Package 'CoRegNet'

May 7, 2023

Type Package

Title CoRegNet : reconstruction and integrated analysis of co-regulatory networks

Version 1.38.0

Date 2015-03-25

Author Remy Nicolle, Thibault Venzac and Mohamed Elati

Maintainer Remy Nicolle <remy.c.nicolle@gmail.com>

Description This package provides methods to identify active transcriptional programs. Methods and classes are provided to import or infer large scale co-regulatory network from transcriptomic data. The specificity of the encoded networks is to model Transcription Factor cooperation. External regulation evidences (TFBS, ChIP,...) can be integrated to assess the inferred network and refine it if necessary. Transcriptional activity of the regulators in the network can be estimated using an measure of their influence in a given sample. Finally, an interactive UI can be used to navigate through the network of cooperative regulators and to visualize their activity in a specific sample or subgroup sample. The proposed visualization tool can be used to integrate gene expression, transcriptional activity, copy number status, sample classification and a transcriptional network including co-regulation information.

License GPL-3

Depends R (>= 2.14), igraph, shiny, arules, methods

Suggests RColorBrewer, gplots, BiocStyle, knitr, rmarkdown

VignetteBuilder knitr

biocViews NetworkInference, NetworkEnrichment, GeneRegulation, GeneExpression, GraphAndNetwork,SystemsBiology, Network, Visualization, Transcription

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

git_branch RELEASE_3_17

git_last_commit c7de07d

git_last_commit_date 2023-04-25

Date/Publication 2023-05-07

R topics documented:

`coRegnet-package` *coRegnet : inference and interrogation co-regulation networks*

Description

coRegnet can be used to infer and analyse regulatory networks. Although it can be used with any network, it has it's own regulatory network inference routine based on a hybrid version of LICORN which combines both a discrete and a statistical model to infer regulatory network with an emphasis on regulator cooperativity.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

References

Chebil I, Nicolle R, Santini G, Rouveirol C & Elati M (2014) Hybrid Method Inference for the Construction of Cooperative Regulatory Network in Human. IEEE Trans Nanobioscience

Nicolle R, Elati M and Radvanyi F (2012) Network Transformation of Gene Expression for Feature Extraction. In pp 108-113.

Elati M, Neuvial P, Bolotin-Fukuhara M, Barillot E, Radvanyi F and Rouveirol C (2007) LICORN: learning cooperative regulation networks from gene expression data. Bioinformatics 23: 2407-2414

addEvidences *Integration of regulatory evidences.*

Description

These functions can be used to integrate external data on gene regulation and co-regulation to enrich an inferred regulatory network. Additional regulatory data sets can include : ChIP-seq data, ChIP on chip, target gene promoter analysis for Transcription Factor Binding Site models, epigenetic marks or even networks inferred by other methods. Cooperative regulation data sets can include : Protein interactions, significant binding site overlap, co-localization. These additional evidence are added to enrich the network and the enrichment of the inferred interaction is tested.

Usage

```
addEvidences(object,...)
addCooperativeEvidences(object,...)
```

Arguments

object	A regulatory network inferred by the hLICORN function or with an externally inferred network importing through the CoRegNet class.
...	A single or several data sets of regulatory data as pairs of Transcription Factor to Target Gene regulation interactions or of pairs of transcription factors for cooperative evidences. These should be given in the form of a two column data.frame or matrix. The characters in the input dataset should correspond to the characters of the regulators and the genes in the network (accessible through regulators targets respectively).

Details

A single or several datasets of regulatory interactions can be added and enrich an inferred regulatory network. Below is an example of the format of the input data A character matrix or data.frame with two columns the first for target genes and the second for Tanscription Factor.

[,1] [,2]

[1,] "TF1" "Gene1"

[2,] "TF1" "Gene2"

[3,] "TF2" "Gene1"

...

Value

A Gene Regulatory Network in a CoRegNet objects with additional informations in the core object.

Note

The variables used as the input are kept and will be used later on to refine the network. The names of target genes and of TF need to be identical to the ones in the expression dataset.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

See Also

refine hLICORN

Examples

```
#Creating a synthetic network. Upper case letters are target genes, lower case letters are regulators
acts=apply(matrix(rep(letters[1:4],4),nrow=2),2,paste,collapse=" ")
reps=apply(matrix(rep(letters[5:8],4),nrow=2),2,paste,collapse=" ")
grn=data.frame("Target"= LETTERS[1:16] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(16),stringsAsFact
co=coregnet(grn)

#Creating dummy regulatory evidence data in a data.frame
evidence1=unique(data.frame(tf=sample(letters[1:8],50, replace=TRUE),target=sample(LETTERS[1:16],50, replace=TR

evidence2=unique(data.frame(tf=sample(letters[1:8],50, replace=TRUE),target=sample(LETTERS[1:16],50, replace=TR


#Creating dummy cooperative vidence data in a data.frame
coregevidence1=unique(data.frame(tf1=sample(letters[1:8],50, replace=TRUE),target=sample(letters[1:8],50, repla



enrichco=addEvidences(co,evidence1,evidence2)
enrichco=addCooperativeEvidences(enrichco,coregevidence1)
```

`coregnet` *Initialize a co-regulatory network object.*

Description

Given a previously built network encoded in a simple data.frame, initialize a co-regulatory network object `coregnet`. This can be used on networks inferred from gene expression data by other algorithms (ARACNe, GENIE3 ...) but also on ChIP-seq data or on network based on Transcription Factor Binding Site analysis. If the input network does not contain a dscription of the type of regulation (activation, inhibition) an expression data set is needed.

