SLGI

October 5, 2010

`Atong`	*Systematic genetic analysis with ordered arrays of yeast deletion Tong et. al. (2004).*

Description

Data from Tong et. al. (2004) buffering experiments using ordered arrays of yeast deletion design by Tong et. al. (2001).

Usage

```
data(Atong)
data(tong2004raw)
```

Format

`tong2004Raw` is dataframe extracted from Table S1 of Tong et al. (2004) online supporting material. We added an extra column, queryGene.sysName, which is the systematic names of the query genes.

queryGene.geneName Column indicates the gene used as query in the synthetic genetic array screen (SGA).

Int.geneName Column indicates the gene identified as an interactor with a particular query.

Int.sysName Column indicates the systematic name of the open reading frame (ORF) that corresponds to the interactor gene.

Score An interaction scored three times in the three runs by visual inspection received a scored of 3. An interaction scored twice in the three in the runs by visual inspection received a scored of 2. An interaction scored by the computer-based image analysis but not visual inspection received a scored of 1. For interactions that scored once in the three runs by visual inspection confirmation was attempted only for those genes pairs related functions. Such confirmed interactions received a score of 0.

RSA Column identifies an interaction that was confirmed by random spore analysis.

Tetrad Column identifies an interaction confirmed by tetrad analysis.

SS Refers to synthetic sick interaction.

SL Refers to synthetic lethal interaction.

Functional.Role Column indicates the assigned GO functional annotation from their defined subset of annotations.All the interactions are identified in this study unless otherwise stated.

References Genetic Interactions that have been previously described.

queryGene.sysName Column indicates the systematic (ORF) name of the gene used as query in a SGA screen.

`Atong` is a 132 by 1008 adjacency matrix of the systematic genetic interactions identified between 132 query genes and the deletion gene set (Tong et al. 2001; see `SGA` for more details). The row names correspond to the systematic (ORF) names for the 132 query genes. The column names correspond to the systematic (ORF) names of the 1011 reporter genes, which showed a synthetic lethal or synthetic sick interaction with at least one query genes. Values are 0 or 1, with a 1 indicating the occurrence of the genetic interaction between the gene pairs.

Source

Tong et al, Science. Vol.303, 2004.

References

Global Mapping of the Yeast Genetic Interaction Network, Tong et al, Science Vol.303, 2004.

http://www.sciencemag.org/cgi/data/303/5659/808/DC1/1

See Also

`SGA`

Examples

```
data(Atong)
dim(Atong)
```

AtongFnDomain *The functional domains shared by the tested pairs in Tong et al experiment.*

Description

Data developed from Tong et. al. buffering experiments.

Usage

```
data(AtongFnDomain)
```

Format

A list containing 3 items.

pairs Dataframe of all the gene pairs and their synthetic lethality status.

SharedPfam List of the Pfam domains shared by each pair. The order of this list is the same as the order of the pairs.

SharedSMART List of the SMART domains shared by each pair. The order of this list is also the same as the order of the pairs.

Author(s)

Z. Jiang

Source

Created from the association matrix reported by Tong et al. and the Pfam (Protein family database http://pfam.janelia.org/) and SMART database of yeast.

References

Global Mapping of the Yeast Genetic Interaction Network, Tong et al, Science Vol.303, 2004 Saccharomyces Genome Database

Examples

```
data(AtongFnDomain)
names(AtongFnDomain)
```

AtongPair *Data frame the pair of yeast gene tested in Tong et al. 2004.*

Description

Data from Tong et. al. buffering experiments (2004) using synthetic genetic arrays (SGA) (Tong et al. 2001).

Usage

```
data(AtongPair)
```

Format

A data frame with 3 columns and 607881 rows.

Details

`AtongPair` stores the yeast gene names for each tested pairs in Tong buffering experiment. Each row represents one pair.

query Query gene name

array Array gene name

interact Logical indicating the synthetic lethal status, if TRUE the genetic interaction is lethal.

Author(s)

N. LeMeur

Source

Created from the association matrix reported by Tong et al (2004) and the genes from the SGA array developed by Tong et al. (2001).

References

Systematic genetic analysis with ordered arrays of yeast deletion mutants, Tong et al, *Science*. Vol. 294, 2001

Global Mapping of the Yeast Genetic Interaction Network, Tong et al, *Science* Vol.303, 2004

Examples

```
data(AtongPair)
dim(AtongPair)
```

`Boeke` *Incidence matrix of Synthetic Lethal interaction from the Boeke Lab*

Description

Data from Pan et al. experiments on DNA integrity Network in the Yeast S. cerevisiae.

Usage

```
data(Boeke2006raw)
data(Boeke2006)
```

Format

`Boeke2006raw` is a data frame with 5775 observations on the following 6 variables.

`Query.ORF` ORF associated with the query gene.

`Query.Gene` Common name of the query gene.

`Target.ORF` ORF for the array gene.

`Target.Gene` Common name of the array gene.

`RSA` Random spore analysis

`Tetrad` Tetrad dissection

`Boeke2006` is an incidence matrix is a 74 by 843 adjacency matrix of the systematic genetic interactions identified between 74 query genes and the deletion gene set in Pan et al.(2004). The row names correspond to the systematic (ORF) names for the 74 query genes. The column names correspond to the systematic (ORF) names of the 843 reporter genes, which showed a synthetic lethal or synthetic sick interaction with at least one query genes. Values are 0 or 1, with a 1 indicating the occurrence of the genetic interaction between the gene pairs.

