joda

October 25, 2011

`damage` *Knockout data for three regulators: ATM, RelA and p53 in two different*

Description

Knockout expression data for the kinase ATM and transcription factors p53 and RelA (Elkon et al., 2005), as well as knowledge about their targets and mutual regulatory relations in two different cell populations: healthy and damaged cells.

Usage

```
data(damage)
```

Format

data.healthy (data.frame): 8463x3, beliefs.healthy (list): 2, model.healthy (matrix): 3x3, data.damage (data.frame): 8463x3, beliefs.damage (list): 1,

model.damage (matrix): 3x3.

Details

The `data.healthy` dataset contains log gene expression ratios for the regulator knockouts versus control. For the genes that are known to be targeted by RelA and genes that are targeted by p53 in normal conditions, `beliefs.healthy` contains certainties (beliefs) that those targets are differentially expressed upon their regulator's knockdown. The `model.healthy` matrix represents mutual signaling relations between the regulators in the healthy cells (here the model reflects that no regulator influences others). The `data.damage` dataset contains log gene expression ratios for the regulator knockouts upon treatment with neocarzinostatin versus treatment with neocarzinostatin alone. For the genes that are known to be targeted by p53 in the damaged cells, `beliefs.damage` contains certainties (beliefs) that they are differentially expressed upon the knockdown of p53. The `model.damage` matrix represents mutual signaling relations between the regulators in the damaged cells (here the model reflects ATM signaling down to RelA and p53).

Author(s)

Ewa Szczurek

References

Elkon R, Rashi-Elkeles S, Lerenthal Y, Linhart C, Tenne T, Amariglio N, Rechavi G, Shamir R, Shiloh Y. Dissection of a DNA-damage-induced transcriptional network using a combination of microarrays, RNA interference and computational promoter analysis. Genome Biol. 2005;6(5):R43.

Examples

```
data(damage)
str(data.damage)
str(beliefs.damage)
print(model.damage)
```

`deregulation.p.values` *Calculating deregulation p-values using resampling method.*

Description

Deregulation p-values based on deregulation scores. They are calculated as fraction of permutations that give more extreme deregulation scores than for original data.

Usage

```
deregulation.p.values(data.1, beliefs.1, model.1, data.2, beliefs.2, model.2, N=
```

Arguments

`data.1, data.2`	Matrices of log expression ratios perturbation vs control, for the genes (rows), in the perturbations of the regulators (columns). See `differential.probs` for more details.
`beliefs.1, beliefs.2`	Lists of beliefs. See `differential.probs` for more details.
`model.1, model.2`	Pathway topologies. See `differential.probs` for more details.
`N`	A number of replications used to calculate p-values
`verbose`	When TRUE, the execution prints informative messages

Details

The deregulation p-values are calculated as fraction of permutations that give more extreme deregulation scores than for original data.

Value

A list with two matrices. This p-values in the slot `deregulation.p.values` and with the original deregulation scores in the slot `deregulationOrg`.

Author(s)

Ewa Szczurek

References

http://joda.molgen.mpg.de

See Also

`differential.probs`, `regulation.scores`, `regulation.scores`

Examples

```
## Not run:
# Step 1
library(joda)
data(damage)

deregulationObj = deregulation.p.values(data.healthy, beliefs.healthy, model.healthy, dat
boxplot(deregulationObj$deregulation.p.values)

## End(Not run)
```

`deregulation.scores` *Calculating deregulation scores.*

Description

Deregulation scores quantify the extent to which the regulatory effect of each regulator changes between the two compared cell populations.

Usage

```
deregulation.scores(reg.scores1, reg.scores2,verbose)
```

Arguments

`reg.scores1`	A matrix of regulation scores of the genes (rows) for the regulators (columns), compued with the `regulation.scores` function. Given for the first cell population.
`reg.scores2`	The same as reg.scores1 but given for the second cell population.
`verbose`	When TRUE, the execution prints informative messages

Details

The deregulation scores are computed by subtracting reg.scores1 from reg.scores2.

Value

A matrix with columns for the regulators, rows for the genes, and entries giving the deregulation scores.