Usage

```
coregnet(GRN, expressionDATA = NULL)
```

Arguments

GRN	A data.frame containing the network. A two column data.frame should contain the regulator in column one and the target gene in column two. A three column data.frame should contain the target in column one, all the co-activators and co-repressors seperated by a space in column two and three respectively.
expressionDATA	If the input GRN is only a two column data.frame, the expression data is needed to set each regulator of a target gene as an activator or a repressor.

Details

In the case of a two column data.frame, the pearson correlation coefficient is used to determine wether a given regulator is an activator (R-squared >= 0) or a repressor (R-squared < 0).

In order to import a network and initialize a new coregnet object the input data.frame should have three columns, the first containing the target genes, then the activators and finally the inhibitors. Target genes can be present in multiple lines. Several regulators can be present in column 2 and 3 if they are separated by a space. Below is an example of a toy network :

A;R1 R2;R3 R4

A;R1 R5;R3 R6

B;R1 R2 R3;NA

...

Value

Returns a `coregnet` object.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

See Also

hLICORN

Examples

```
acts=apply(matrix(rep(letters[1:4],4),nrow=2),2,paste,collapse=" ")
reps=apply(matrix(rep(letters[5:8],4),nrow=2),2,paste,collapse=" ")
grn=data.frame("Target"= LETTERS[1:16] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(16),stringsAsFact
co=coregnet(grn)
```

`coregnet-class` *Class coregnet — Specifying the structure of the network used throughout the package.*

Description

`coregnet` is an S4 type class which specifies the structure containing the necessary data of a Co-Regulatory network. `coregnet` objects are constructed by the `hLICORN` function which infers a regulatory network from gene expression data. A `coregnet` object can also be initiliazed by a network constructed by any other reverse engeneering method as specified in the manual of the `coregnet` function.

Initialization

An object of type CoRegNet can be either built from a gene expression data set using the `hLICORN` function, using a data.frame containing a network specifying the activators and inhibitors of each gene or using an adjacency matrix (see the coregnet function).

Slots

`GRN`**:** A data.frame with three columns, Target Gene, co-activators and co-repressors. Co-regulators are seperated by spaces.

`adjacencyList`**:** A description of the network through a adjacency matrix.

bytf A named list of lists. The first entry point, the names of the list, are the Transcription Factors of the network. For each TF a list with two entries, act and rep, contains the set of activated and repressed genes respectively.

bygene A named list of lists. The first entry point, the names of the list, are the target genes of the network. For each gene a list with two entries, act and rep, contains the set of activators and repressors of the genes respectively.

`coRegulators`**:** A data.frame specifying the set of inferred co-regulators with several measures and statistics for each pairs.

`evidences`**:** A list containing all the added regulatory or co-regulatory evidences if any.

`evidenceDescription`**:** A list containing all the added regulatory or co-regulatory evidences if any.

`inferenceParameters`**:** A list of parameters of hLICORN.

Methods

Generic methods to be used with this class :

print Prints the number of genes, transcription factors and regulatory interactions.

summary Same as print

length outputs the number of genes, transcription factors and regulatory interactions in the form of a numeric vector

dim same as length

targets Returns the set of target genes or the targets of a given list of regulators

regulators Returns the set of regulators of the entwork or the set of regulators of a given list of genes.

activators Same as regulators but returns only activators of a gene

repressors Same as activtors for repressor regulators

as.list Gets the network in the form of an adjacency list

as.data.frame Get the network in the form of an edge list

addEvidences Enriches the network with external regulation evidences

addCooperativeEvidences Enriches the network with external evidence of cooperative interaction between regulators

refine Refine the network based on a score which can take into account external regulation or co-regulation data.

regulatorInfluence Predicts the influence of the regulators in the network. Returns a predicted regulatory activity sample by sample

fitCoregnet Returns the fitness of the regulatory network given an expression dataset

display Starts a shiny application to display the co-regulation graph

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

See Also

hLICORN, coregnet

`coregulators` *Extract all co-regulators of a regulatory network.*

Description

Based on the frequency and specificity of co-regulation, this functions extracts from a coregnet network all the cooperative regulators.