Details

In Pan et al (2006), the authors provide this note. Note: SL - synthetically lethal; SF/SL-very severe synthetic fitness defects; SF-obvious but modest synthetic fitness defects; SF (slight) - slight synthetic fitness defect. Approximately 10% of the positive interactions presented here were not scored as positive in the dSLAM screens. These were individually tested because we wanted to make sure that they were indeed false negatives in the dSLAM screens. We also note that there is a small chance that the interactions scored as positive in RSA (random spore analysis) might not reflect direct growth defects of the double mutants but rather, the double mutants are defective in expressing the MFA1pr-HIS3 reporter.

Source

The data were extracted from Pan et al (2004) and Table S1 of Pan et al. (2006).

References

Pan X, Ye P, Yuan DS, Wang X, Bader JS, Boeke JD. A DNA integrity network in the yeast Saccharomyces cerevisiae. Cell. 2006 Mar 10;124(5):1069-81

Pan X, Yuan DS, Xiang D, Wang X, Sookhai-Mahadeo S, Bader JS, Hieter P, Spencer F, Boeke JD. A robust toolkit for functional profiling of the yeast genome. Mol Cell. 2004 Nov 5;16(3):487-96.

See Also

dSLAM.GPL1444, and dSLAM

SDL *The Association matrix for the synthetic dosage lethal screens in Yeast.*

Description

The data reported in Table 6 of the supplementary data of Measday et. al.

Usage

```
data(SDL)
data(SLchr)
```

Format

SDL is a matrix with 141 rows and 9 columns. The columns represent 3 genes at each of 3 temperatures (16, 25, 37 Celsius). The gene names and temperatures are combined in the column names. The row names are yeast standard names. The values are NA, no effect, SDS for synthetic dosage sick, SL for synthetic lethal and SDL for synthetic dosage lethal.

SLchr is a matrix with 84 rows and 14 columns. Each column represents a query strain which was tested against the genome wide set of deletion strains. The entries can be NA for no effect, SL for synthetic lethal and SS for synthetic sick.

Source

Supplementary Table 6 of the reference given below.

References

Systematic yeast synthetic lethal and synthetic dosage lethal screens identify genes required for chromosome segregation. Measday et al, PNAS, 2005, 13956-13961.

Examples

```
data(SDL)
table(SDL)
```

SGA *Systematic genetic analysis with ordered arrays of yeast deletion.*

Description

Listed of yeast deletion genes used as array probes in the Systematic Genetic Analysis (SGA) of yeast deletion Tong et. al. (2001).

Usage

```
data(SGAraw)
data(SGA)
```

Details

`SGAraw` is a `character` vector of length 4672, corresponding to the original yeast deletion genes set on the array. Note that some of those genes correspond to ORFS that have subsequently been rejected.

`SGA` is a `character` vector of length 4655, corresponding to the updated yeast deletion genes set on the array. The gene names have been updated from common gene name or alias to systemantic names (last update Feb. 2006).

Source

Table S1 from Tong et al . (2001) online supporting material. http://www.sciencemag.org/cgi/content/full/294/5550/2364/DC1

References

Systematic genetic analysis with ordered arrays of yeast deletion mutants, Tong et al, Science. Vol. 294, 2001

Examples

```
data(SGAraw)
length(SGAraw)

if(require("YEAST"))
updateSGA <- mget(SGAraw, YEASTCOMMON2SYSTEMATIC, ifnotfound = SGAraw)
```

SGD.SL *Interaction data from the Saccharomyces Genome Database*

Description

The Saccharomyces Genome Database (SGD) provides, for download a table listing all known interactions in yeast. This table was downloaded on Jan 25, 2007 and three subsets were extracted. The synthetic lethal interactions, `SGD.SL`, the synthetic grow defect interactions, `SGD.SynGrowthDefect` and the synthetic rescue interactions, `SGD.SynRescue`. No other processing has been done.

Usage

```
data(SGD.SL)
data(SGD.SynRescue)
data(SGD.SynGrowthDefect)
```

Format

Each data set is a data frame with the following 7 variables.

`V1` Factor, indicating the type of data.

`V2` Factor describing the interaction, in particular naming bait and prey and interactors.

`V3` Factor indicating whether the cells were viable.

`V4` Factor which is always NA for these data.

`V5` Factor naming the reference for the interaction.

`V6` Factor with levels indicating the PubMed ID for the publication in V5.

`V7` Factor with level `BioGRID`, probably indicating the source.

Details

SGD says this about the file:

```
Contains interaction data.  Tab-separated columns are:
1) interaction_type (mandatory)
2) genes involved and their mutation type, in the format: ORF
(mutation_type, action), with multiples separated by a |
3) phenotype (optional, multiples separated by |)
4) description (optional)
5) citation (multiples separated by |)
6) PubMed ID (optional, multiples separated by |)

This file is updated weekly.
```

Author(s)

Z. Jiang

Source

The file can be downloaded from, `ftp://genome-ftp.stanford.edu/pub/yeast/literature_curation`.

Examples

```
data(SGD.SL)
```

TFmat *Transcription Factor Binding Affinities*

Description

The data are from Lee et al, the rows of the matrix represent genes in S. cerevisiae, the columns known transcription factor. The value in each entry represents the p-value, as reported by Lee et al, for the transcription factor (TF) binding upstream of the gene.

Usage

```
data(TFmat)
```

Format

`TFmat` is a matrix, rows represent genes, columns transcription factors and the elements are p-values representing some notion of the likelihood that the transcription factor binds up stream of the gene.

