simulatorAPMS

April 20, 2011

`APComEX` *An example of an ISI using apComplex on simEX*

Description

Using simEX, vBaitsEX, no deformed nor sticky baits, APComEX is an example output of apComplex. This is the bipartite graph matrix representing proteins (rows) and protein complexex (columns). Each row is indexed by a particular protein and similarly for each column for complexes. An entry of 1 means the protein indexing the row is in the complex indexing the column.

Usage

```
data(APComEX)
```

Format

The format is: num [1:28, 1:6] 1 0 1 0 0 1 0 0 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:28] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:6] "Complex1" "Complex2" "Complex3" "Complex4" ...

Details

The APComEX is an example ISI for use as an example to try the simulatorAPMS package.

Examples

```
data(APComEX)
```

`applyDeformationError` *Function that handles deformed bait proteins*

Description

This function simulates False Negative (FN) observations on those bait proteins known to have been deformed during the AP-MS process.

Usage

```
applyDeformationError(foundPrey, rateDeform)
```

Arguments

`foundPrey`	A character vector of proteins names that interact with the a particular deformed protein P
`rateDeform`	The rate at which the deformation in P produces false negative observations within the AP-MS technology

Details

If there is a non-trivial number of deformed bait proteins, the function `runSimulators` calls the intermediary function `simulatorD`. This latter function controls `applyDeformationError` by examining the arguments and parameters. For each deformed bait, `simulatorD` calls this function to generate deformation errors.

This is essentially a function for recording more FN interactions among baits and prey. The function takes in a bait that is known to have some deformation as a result of the experimentation.

In essence, the bait loses some capacity to pull other proteins with which it would normally interact.

Value

The character vector, missedPrey:

The return value missedPrey is a vector which is a subset of the vector foundPrey. This subset will be the proteins which will be included as FN observations due to the systematic errors via the deformation of protein P.

Author(s)

T. Chiang

See Also

`applyFNErrors`, `runSimulators`

Examples

```
data(TSNMatrix)
exampleFoundPrey <- rownames(TSNMatrix)[1:50]
applyDeformationError(exampleFoundPrey, 0.5)
```

applyFNErrors *Function that generates stochastic False Negative Errors*

Description

This function generates the stochastic False Negative (FN) observations for each bait protein used in the AP-MS simulation.

Usage

```
applyFNErrors(foundPrey, rateFN)
```

Arguments

foundPrey	A character vector of proteins.
rateFN	A scalar within the unit interval indicating the rate at which stochastic FN obersevations are observed.

Details

This function tries to simulate the stochastic nature of the AP-MS technology. The overall simulation function, runSimulators, will call the simulator function on each bait protein along with some uniform random rate at which the technology is estimated to generate FN observations.

Value

A character vector, proteinFN:

The proteinsFN is a vector consisting of a subset of the foundPrey proteins which will serve as the FN observations for bait protein B.

Author(s)

T. Chiang

See Also

applyFPErrors,applyDeformationError

Examples

```
data(TSNMatrix)
exampleFoundPrey <- rownames(TSNMatrix)[1:50]
applyFNErrors(exampleFoundPrey, 0.3)
```

`applyFPErrors` *Function to generate the False Postive Observations*

Description

This function generates the stochastic False Positive (FP) observations for each bait protein used in the AP-MS simulation.

Usage

```
applyFPErrors(notFoundProt, rateFP)
```

Arguments

`notFoundProt`	A vector of proteins
`rateFP`	A scalar within the unit interval indicating the rate at which stochastic FN obersevations are observed.

Details

This function tries to simulate the stochastic nature of the AP-MS technology. The overall simulation function, `runSimulators`, will call the `simulator` function on each bait protein along with some uniform random rate at which the technology is estimated to generate FP observations.

Value

A character vector of proteins, proteinFP:

The proteinsFP is a vector consisting of a subset of the notFoundProt proteins which will serve as the FP output for bait B

Author(s)

T.Chiang

See Also

`applyFNErrors`, `applyStickyError`

Examples

```
data(TSNMatrix)
exampleNotFoundPrey <- rownames(TSNMatrix)[50:100]
applyFPErrors(exampleNotFoundPrey, 0.04)
```

`applyStickyError` *Function that takes sticky proteins and calculates FP's*

Description

This function simulates False Positive (FP) observations on those bait proteins known to have affinity towards a large number of prey proteins during the AP-MS process.

