Package 'netresponse'

October 9, 2013

Type Package

Title NetResponse: functional network analysis

Version 1.12.0

Date 2013-03-09

Author Leo Lahti, Olli-Pekka Huovilainen, Antonio Gusmao and Juuso Parkkinen

Maintainer Leo Lahti <leo.lahti@iki.fi>

Description Algorithms for functional network analysis. Includes an implementation of a variational Dirichlet process Gaussian mixture model for nonparametric mixture modeling.

License GPL (>=2)

Depends R (>= 2.15.1), dmt, igraph0, infotheo, ggplot2, graph, mclust,methods, minet, parallel, qvalue, RColorBrewer, reshape,Rgraphviz

URL http://netpro.r-forge.r-project.org/

LazyLoad yes

biocViews CellBiology, Clustering, GeneExpression, Genetics,NetworkAnalysis, GraphsAndNetworks, DifferentialExpression,Microarray, Transcription

Collate
'AllClasses.R' 'AllGenerics.R' 'deprecated.R''detect.responses.R' 'firstlib.R' 'toydata.R''ICMg.internals.R' 'information.
accessors.R' 'order.responses.R''plot-
methods.R' 'read.network.R' 'response2sample.R''response.enrichment.R' 'sample2response.R' 'show-
methods.R''utilities.R' 'vdp.mixt.R' 'pkg-
package.R' 'ICMg.R''visualization.R' 'check.matrix.R' 'check.network.R''filter.netw.R' 'get.mis.R' 'ICMg.get.comp.mem

R topics documented:

`netresponse-package` *NetResponse: Global modeling of transcriptional responses in interaction networks*

Description

Global modeling of transcriptional responses in interaction networks.

Details

Package:	netresponse
Type:	Package
Version:	See sessionInfo() or DESCRIPTION file
Date:	2011-02-03
License:	GNU GPL >=2
LazyLoad:	yes

Author(s)

Leo Lahti, Olli-Pekka Huovilainen, Antonio Gusmao and Juuso Parkkinen. Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for details.

Examples

```
# Load the package
library(netresponse)

# Define parameters for toy data
Ns <- 200  # number of samples (conditions)
Nf <- 10   # number of features (nodes)
feature.names <- paste("feat", seq(Nf), sep="")
sample.names  <- paste("sample", seq(Ns), sep="")

# random seed
set.seed( 123 )
```

```
# Random network
netw <- pmax(array(sign(rnorm(Nf^2)), dim = c(Nf, Nf)), 0)
# in pathway analysis nodes correspond to genes
rownames(netw) <- colnames(netw) <- feature.names

# Random responses of the nodes across conditions
D <- array(rnorm(Ns*Nf), dim = c(Ns,Nf), dimnames = list(sample.names, feature.names))
D[1:100, 4:6]  <- t(sapply(1:(Ns/2),function(x){rnorm(3, mean = 1:3)}))
D[101:Ns, 4:6] <- t(sapply(1:(Ns/2),function(x){rnorm(3, mean = 7:9)}))

# Calculate the model
model <- detect.responses(D, netw)

# Subnets (each is a list of nodes)
get.subnets( model )

# Retrieve model for one subnetwork
# means, standard devations and weights for the components
inds <- which(sapply(model@last.grouping, length) > 2)
subnet.id <- names(model@subnets)[[1]]
m <- get.model.parameters(model, subnet.id)
print(m)
```

`add.ellipse` *Add ellipse to an existing plot.*

Description

Calculates and plots ellipse corresponding to specified confidence interval in 2-dimensional plot.

Usage

```
add.ellipse(centroid, covmat, confidence = 0.95, npoints
  = 100, col = "black", ...)
```

Arguments

`centroid`	Vector with two elements defining the ellipse centroid.
`covmat`	Covariance matrix for the investigated data. Only diagonal covariances supported.
`confidence`	Confidence level determining the ellipse borders based on the covariance matrix.
`npoints`	Number of plotting points.
`col`	Color.
`...`	Other arguments to be passed.

Value

Used for plotting side effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

Examples

```
#add.ellipse(centroid = c(0, 0), covmat = diag(c(1,2)))
```

`bic.mixture` *Description: Latent class analysis based on (infinite) Gaussian mixture model. If the input is data matrix, a multivariate model is fitted; if the input is a vector, a univariate model is fitted*

Description

Arguments:

Usage

```
bic.mixture(x, max.modes, bic.threshold = 0,
  min.modes = 1, ...)
```

Arguments

`x`	samples x features matrix for multivariate analysis, or a vector for univariate analysis
`max.modes`	Maximum number of modes to be checked for mixture model selection
`bic.threshold`	BIC threshold which needs to be exceeded before a new mode is added to the mixture.
`min.modes`	minimum number of modes
`...`	Further optional arguments to be passed
	Returns:

Value

Fitted latent class model (parameters and free energy)

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`bic.mixture.multivariate` *Description: Latent class analysis based on (infinite) Gaussian mixture model. If the input (dat) is data matrix, a multivariate model is fitted.*

Description

Arguments:

Usage

```
bic.mixture.multivariate(x, max.modes, bic.threshold = 0,
  ...)
```

Arguments

`x`	matrix (for multivariate analysis)
`max.modes`	Maximum number of modes to be checked for mixture model selection
`bic.threshold`	BIC threshold which needs to be exceeded before a new mode is added to the mixture.
`...`	Further optional arguments to be passed
	Returns:

Value

Fitted latent class model (parameters and free energy)

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`bic.mixture.univariate`

Description: Latent class analysis based on (infinite) Gaussian mixture model. If the input (dat) is data matrix, a multivariate model is fitted. If the input is a vector or a 1-dimensional matrix, a univariate model is fitted.