Usage

```
coregulators(object, maxcoreg = 2, verbose = TRUE, minCommonGenes = ifelse(maxcoreg == 2, 1, 10), adjust
```

Arguments

object	The coregnet object containing the co-regulatory network
maxcoreg	The maximum size of co-regulator sets to extract. The default is 2 to extract pairs of co-regulators
verbose	To output information during the extraction process. By default does not ouput anything except warnings.
minCommonGenes	The minimum number of commmon genes, co-regulated genes, to consider two regulators for further co-regulation analysis. Default is 1 for pairs and 10 for bigger sets.
adjustMethod	The p-value adjustment method to extract significant pairs of co-regulators. Default is FDR correction. Anything that is accepted by the p.adjust method is fine.
alpha	The threshold to consider a pair of co-regulator significant (after pvalue correction). Default is 1%

Value

For maxcoreg set to 2, a data.frame with pairs of regulators in the two first columns (Reg1 and Reg2), the Support (the portion of coregulated gene networks in which this pair is found), nGRN for the number of times the pair was found as cooperative regulators in the net, a p-value of a FisherTest testing the specificty of the shared targets and the adjusted pvalue of this test.

For maxcoreg higher than 2, a two column data.frame with the set of co-regulators (collapsed in one character seperated by a space) and the Support.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

See Also

regulators,hLICORN

Examples

```
acts=apply(rbind(rep("z",14),matrix(rep(letters[1:4],7),nrow=2)),2,paste,collapse=" ")[1:13]
reps=apply(matrix(rep(letters[5:8],7),nrow=2),2,paste,collapse=" ")[1:13]
grn=data.frame("Target"= LETTERS[1:26] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(26),stringsAsFact
co=coregnet(grn)

coregulators(co)
coregulators(co,maxcoreg=3,minCommonGenes=3)

coregulators(co,adjustMethod="bonferroni")

coregulators(co,alpha=1)
```

discretizeExpressionData *Three-value discretization of gene expression data.*

Description

Pre-process step to transform log2 numerical expression data into a three value categorical : over-expression (+1), under-expression (-1) and no change (0).

Usage

```
discretizeExpressionData(numericalExpression, threshold = NULL,
refSamples = NULL, standardDeviationThreshold = 1)

discretizeExpressionData(numericalExpression, threshold = NULL,
refSamples = NULL, standardDeviationThreshold = 1)
```

Arguments

numericalExpression	A matrix of continuous log2 gene expression data with genes in rows and samples in column.
threshold	A numeric value used as a fixed fold change threshold for brut discretization. Can be NULL if a standardDeviationThreshold is not.
refSamples	A vector of column names used as a set of reference samples to be used to compute fold changes. Can be NULL if the input data is already centered on a reference values (tested by the presence of negative values) or if the mean if each gene should be used to center (not scale) each expression values.
standardDeviationThreshold	The multplicator of the whole data set standard deviation to be used as the threshold.

Details

Given a continuous log2 gene epression matrix this function aims at producing a matrix of discretized expression values. The numerical data must be in some form of fold change to compare each value of a gene in a sample with a reference value of the same gene. The behavior of the function will therefore depend on the form of the input data.

Given a matrix with negative values, the function will consider that the data is already in the right format and will simply apply a hard threshold to discetize the data.

Given a matrix with only positive values, which is the case for normalized RNAseq or single color microarrays, the function will center each gene based on it's mean expression in all samples or based on the mean of expression of a set of reference sample (normal samples in a study of a particular disease for example).

In either case, the threshold will be used to transform the data in +1s if the value of a gene in a sample is above or equal to the threshold, -1s if the value is below the the negative value of the threshold and 0 otherwise.

The default is to compute a threshold based on the overall distribution of the numerical values in the dataset. This was choosen over a default fold change example (usually 1 or 2 corresponding to a two-fold or four-fold increase/decrease) after observing a large difference between technologies. However, the choice between a simple hard fold change threshold or a threshold as a multiplicator of the global standard deviation remains.

Value

A matrix of integers with the same number of rows and the same number of column as the input numericalExpression. Values in the output matrix are in -1,0,1. The reference samples are removed if given.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

Examples

```
# Use mean of each gene as a reference
expression=matrix(2*rnorm(200),nrow=2,dimnames=list(paste("gene",1:2,sep=""),paste("sample",1:100,sep="")))
discExp=discretizeExpressionData(expression)
boxplot(expression~discExp,xlab="Discrete values",ylab="Continuous values")
pie(table(discExp))
discExp=discretizeExpressionData(expression,standardDeviationThreshold=2)
pie(table(discExp))
discExp=discretizeExpressionData(expression,threshold=1)
pie(table(discExp))

# Use of reference sample
expression=matrix(2*rnorm(200),nrow=2,dimnames=list(paste("gene",1:2,sep=""),paste("sample",1:100,sep="")))
discExp=discretizeExpressionData(expression,refSamples=1:10)
boxplot(expression~discExp,xlab="Discrete values",ylab="Continuous values")
pie(table(discExp))
discExp=discretizeExpressionData(expression,standardDeviationThreshold=2,refSamples=1:10)
pie(table(discExp))
discExp=discretizeExpressionData(expression,threshold=1,refSamples=paste("sample",1:10,sep=""))
pie(table(discExp))
```

`display` *Display a shiny interactive web interface to*

Description

Launches a shiny webpage for interactive viewing and analysis of the co-regulation network using a javascript cytoscape network.