Author(s)

Z. Jiang

Source

Supplementary material from http://web.wi.mit.edu/young/regulator_network/

References

Transcriptional Regulatory Networks in Saccharomyces cerevisiae, Lee et al. Science 298:799-804 (2002).

Examples

```
data(TFmat)
```

byComplex *Evaluate protein co-membership within cellular organizational units*

Description

Count the protein co-members of one (or more) cellular organizational units such as complex(es). This co-membership can be characterized by a synthetic lethal interaction if bpL is the list of observed synthetic lethal interactions or it can be characterized by the number of all the expected interactions within that complexes if bpL is all the interactions tested.

Usage

```
byComplex(bpL,interactome)
```

Arguments

bpL	List of tested genes (or reported as synthetic lethal) per bait.
interactome	Adjacency matrix where the rows are the genes and the columns represent the cellular organizational units, e.g., ScISI

Value

Vector of the number of genes(proteins) co-member in one or more biological complexes or pathways.

Author(s)

N. LeMeur and R. Gentleman

See Also

withinComplex

Examples

```
data(ScISIC)
data(AtongPair)
pairSL <-  AtongPair[ AtongPair[,3],]
SLlist <- split(as.character(pairSL[,2]),as.character(pairSL[,1]))
##Number of synthetic lethal pairs within the same complexe
bySL <-byComplex(SLlist, ScISIC)
```

---

comemberIn *Retrieve the biological complexes.*

Description

Retrieve the biological complexes within which two proteins are comembers.

Usage

```
comemberIn(iMat,interactome)
```

Arguments

iMat	Comembership matrix of genes(proteins) that linked to other genes(proteins) by any biological experiment
interactome	Adjacency matrix composed of genes (rows) and biological complexes (columns) ScISI

Value

Dataframe of pairs of genes(proteins) and their common biological complexes.

Author(s)

N. LeMeur

See Also

withinComplex

Examples

```
data(Atong)
data(ScISI)
coMember<-withinComplex(Atong,ScISI)
SLpairWithinComplex <- comemberIn(coMember,ScISI)
```

---

compare *Compare observed data to expected in permutation models*

Description

This method summarizes the result of the modelSLGI function.

Usage

```
## S4 method for signature 'siResult':
compare(x)
```

Arguments

x	a siResult object to summarize

Details

This compares the number of observed interactions to the number of expected interactions in each permutation model. It counts how many times the number of observed interactions is greater than the number of expected interactions (from the permutations) and divides by the number of permutations applied.

Value

Numerical vector

Author(s)

N. LeMeur

See Also

modelSLGI

Examples

```
data(ScISIC)
data(Atong)
data(SGA)
model <-  modelSLGI(Atong, universe= SGA, interactome=ScISIC, type="intM", perm=2)
ans <- compare(model)
```

`congruence` *Calculate congruence score between pairs of of genes sharing pattern of synthetic genetic interactions (Ye et al. (2005)).*

Description

The `congruence` score represents the number of common synthetic genetic interacting partners between two genes. The higher is the score the more overlap there is between the synthetic genetic partners of those genes.

Usage

```
congruence(iMat, sharedInt, mode="query", universe, padjust=FALSE)
```

Arguments

`iMat`	Adjacency matrix reporting genetic Interactions. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
`sharedInt`	numeric vector representing the number of common genetic interactions between a pair of query or target genes. See `getSharedInteraction` for more details
`mode`	character vector of value "query" or "target"
`universe`	total number of genes tested
`padjust`	adjust by the number of genes tested that show at least one synthetic genetic interaction.

Value

A numeric vector of the congruence score values.

Author(s)

N. LeMeur

References

Ye P. et al. (2005). Gene function prediction from congruent synthetic lethal interactions in yeast. Molecular Systems Biology 1:2005.0026.

See Also

`getSharedInteraction`

Examples

```
intM <- matrix(c(0,1,0,0,1,1,1,0,1,1,1,1,1,0,1,1),
                nrow=4, ncol=4,
                dimnames=list(c("p1","p2","p3","p4"),
                  c("p1","p3","p5","p7")))
sharedInt <- getSharedInteraction(intM)
score <- congruence(intM, sharedInt, mode="query", universe=15, padjust=FALSE)
```

---

createSquareMatrix *Create a square matrix*

Description

Create a square matrix based on row and column names. The new matrix is created so that the row and column names are a perfect match and the added values are zero.

In the case of genetic interactions, for example it could be useful that the matrix of all the interactions tested and not tested.

Usage

```
createSquareMatrix(data)
```

Arguments

`data`	Matrix

Value

matrix.

Author(s)

N. LeMeur

Examples

```
data(Atong)
dim(Atong)

Tong<- createSquareMatrix(Atong)
dim(Tong)
```

dSLAM.GPL1444 *dSLAM platform used for Synthetic Lethal screens in the Boeke Lab*

Description

These data are the 21991 probes spotted on the dSLAM array (heterozygote diploid-based synthetic lethality analyzed by microarray) used to test synthetic lethal interactions by Pan et al (2006).

Usage

```
data(dSLAM.GPL1444)

data(dSLAM)
```

Format

dSLAM.GPL1444 is a data frame with 21991 observations on the following 10 variables.

`ID` Serial identifier for probe.

`ROW` Row number in the array as scanned with GenePix scanner.

`COLUMN` Column number in the array as scanned with GenePix scanner.