Description

Arguments:

Usage

```
bic.mixture.univariate(x, max.modes, bic.threshold = 0,
  min.modes = 1, ...)
```

Arguments

`x`	dat vector (for univariate analysis) or a matrix (for multivariate analysis)
`max.modes`	Maximum number of modes to be checked for mixture model selection
`bic.threshold`	BIC threshold which needs to be exceeded before a new mode is added to the mixture.
`min.modes`	minimum number of modes
`...`	Further optional arguments to be passed
	Returns:

Value

Fitted latent class model (parameters and free energy)

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`bic.select.best.mode` *Description: Select optimal number of mixture components by adding components until the increase in objective function is below threshold.*

Description

Arguments:

Usage

```
bic.select.best.mode(x, max.modes, bic.threshold,
  min.modes = 1)
```

Arguments

`x`	dat vector (for univariate analysis) or a matrix (for multivariate analysis)
`max.modes`	Maximum number of modes to be checked for mixture model selection
`bic.threshold`	BIC threshold which needs to be exceeded before a new mode is added to the mixture.
`min.modes`	Optiomal. Minimum number of modes.
	Returns:

Value

Fitted latent class model (parameters and free energy)

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`centerData` *Center data matrix.*

Description

Center data matrix to 0 for each variable by removing the means.

Usage

```
centerData(X, rm.na = TRUE, meanvalue = NULL)
```

Arguments

X	The data set: samples x features. Each feature will be centered.
rm.na	Ignore NAs.
meanvalue	Can be used to set a desired center value. The default is 0.

Value

Centered data matrix.

Note

Note that the model assumes samples x features matrix, and centers each feature.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse").

Examples

```
#centerData(X)
```

---

continuous.responses *Description: Quantify association between modes and continuous variable*

Description

Arguments:

Usage

```
continuous.responses(annotation.vector, model,
  method = "t-test", min.size = 2, data = NULL)
```

Arguments

annotation.vector	annotation vector with discrete factor levels, and named by the samples
model	NetResponse model object
method	method for enrichment calculation
min.size	minimum sample size for a response
data	data matrix (samples x features)
	Returns:

Value

List with each element corresponding to one variable and listing the responses according to association strength

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`detect.responses` *detect.responses*

Description

Main function of the NetResponse algorithm. Detect condition-specific network responses, given network and a set of measurements of node activity in a set of conditions. Returns a set of subnetworks and their estimated context-specific responses.

Usage

```
detect.responses(datamatrix, network = NULL,
  initial.responses = 1, max.responses = 10,
  max.subnet.size = 10, verbose = TRUE, prior.alpha = 1,
  prior.alphaKsi = 0.01, prior.betaKsi = 0.01,
  update.hyperparams = 0, implicit.noise = 0,
  vdp.threshold = 1e-05, merging.threshold = 0,
  ite = Inf, information.criterion = "BIC",
  speedup = TRUE, speedup.max.edges = 10,
  positive.edges = FALSE, mc.cores = 1,
  mixture.method = "vdp", bic.threshold = 0,
  pca.basis = FALSE, ...)
```

Arguments

`datamatrix`	Matrix of samples x features. For example, gene expression matrix with conditions on the rows, and genes on the columns. The matrix contains same features than the 'network' object, characterizing the network states across the different samples.
`network`	Binary network describing undirected pairwise interactions between features of 'datamatrix'. The following formats are supported: binary matrix, graphNEL, igraph, graphAM, Matrix, dgCMatrix, dgeMatrix
`initial.responses`	Initial number of components for each subnetwork model. Used to initialize calculations.

`max.responses` Maximum number of responses for each subnetwork. Can be used to limit the potential number of network states.

`max.subnet.size`
Numeric. Maximum allowed subnetwork size.

`verbose` Logical. Verbose parameter.

`implicit.noise` Implicit noise parameter. Add implicit noise to vdp mixture model. Can help to avoid overfitting to local optima, if this appears to be a problem.

`update.hyperparams`
Logical. Indicate whether to update hyperparameters during modeling.

`prior.alpha,prior.alphaKsi,prior.betaKsi`
Prior parameters for Gaussian mixture model that is calculated for each subnetwork (normal-inverse-Gamma prior). alpha tunes the mean; alphaKsi and betaKsi are the shape and scale parameters of the inverse Gamma function, respectively.

`vdp.threshold` Minimal free energy improvement after which the variational Gaussian mixture algorithm is deemed converged.

`merging.threshold`
Minimal cost value improvement required for merging two subnetworks.

`ite` Defines maximum number of iterations on posterior update (updatePosterior). Increasing this can potentially lead to more accurate results, but computation may take longer.

`information.criterion`
Information criterion for model selection. Default is BIC (Bayesian Information Criterion); other options include AIC and AICc.

`speedup` Takes advantage of approximations to PCA, mutual information etc in various places to speed up calculations. Particularly useful with large and densely connected networks and/or large sample size.

`speedup.max.edges`
Used if speedup = TRUE. Applies prefiltering of edges for calculating new joint models between subnetwork pairs when potential cost changes (delta) are updated for a newly merged subnetwork and its neighborghs. Empirical mutual information between each such subnetwork pair is calculated based on their first principal components, and joint models will be calculated only for the top candidates up to the number specified by speedup.max.edges. It is expected that the subnetwork pair that will benefit most from joint modeling will be among the top mutual infomation candidates. This way it is possible to avoid calculating exhaustive many models on the network hubs.

`positive.edges` Consider only the edges with positive association. Currently measured with Spearman correlation.

`mc.cores` Number of cores to be used in parallelization. See help(mclapply) for details.

`mixture.method` Specify the approach to use in mixture modeling. Options. vdp (nonparametric Variational Dirichlet process mixture model); bic (based on Gaussian mixture modeling with EM, using BIC to select the optimal number of components)

`bic.threshold` BIC threshold which needs to be exceeded before a new mode is added to the mixture with mixture.method = "bic"

pca.basis	Transform data first onto PCA basis to try to avoid problems with non-diagonal covariances.
...	Further optional arguments to be passed.

Value

NetResponseModel object.

Author(s)

Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse").

Examples

```
library(netresponse)
data( toydata )        # Load toy data set
D    <- toydata$emat   # Response matrix (for example, gene expression)
netw <- toydata$netw   # Network

# Run NetReponse algorithm
model <- detect.responses(D, netw, verbose = FALSE)
```

---

dna *Dna damage data set (PPI and expression)*

Description

A combined yeast data set with protein-protein interactions and gene expression (dna damage). Gene expression profiles are transformed into links by computing a Pearson correlation for all pairs of genes and treating all correlations above 0.85 as additional links.