Usage

```
display(coregnetwork, expressionData = NULL, TFA = NULL, alterationData = NULL, clinicalData = NULL, TFr
```

Arguments

coregnetwork	A coregnet object
expressionData	A matrix or data.frame object with named columns (samples) and named rows (genes) containing gene expression data
TFA	A matrix or data.frame object with named columns (samples) and named rows (genes) containing transcription factor activity data such as the data obtained using the regulatorInfluence function. (unnecessary if expression data is given, makes the function run faster)
alterationData	optional. A matrix or data.frame object with named columns (samples) and named rows (genes) containing gene alteration data
clinicalData	optional. Either a list or a factor describing clinical information about samples. A list must be named and each entry should contain a vector of samples.
TFnotes	optional. A factor describing TFs.
allTFplot	A function ploting information by default. Default functions are implemented.
oneTFplot	A function ploting information about a TF which will be used when a single node is selected on the network. Default function are implemented.

Value

Does not return anything.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

Examples

```
acts=apply(matrix(rep(letters[1:4],7),nrow=2),2,paste,collapse=" ")[1:13]
reps=apply(matrix(rep(letters[5:8],7),nrow=2),2,paste,collapse=" ")[1:13]
grn=data.frame("Target"= LETTERS[1:26] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(26),stringsAsFact
co=coregnet(grn)
samples= paste("S",1:100,sep="")
expression=matrix(rnorm(3400),ncol=100)
dimnames(expression) = list(c(grn$Target,names(regulators(co))),samples)

TFA = regulatorInfluence(co,expression,minTarg=4)
colnames(TFA) = samples
if(interactive()){
 display(co,TFA=TFA,expressionData=expression)
}

CNA =matrix( sample(-2:2,800,replace=TRUE),ncol=100)
dimnames(CNA) = list(names(regulators(co)),samples)
if(interactive()){
```

```
  display(co,TFA=TFA,expressionData=expression,alteration=CNA)
}

clinicGrp = factor(paste("grp",sample(1:3,100,replace=TRUE),sep=""))
names(clinicGrp) =samples
if(interactive()){
 display(co,TFA=TFA,expressionData=expression,alteration=CNA,clinicalData=clinicGrp)
}
```

---

hLICORN *Hybrid Learning of co-operative regulation network.*

Description

Parallelized inference of co-regulatory network from gene expression only.

Usage

```
hLICORN(numericalExpression, discreteExpression =
discretizeExpressionData(numericalExpression),
TFlist, GeneList = setdiff(rownames(numericalExpression), TFlist),
parallel = c("multicore", "no", "snow"), cluster = NULL,
minGeneSupport=0.1,minCoregSupport = 0.1,
maxCoreg=length(TFlist),
searchThresh=1/3,nGRN=100,
verbose=FALSE)
```

Arguments

`numericalExpression`
: A numerical Matrix containing the expression of genes and of transcription factors that will be used to inferred the network. Rownames should contain the Gene names/identifiers. Samples should be in columns but Colnames are not important. The data will be gene-centered but not scaled.

`discreteExpression`
: optional. Should be in exactly the same format as `numericalExpression` (dimensions, colnames and rownames) and should contain value only in -1,0,1 with -1 for under-expressed, 0 for no change and 1 for over expressed. For default value see details.

`TFlist`
: A character vector containing the names of the genes that are designated as Transcription Factor or Regulators. These should be contained in the `rownames` of the `numericalExpression` argument. Use `data(HumanTF)` for gene symbols of Human transcription factor

`GeneList`
: optional. The list of genes for which Gene Regulatory Networks should be inferred. Should be in the rownames of the expression data. If not provided will be taken as all the genes in the rownames of the expression data that are not annotated as TF in `TFlist`.

parallel	optional. The type of parallel method to use to run the process on several threads. Should be a value in "no" for single thread calculation, "multicore" to use several threads on the same local machine or "snow" to use the snow package in which case the cluster argument should contain a cluster object (e.g. makeCluster)
cluster	A object of type cluster from the snow package which describes a cluster of computer to run the inference of each gene in parallel. Needed only when parallel is set to snow.
minGeneSupport	A float between 0 and 1. Minimum number of samples in which a gene has to have non-zero discretized expression value to be considered for regulatory network inference.
minCoregSupport	A float between 0 and 1. Minimum number of samples in which a set of co-regulators have the same non-zero discretized expression value to be considered as a potential set of co-regulator. Default is 0.1 . Can be increased in case of limitations on memmory.
maxCoreg	A integer. Maximum size of co-regulator to consider. Default is set to the number of TF. Can be decreased in case of limitations on memmory.
searchThresh	A float between 0 and 1. Minimum proportion of sample in which a gene and a set of co-regulators must have the same non-zero discretized value in order to be considered as a potential co-regulator of the gene.
nGRN	if NA, takes only the best GRN models.
verbose	Sets wether information will appear in console during computation.