Usage

```
applyStickyError(notFoundProt, rateSticky)
```

Arguments

`notFoundProt`	A character vector of proteins that have no known interactions with the bait protein B.
`rateSticky`	The rate at which bait B attracts other proteins ands records FP observations due to its systematic affinity to interact with other proteins.

Details

If there is a non-trivial number of sticky bait proteins, the function `runSimulators` calls the intermediary function `simulatorS`. This latter function controls `applyStickyError` by examining the arguments and parameters. For each sticky bait, `simulatorS` calls this function to generate un-natural interactions.

This is essentially a function for recording more FP interactions among baits and non-prey. The function takes in a bait that is known to have some high level of finding prey as a result of the experimentation.

In essence, the bait attracts and binds to other proteins with which it would normally not interact.

Value

The character vector, ProtInErr:

The return value, ProtInErr, is a vector consisting of a subset of proteins from the set notFoundProt. This subset will be those proteins which form systematic FP observations for the simulation.

Author(s)

T. Chiang

See Also

`applyFPErrors`, `applyDeformationError`

Examples

```
data(TSNMatrix)
exampleNotFoundPrey <- rownames(TSNMatrix)[1:50]
applyStickyError(exampleNotFoundPrey, 0.9)
```

`compIndep` *This function determines probability distribution independence between two bi-partite graph incidence matrices.*

Description

This function calculates the probability that a protein is in Cluster C-i and the probability that a protein is in cluster K-j and then the joint probability that the protein is in both C-i and K-j. The function compares the joint distributions with the marginals.

Usage

```
compIndep(TSNMat, erMat, intersectMat)
```

Arguments

`TSNMat`	The first bi-partite graph matrix; usually a representation of the true state of nature.
`erMat`	The second bi-partite graph mattrix; usually an estimate of the true state of nature based on some error model.
`intersectMat`	A matrix of of integers. The (i,j)th entry is the cardinality of the intersection between the i-th complex of TSNMat and the j-th complex of erMat.

Value

Matrix of real numbers:

The return value is a matrix of real numbers that account for how independent the marginals are from the joint distribution of protein membership to complexes.

Author(s)

Tony Chiang

Examples

```
data(simEX)
data(APComEX)
CC = runCompareComplex(simEX, APComEX)
compIndep(simEX, APComEX, CC[[1]])
```

filteredProt *A data file containing unusable proteins*

Description

This data file contains a character vector of genes known to either be sticky or otherwise unusable in the AP-MS technologies.

Usage

```
data(filteredProt)
```

Format

The format is: chr "filteredProt"

Examples

```
data(filteredProt)
```

fullSimEX *In Silico Interactome*

Description

This is an ISI with proteins not participating in any protein complexes append to the bipartite graph. Usually, these non-participating proteins are not recorded in the ISI, but are needed in experimentation and simulation.

Usage

```
data(fullSimEX)
```

Format

The format is: num [1:35, 1:10] 1 0 0 0 0 0 0 0 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:35] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:10] "MBME1" "MBME2" "MBME3" "MBME4" ...

Details

The fullSimEX is the ISI that should be acted upon by the simulation methods of simulatorAPMS. The non-participating proteins will be key in the false positive observations.

Examples

```
data(fullSimEX)
```

missedProtEX *Vector of Protein not recorded in simEX*

Description

Vector of Proteins used to append to the example ISI

Usage

```
data(missedProtEX)
```

Format

The format is: chr [1:10] "YPR181C" "YML014W" "YNL042W" "YPR180W" "YHR050W" "YIL069C" ...

Details

These are example proteins that are present in the organism or cell under some conditions, but do not share co-membership with any other proteins, and hence were not recorded in the ISI

Examples

```
data(missedProtEX)
```

missedProtHMSPCI *The proteins not recorded by the ISI TSNMatrix*

Description

The list of proteins not recorded in the ISI TSNMatrix because they were non-participatory in any complexes.

Usage

```
data(missedProtHMSPCI)
```

Format

The format is: chr [1:5303] "YPR181C" "YML014W" "YNL042W" "YPR180W" "YHR050W" "YIL069C" ...

Examples

```
data(missedProtHMSPCI)
```

PPIEX *Protein-protein co-membership graph matrix*

Description

From the matrix of ISI, we can derive the PPCM matrix by the multiplication: ISI

Usage

```
data(PPIEX)
```

Format

The format is: num [1:35, 1:35] 1 0 0 0 0 0 0 0 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:35] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:35] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ...