Format

List of following objects:

ppi PPI data matrix

exp gene expression profiles data matrix

gids Vector of gene ids corresponding to indices used in data matrices

obs Gene expression observation details

combined.links pooled matrix of PPI and expression links

Details

Number of genes: 1823, number of interactions: 12382, number of gene expression observations: 52, number of total links with PPI and expression links: 15547.

Source

PPI data pooled from yeast data sets of [1] and [2]. Dna damage expression set of [3].

References

Ulitsky, I. and Shamir, R. *Identification of functional modules using network topology and high-throughput data.* BMC Systems Biology 2007, 1:8.

Nariai, N., Kolaczyk, E. D. and Kasif, S. *Probabilistic Protein Function Predition from Heterogenous Genome-Wide Data*. PLoS ONE 2007, 2(3):e337.

Gasch, A., Huang, M., Metzner, S., Botstein, D. and Elledge, S. *Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mex1p.* Molecular Biology of the Cell 2001, 12:2987-3003.

Examples

```
data(dna)
```

---

enrichment.list.factor

Description: enrichment.list.factor

Description

Orders the responses by association strength (enrichment score) to a given sample set. For instance, if the samples correspond to a particular experimental factor, this function can be used to prioritize the responses according to their association strength to this factor.

Usage

```
enrichment.list.factor(models, level.samples, method,
  verbose = FALSE)
```

Arguments

`models`	List of models. Each model should have a sample-cluster assignment matrix qofz.
`level.samples`	Measure enrichment of this sample (set) across the observed responses.
`method`	'hypergeometric' measures enrichment of factor levels in this response; 'precision' measures response purity for each factor level; 'dependency' measures logarithm of the joint density between response and factor level vs. their marginal densities: log(P(r,s)/(P(r)P(s)))
`verbose`	Follow progress by intermediate messages.
	Returns:

Details

Arguments:

Value

A data frame which gives a data frame of responses ordered by enrichment score for the investigated sample. The model, response id and enrichment score are shown. The method field indicates the enrichment calculation method. The sample field lists the samples et for which the enrichments were calculated. The info field lists additional information on enrichment statistics.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

#

---

factor.responses *Description: List responses for each level of the given factor*

Description

Arguments:

Usage

```
factor.responses(annotation.vector, models,
  method = "hypergeometric", min.size = 2, data = NULL)
```

Arguments

annotation.vector	annotation vector with discrete factor levels, and named by the samples
models	List of models. Each model should have a sample-cluster assignment matrix qofz.
method	method for enrichment calculation
min.size	minimum sample size for a response
data	data (samples x features; or a vector in univariate case)

Returns:

Value

List with each element corresponding to one factor level and listing the responses according to association strength

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`find.similar.features` *Find similar features with a given subnetwork.*

Description

Given subnetwork, orders the remaining features (genes) in the input data based on similarity with the subnetwork. Allows the identification of similar features that are not directly connected in the input network.

Usage

```
find.similar.features(model, subnet.id, datamatrix =
  NULL, verbose = FALSE, information.criterion = NULL)
```

Arguments

`model`	NetResponseModel object.
`subnet.id`	Investigated subnetwork.
`datamatrix`	Optional. Can be used to compare subnetwork similarity with new data which was not used for learning the subnetworks.
`verbose`	Logical indicating whether progress of the algorithm should be indicated on the screen.
`information.criterion`	Information criterion for model selection. By default uses the same than in the 'model' object.

Details

The same similarity measure is used as when agglomerating the subnetworks: the features are ordered by delta (change) in the cost function, assuming that the feature would be merged in the subnetwork. The smaller the change, the more similar the feature is (change would minimize the new cost function value). Negative values of delta mean that the cost function would be improved by merging the new feature in the subnetwork, indicating features having coordinated response.

Value

A data frame with elements feature.names (e.g. gene IDs) and delta, which indicates similarity level. See details for details. The smaller, the more similar. The data frame is ordered such that the features are listed by decreasing similarity.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for reference details.

Examples

```
data(toydata)
model <- toydata$model
subnet.id <- "Subnet-1"
g <- find.similar.features(model, subnet.id)
# List features that are similar to this subnetwork (delta < 0)
# (ordered by decreasing similarity)
subset(g, delta < 0)
```

---

get.dat *get.dat*

Description

Retrieve data for a given subnetwork.

Arguments

model	Result from NetResponse (detect.responses function).
subnet.id	Subnet identifier. A natural number which specifies one of the subnetworks within the 'model' object.
sample	Specify samples for which the data will be retrieved.

Value

Data matrix features x samples.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

Examples

```
# Load toy data
#data( toydata )          # Load toy data set
#D     <- toydata$emat    # Response matrix (for example, gene expression)
#model <- toydata$model   # Pre-calculated model
# Get model parameters for a given subnet
# (Gaussian mixture: mean, covariance diagonal, mixture proportions)
#get.dat(model, subnet.id = 1)
```

`get.model.parameters` *get.model.parameters*

Description

Retrieve the mixture model parameters of the NetResponse algorithm for a given subnetwork.

Usage

```
get.model.parameters(model, subnet.id = NULL)
```

Arguments

`model`	Result from NetResponse (detect.responses function).
`subnet.id`	Subnet identifier. A natural number which specifies one of the subnetworks within the 'model' object.

Details

Only the non-empty components are returned. Note: the original data matrix needs to be provided for function call separately.

Value

A list with the following elements:

`mu`	Centroids for the mixture components. Components x nodes.
`sd`	Standard deviations for the mixture components. A vector over the nodes for each component, implying the diagonal covariance matrix of the model (i.e. diag(std^2)). Components x nodes
`w`	Vector of component weights.
`nodes`	List of nodes in the subnetwork.
`K`	Number of mixture components.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for details.