`TAGTYPE` Code for whether tag is 5' (Up) or 3' (Dn) relative to the open reading frame (ORF).

`PROBE` Code for singleton probes arrayed in ORF order (ArrA, ArrB), five-fold replicate probes arrayed in randomized order (Rpts), systematic mutations arrayed across the center of the array (Muts), negative controls (NegT), or probes peripheral to the array as specified by the manufacturer (Edge)

`ORF` Systematic ORF name (from SGD, Feb 2003) (http://genome-www4.stanford.edu/cgi-bin/SGD/locus.pl?locus=

`GENE` Standard gene name (SGD) (or ORF if not available)

`SEQUENCE` DNA sequence of probe (includes custom-designed sequences for 193 YA* and YM* ORFs missing DnTags)

`SGDID` Unique ORF identifier from SGD; 'S000000000' denotes missing value

`SPOT_ID` spot identifier; ('YQL' ORFs denote custom-designed sequences; 'NegA', 'NegB', 'PosA', 'PosB' denote proprietary sequences specified by the manufacturer)

dSLAM is a character vector of length 5641 that contains the unique and valid systematic ORF names.

Details

The dSLAM.GPL1444 were directly obtain from parsing the GPL1444_family.soft.gz available at http://www.ncbi.nlm.nih.gov/projects/geo/query/acc.cgi?acc=GPL1444

dSLAM is a vector of length 5641, extracted from the dSLAM.GPL1444 ORF, and that contains the unique and valid systematic ORF names. This vector was built in three steps. First the ORFs with SGDID equals to S000000000 in the dSLAM.GPL1444 data frame were removed as some correspond to custom sequences and other were dubious ORFs that have been deleted from SGD or merged with other ORFs. Secondly, the duplicated names were removed. Then, the systematic ORF names were verified against the YEAST data package.

Source

The data were extracted from the Gene Expression Omnibus (GEO) website: http://www.ncbi.nlm.nih.gov/projects/geo/query/acc.cgi?acc=GPL1444

References

Pan X, Ye P, Yuan DS, Wang X, Bader JS, Boeke JD. A DNA integrity network in the yeast Saccharomyces cerevisiae. Cell. 2006 Mar 10;124(5):1069-81

See Also

Boeke2006raw, and Boeke2006

---

domainDist *Finds the number of gene sets for each shared domain*

Description

domainDist takes a list of shared domains, and compute for each distinct domain how many gene sets share it.

Usage

```
domainDist(domainL)
```

Arguments

domainL	Each element of the list is a vector of functional domains.

Details

For each domain that appears in the domain list, domainDist counts the number of elements that have this domain.

Value

Returns a frequency table with descending order.

Author(s)

Z. Jiang

See Also

getSharedDomains, sharedBy

Examples

```
data(AtongFnDomain)
domainDist(AtongFnDomain$SharedPfam[1:20])
```

essglist *The list of yeast essential genes*

Description

List of systematic names and common names of the yeast essential genes.

Usage

```
data(essglist)
```

Format

`essglist` is a list with 1103 elements (last download 03/17/2006). The name of each element is the systematic gene name. The value of each element is its corresponding common (standard) name.

Details

The aliases of the yeast gene names can be retrieved with the `YEASTALIAS` environment of the `YEAST` package.

Source

Saccharomyces Genome Database `ftp://genome-ftp.stanford.edu/pub/yeast/data_download/literature_curation/phenotypes.tab` (last download 03/17/2006)

References

Saccharomyces Genome Database `http://www.yeastgenome.org/`

Examples

```
data(essglist)
essglist[[1]]
names(essglist)
```

getFASTAname *Obtain sequence name from FASTA object*

Description

Extract the name of a sequence from a FASTA object that created by `readFASTA` function from `Biostrings` package.

Usage

```
getFASTAname(Fobj)
```

Arguments

Fobj	is a FASTA object created by readFASTA function from **Biostrings** package.

Details

The function gets the first string between ">" and space the "desc" element of the Fobj, which is the names of the sequence.

Value

A character string.

Author(s)

Z. Jiang

See Also

readFASTA

Examples

```
f <- gzfile(file.path(.path.package("SLGI"),
            "extdata/orf_trans.fasta.gz"), open = "rt")
library(Biostrings)
yeastF <- readFASTA(f)
sapply(yeastF[1:5], getFASTAname)
```

---

getInteraction *Count genetic interactions within and between cellular organizational units*

Description

Count the number of genetic interactions within and between the elements of the interactome.

Usage

```
getInteraction(iMat, universe, interactome)
```

Arguments

iMat	Interaction matrix. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
universe	Character vector of gene names, e.g., array genes used in synthetic genetic array experiments (SGA)
interactome	Adjacency matrix where row are gene names and columns are cellular organizational units.

Value

The returned value is a list of 2 matrices:

bwMat	A interaction matrix that corresponds to the cellular organizational units interaction matrix where row and columns a organizational units names and the value inside the matrix are the number of genetic interactions they share.
CDs	Subset of the input interactome that shares interactions.