Details

This is an example PPCM where both the rows and the columns are indexed by the proteins present in the organism or cell. Here an entry of 1 means that two proteins share co-membership in some complex.

Examples

```
data(PPIEX)
```

recordSticky *This function records proteins known to be sticky in the simulation of AP-MS.*

Description

In the actual AP-MS technology, sticky proteins are identified and then removed from the process so sticky proteins do not influence the outcome of the protein-protein co-membership matrix nor the estimate from complex composition algorithms such as apComplex.

Usage

```
recordSticky(simMat, vSticky, vBaits)
```

Arguments

simMat	An adjcency matrix: the rows are indexed by bait proteins and the colunms are indexed by prey proteins
vSticky	The pre-determined vector of sticky proteins
vBaits	A character vector of the proteins used as baits

Details

The recordSticky function will return both the sticky proteins that are put into the function at the beginning as well as proteins which pull down a large number of prey proteins.

Value

A character vector:

The return value is a vector of all the proteins that are considered to be sticky in the simulation of AP-MS

Author(s)

Tony Chiang

Examples

```
data(fullSimEX)
data(vBaits)
recordSticky(fullSimEX, vSticky = vBaits[2], vBaits = vBaits[2:4])
```

---

runAPComplex *A function that calls the complex estimation function, apComplex on the simulated output from runSimulators*

Description

The runAPComplex function takes the simulated AP-MS experimental data (represented as an adjacency matrix) and deletes the proteins from this matrix that is either not used as a bait nor ever seen as a prey in the experiment. After the deletion, the apComplex is called on the resulting matrix.

Usage

```
runAPComplex(errorModel, vBaits)
```

Arguments

errorModel	An adjacency matrix: the rows are indexed by the baits and the columns are indexed by all the proteins found in the organism or cell under certain conditions
vBaits	A character vector: baits used in the simulated AP-MS experiment

Details

This function runs the estimation algorithm apComplex. The algorithm accomplishes two things: it makes decision on if a directed edge is a True Positive (TP) or False Positive (FP) and if a missing directed edge is a True Negative (TN) or False Negative; after total reciprocity is decided, apComplex estimates the protein complex bi-partite graph matrix.

Value

The return value errorComplex is the Bi-partite Graph Incidence Matrix estimated by apComplex.

Author(s)

Tony Chiang

Examples

```
#data(TSNMatrix)
#data(vBaits)
#data(missedProtEX)
#sim = runSimulators(TSNMatrix, vBaits, vBaits[2], vBaits[5], 0.0003,
#0.1, 0, 0, missedProtEX, 357)
#runAPComplex(sim, vBaits)
```

---

runCompIndep *Function checks the arguments and parameters for compIndep.*

Description

Makes sure that the same set of proteins are used for both each of the first two arguments.

Usage

```
runCompIndep(TSNMat, estMat, intersectMat)
```

Arguments

TSNMat	Incidence Matrix for the True State of Nature (TSN) bipartite graph
estMat	Incidence Matrix for the estimate bipartite graph
intersectMat	A matrix of integers: the (i,j)th entry of this matrix is the cardinality of the intersection between the i-th complex of TSN and the j-th complex of the estimate.

Details

See above

Value

A numeric matrix:

Matrix that calculates the probablistic independence that a protein will be in both complex C-i of the TSN and K-j in the estimate.

Author(s)

T. Chiang

Examples

```
data(simEX)
data(APComEX)
CC = runCompareComplex(simEX, APComEX)
runCompIndep(simEX, APComEX, CC[[1]])
```

runSimEX — *The output of the simulation with fullSimEX as the input*

Description

The runSimEX is a matrix representing PPCM graph with error. It is the output of the simulation function.

Usage

```
data(runSimEX)
```

Format

The format is: num [1:6, 1:35] 1 0 0 0 0 1 0 1 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:6] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:35] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ...

Details

This is first input for runAPComplex function

Examples

```
data(runSimEX)
```

runSimulators — *Function that sets up the ISI et al to run the simulations*

Description

The runSimulators function is the primary function for the simulation tool. It first takes in an in the incidence matrix of the silico interactome's (ISI) bipartite graph representation. Since not all the proteins are documented in the ISI, it also appends all proteins found in the organism to the ISI. From there, we derive the Protein-Protein Co-Membership (PPCM) matrix. We then only take the rows indexed by baits, and apply all the simulated errors to this matrix. The result is an error model simulation of an AP-MS experiment.