Details

A parallelized implementation of the h-LICORN alogrithm. The inference of the network can be run in parallel using either a single machine or on a cluster using the make cluster in the parallel package (now a base R package).

The two required inputs are the continuous gene expression matrix and the list of transcription factor in the rows of the dataset. For Human, a list of TF is included as default data set using data(HumanTF) for symbols or data(HumanTF_entrezgene) for entrez gene IDs.

In case of memmory overflow, a higher minCoregSupport (0.2 to 0.4) will perform more shallow search of co-regulators but will limit the quantity of memmory used.

In some cases, in particular for very large datasets (several hundreds and thousands of samples), a lower minCoregSupport (down to 0.01) can be used for deep search.

Value

A object of type CoRegNet with gene regulatory network (GRN) containing several solution per gene and specifying the co-regulation interactions, the same network in the form of an adjacency list (adjacencyList) and the inferred co-regulators (coRegulators).

Note

The regulatory network inference process is done in several steps. The speed of the inference is dependant on several factors. The number of genes, the density of the discrete expression matrix (number of 1 and -1).

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

References

Elati M, Neuvial P, Bolotin-Fukuhara M, Barillot E, Radvanyi F and Rouveirol C (2007) LICORN: learning cooperative regulation networks from gene expression data. Bioinformatics 23: 2407-2414

Chebil I, Nicolle R, Santini G, Rouveirol C and Elati M (2014) Hybrid Method Inference for the Construction of Cooperative Regulatory Network in Human. IEEE Trans Nanobioscience

See Also

discretizeExpressionData,coregnet-class

Examples

```
# Dummy expression data
gexp=matrix(rnorm(2600,sd=3),ncol=100)

gexp=rbind(gexp,do.call(rbind,lapply(1:26,function(i){
tf = sample(1:26,4)
 return((gexp[tf[1],]+gexp[tf[2],] -gexp[tf[3],]-gexp[tf[4],] +rnorm(100,sd=0.5))/2)})))
dimnames(gexp)=list(c(letters,LETTERS),paste("s",1:100,sep=""))

## Simple example of network inference
dummyNet=hLICORN(gexp,TFlist = letters)

## Infer a network only on a subset of genes
subgene = unique(dummyNet@GRN$Target)[1:2]
dummyNet=hLICORN(gexp,TFlist = letters,GeneList=subgene)

## Discretize data based on a set of reference samples (here 10 first)
discexp = discretizeExpressionData(gexp,refSamples=1:10)
dummyNet=hLICORN(gexp,TFlist = letters,discreteExpression=discexp)


## The network can be queried using the following functions
# returns the hub regulators
regulators(dummyNet)
# get the regulators of a given gene
regulators(dummyNet,"A")
activators(dummyNet,"A")
targets(dummyNet)
targets(dummyNet,"b")


# or transformed into a data.frame
coregnetToDataframe(dummyNet)
```

Human Data Examples *Human Transcription Factor data and Bladder cancer dataset.*

Description

Internal datasets for the use of the package containing : A list of 2020 Human Transcription Factors in EntrezGene or Official Gene Symbol (HUGO) from the FANTOM consortium (Ravasi et al, 2010), A gene expression and copy number dataset of bladder cancer, Datasets of human TF protein interactions and regulatory interactions.

Usage

```
data(HumanTF)
data(HumanTF_entrezgene)
data(CIT_BLCA_EXP)
data(CIT_BLCA_CNV)
data(CIT_BLCA_Subgroup)
data(ENCODE_sub)
data(CHEA_sub)
data(STRING_sub)
data(HIPPIE_sub)
```

Format

`HumanTF` A character vector of Official (HUGO) gene symbols

`HumanTF_entrezgene` A character vector of EntrezGene

`CIT_BLCA_EXP` A matrix of bladder cancer gene expression from the CIT program. Only contains the 1000 genes with highest standard deviation. rownames : genes, colnames : samples. (from Rebouissou et al. 2014)

`CIT_BLCA_CNV` A matrix of bladder cancer DNA copy number aberrations from the CIT program. Only contains the TF and the tumor samples of `CIT_BLCA_EXP`. rownames : TF, colnames : samples.

`CIT_BLCA_Subgroup` A data.frame with sample name in first column and classification from the TCGA in the second column.

`CIT_BLCA_smallGRN` An example CoRegNet object.

`ENCODE_sub` A data.frame of regulatory interactions from the ENCODE project.

