases or regulates. All three columns must be of type character.

evidence A data frame of entities which are target nodes in the causal network and have new experimental values (e.g gene expression data). The evidence data frame must have three columns entrez, fc and pvalue. entrez denotes the entrez id of a given gene, fc denotes the fold change of a gene, and pvalue denotes the p-value. The entrez column must be of type character, and the fc and pvalue columns must be numeric values.

entities A data frame of mappings for all entities present in data frame relations. entities must contain four columns: uid, id, symbol and type respective of order. All four columns must be of type character. uid includes every source and target node in the network (i.e relations), id is the id of uid (e.g entrez id of an mRNA), symbol is the symbol of id and type is the type of entity of id which can be one of mRNA, protein, drug or compound. All target nodes must be of type mRNA.

method Choose one of Quaternary, Ternary or Enrichment. Default is Quaternary.

fc.thresh Threshold for fold change in evidence data frame. Any row in evidence with absolute value fc smaller than fc.thresh will be ignored. Default value is log2(1.3). If is.Logfc = FALSE then fc column in evidence is converted to log scale.

is.Logfc Boolean value to indicate if the fold change is in log scale. Default value is TRUE.

pval.thresh Threshold for p-values in evidence data frame. All rows in evidence with p-values greater than pval.thresh will be ignored. Default value is 0.05.
Value

This function returns a data frame containing parameters concerning the Quaternary Dot Product Scoring Statistic. The p-values of each of the source nodes is also computed, and the data frame is in increasing order of p-values of the goodness of fit score for the given source nodes. The column names of the data frame are:

- `uid` The source node in the network.
- `name` symbols of the source nodes.
- `regulation` Direction of change of source node.
- `correct.pred` Number of correct predictions in `evidence` when compared to predictions made by the network.
- `incorrect.pred` Number of incorrect predictions in `evidence` when compared to predictions made by the network.
- `score` The number of correct predictions minus the number of incorrect predictions.
- `total.reachable` Total number of `trguids` connected to a `srcuid`.
- `significant.reachable` number of `trguids` connected to a `srcuid` that are also regulated in `evidence`.
- `total.ambiguous` Total number of children of a given `srcuid` with relation type `regulates` or children which share both `increase` and `decrease` relation with `srcuid`.
- `significant.ambiguous` Total number of similar type of relations as with `total.ambiguous` but with the restriction that the children are regulated in `evidence`.
- `unknown` Number of `trguids` which do not interact with the given `srcuid`.
- `pvalue` P-value of the score.

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam

References


Examples

```r
# We provide an example of one possible way to parse the Stringdb
# Homo Sapien protein actions network and prepare
# it to be used with our package. First, we need to
# upload the network which is attached to QuaternaryProd
# for convenience.
library(readr)
library(org.Hs.eg.db)
library(dplyr)
library(stringr)
```
library(fdrtool)

# Get the full file name containing the STRINGdb relations
ff <- system.file("extdata", "9606.protein.actions.v10.txt.gz", package="QuaternaryProd")
all_rels <- read_tsv(gzfile(ff), col_names = TRUE)

# Next, we filter out the important columns and important relations. We remove all rows which do not have a relation activation, inhibition and expression. Moreover, we also consider reverse causality for any relation which has a direction value equal to 0.

# Set new names for columns
names(all_rels) <- c("srcuid", "trguid", "mode", "action", "direction","score")
Rels <- all_rels[, c("srcuid", "trguid", "mode", "direction")]

# Get all rows with causal relations
Rels <- Rels[Rels$mode %in% c("activation", "inhibition","expression"),]

# Get causal relations where direction is not specified, and consider reversed direction of causality as a valid causal relation
Bidirectional <- Rels[Rels$direction == 0 ,
  c("trguid", "srcuid", "mode", "direction")]
names(Bidirectional) <- c("srcuid", "trguid", "mode", "direction")
Rels <- unique(bind_rows(Rels, Bidirectional))
Rels$direction <- NULL

# Rename activation as increases, inhibition as decreases, expression as regulates
Rels$mode <- sub("activation", "increases", Rels$mode)
Rels$mode <- sub("inhibition", "decreases", Rels$mode)
Rels$mode <- sub("expression", "regulates", Rels$mode)
Rels <- unique(Rels)

# Get a subset of the network: Skip this step if you want the p-values of the scores corresponding to the source nodes computed over the entire network.
Rels <- Rels[sample(1:nrow(Rels), 40000, replace=FALSE),]

# Third, we extract the protein entities from the network, and we map them to their respective genes. Note, the entities could have been possibly a drug or compound, but we are working with this protein interactions network for the purpose of providing a nontrivial example.