Author(s)

N. LeMeur

Examples

```
##Create the genetic interaction matrix
gInt <- sample(c(0, 1), 25, TRUE)
iMat  <- matrix(gInt, nrow=5, ncol=5, dimnames=list(letters[1:5],letters[4:8]))

##Create the interactome
cInt <- sample(c(0,1),30, TRUE)
interactome  <- matrix(cInt, nrow=6, ncol=5,dimnames=list(letters[2:7],LETTERS[1:5]))

## Reduce the genetic interaction matrix to match the gene present in
## the interactome
reducediMat <- gi2Interactome(iMat, interactome)

## Get the interaction
prey <- letters[1:20]
myInteraction <- getInteraction(reducediMat, prey, interactome)
```

---

getSharedDomains *Find domains shared by a given list of gene names.*

Description

getSharedDomains finds domains in the provided environment that are shared by a list of genes.

Usage

```
getSharedDomains(geneNameV, env)
```

Arguments

geneNameV	Character vector of gene names.
env	R object that provides mappings between an entrez gene identifier and the associated Pfam identifiers.

Value

getSharedDomains returns a vector of the names of the shared domains.

Author(s)

Z. Jiang

See Also

domainDist, sharedBy

Examples

```
library("org.Sc.sgd.db")
getSharedDomains(c("YEL003W","YLR200W"),  org.Sc.sgdPFAM)
```

---

getSharedInteraction

Calculate the number of shared synthetic genetic interactions between pairs of genes.

Description

The number of common synthetic genetic interacting partners between two genes.

Usage

```
getSharedInteraction(iMat, mode="query")
```

Arguments

iMat	Adjacency matrix reporting genetic Interactions. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
mode	Character vector of value "query" or "target"

Value

A numeric vector of the number of common genetic interactions between a pair of query or target genes.

Author(s)

N. LeMeur

See Also

congruence

Examples

```
intM <- matrix(c(0,1,0,0,1,1,1,0,1,0,0,1,1,0,1,0),
               nrow=4, ncol=4,
               dimnames=list(c("p1","p2","p3","p4"),
                 c("p1","p3","p5","p7")))

sharedInt <- getSharedInteraction(intM)
```

`getTestedPairs` *Find interacting and non-interacting tested pairs from an genetic interaction matrix.*

Description

`getTestedPairs` find all the pairs from an interaction matrix and a list of tested genes.

Usage

```
getTestedPairs(iMat, respV)
```

Arguments

`iMat`	Adjacency matrix reporting genetic Interactions. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
`respV`	Character vector of all gene names that were tested (found to interact or not)

Value

A data.frame with 4 columns:

`query`	gene names of the query genes
`array`	gene names og the tested genes (e.g., array genes)
`interact`	numeric vector of the number of observed interactions (0: no interaction; 1: one interaction; 2: two interactions when the query genes were also on the array)
`recip`	logical to indicate whether the reported genes were both query and array genes (TRUE: both genes were query and array genes)

.

Author(s)

N. LeMeur

See Also

`getSharedDomains` `getUniquePairs`

Examples

```
intM <- c(0,1,0,0,1,1,0,0,1,0,0,1,1,0,1,0)
dim(intM) <- c(4,4)
dimnames(intM) <- list(c("p1","p2","p3","p4"),c("p1","p3","p5","p7") )
respV <- c("p6","p8")
intM
getTestedPairs(intM,respV)
```

`getUniquePairs` *Find unique pairs from an genetic interaction matrix.*

Description

`getUniquePairs` can find all the unique pairs from an interaction matrix and supplementary array genes, or finds only the unique pairs that shows positive interaction.

Usage

```
getUniquePairs(iMat, respV = character(0), only = FALSE)
```

Arguments

`iMat`	Adjacency matrix reporting genetic Interactions. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
`respV`	Character vector of all gene names that were tested (found to interact or not)
`only`	has default value `FALSE`, if `TRUE`, then only reports the positively interacted pairs.

Value

A data.frame with two or three columns. The first two columns are the query gene name and the array gene name, respectively. If `only` is `TRUE`, the third column shows the interaction status.

Author(s)

Z. Jiang

See Also

`getSharedDomains`

Examples

```
intM <- c(0,1,0,0,1,1,0,0,1,0,0,1,1,0,1,0)
dim(intM) <- c(4,4)
dimnames(intM) <- list(c("p1","p2","p3","p4"),c("p1","p3","p5","p7") )
respV <- c("p6","p8")
intM
getUniquePairs(intM,respV,only=FALSE)
getUniquePairs(intM,respV,only=TRUE)
getUniquePairs(intM,only=FALSE)
getUniquePairs(intM,only=TRUE)
```

`gi2005` *Genetic Interaction Data (EMAP) from the yeast early secretory pathway*

Description

The data are in the form of a 424 by 424 array which contains the scores from using the EMAP procedure on yeast strains which are ideally double mutants, each strain with a different pair of genes knocked out. For each row, the gene named in the row label is knocked out in all pairs, and the same holds true for each column.

Usage

```
data(gi2005)
data(gi2005.metadata)
```

Format

`gi2005` is a 424 by 424 array of real values. `gi2005.metadata` is a vector of length 424 which contains the common names for the genes that were knocked out. The row and column names of `gi2005` are standard names.

Details

NA values in `gi2005` are interactions that were not scored.

Source

Data were obtained as supplementary material from the publication listed below.

References

Schuldiner et al, Exploration of the function and organization of the yeast early secretory pathway through an epistatic miniarray profile. Cell, 2005, 123:507-519.

Collins et al, A strategy for extracting and analyzing large-scale quantitative epistatic interaction data. Genome Biology, 2006, 7:R63.

Examples

```
data(gi2005)
data(gi2005.metadata)
```

gi2007 *Synthetic Genetic Interaction data from Collins et al*

Description

The data gi2007 are a 754 by 754 set of genetic interactions that were tested pairwise by either deletion or decreased abundance messenger RNA perturbation.