Usage

```
runSimulators(TSNMat, vBaits, vDeform, vStky, rateFP, rateFN, rateD, rateS, miss
```

Arguments

TSNMat	Incidence Matrix of ISI's bipartite graph representation
vBaits	The vector of baits used in the simulated AP-MS experiment
vDeform	The vector of deformed baits
vStky	The vector of sticky baits
rateFP	Rate at which AP-MS returns FP's
rateFN	Rate at which AP-MS returns FN's
rateD	Rate(s) at which the deformed baits return FN's
rateS	Rate(s) at which the sticky baits return FP's
missedProt	The vector of proteins that have trivial co-membership and not recorded by the ISI
seedIn	Seed used in the random generator which allows the user to set the seed for reproducible experiments

Details

The details of the function are as follows. TSNMat and vBaits have been described. The vStky and vDeformed are baits that account for systematic errors called sticky protein errors and deformed protein error, and the rateS and rateD are the rates for the respective systematic errors. The missedProt are those who are in no protein complex and thus not recorded in the ISI, but needed for simulation.

Value

The return value for this function is a rectangular matrix with rows indexed by the baits and the columns indexed by all proteins in the organism. This matrix will differ from the PPCM matrix in that it will have FN/FP's embedded from the simulated error functions.

Author(s)

Tony Chiang

See Also

simulator

Examples

```
data(simEX)
data(vBaits)
data(missedProtEX)
runSimulators(simEX, vBaits[2:4], vBaits[2], vBaits[3], 0.05, 0.1, 0, 0, missedProtEX, 35
```

simEX *An example ISI*

Description

An example of a bipartite graph matrix of some in silico interactome.

Usage

```
data(simEX)
```

Format

The format is: num [1:25, 1:10] 1 0 0 0 0 0 0 0 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:25] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:10] "MBME1" "MBME2" "MBME3" "MBME4" ...

Details

The matrix is indexed by proteins for the rows and complexes for the colunms where incidence is defined in the obvious way.

Examples

```
data(simEX)
```

simulatorD *Function that simulates deformation errors*

Description

The simulatorD function is the function that operates on deformed baits. It takes in a vector of deformed baits, and for each bait B, it finds all the proteins interacting with it in the ISI. When it has calculated this set of proteins, it calls the applyDeformationErrors function and gets the subset of proteins as the FN hits. It changes these interactions from 1 to 0 in the PPCM matrix creating the error model.

Usage

```
simulatorD(deformedBaits, rateD, TSNMat, exMat)
```

Arguments

deformedBaits	Vector of deformed baits
rateD	Either a single rate or a vector of rates which the experiment will record FN's
TSNMat	The PPCM matrix
exMat	The experimental matrix that carries the FN/FP

Details

One interesting aspect of this function is that the rateD can either be a vector of the same length as the vector of deformed baits, or it can be a single rate which is applied to all the deformed baits. The function uses each rate to decide how deformed the bait has become and will create FN observations based on this data.

Value

The return value of this function is an PPCM matrix with error to reflect the FN's that have been induced by applyDeformationErrors

Author(s)

Tony Chiang

See Also

applyDeformationError

Examples

```
data(MBMEcHMSPCI)
data(HMSPCI)
vBaits = rownames(MBMEcHMSPCI)[1:10]
PPI = MBMEcHMSPCI
mode(PPI) = "logical"
mode(PPI) = "numeric"
exMat = PPI[vBaits[2:4],]
deformedBaits = vBaits[3]
rateD = 0.9
test = simulatorD(deformedBaits, rateD, MBMEcHMSPCI, exMat)
test[3,]
```

---

simulator *Function sets up inputs to call all FN/FP errors on the PPCM*

Description

For each bait, the simulator function first partitions all the proteins into two groups: (1) those that share co-membership with the bait in the ISI and (2) those that do not. From the first group, simulator calls the function applyFNErrors to generate the FN's for the error model; it calls applyFPErrors on the second group.

Usage

```
simulator(TSNMat, exMat, bait, rateFP, rateFN)
```

Arguments

TSNMat	The PPCM matrix
exMat	The error model matrix
bait	A bait protein used in the simulated experiment
rateFP	The rate for which FP's are seen in the experiment
rateFN	The rate for which FN's are seen in the experiment

Details

See description

Value

The return value is an directed graph matrix of PPCM with the embedded FN/FP's incorporated.