# Get all unique protein ensemble ids in the causal network
allEns <- unique(c(Rels$srcuid, Rels$trguid))

# Map ensemble protein ids to entrez gene ids
map <- org.Hs.egENSEMBLPROT2EG
id <- unlist(mget(sub("9606.","", allEns), map, ifnotfound=NA))
id[is.na(id)] <- "-1"
uid <- paste("9606." , names(id), sep="")
# Function to map entrez ids to gene symbols
map <- org.Hs.egSYMBOL
symbol <- unlist(mget(id, map, ifnotfound=NA))
symbol[is.na(symbol)] <- -1

# Create data frame of STRINGdb protein Id, entrez id and gene symbol and type of entity
Ents <- data_frame(uid, id, symbol, type="protein")
Ents <- Ents[Ents$uid %in% allEns,]

# Remove ensemble ids in entities with duplicated entrez id
Ents <- Ents[!duplicated(Ents$id),]

# Add mRNAs to entities
uid <- paste("mRNA_", Ents$uid, sep = "")
mRNAs <- data_frame(uid=uid, id=Ents$id, symbol=Ents$symbol, type="mRNA")
Ents <- bind_rows(Ents, mRNAs)

# Get all unique relations
Rels$trguid <- paste("mRNA_", Rels$trguid, sep="")
Rels <- Rels[Rels$srcuid %in% Ents$uid & Rels$trguid %in% Ents$uid,]
Rels <- unique(Rels)

# Leave source proteins which contain at least 10 edges
sufficientRels <- group_by(Rels, srcuid) %>% summarise(count=n())
sufficientRels <- sufficientRels %>% filter(count > 10)
Rels <- Rels %>% filter(srcuid %in% sufficientRels$srcuid)

# Given new gene expression data, we can compute the scores and p-values
# for all source nodes in the network. BioQCREtoNet is a specialized function for this purpose.

# Gene expression data
evidence1 <- system.file("extdata", "e2f3_sig.txt", package = "QuaternaryProd")
evidence1 <- read.table(evidence1, sep = "\t", header = TRUE
, stringsAsFactors = FALSE)

# Remove duplicated entrez ids in evidence and rename column names appropriately
evidence1 <- evidence1[!duplicated(evidence1$entrez),]
names(evidence1) <- c("entrez", "pvalue", "fc")

# Run Quaternary CRE for entire Knowledge base on new evidence
# which computes the statistic for each of the source proteins
CRE_results <- BioQCREtoNet(Rels, evidence1, Ents, is.Logfc = TRUE)

QP_Pmf

Computes the probability mass function of the scores.

Description

This function computes the probability mass function for the Quaternary Dot Product Scoring Statistic for signed causal graphs. This includes scores with probabilities strictly greater than zero.
**Usage**

```
QP_Pmf(q_p, q_m, q_z, q_r, n_p, n_m, n_z, epsilon = 1e-16)
```

**Arguments**

- `q_p`: Expected number of positive predictions.
- `q_m`: Expected number of negative predictions.
- `q_z`: Expected number of nil predictions.
- `q_r`: Expected number of regulated predictions.
- `n_p`: Number of positive predictions from experiments.
- `n_m`: Number of negative predictions from experiments.
- `n_z`: Number of nil predictions from experiments.
- `epsilon`: parameter for thresholding probabilities of matrices. Default value is 1e-16.

**Details**

This function computes the probability for each score in the support of the distribution. The returned value is a vector of probabilities where the returned vector has names set equal to the corresponding scores.

Setting epsilon to zero will compute the probability mass function without ignoring any matrices with probabilities smaller than epsilon*D_max (D_max is the numerator associated with the matrix of highest probability for the given constraints). The default value of 1e-16 is experimentally validated to be a very reasonable threshold. Setting the threshold to higher values which are smaller than 1 will lead to underestimating the probabilities of each score since more tables will be ignored.

**Value**

Vector of probabilities for scores in the support.

**Author(s)**

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam

**References**


**See Also**

`QP_Pvalue, QP_Support`
Examples

```r
# Compute the probability mass function of the Quaternary Dot Product Scoring Statistic for the given table margins.
pmf <- QP_Pmf(50,50,50,0,50,50,50)
```

Description

This function computes the probability of a score in the Quaternary Dot Product scoring distribution.

Usage

```r
QP_Probability(score, q_p, q_m, q_z, q_r, n_p, n_m, n_z, epsilon = 1e-16)
```

Arguments

- `score`: The score for which the probability will be computed.
- `q_p`: Expected number of positive predictions.
- `q_m`: Expected number of negative predictions.
- `q_z`: Expected number of nil predictions.
- `q_r`: Expected number of regulated predictions.
- `n_p`: Number of positive predictions from experiments.
- `n_m`: Number of negative predictions from experiments.
- `n_z`: Number of nil predictions from experiments.
- `epsilon`: Threshold for probabilities of matrices. Default value is 1e-16.