Examples

```
# Load toy data
data( toydata )          # Load toy data set
D     <- toydata$emat    # Response matrix (for example, gene expression)
model <- toydata$model   # Pre-calculated model

# Get model parameters for a given subnet
# (Gaussian mixture: mean, covariance diagonal, mixture proportions)
get.model.parameters(model, subnet.id = 1)
```

`get.subnets` *get.subnets*

Description

List the detected subnetworks (each is a list of nodes in the corresponding subnetwork).

Arguments

`model`	Output from the detect.responses function. An object of NetResponseModel class.
`get.names`	Logical. Indicate whether to return subnetwork nodes using node names (TRUE) or node indices (FALSE).
`min.size,max.size`	Numeric. Filter out subnetworks whose size is not within the limits specified here.
`min.responses`	Numeric. Filter out subnetworks with less responses (mixture components) than specified here.

Value

A list of subnetworks.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for details.

Examples

```
# library(netresponse)
## Load a pre-calculated netresponse model obtained with
# model <- detect.responses(toydata$emat, toydata$netw, verbose = FALSE)
# data( toydata ); get.subnets(toydata$model)
```

`ICMg.combined.sampler` *ICMg.combined.sampler*

Description

Main function of the ICMg algorithm. ICMg.combined.sampler computes samples from the posterior of the assignments of datapoints (interactions and expression profiles) to latent components. From these we can then obtain component membership distributions and clusterings for genes.

Usage

```
ICMg.combined.sampler(L, X, C, alpha=10, beta=0.01,
  pm0=0, V0=1, V=0.1, B.num=8, B.size=100, S.num=20,
  S.size=10, C.boost=1)
```

Arguments

`L`	N x 2 matrix of link endpoints (N = number of links).
`X`	M x D matrix of gene expression profiles (M = number of nodes, D = number of observations).
`C`	Number of components.
`alpha`	Hyperparameter describing the global distribution over components, larger alpha gives a more uniform distribution.
`beta`	Hyperparameter describing the component-wise distributions over nodes, larger beta gives a more uniform distribution.
`pm0`	Hyperparameter describing the prior mean of the expression profiles, should be zero.
`V0`	Hyperparameter describing the variation of the component-wise expression profiles means around pm0.
`V`	Hyperparameter describing the variation of gene-specific expression profiles around the component-wise means.
`B.num`	Number of burnin rounds.*
`B.size`	Size of one burnin round.*
`S.num`	Number of sample rounds.*
`S.size`	Size of one sample round.*
`C.boost`	Set to 1 to use faster iteration with C, set to 0 to use slower R functions.

Details

One run consists of two parts, during burnin the sampler is expected to mix, after which the samples are taken. Information about convergence (convN and convL are estimates of convergence for link and node sampling, respectively) and component sizes are printed after each burnin/sample round. For example: B.num=8, B.size=100, S.num=20, S.size=10, runs 800 burnin iterations in 8 rounds and then takes 20 samples with an interval of 10 iterations.

Value

Returns samples as a list:

`z`	S.num x N matrix of samples of component assignments for links.
`w`	S.num x M matrix of samples of component assignments for gene expression profiles.
`convl`	Vector of length (B.num + S.num) with convergence estimator values for link sampling.
`convn`	Vector of length (B.num + S.num) with convergence estimator values for node sampling.
`countsl`	(B.num + S.num) x C matrix of link component sizes.
`countsn`	(B.num + S.num) x C matrix of node component sizes.

additionally all parameters of the run are included in the list.

Author(s)

Juuso Parkkinen

References

Parkkinen, J. and Kaski, S. Searching for functional gene modules with interaction component models. BMC Systems Biology 4 (2010), 4.

See Also

`ICMg.links.sampler`

Examples

```
library(netresponse)
data(osmo) # Load data set

## Run ICMg combined sampler
res = ICMg.combined.sampler(osmo$ppi, osmo$exp, C=10)
```

ICMg.get.comp.memberships *ICMg.get.comp.memberships*

Description

Function for computing the component memberships for each data point from the MCMC samples.

Usage

```
ICMg.get.comp.memberships(links, samples)
```

Arguments

links	N x 2 matrix of link endpoints (N = number of links).
samples	Posterior samples, as given by either ICMg.combined.sampler or ICMg.links.sampler.

Value

A matrix containing the component memberships for each data point (node).

Author(s)

Juuso Parkkinen

References

Parkkinen, J. and Kaski, S. Searching for functional gene modules with interaction component models. BMC Systems Biology 4 (2010), 4.

See Also

ICMg.combined.sampler, ICMg.links.sampler

ICMg.links.sampler *ICMg.links.sampler*

Description

ICMg.links.sampler computes samples from the posterior of the assignments of datapoints (interactions) to latent components. From these we can then obtain component membership distributions and clusterings for genes.

Usage

```
ICMg.links.sampler(L, C, alpha=10, beta=0.01, B.num=8,
  B.size=100, S.num=20, S.size=10, C.boost=1)
```

Arguments

L	N x 2 matrix of link endpoints (N = number of links).
C	Number of components.
alpha	Hyperparameter describing the global distribution over components, larger alpha gives a more uniform distribution.
beta	Hyperparameter describing the component-wise distributions over nodes, larger beta gives a more uniform distribution.
B.num	Number of burnin rounds.*
B.size	Size of one burnin round.*
S.num	Number of sample rounds.*
S.size	Size of one sample round.*
C.boost	Set to 1 to use faster iteration with C, set to 0 to use slower R functions.

Details

One run consists of two parts, during burnin the sampler is expected to mix, after which the samples are taken. Information about convergence (convN and convL are estimates of convergence for link and node sampling, respectively) and component sizes are printed after each burnin/sample round. For example: B.num=8, B.size=100, S.num=20, S.size=10, runs 800 burnin iterations in 8 rounds and then takes 20 samples with an interval of 10 iterations.

Value

Returns samples as a list:

z	S.num x N matrix of samples of component assignments for links.
conv	Vector of length (B.num + S.num) with convergence estimator values for link sampling.
counts	(B.num + S.num) x C matrix of link component sizes.

additionally all parameters of the run are included in the list.