Author(s)

Ewa Szczurek

References

http://joda.molgen.mpg.de

See Also

`differential.probs`, `regulation.scores`

Examples

```
data(damage)

# Step 1
# Get the probabilities of differential expression
# for the knockout of ATM in the healthy cells
probs.healthy.ATM= differential.probs(data.healthy[,"ATM",FALSE], NULL)

# Get the probabilities of differential expression
# for the knockout of ATM in the damaged cells
probs.damage.ATM= differential.probs(data.damage[,"ATM",FALSE], NULL)

# Step 2
# Regulation scores for a dataset with only one regulator
# equal the signed probabilities

# Step 3
# Get the deregulation scores
deregulation.ATM= deregulation.scores(probs.healthy.ATM, probs.damage.ATM, TRUE)

## Not run:
# Step 1
probs.healthy= differential.probs(data.healthy, beliefs.healthy)
probs.damage= differential.probs(data.damage, beliefs.damage)

# Step 2
regulation.healthy= regulation.scores(probs.healthy, model.healthy)
regulation.damage= regulation.scores(probs.damage, model.damage)

# Step 3
deregulation= deregulation.scores(regulation.healthy, regulation.damage, TRUE)

## End(Not run)
```

`differential.probs` *Calculating probabilities of differential expression in perturbation*

Description

Returns probabilities of differential expression for genes under perturbation of a set of regulators. Takes as input perturbation data and beliefs about known genes.

Usage

```
differential.probs(data, beliefs, verbose, plot.it)
```

Arguments

`data`	A matrix of log expression ratios perrturbation vs control, for the genes (rows), in the perturbations of the regulators (columns). The data has to have row and colnames specified by the user.
`beliefs`	A list with names being a subset of the regulators (i.e., the names of `beliefs` have to be a subset of the columns of the `data`). Each list entry for a given regulator is a matrix with rows corresponding to the genes that are known to respond in some way to the perturbation of this regulator. The rownames of the matrix must be a subset of the rows in the `data`. The matrix can have either two or three columns. Each row is a distribution over the differential and unchanged cluster (2 columns) or over down, up-regulated and unchanged cluster of genes (3 columns). This distribution reflects the certainties with which a gene that corresponds to this row belongs to each of those clusters.
`verbose`	When TRUE, for each regulator and its perturbation data, the execution prints out the parameters of the fitted model(s), indicating which components are differential and which are unchanged.
`plot.it`	When TRUE, for each regulator and its perturbation data, the execution plots the Gaussian components of the fitted model(s), indicating which components are differential and which are unchanged.

Details

For each regulator, a belief-based mixture model is fitted to the observations in the `data`. The fitted models have the number of model components equal to the number of columns in the corresponding beliefs. If no beliefs are given, unsupervised two-component mixture modeling is applied.

Value

A matrix with columns for the regulators, rows for the genes, and entries giving the signed probabilities of differential expression.

Author(s)

Ewa Szczurek

References

http://joda.molgen.mpg.de

See Also

`regulation.scores`, `deregulation.scores`

Examples

```
data(damage)

# Get the probabilities of differential expression
# for the knockout of p53 in healthy cells
probs.healthy.p53= differential.probs(data.healthy[,"p53",FALSE],
beliefs.healthy["p53"], TRUE,TRUE)
```

```
# Get the probabilities of differential expression
# for the knockout of Ste12 under pheromone treatment
library(bgmm)
data(Ste12)
data=as.matrix(Ste12Data)
colnames(data)="Ste12"
beliefs=list(Ste12=Ste12Beliefs)
diff.p=differential.probs(data,beliefs,TRUE,TRUE)

## Not run:
probs.healthy= differential.probs(data.healthy, beliefs.healthy, TRUE,TRUE)

## End(Not run)
```

`regulation.scores` *Calculating regulation scores.*

Description

Regulation scores reflect the actual regulatory influence of the regulators on the genes. For each regulator, these are probabilities of differential expression averaged over all perturbation experiments that affect the regulator.

Usage

```
regulation.scores(probs, model, verbose)
```

Arguments

`probs`	A matrix of probabilities of differential expression of the genes (rows) under perturbations of regulators (columns). Obtained with the `differential.probs` function
`model`	A pathway `model` is a matrix with rows and columns equal to the names of the regulators (i.e., the columns of probs). Each `model` has an entry 1 where the regulator in the corresponding row influences the regulator in the corresponding column. A transitive closure of the input `model` is computed to get information about the experiments affecting each regulator.
`verbose`	When TRUE, the execution prints informative messages

Value

A matrix with columns for the regulators, rows for the genes, and entries giving the regulation scores.

Author(s)

Ewa Szczurek

References

http://joda.molgen.mpg.de

See Also

differential.probs, deregulation.scores

Examples

```
data(damage)

# Get the probabilities of differential expression
# for the knockouts of ATM, RelA and p53 in healthy cells
probs.healthy= differential.probs(data.healthy, beliefs.healthy)
#Get the regulation scores for ATM, RelA and for p53
regulation.healthy= regulation.scores(probs.healthy, model.healthy, TRUE)
```

Index