Details

Setting epsilon to zero will compute the probability mass function without ignoring any matrices with probabilities smaller than epsilon*D_max (D_max is the numerator associated with the matrix of highest probability for the given constraints). The default value of 1e-16 is experimentally validated to be a very reasonable threshold. Setting the threshold to higher values which are smaller than 1 will lead to underestimating the probabilities of each score since more tables will be ignored.

For computing p-values, the user is advised to use the p-value function which is optimized for such purposes.

Value

This function returns a numerical value, where the numerical value is the probability of the score.

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam
**QP_Pvalue**

**References**


**See Also**

QP_Pmf, QP_Pvalue, QP_SigPvalue

**Examples**

```r
# Computing The probability of score 50
# for the given table margins.
prob <- QP_Probability(0, 50, 50, 50, 0, 50, 50, 50)
```

**QP_Pvalue**

*Computes the p-value of a score.*

**Description**

This function computes the right sided p-value for the Quaternary Dot Product Scoring Statistic.

**Usage**

```r
QP_Pvalue(score, q_p, q_m, q_z, q_r, n_p, n_m, n_z, epsilon = 1e-16)
```

**Arguments**

- `score`: The score for which the p-value will be computed.
- `q_p`: Expected number of positive predictions.
- `q_m`: Expected number of negative predictions.
- `q_z`: Expected number of nil predictions.
- `q_r`: Expected number of regulated predictions.
- `n_p`: Number of positive predictions from experiments.
- `n_m`: Number of negative predictions from experiments.
- `n_z`: Number of nil predictions from experiments.
- `epsilon`: Threshold for probabilities of matrices. Default value is 1e-16.

**Details**

Setting epsilon to zero will compute the probability mass function without ignoring any matrices with probabilities smaller than epsilon*D_max (D_max is the numerator associated with the matrix of highest probability for the given constraints). The default value of 1e-16 is experimentally validated to be a very reasonable threshold. Setting the threshold to higher values which are smaller than 1 will lead to underestimating the probabilities of each score since more tables will be ignored.
Value

This function returns a numerical value, where the numerical value is the p-value of the score.

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam

References


See Also

QP_SigPvalue

Examples

# Computing The p-value of score 50
# for the given table margins.
pval <- QP_Pvalue(50,50,50,50,0,50,50,50)

QP_SigPvalue Computes the p-value for a statistically significant score.

Description

This function computes the right sided p-value for the Quaternary Dot Product Scoring Statistic for statistically significant scores.

Usage

QP_SigPvalue(score, q_p, q_m, q_z, q_r, n_p, n_m, n_z, epsilon = 1e-16, sig_level = 0.05)

Arguments

score The score for which the p-value will be computed.
q_p Expected number of positive predictions.
q_m Expected number of negative predictions.
q_z Expected number of nil predictions.
q_r Expected number of regulated predictions.
n_p Number of positive predictions from experiments.
n_m Number of negative predictions from experiments.
n_z Number of nil predictions from experiments.
epsilon Threshold for probabilities of matrices. Default value is 1e-16.
sig_level Significance level of test hypothesis. Default value is 0.05.
Details

Setting epsilon to zero will compute the probability mass function without ignoring any matrices with probabilities smaller than epsilon*D_max (D_max is the numerator associated with the matrix of highest probability for the given constraints). The default value of 1e-16 is experimentally validated to be a very reasonable threshold. Setting the threshold to higher values which are smaller than 1 will lead to underestimating the probabilities of each score since more tables will be ignored. If the score is not statistically significant, then a value of -1 will be returned.

Value

This function returns a numerical value, where the numerical value is the p-value of a score if the score is statistically significant otherwise it returns -1.

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam

References


See Also

QP_Pvalue

Examples

# Computing The p-value of score 50
# for the given table margins.
pval <- QP_SigPvalue(50,50,50,50,0,50,50,50)

QP_Support

Computes the support for the scores.

Description

This function computes the support of the Quaternary Dot Product Scoring distribution for signed causal graphs. This includes all scores which have probabilities strictly greater than 0.

Usage

QP_Support(q_p, q_m, q_z, q_r, n_p, n_m, n_z)
Arguments

- `q_p`: Expected number of positive predictions.
- `q_m`: Expected number of negative predictions.
- `q_z`: Expected number of nil predictions.
- `q_r`: Expected number of regulated predictions.
- `n_p`: Number of positive predictions from experiments.
- `n_m`: Number of negative predictions from experiments.
- `n_z`: Number of nil predictions from experiments.

Value

Integer vector of support.

Author(s)

Carl Tony Fakhry, Ping Chen and Kourosh Zarringhalam

References


Examples

# Compute the support of the Quaternary Dot Product Scoring distribution with the given margins.
QP_Support(50,50,50,0,50,50,50)
Index

BioQCREtoNet, 3
QP_Pmf, 6, 9
QP_Probability, 8
QP_Pvalue, 7, 9, 9, 11
QP_SigPvalue, 9, 10, 10
QP_Support, 7, 11
QuaternaryProd-package, 2