Author(s)

Juuso Parkkinen

References

Parkkinen, J. and Kaski, S. Searching for functional gene modules with interaction component models. BMC Systems Biology 4 (2010), 4.

See Also

ICMg.combined.sampler

Examples

```
#
library(netresponse)
data(osmo) # Load data

## Run ICMg links sampler
res = ICMg.links.sampler(osmo$ppi, C=10)
```

`list.responses.continuous`

Description: Investigate association of a continuous variable and the modes

---

Description

Arguments:

Usage

```
list.responses.continuous(annotation.df, models,
  method = "t-test", qth = Inf, verbose = TRUE,
  rounding = NULL)
```

Arguments

`annotation.df`	annotation data.frame with discrete factor levels, rows named by the samples
`models`	List of models. Each model should have a sample-cluster assignment matrix qofz.
`method`	method for quantifying the association
`qth`	q-value threshold
`verbose`	verbose
`rounding`	rounding digits
	Returns:

Value

Table listing all associations between the factor levels and responses

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`list.responses.factor` *Description: List responses for all factors and levels in the given annotation matrix*

Description

Arguments:

Usage

```
list.responses.factor(annotation.df, models,
  method = "hypergeometric", min.size = 2, qth = Inf,
  verbose = TRUE, data = NULL, rounding = NULL)
```

Arguments

`annotation.df`	annotation data.frame with discrete factor levels, rows named by the samples
`models`	List of models. Each model should have a sample-cluster assignment matrix qofz.
`method`	method for enrichment calculation
`min.size`	minimum sample size for a response
`qth`	q-value threshold
`verbose`	verbose
`data`	data (samples x features; or a vector in univariate case)
`rounding`	rounding digits
	Returns:

Value

Table listing all associations between the factor levels and responses

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

`list.significant.responses` *List responses with significant associations to a given sample group.*

Description

List responses with significant associations to a given sample group.

Usage

```
list.significant.responses(model, sample, qth = 1,
  method = "hypergeometric")
```

Arguments

`model`	NetResponseModel object.
`sample`	User-specified samples group for which the enrichments are calculated. For instance, an annotation category.
`qth`	q-value threshold for enrichments
`method`	Enrichment method.

Value

Table containing statistics of the significantly associated responses.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

See Also

response.enrichment

Examples

```
#
```

mixture.model *Description: Fit Gaussian mixture model*

Description

Arguments:

Usage

```
mixture.model(x, mixture.method = "vdp",
  max.responses = 10, implicit.noise = 0,
  prior.alpha = 1, prior.alphaKsi = 0.01,
  prior.betaKsi = 0.01, vdp.threshold = 1e-05,
  initial.responses = 1, ite = Inf, speedup = TRUE,
  bic.threshold = 0, pca.basis = FALSE,
  min.responses = 1, ...)
```

Arguments

x	data matrix (samples x features, for multivariate analysis) or a vector (for univariate analysis)
mixture.method	Specify the approach to use in mixture modeling. Options. vdp (nonparametric Variational Dirichlet process mixture model); bic (based on Gaussian mixture modeling with EM, using BIC to select the optimal number of components)
max.responses	Maximum number of responses for each subnetwork. Can be used to limit the potential number of network states.
implicit.noise	Implicit noise parameter. Add implicit noise to vdp mixture model. Can help to avoid overfitting to local optima, if this appears to be a problem.
prior.alpha,prior.alphaKsi,prior.betaKsi	Prior parameters for Gaussian mixture model that is calculated for each subnetwork (normal-inverse-Gamma prior). alpha tunes the mean; alphaKsi and betaKsi are the shape and scale parameters of the inverse Gamma function, respectively.
vdp.threshold	Minimal free energy improvement after which the variational Gaussian mixture algorithm is deemed converged.
initial.responses	Initial number of components for each subnetwork model. Used to initialize calculations.
ite	Maximum number of iterations on posterior update (updatePosterior). Increasing this can potentially lead to more accurate results, but computation may take longer.
speedup	Takes advantage of approximations to PCA, mutual information etc in various places to speed up calculations. Particularly useful with large and densely connected networks and/or large sample size.

bic.threshold	BIC threshold which needs to be exceeded before a new mode is added to the mixture with mixture.method = "bic"
pca.basis	pca.basis
min.responses	minimum number of responses
...	Further optional arguments to be passed.
	Returns:

Value

List with two elements: model: fitted mixture model (parameters and free energy); model.params: model parameters

Author(s)

Contact: Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

---

model.stats *model.stats*

Description

Subnetwork statistics: size and number of distinct responses for each subnet.

Usage

```
model.stats(models)
```

Arguments

models	NetResponse object or list of models

Value

A 'subnetworks x properties' data frame containing the following elements.

subnet.size:	Vector of subnetwork sizes.
subnet.responses:	Vector giving the number of responses in each subnetwork.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for reference details.

Examples

```
library(netresponse)

# Load a pre-calculated netresponse model obtained with
# model <- detect.responses(toydata$emat, toydata$netw, verbose = FALSE)
data( toydata )
# Calculate summary statistics for the model
stat <- model.stats(toydata$model)
```

NetResponseModel-class *Class "NetResponseModel"*

Description

A NetResponse model.

Objects from the Class

Returned by `detect.responses` function.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

Examples

```
showClass("NetResponseModel")
```

---

order.responses *order.responses*

Description

Orders the responses by association strength (enrichment score) to a given sample set. For instance, if the samples correspond to a particular experimental factor, this function can be used to prioritize the responses according to their association strength to this factor.