`CHEA_sub` A data.frame of regulatory interactions from the ChEA2 database. (http://amp.pharm.mssm.edu/ChEA2)

`STRING_sub` A data.frame of Protein interaction between TF from the STRING database. (http://string.embl.de)

`HIPPIE_sub` A data.frame of Protein interaction between TF from the HIPPIE database. (http://cbdm.mdc-berlin.de/tools/hippie)

References

Rebouissou S, Bernard-Pierrot I, de Reynies A, Lepage M-L, Krucker C, Chapeaublanc E, Herault A, Kamoun A, Caillault A, Letouze E, Elarouci N, Neuzillet Y, Denoux Y, Molinie V, Vordos D, Laplanche A, Maille P, Soyeux P, Ofualuka K, Reyal F, et al (2014) EGFR as a potential therapeutic target for a subset of muscle-invasive bladder cancers presenting a basal-like phenotype. Science Translational Medicine 6: 244ra91-244ra91

Ravasi T, Suzuki H, Cannistraci CV, Katayama S, Bajic VB, Tan K, Akalin A, Schmeier S, Kanamori-Katayama M, Bertin N, Carninci , Daub et al (2010) An Atlas of Combinatorial Transcriptional Regulation in Mouse and Man. Cell 140: 744-752

Kou Y, Chen EY, Clark NR, Duan Q, Tan CM and Ma ayan A (2013) ChEA2: Gene-Set Libraries from ChIP-X Experiments to Decode the Transcription Regulome. In ... pp 416-430. Berlin, Heidelberg: Springer Berlin Heidelberg

Franceschini A, Szklarczyk D, Frankild S, Kuhn M, Simonovic M, Roth A, Lin J, Minguez P, Bork P, Mering von C and Jensen LJ (2012) STRING v9.1: protein-protein interaction networks, with increased coverage and integration. Nucleic Acids Research 41: D808-D815

Schaefer MH, Fontaine J-F, Vinayagam A, Porras P, Wanker EE and Andrade-Navarro MA (2012) HIPPIE: Integrating Protein Interaction Networks with Experiment Based Quality Scores. PLoS ONE 7: e31826

Examples

```
data(HumanTF)
head(HumanTF)
data(HumanTF_entrezgene)
head(HumanTF_entrezgene)
```

`HumanTF` *List of Human Transcription Factors.*

Description

A list of 2020 Human Transcription Factors in EntrezGene or Official Gene Symbol (HUGO) from the FANTOM consortium (Ravasi et al, 2010).

Usage

```
data(HumanTF)
```

Format

Two different objects :

`HumanTF_entrezgene` A character vector of EntrezGene

`HumanTF` A character vector of Official (HUGO) gene symbols

References

Ravasi T, Suzuki H, Cannistraci CV, Katayama S, Bajic VB, Tan K, Akalin A, Schmeier S, Kanamori-Katayama M, Bertin N, Carninci , Daub et al (2010) An Atlas of Combinatorial Transcriptional Regulation in Mouse and Man. Cell 140: 744-752

Examples

```
data(HumanTF)
head(HumanTF)
data(HumanTF_entrezgene)
head(HumanTF_entrezgene)
```

`masterRegulator` *Identify phenotype related Master Regulators.*

Description

This function implements methods to identify Master Regulators based on a regulatory network and on various type of input representing the implication of sets of gene in a phenotype of interest. These are derived from Celine Lefebvre's algorithm MARINa (Lefebvre, 2010).

Usage

```
masterRegulator(coregnet,targetGenes,method=c("set.overlap","merge.pvalues","list.enriched"))
```

Arguments

`coregnet`	A large scale co-regulatory network of type coregnet
`targetGenes`	A liste of target genes. Can be either given as a character vector containing the genes or a named numeric vector containing weights (*e.g.* fold change) or p-values with gene as names.
`method`	The method to use to find the Master Regulators of the input target genes. The default is `set.overlap`. See details.

Details

Three types of input can be used depending on the objective.

To identify Master Regulators of a given set of genes (from a pathway, a liste of differentially expressed genes etc ...), `MasterRegulatorInference` simply needs a CoRegNet network object and a character vector describing the Target Genes of interest. Fisher's exact test will be used to identify TF with the set of target genes that is the most specific to these genes of interest.

To identify Master Regulators of a phenotype of interest, the p-values of the comparison with a reference phenotype (using a moderate or unmoderate t-test) or the Fold change can be used with a combined Fisher's test or a Kolmogorov-Smirnov test to identify significant TF of these genes.

Value

The sorted list of TF in the input network with it's associated p-value.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

References

Lefebvre C, Rajbhandari P, Alvarez MJ, Bandaru P, Lim WK, Sato M, Wang K, Sumazin P, Kustagi M, Bisikirska BC, Basso K, Beltrao P, Krogan N, Gautier J, Dalla-Favera R and Califano A (2010) A human B-cell interactome identifies MYB and FOXM1 as master regulators of proliferation in germinal centers. Molecular Systems Biology 6: 1-10

Examples

```
# Dummy expression data and network
gexp=matrix(rnorm(2600,sd=3),ncol=100)
gexp=rbind(gexp,do.call(rbind,lapply(1:26,function(i){
tf = sample(1:26,4)
  return((gexp[tf[1],]+gexp[tf[2],] -gexp[tf[3],]-gexp[tf[4],] +rnorm(100,sd=3))/2)})))
dimnames(gexp)=list(c(letters,LETTERS),paste("s",1:100,sep=""))
GRN=hLICORN(gexp,TFlist = letters)


MR=masterRegulator(GRN,LETTERS[1:10])
head(MR)

exampleWeight = rnorm(26)
names(exampleWeight) = LETTERS[1:26]
MR=masterRegulator(GRN,exampleWeight,"list")
head(MR)

examplePvalue = 10^-(0.1+runif(26))
names(examplePvalue) = LETTERS[1:26]
MR=masterRegulator(GRN,examplePvalue,"merg")
head(MR)
```