Usage

```
data(gi2007)
data(gi2007.metadata)
```

Format

The gi2007 data are a 754 by 754 matrix where values indicate a score for a synthetic genetic interaction. An NA indicates that the genetic interaction was not measured.

gi2007.metadata is a data.frame of dimensions 754 rows and two columns. The columns are the systematic names and the mutation (which is typically either DAMP, DELETION or the name of the alternate allele that was tested. In 11 cases an alternative allele was tested.

References

Collins et al. Nature, 2007, Vol 446, p. 806-810. Data are available as supplementary material.

Examples

```
data(gi2007)
data(gi2007.metadata)
```

gi2Interactome *Reduce genetic interactions matrix*

Description

Reduce genetic interactions matrix to the pairs that genetically interact and that are present in the interactome of interest.

Usage

```
gi2Interactome(iMat, interactome, threshold=0)
```

Arguments

iMat	Genetic interaction matrix. Each entry has usually a value of 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
interactome	Interactome matrix, e.g. ScISIC.
threshold	Integer

Value

The returned value is the genetic interaction matrix reduced to the row and column (genes) names that are present in the interactome and where the row and column sums are higher than the specified threshold.

Author(s)

N. LeMeur

Examples

```
##Create the genetic interaction matrix
gInt <- sample(c(0, 1), 25, TRUE)
iMat  <- matrix(gInt, nrow=5, ncol=5, dimnames=list(letters[1:5],letters[4:8]))

##Create the interactome
cInt <- sample(c(0,1),30, TRUE)
interactome  <- matrix(cInt, nrow=6, ncol=5,dimnames=list(letters[2:7],LETTERS[1:5]))

## Reduce the genetic interaction matrix to match the gene present in
## the interactome
reducediMat <- gi2Interactome(iMat, interactome)
```

---

hyperG *Hypergeometric test*

Description

a hypergeometric test for genetic interaction data.

Usage

```
hyperG(data, nbTested, universe)
```

Arguments

`data`	Matrix with 2 columns the first one corresponds to the number of interactions per pair of interacting complexes and the second one to number of tested interactions. This could be the first two columns resulting from a call to the `test2Interact` function.
`nbTested`	Number of interacting pairs
`universe`	Total Number of tested pairs

Author(s)

N. LeMeur

See Also

`phyper`

Examples

```
## Create matrix interaction x tested matrix
interact <-  c(1, 3, 2, 2, 6, 5, 2, 4, 1, 3)
tested <-  c(3, 3, 5, 4, 8, 5, 3, 4, 2, 3)
mat <- cbind(interact, tested)

## Perform test
res <- hyperG(mat, 1000, 10000)
summary(res$P)
```

---

iSummary *Summarize cellular organizational units sharing genetic interaction*

Description

Summarize the cellular organizational units sharing genetic interactions and display their GO annotation if available

Usage

```
iSummary(iMat, n=10, reverse=FALSE)
```

Arguments

iMat	Comembership matrix of genes(proteins) that linked to other genes(proteins) by any biological experiment, e.g., output of the `getInteraction` function.
n	Numeric threshold indicating the minimum number of genetic interactions that a pair of cellular organizational unit must share.
reverse	Logical, by default the function return a list of pair of cellular organizational units where the name of each element is the number of genetic interactions they share. If reverse is TRUE, the output is a vector where the values are the number of interactions and the names are the combination of the 2 cellular organizational units.

Value

The function print the result in the standard output but can also save it in variable.

If `reverse` is FALSE the output is a list of pairs of cellular organizational units where the name of each element is the number of genetic interactions they share.

If `reverse` is TRUE the output is a vector where the values are the number of interactions and the names are the combination of the 2 cellular organizational units.

Author(s)

N. LeMeur

Examples

```
data(Atong)
data(ScISIC)
data(SGA)
SLa2 <- gi2Interactome(Atong, ScISIC)
## Search for synthetic lethal interaction
compM <- getInteraction(SLa2, SGA, ScISIC)
## Display the tightly interacting pairs
largeInt <- iSummary(compM$bwMat,n=15)
```

`modelSLGI` *Permutation model for assessing synthetic genetic interactions in cellular organizational units.*

Description

Permutation model for assessing synthetic genetic interactions within and between cellular organizational units such as multi-protein complexes.

Usage

```
modelSLGI(iMat, universe, interactome,  type="intM", perm=50)
```

Arguments

`iMat`	Adjacency matrix reporting genetic interactions. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column.
`universe`	character vector of the names of the tested genes, e.g., names of the genes on the synthetic genetic array (SGA) used by Tong et al.
`interactome`	Adjacency matrix where row are genes and columns are cellular organizational units. Each entry has value 0 or 1, for absence or presence of a gene in a complex.
`type`	Character vector of value "intM" (Default) or "interactome" to either perform the test based on to the genetic interaction matrix or the interactome, respectively.
`perm`	Number of permutations to apply. Default is 50.

Value

Interaction matrix between cellular organizational units.

Author(s)

N. LeMeur

See Also

`getInteraction`

Examples

```
 data(ScISIC)
 data(Atong)
 data(SGA)
 model <-  modelSLGI(Atong, universe= SGA, interactome=ScISIC,
type="intM", perm=2)
```

`normInteraction` *Normalize a matrix of biological interactions*

Description

Normalize a square matrix of biological interactions according to the number of possible interactions between each biological complex.