Usage

```
order.responses(models, sample,
  method = "hypergeometric", min.size = 2,
  max.size = Inf, min.responses = 2, subnet.ids = NULL,
  verbose = FALSE, data = NULL)
```

Arguments

models	List of models. Each model should have a sample-cluster assignment matrix qofz.
sample	Measure enrichment of this sample (set) across the observed responses.
method	'hypergeometric' measures enrichment of factor levels in this response; 'precision' measures response purity for each factor level; 'dependency' measures logarithm of the joint density between response and factor level vs. their marginal densities: log(P(r,s)/(P(r)P(s)))
min.size,max.size,min.responses	Optional parameters to filter the results based on subnet size and number of responses.
subnet.ids	Specify subnets for which the responses shall be ordered. By default, use all subnets.
verbose	Follow progress by intermediate messages.
data	data (samples x features; or a vector in univariate case)

Value

A data frame with elements 'ordered.responses' which gives a data frame of responses ordered by enrichment score for the investigated sample. The subnetwork, response id and enrichment score are shown. The method field indicates the enrichment calculation method. The sample field lists the samples et for which the enrichments were calculated. The info field lists additional information on enrichment statistics.

Note

Tools for analyzing end results of the model.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
#
# - for given sample/s (factor level), order responses (across all subnets) by association strength (enrichment scor
#order.responses(model, sample, method  = "hypergeometric") # overrepresentation
```

`osmo` *Osmoshock data set (PPI and expression)*

Description

A combined yeast data set with protein-protein interactions and gene expression (osmotick shock response). Gene expression profiles are transformed into links by computing a Pearson correlation for all pairs of genes and treating all correlations above 0.85 as additional links.

Format

List of following objects:

ppi PPI data matrix

exp gene expression profiles data matrix

gids Vector of gene ids corresponding to indices used in data matrices

obs Gene expression observation details

combined.links pooled matrix of PPI and expression links

Details

Number of genes: 1711, number of interactions: 10250, number of gene expression observations: 133, number of total links with PPI and expression links: 14256.

Source

PPI data pooled from yeast data sets of [1] and [2]. Dna damage expression set of [3].

References

Ulitsky, I. and Shamir, R. *Identification of functional modules using network topology and high-throughput data.* BMC Systems Biology 2007, 1:8.

Nariai, N., Kolaczyk, E. D. and Kasif, S. *Probabilistic Protein Function Predition from Heterogenous Genome-Wide Data*. PLoS ONE 2007, 2(3):e337.

O'Rourke, S. and Herskowitz, I. *Unique and redundant roles for Hog MAPK pathway components as revealed by whole-genome expression analysis.* Molecular Biology of the Cell 2004, 15:532-42.

Examples

```
data(osmo)
```

plot.data *Plot observed data.*

Description

Plotting tool for measurement data.

Usage

```
  ## S3 method for class 'data'
plot(x, subnet.id, labels, ...)
```

Arguments

x	NetResponseModel object.
subnet.id	Specify the subnetwork.
labels	Annotation categories.
...	Further arguments for plot function.
	Return:

Details

Produces boxplot for each feature in each annotation category for the selected subnetwork.

Value

ggplot2 plot object

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

See Also

plot.responses

Examples

#

plot.expression *plot.expression*

Description

Plot expression matrix in color scale. For one-channel data; plot expression of each gene relative to its mean expression level over all samples. Blue indicates decreased expression and red indicates increased expression. Brightness of the color indicates magnitude of the change. Black denotes no change.

Usage

```
  ## S3 method for class 'expression'
plot(x, maintext, ...)
```

Arguments

x	samples x features matrix
maintext	main title
...	optional arguments

Value

Used for its side effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse").

See Also

plot.scale

Examples

```
#

#plot.expression(x)
```

plot.response *plot.response*

Description

Plot a specific transcriptional response for a given subnetwork.

Usage

```
  ## S3 method for class 'response'
plot(x, mynet, mybreaks, mypalette,
    plot.names = TRUE, colors = TRUE, plot.type = "twopi",
    ...)
```

Arguments

x	A numerical vector, or NULL.
mynet	Binary matrix specifying the interactions between nodes.
mybreaks	Specify breakpoints for color plot.
mypalette	Specify palette for color plot.
plot.names	Plot node names (TRUE) or indices (FALSE).
colors	Plot colors. Logical.
plot.type	Network plot mode. For instance, 'neato' or 'twopi'.
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti, Olli-Pekka Huovilainen and Antonio Gusmao. Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

L. Lahti et al.: Global modeling of transcriptional responses in interaction networks. Submitted.

Examples

```
#

#tmp <- plot.response(model, mynet, maintext = paste("Subnetwork", subnet.id))
```

plot.responses *plot.responses*

Description

Plot the detected transcriptional responses for a given subnetwork.

Usage

```
  ## S3 method for class 'responses'
plot(x, subnet.id, nc = 3, plot.names
    = TRUE, plot.mode = "network", xaxis = TRUE, yaxis =
    TRUE, plot.type = "twopi", mar = c(5, 4, 4, 2), horiz =
    TRUE, datamatrix = NULL, scale = FALSE, ...)
```

Arguments

x	Result from NetResponse (detect.responses function).
subnet.id	Subnet id.
nc	Number of columns for an array of images.
plot.names	Plot node names (TRUE) or indices (FALSE).
plot.mode	network: plot responses as a subnetwork graph; matrix, heatmap: plot subnetwork expression matrix. For both, expression of each gene is shown relative to the mean expression level of the gene; boxplot.data: feature-wise boxplots for hard sample-to-response assignments; response.barplot: estimated response centroids as barplot including 95 confidence intervals for the means; pca: PCA projection with estimated centroids and 95 1-dimensional case a histogram is drawn. In two-dimensional case the original coordinates are used.
xaxis,yaxis	Logical. Plot row/column names.
plot.type	Network plot mode. For instance, 'neato' or 'twopi'.
mar	Figure margins.
horiz	Logical. Horizontal barplot.
datamatrix	datamatrix
scale	scale the phylotypes to unit length (only implemented for plot.mode = "matrix"
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

See Also

plot.scale

Examples

```
#
#res <- detect.responses(D, netw)
#vis <- plot.responses(res, subnet.id)
```

---

plot.scale *plot.scale*

Description

Plot the color scale used in visualization.

Usage