`refine` *Refine an inferred regulatory network using external evidence.*

Description

Refines the inferred network using the integrated external evidences added by addEvidences or addCooperativeEvidences. Several strategies can be applied depending on the number and type of added evidences. These include supervised and unsupervised processes to use all the integrated data set with the inferred network and select the best Gene Regulatory Networks (GRN). Can also be used when no additional dataset has been integrated.

Usage

```
refine(object,GRNselection=c("best","maximize","threshold"),
  integration=c("unsupervised","supervised"),
  referenceEvidence=NULL,evidenceToMaximize="R2",threshold=NULL,verbose=TRUE)
```

Arguments

`object`	A regulatory network inferred by the hLICORN function which can also have been enriched by external regulatory data sets using addEvidences or addCooperativeEvidences.
`GRNselection`	The type of Gene Regulatory Network (GRN) selection method to apply. Default is to select the best regulatory model per gene. See details for other possibilities.
`integration`	Defines the method to merge all the available data sets into a single score which will define the quality of a given GRN. Default is unsupervised. See details.
`referenceEvidence`	To be specified when using the supervised integration method. Specifies one of the integrated data set as a Gold standard to learn the best weight of each evidences to maximize the number of reference evidence in the final network.
`evidenceToMaximize`	To be specified when using the `maximize` GRN selection scheme. Instead of selecting one GRN per gene, this method will choose a threshold for the merged score that will maximize the number of interaction from a given evidence data set. When using the `supervised` integration method, the default is to use the `referenceEvidence` to be maximized.
`threshold`	When the automatically choosen threshold is not satisfactory, a user given threshold can be applied.
`verbose`	If set to TRUE (the default) sends messages at each step of the process.

Details

This function implements several strategies to select the best large scale regulatory network. Depending on the number and type of added evidences the strategies and some recommandations are detailed below.

The first step of the refinement is the integration of the different external evidences into a *merged score*. If no evidence data set has been added, the score given by the inference algorithm is used by its own (an adjusted R2). In the `unsupervised` method, the default, the merged score is simply the mean of each of the evidences, including the score given by hLICORN. For the `supervised` process, a weight is given to each of the evidences which is learned using a generalized linear model which will be fitted to predict the regulatory interactions of a user defined reference evidence data set.

These two (`unsupervised` or `supervised`) integration methods are derived from the network learning process used by the modENCODE consortium (Marbach et al, 2012). Once the merged score is obtained, the default is to select the best GRN per gene. However, another possibility is to select all the "good" networks by choosing a threshold on the merged score. This can be done either by a user defined threshold between 0 and 1 or by choosing automatically a threshold that will maximize the interactions originating from a user defined evidence data set.

The default behavior of the function is to integrate the data set in an unsupervised way and selecting the best GRN per gene.

It is recommanded that when no additional data has been integrated and the selection of the network only needs to be done based on the inference score, a bootstraped regression coefficient, then the simplest strategies is to use the default parameters which will select the best GRN per gene.

Value

A coRegNet object specifying a refined large scale co-regulatory network.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

References

Marbach D, Roy S, Ay F, Meyer PE, Candeias R, Kahveci T, Bristow CA & Kellis M (2012) Predictive regulatory models in Drosophila melanogaster by integrative inference of transcriptional networks. Genome Research 22: 1334-1349

See Also

addEvidences and addCooperativeEvidences

Examples

```
#Dummy network and evidence data examples
acts=apply(rbind(rep("z",14),matrix(rep(letters[1:4],7),nrow=2)),2,paste,collapse=" ")[1:13]
reps=apply(matrix(rep(letters[5:8],7),nrow=2),2,paste,collapse=" ")[1:13]
grn=data.frame("Target"= LETTERS[1:26] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(26),stringsAsFact
GRN=coregnet(grn)

tfs=letters
genes=LETTERS
evidence1=unique(data.frame(tf=sample(tfs,100,replace=TRUE),target=sample(genes,100,replace=TRUE),stringsAsFac
evidence2=unique(data.frame(tf=sample(tfs,100,replace=TRUE),target=sample(genes,100,replace=TRUE),stringsAsFac
evidence3=unique(data.frame(tf=sample(tfs,100,replace=TRUE),target=sample(genes,100,replace=TRUE),stringsAsFac

GRNenrich=addEvidences(GRN,evidence1,evidence2,evidence3)
print(GRNenrich)



unsupervisedNet=refine(GRNenrich)
supervisedNet=refine(GRNenrich,integration="sup",referenceEvidence="evidence1")

# The following usually gives poor results...
#supervisedNet=refine(GRNenrich,integration="sup",referenceEvidence="evidence1",evidenceToMaximize="evidence1"
```

regulatorInfluence *Regulator Influence, estimating the sample specific activity of Transcription Factors.*

Description

Uses a network in the form of a coregnet object to compute regulatory influence to estimate the transcriptional activity of each regulators in each sample of the given expression data.