Usage

```
normInteraction(data, genename, interactome)
```

Arguments

`data`	Square Matrix of biological complexes that shares one or more genes(proteins)
`genename`	Character vector of the gene names that possibly create interactions between complexes
`interactome`	Adjacency matrix where row are genes and columns are cellular organizational units. Each entry has value 0 or 1, for absence or presence of a gene in a complex, e.g., `ScISI`

Value

Square matrix of biological complexes linked by one or more interacting proteins and normalized by the possible number of interactions between each complex.

Author(s)

N. LeMeur

See Also

`getInteraction`

Examples

```
data(Atong)
data(ScISIC)
data(SGA)
SLa2 <- gi2Interactome(Atong, ScISIC)
## Search for synthetic lethal interaction
compM <- getInteraction(SLa2, SGA, ScISIC)
## Normalize
normIntComplex<- normInteraction(compM$bwMat, SGA, ScISIC)
```

plot *Graphical method to represent the result of the modelSLGI .*

Description

a plot method for siResult.

Usage

```
## S4 method for signature 'siResult':
plot(x,...)
```

Arguments

x	the siResult object to plot.
...	general commands to be sent to plot.

Details

The plot generated from a siResult object is a dotplot with the observed and expected data average of interaction represented in 2 different colors.

Author(s)

N. LeMeur

See Also

ScISI

Examples

```
data(ScISIC)
data(Atong)
data(SGA)
model <-  modelSLGI(Atong, universe= SGA, interactome=ScISIC,
type="intM", perm=2)
plot(model)
```

seqMatcherAlign *Functions to do local alignment of two sequences using EMBOSS matcher*

Description

seqMatcherAlign matches two sequences using the EMBOSS matcher program.

getAlignStats extract the statistics from the alignment result data.

Usage

```
seqMatcherAlign(pairNameV,BankIDV,seqBank)
getAlignStats(alignRes)
```

Arguments

pairNameV	a vector of gene pair names
BankIDV	a vector of the sequence IDs in the sequence Bank.
seqBank	a database of all the sequences
alignRes	object returned by seqMatcherAlign

Details

seqMatcherAlign matches the gene pair names with the sequence bank IDs and export the two sequences in to two files: seq1.new and seq2.new. Then uses system calls to run EMBOSS matcher program to align the two sequences. The result from matcher is store in file "out.matcher". seqMatcherAlign read in this file and create a R object summarize the alignment results.

getAlignStats takes the alignment result data and extract the statistics of the result in to data.frame.

Value

names	contains the names of the gene pair
results	contains the alignment statistics: the aligned total length, the number of identical match, the number of similar match, the number of gaps, and the alignment score
seq	displays the aligned sequences

Note

pairMatcherAlign use system calls to run EMBOSS matcher program. You must have EMBOSS matcher installed on your computer.

Author(s)

Z. Jiang

References

EMBOSS: The European Molecular Biology Open Software Suite (2000) Rice,P. Longden,I. and Bleasby,A. Trends in Genetics 16, (6) pp276–277

Examples

```
seq1 <- "RPHEDEKEAIDEAKKMKVPGENEDESKEEEKSQELEEAIDSKEKSTDARDEQGDEGDNEEENNEEDNENENEHTAPPALV
seq2 <- "QKYLLKKAIRNFSEYPFYAQNKLIHQQATGLILTEEEKSQELEEKIISKIKKEEHLKKINLKHDYFDLQKKYEKECEILI
seq3 <- "IHQQATGLILTKIISKIKKEEHVPGENEDLKKINLKHDYFDLQKKYEKECEILTKLSENLRKEEIENKRKEHELMEQKRF
seqBank <- list(seq1=list(seq=seq1),seq2=list(seq=seq2),seq3=list(seq=seq3))
bid <- names(seqBank)
pnames <- c("seq1","seq3")
## Not run:
ar <- seqMatcherAlign(pnames, bid, seqBank)
```

```
ar
getAlignStats(ar)

## End(Not run)
```

`sharedBy` *Find the gene pairs that share a domain.*

Description

`sharedBy` finds whether the given domain is in each of the elements of the domain list.

Usage

```
sharedBy(domainL)
```

Arguments

`domainL`	is a list, each element of the list is a vector of domains.

Details

`sharedBy` first remove all the elements with length 0 or have value 'NA'. Then apply the reverseSplit on the remaining list.

Value

A list with each element represent a domain, and the values of the element are the pairs that share this domain.

Author(s)

Z. Jiang

See Also

`reverseSplit`, `domainDist`, `getSharedDomains`

Examples

```
## Load PFAM and SMART domains shared between Tong's Synthetic lethal data
data(AtongFnDomain)
## Find pair that share identical domain
sharedBy(AtongFnDomain$SharedPfam[1:20])
```

`sharedInt` *List shared genetic interactions between genes*

Description

List shared interactions and cellular organizational units names between genes.

Usage

```
sharedInt(pairL, interactome, threshold=0)
```

Arguments

`pairL`	Dataframe with 3 columns. The first columns are the pair of genes tested i.e., the query and array genes. The third columns in a logical: TRUE when the 2 genes genetically interact and FALSE when they do not.(see `AtongPair` dataset as example)
`interactome`	Adjacency matrix where row are gene names and columns are cellular organizational units names. Each entry has value 0 or 1, for absence or presence of a gene in the complex.
`threshold`	Numeric. Indicate the minimum number of interactions that 2 genes must share

Value

The return value is a list. Each element of the list has for name 2 genes that genetically interact. Each element of the list corresponds to the list of cellular organizational units where the interacting genes are found (independently or together).