Author(s)

Tony Chiang

See Also

runSimulators

Examples

```
data(simEX)
data(vBaits)
PPCM = simEX
mode(PPCM) = "logical"
mode(PPCM) = "numeric"
estMat = PPCM[vBaits[2:4],]
rateFP = 0.025
rateFN = 0.20
simulator(simEX, estMat, vBaits[2], rateFP, rateFN)[2,]
```

---

simulatorS *Function that calculates the FP sticky errors*

Description

The simulatorS function is the function that operates on sticky baits. It takes in a vector of sticky baits, and for each bait B, it finds all the proteins not interacting with it in the ISI. When it has calculated this set of proteins, it calls the applyStickyErrors function and gets the subset of proteins as the FP hits. It changes these interactions from 0 to 1 in the PPCM matrix creating an directed graph PPCM matrix with error.

Usage

```
simulatorS(sticky, rateS, exMat)
```

Arguments

sticky	A vector of baits known to be sticky
rateS	A single rate of a vector of rates which the experiment will record FP's due to the sticky proteins
exMat	The experimental matrix that carries the FP/FN

Details

One interesting aspect of this function is that the rateS can either be a vector of the same length as the vector of sticky baits, or it can be a single rate which is applied to all the sticky baits. The function calls applyStickyErrors on the prescribed sticky baits to assign FP observations onto them. This FP rate is different than the general rateFP and is determined by rateS

Value

The return value of this function is the adjusted PPCM matrix with error to reflect the FP's that have been induced by applyStickyErrors

Author(s)

Tony Chiang

See Also

simulatorS

Examples

```
data(MBMEcHMSPCI)
data(HMSPCI)
vBaits = rownames(MBMEcHMSPCI)[1:10]
PPI = MBMEcHMSPCI
mode(PPI) = "logical"
mode(PPI) = "numeric"
exMat = PPI[vBaits,]
stickyBaits = vBaits[2:3]
rateS = 0.9
#test = simulatorS(stickyBaits, rateS, exMat)
#test[2:3,]
```

TSNMatrix *ISI*

Description

An in silico interactome created by using apComplex on the data set of Ho et al.

Usage

```
data(TSNMatrix)
```

Format

The format is: num [1:1578, 1:242] 1 0 0 0 0 0 0 0 0 0 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:1578] "YAL015C" "YAL017W" "YAL021C" "YAL036C" ... ..$: chr [1:242] "MBME1" "MBME2" "MBME3" "MBME4" ...

Source

~~ reference to a publication or URL from which the data were obtained ~~

References

~~ possibly secondary sources and usages ~~

Examples

```
data(TSNMatrix)
```

---

vBaitsEX *Example Vector of Baits*

Description

This is a vector of proteins that will be used in the AP-MS technology

Usage

```
data(vBaitsEX)
```

Format

The format is: chr [1:6] "YAL015C" "YAL017W" "YAL021C" "YAL036C" "YAR003W" "YAR007C"

Details

Vector of pre-selected baits used in the simulation.

Examples

```
data(vBaitsEX)
```

vBaits *The baits used in HMSPCI*

Description

The actual proteins used as baits by Ho et al

Usage

```
data(vBaits)
```

Format

The format is: chr [1:493] "YAL015C" "YAL017W" "YAL021C" "YAL036C" "YAR003W" "YAR007C" ...

Examples

```
data(vBaits)
```

vDeformedEX *Example vector of deformed baits*

Description

Example vector of deformed baits used in the Vignette

Usage

```
data(vDeformedEX)
```

Format

The format is: chr "YAL021C"

Examples

```
data(YAL021C)
```

vDeformed *A deformed protein*

Description

An example of an input for a defored protein

Usage

```
data(vDeformed)
```

Format

The format is: chr "YAL021C"

Examples

```
data(vDeformed)
```

vDeformEX *Doformed Protein*

Description

Another deformed protein

Usage

```
data(vDeformEX)
```

Format

The format is: chr "YAL017W"

Examples

```
data(YAL017W)
```

vStickyEX *Sticky Proteins*

Description

An example of sticky protein

Usage

```
data(vStickyEX)
```

Format

The format is: chr "YAL017W"

Examples

```
data(YAL017W)
```

Index