```
  ## S3 method for class 'scale'
plot(x, y, m = NULL, cex.axis = 1.5,
    label.step = 2, interval = 0.1, two.sided = TRUE,
    label.start = 1, Nlab = 3, ...)
```

Arguments

x	Breakpoints for the plot.
y	Color palette.
m	Breakpoints' upper limit.
cex.axis	Axis scale.
label.step	Density of the labels.
interval	Interval.
two.sided	Plot two-sided (TRUE) or one-sided (FALSE) visualization.
label.start	Label starting point.
Nlab	Number of labels to plot.
...	Further arguments for plot function.

Value

Used for its side-effects.

Note

Depends on Rgraphviz and igraph0 packages.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

Examples

```
#
  #res <- detect.responses(D, netw, verbose = FALSE)
  #vis <- plot.responses(res, subnet.idx)
  #plot.scale(vis$breaks, vis$palette)
```

---

plot.subnet *plot.subnet*

Description

Plot the given subnetwork.

Usage

```
  ## S3 method for class 'subnet'
plot(x, subnet.id, network, plot.names =
    TRUE, ...)
```

Arguments

x	Result from NetResponse (detect.responses function).
subnet.id	Subnet id.
network	Original network used in the modelling.
plot.names	Plot node names (TRUE) or indices (FALSE).
...	Further arguments for plot function.

Value

Used for its side-effects. Returns a matrix that describes the investigated subnetwork.

Author(s)

Leo Lahti, Olli-Pekka Huovilainen and Antonio Gusmao. Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

L. Lahti et al.: Global modeling of transcriptional responses in interaction networks. Submitted.

Examples

```
#

# res <- detect.responses(D, netw, verbose = FALSE)
# net <- plot.subnet(res, subnet.idx = 1)
```

---

plotAssociations *Association strength between category labels and responses.*

Description

Plot association strength between user-defined category labels and responses in a selected subnetwork.

Usage

```
plotAssociations(x, subnet.id, labels,
  method = "hypergeometric", mode = "group.by.classes",
  ...)
```

Arguments

x	NetResponseModel object
subnet.id	Subnetwork.
labels	Factor. Labels for the data samples. Name by samples, or provide in the same order as in the original data.
method	Method to calculate association strength.
mode	group.by.responses or group.by.classes: indicate barplot grouping type.
...	Other arguments to be passed for plot.

Details

Associations are showm in terms -log10(p) enrichment values for the annotation categories for the responses within the specified subnetwork. No correction for multiple testing.

Value

Used for side effect (plotting).

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse").

See Also

plot.responses

Examples

```
#
```

`PlotMixture` *PlotMixture*

Description

Arguments:

Usage

```
PlotMixture(x, qofz, binwidth = 0.05, xlab.text = NULL,
  ylab.text = NULL, title.text = NULL)
```

Arguments

`x`	data vector
`qofz`	Mode assignment probabilities for each sample. Samples x modes.
`binwidth`	binwidth for histogram
`xlab.text`	xlab.text
`ylab.text`	ylab.text
`title.text`	title.text
	Return:

Value

Used for its side-effects

Author(s)

Leo Lahti `<leo.lahti@iki.fi>`

References

See citation("netresponse") for citation details.

Examples

```
# PlotMixture(x, qofz)
```

PlotMixtureBivariate *PlotMixtureBivariate*

Description

Visualize data, centroids and response confidence intervals for a given Gaussian mixture model in two-dimensional (bivariate) case. Optionally, color the samples according to annotations labels.

Usage

```
PlotMixtureBivariate(x, means, sds, ws, labels = NULL,
  confidence = 0.95, main = "", ...)
```

Arguments

x	data matrix (samples x features)
means	mode centroids (modes x features)
sds	mode standard deviations, assuming diagonal covariance matrices (modes x features, each row giving the sqrt of covariance diagonal for the corresponding mode)
ws	weight for each mode
labels	Optional: sample class labels to be indicated in colors.
confidence	Confidence interval for the responses based on the covariances of each response. If NULL, no plotting.
main	title text
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
#plotMixture(dat, means, sds, ws)
```

PlotMixtureMultivariate *PlotMixtureMultivariate*

Description

Visualize data, centroids and response confidence intervals for a given Gaussian mixture model with PCA. Optionally, color the samples according to annotations labels.

Usage

```
PlotMixtureMultivariate(x, means, sds, ws, labels = NULL,
  title = NULL, modes = NULL, pca = FALSE, qofz = NULL,
  ...)
```

Arguments

x	data matrix (samples x features)
means	mode centroids (modes x features)
sds	mode standard deviations, assuming diagonal covariance matrices (modes x features, each row giving the sqrt of covariance diagonal for the corresponding mode)
ws	weight for each mode
labels	Optional: sample class labels to be indicated in colors.
title	title
modes	Optional: provide sample modes for visualization already in the input
pca	The data is projected on PCA plane by default (pca = TRUE). By setting this off (pca = FALSE) it is possible to visualize two-dimensional data in the original domain.
qofz	Sample-response probabilistic assignments matrix (samples x responses)
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
#plotMixture(dat, means, sds, ws)
```

PlotMixtureMultivariate.deprecated
PlotMixtureMultivariate.deprecated

Description

Visualize data, centroids and response confidence intervals for a given Gaussian mixture model with PCA. Optionally, color the samples according to annotations labels.

Usage

```
PlotMixtureMultivariate.deprecated(x, means, sds, ws,
  labels = NULL, confidence = 0.95, ...)
```

Arguments

x	data matrix (samples x features)
means	mode centroids (modes x features)
sds	mode standard deviations, assuming diagonal covariance matrices (modes x features, each row giving the sqrt of covariance diagonal for the corresponding mode)
ws	weight for each mode
labels	Optional: sample class labels to be indicated in colors.
confidence	Confidence interval for the responses based on the covariances of each response. If NULL, no plotting.
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
#plotMixture(dat, means, sds, ws)
```

PlotMixtureUnivariate *PlotMixtureUnivariate*

Description

Visualize data, centroids and stds for a given univariate Gaussian mixture model with PCA.