Author(s)

N. LeMeur

Examples

```
## Synthetic genetic interactions
dat <- data.frame("query" = LETTERS[1:5], "array" = LETTERS[2:6], "interact" = as.logical
## interactome
interA <- matrix(sample(c(0, 1), 30,TRUE), nrow=6, ncol=5,dimnames = list(LETTERS[1:6], l

sharedInt(dat, interA, threshold=1)
```

siResult-class — *A class for representing the result of the SLGI graph permutation model.*

Description

A class for representing the result of the `modelSLGI` function.

Slots

`Observed`: Return a `"numeric"` vector: the observed number of synthetic genetic interactions between components of one or two cellular organizational units

`Expected`: Return a matrix: the expected number of synthetic genetic interactions between components of one or two cellular organizational units

Methods

`plot` Graphical representation of the permutation model result

`compare` Summarizes the result of the `modelSLGI` function

Author(s)

N. LeMeur

See Also

`modelSLGI`,`plot`

Examples

```
## apply a permutation model
 data(ScISIC)
 data(Atong)
 data(SGA)
 model <-  modelSLGI(Atong, universe= SGA, interactome=ScISIC,
  type="intM", perm=2)

model
```

test2Interact — *Summarize genetic interactions within or between cellular organizational units*

Description

Summarize the genetic interactions within one cellular organizational unit or between 2 cellular organizational units.

Usage

```
test2Interact(iMat, tMat, interactome)
```

Arguments

iMat	Genetic interaction matrix. Each entry has value 0 or 1, representing positive or negative interaction of corresponding pairs of row and column, respectively.
tMat	Adjacency matrix of tested object. Each entry has value 0 or 1, representing the fact that the corresponding pairs of row and column have been tested for interaction or not.
interactome	Adjacency matrix where row are gene names and columns are cellular organizational units names. Each entry has value 0 or 1, for absence or presence of a gene in the complex.

Value

the return value is a data.frame with 6 columns.

unit1, unit2	cellular organizational units tested and interacting
tested	Number of interactions tested between unit1 and unit2
interact	Number of interactions found between unit1 and unit2
sizeC1, sizeC2	Number of genes in unit1 and unit2

Author(s)

N. LeMeur

Examples

```
set.seed(123)
##Create the interactome
cInt <- sample(c(0,1),30, TRUE)
interactome  <- matrix(cInt, nrow=6, ncol=5,dimnames=list(letters[2:7],LETTERS[1:5]))

## Create cellular organizational units interaction matrix
gInt <- sample(c(1:8), 25, TRUE)
gInt  <- matrix(gInt, nrow=5, ncol=5, dimnames=list(LETTERS[1:5],LETTERS[1:5]))

## All interactome tested
gTest <- matrix(sample(c(0:3), 25, TRUE), nrow=5, ncol=5)
gTested <- gInt+gTest

val <- test2Interact(iMat=gInt, tMat=gTested, interactome=interactome)
```

---

topInteraction *Extract interacting biological complexes*

Description

Extract the top X interacting biological complexes.

Usage

```
topInteraction(data,top=10)
```

Arguments

data	Square matrix of biological complexes that shares one or more genes(proteins)
top	Interger that represents the percentage of interacting complexe

Value

Data frame of biological complexes that interact. The first two columns are the cellular organizational units names and the third column indicates the number of interactions.

Author(s)

N. LeMeur

Examples

```
data(Atong)
data(ScISIC)
data(SGA)
SLa2 <- gi2Interactome(Atong, ScISIC)
## Search for synthetic lethal interaction
compM <- getInteraction(SLa2, SGA, ScISIC)
top10Interaction<- topInteraction(compM$bwMat,top=10)
```

---

twoWayTable *Generate two-way table for genetic interaction data*

Description

Generate two-way table from a vector of genetic interaction status and a vector of the pairs that share a functional domain.

Usage

```
twoWayTable(var1, var2idx)
```

Arguments

var1	Vector of the status of the first property.
var2idx	Vector of the index in var1 that have the second property.

Details

Calculates the count numbers from the given vectors. Then put them into a matrix format.

Value

A two-way contingency table of genetic interaction and whether sharing a functional domain.

Author(s)

Z. Jiang

See Also

sharedBy, getUniquePairs

Examples

```
var1 <- c(0,1,1,0,0,0,1,0,1,1)
var2idx <- c(3,5,7)
twoWayTable(var1,var2idx)

data("AtongFnDomain")
pf <- Biobase::reverseSplit(AtongFnDomain$SharedPfam)
idx <- which(rownames(AtongFnDomain$pairs) %in% pf$PF00478)
twoWayTable(AtongFnDomain$pairs[,"interact"],idx)
```

---

withinComplex *Search for protein co-membership within complexes.*

Description

Search for protein co-membership within one (or more) complex(es).

Usage

```
withinComplex(data,interactome)
```

Arguments

data	Binary matrix of genes(proteins) linked to other genes(protein) by any biological experiment
interactome	Binary matrix composed of genes (rows) and biological complexes (columns) ScISI

Value

Matrix of genes(proteins) co-member of one or more biological complexes.

Author(s)

N. LeMeur

See Also

byComplex

Examples

```
data(Atong)
data(ScISIC)
coMember <- withinComplex(Atong, ScISIC)
table(coMember)
```

Index