Usage

```
regulatorInfluence(object,expData,minTarg = 10,withEvidences=FALSE,addCoregulators=FALSE, is.scaled
```

Arguments

object	A network in the form of a coregnet object.
expData	An expression data matrix or data.frame.
addCoregulators	Compute influence for coregulators with sufficient number of targets. Default to FALSE.
minTarg	The minimum number of targets for a regulator to be considered for actvity prediction. Default set to 10.
withEvidences	Use only the target genes which are validated by an external validation dataset (ChIP-seq for example). This is only possible if external evidence was added using addEvidences. Default set to False.
is.scaled	Wether the input expression data is scaled, if not it will be.

Value

An N by R matrix with N columns the number of sample in the original expression data and R rows the number of regulators with sufficient targets to compute their influence.

The expression data is centered by default but not scaled.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

References

Nicolle R, Elati M and Radvanyi F (2012) Network Transformation of Gene Expression for Feature Extraction. In pp 108-113.

See Also

hLICORN and coregnet-class to create the network. addEvidences to add external evidences.

Examples

```
acts=apply(matrix(rep(letters[1:4],7),nrow=2),2,paste,collapse=" ")[1:13]
reps=apply(matrix(rep(letters[5:8],7),nrow=2),2,paste,collapse=" ")[1:13]
grn=data.frame("Target"= LETTERS[1:26] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(26),stringsAsFact
co=coregnet(grn)
samples= paste("S",1:100,sep="")
expression=matrix(rnorm(3400),ncol=100)
dimnames(expression) = list(c(grn$Target,names(regulators(co))),samples)

#Minimum number of targets is adjusted because of the small size of the network
TFA = regulatorInfluence(co,expression,minTarg=4)
```

`regulators` *Interogate a coregnet object.*

Description

Query the network for regulators of specific targets and targets of specific genes.

Usage

```
regulators(object, target = NULL, type = c("single", "coregulators"))
activators(object,target,type=c("single","coregulators"))
repressors(object,target,type=c("single","coregulators"))
targets(object,regulator=NULL,type=c("regulating","activating","repressing"))
```

Arguments

`object`	The network in the form of a `coregnet` object to query.
`target`	The target gene to query.
`regulator`	The regulator to query.
`type`	The type of regulation to obtain. Differs depending on the function used.

Value

For `regulators` if no target is given, returns integer vector with the number of targets for each regulators of the network. Given a non null vector, a vector of the union of the regulators of all the genes is returned. For `activators` and `repressors` the behavior is similar except that a target gene is needed. If `type = "coregulators"` then only the regulators, activators or repressors which are found to be co-regulators, co-activators or co-repressors of the target genes are given.

`targets` with no given regulator returns a character vector of all the target genes in the network. Specifying a vector of regulators will return a vector of the union of the targets of all these regulators. The type of regulation can be specified to return only the activated or repressed targets.

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

Examples

```
acts=apply(matrix(rep(letters[1:4],7),nrow=2),2,paste,collapse=" ")[1:13]
reps=apply(matrix(rep(letters[5:8],7),nrow=2),2,paste,collapse=" ")[1:13]
grn=data.frame("Target"= LETTERS[1:26] ,"coact"=c(acts,reps),"corep"= c(reps,acts),"R2"=runif(26),stringsAsFact
co=coregnet(grn)

regulators(co)
regulators(co,"A")
regulators(co,"A","coregulators")


activators(co,"A")
activators(co,"A","coregulators")

repressors(co,"A")
repressors(co,"A","coregulators")

targets(co)
targets(co,"a")
targets(co,"a","reg")
targets(co,"a","act")
targets(co,"a","rep")
targets(co,c("a","b"),"act")
```

---

summary *Summaries and info coregnet*

Description

Several functions to print and view info about the network enclosed in a coregnet object.

Usage

```
summary(object,...)
## S4 method for signature 'coregnet'
show(object)
## S4 method for signature 'coregnet'
dim(x)
## S4 method for signature 'coregnet'
length(x)
## S4 method for signature 'coregnet'
print(x)
## S4 method for signature 'coregnet'
```

```
coregnetToDataframe(network)
## S4 method for signature 'coregnet'
coregnetToList(network)
```

Arguments

network	a coregnet network object.
object	a coregnet network object.
x	a coregnet network object.
...	unused argument

Author(s)

Remy Nicolle <remy.c.nicolle AT gmail.com>

Examples

```
regs=sample(letters,7)
grn=data.frame("Target"= LETTERS ,"activators"= sample(rep(regs,4))[1:26],"repressors"= sample(rep(regs,4))[1:26
co=coregnet(grn)
print(co)
length(co)
dim(co)
co
coregnetToDataframe(co)
coregnetToList(co)
```

Index