Usage

```
PlotMixtureUnivariate(x, means, sds, ws,
  title.text = NULL, xlab.text = NULL, ylab.text = NULL,
  binwidth = 0.05, qofz = NULL,
  density.color = "darkgray", ...)
```

Arguments

x	data vector
means	mode centroids
sds	mode standard deviations
ws	weight for each mode
title.text	Plot title
xlab.text	xlab.text
ylab.text	ylab.text
binwidth	binwidth for histogram
qofz	Mode assignment probabilities for each sample. Samples x modes.
density.color	Color for density lines
...	Further arguments for plot function.
	Return:

Details

Arguments:

Value

Used for its side-effects

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
# plotMixtureUnivariate(dat, means, sds, ws)
```

---

plotPCA *plotPCA*

Description

Visualize data, centroids and response confidence intervals for a given subnetwork with PCA. Optionally, color the samples according to annotations labels.

Usage

```
plotPCA(x, subnet.id, labels = NULL, confidence = 0.95,
  ...)
```

Arguments

x	NetResponseModel object. Output from the detect.responses function.
subnet.id	Subnetwork id. Either character as 'Subnetwork-2' or numeric as 2, which is then converted to character.
labels	Optional: sample class labels to be indicated in colors.
confidence	Confidence interval for the responses based on the covariances of each response. If NULL, no plotting.
...	Further arguments for plot function.

Value

Used for its side-effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse") for citation details.

Examples

```
#plotPCA(x, subnet.id)
```

`read.sif` *Reading network files*

Description

Function to read network files.

Usage

```
read.sif(sif.file, format = "graphNEL", directed = FALSE,
  header = TRUE, sep = "\t", ...)
```

Arguments

`sif.file`	Name of network file in SIF format.
`format`	Output format: igraph0 or graphNEL
`directed`	Logical. Directed/undirected graph. Not used in the current model.
`header`	Logical. Indicate whether the SIF file has header or not.
`sep`	Field separator.
`...`	Further optional arguments to be passed for file reading.

Details

Read in SIF network file, return R graph object in igraph or graphNEL format.

Value

R graph object in igraph0 or graphNEL format.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

Examples

```
#net <- read.sif("network.sif")
```

response.enrichment *Enrichment for a specified sample group in the given response.*

Description

Calculate enrichment values for a specified sample group in the given response.

Usage

```
response.enrichment(qofz, annotation.sample, response,
  method = "hypergeometric")
```

Arguments

qofz	sample-mode assignment matrix
annotation.sample	User-defined sample group. For instance, samples belonging to a particular annotation class.
response	Response id (integer) within the subnet.
method	Enrichment method.

Value

List with enrichment statistics, depending on enrichment method.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

See Also

order.responses

Examples

```
#enr <- response.enrichment(subnet.id, models, sample, response, method)
```

response2sample *response2sample*

Description

List the most strongly associated response of a given subnetwork for each sample.

Usage

```
response2sample(model, subnet.id = NULL,
  component.list = TRUE, verbose = FALSE, data = NULL)
```

Arguments

model	A NetResponseModel object or list.
subnet.id	Subnet id. A natural number which specifies one of the subnetworks within the 'model' object.
component.list	List samples separately for each mixture component (TRUE). Else list the most strongly associated component for each sample (FALSE).
verbose	Follow progress by intermediate messages.
data	Data (features x samples; or a vector for univariate case) to predict response for given data points (currently implemented only for mixture.model output)
	Return:

Value

A list. Each element corresponds to one subnetwork response, and contains a list of samples that are associated with the response (samples for which this response has the highest probability P(response | sample)).

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for citation details.

Examples

```
library( netresponse )

# Load example data
data( toydata )         # Load toy data set
D    <- toydata$emat    # Response matrix (for example, gene expression)
model <- toydata$model  # Pre-calculated model
```

```
# Find the samples for each response (for a given subnetwork)
response2sample(model, subnet.id = 1)
```

`sample2response` *sample2response*

Description

Probabilistic sample-response assignments for given subnet.

Usage

```
sample2response(model, subnet.id, mode = "soft")
```

Arguments

`model`	Result from NetResponse (detect.responses function).
`subnet.id`	Subnet identifier. A natural number which specifies one of the subnetworks within the 'model' object.
`mode`	soft: gives samples x responses probabilistic assignment matrix; hard: gives the most likely response for each sample

Value

A matrix of probabilities. Sample-response assignments for given subnet, listing the probability of each response, given a sample.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010). See citation("netresponse") for citation details.

Examples

```
#library(netresponse)
#data( toydata )        # Load toy data set
#D    <- toydata$emat   # Response matrix (for example, gene expression)
#netw <- toydata$netw   # Network

# Detect network responses
#model <- detect.responses(D, netw, verbose = FALSE)

# Assign samples to responses (soft, probabilistic assignments sum to 1)
#response.probabilities <- sample2response(model, subnet.id = "Subnet-1")
```

set.breaks *set.breaks*

Description

Set breakpoints for two-way color palette.

Usage

```
set.breaks(mat, interval = 0.1)
```

Arguments

mat	Matrix to visualize.
interval	Density of color breakpoints.

Value

A vector listing the color breakpoints.

Author(s)

Leo Lahti, Olli-Pekka Huovilainen and Antonio Gusmao. Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

L. Lahti et al.: Global modeling of transcriptional responses in interaction networks. Submitted.

Examples

```
set.breaks(array(rnorm(100), dim = c(10, 10)), interval = .1)
```

toydata *toydata*

Description

Toy data for NetResponse examples.

Format

Toy data: a list with three elements:

emat: Data matrix (samples x features). This contains the same features that are provided in the network (toydata$netw). The matrix characterizes measurements of network states across different conditions.

netw: Binary matrix that describes pairwise interactions between features. This defines an undirected network over the features. A link between two nodes is denoted by 1.

model: A pre-calculated model. Object of NetResponseModel class, resulting from applying the netresponse algorithm on the toydata with model <- detect.responses(D, netw).

References

Leo Lahti et al.: Global modeling of transcriptional responses in interaction networks. Bioinformatics (2010).

Examples

```
data(toydata)
  D    <- toydata$emat   # Response matrix (samples x features)
  netw <- toydata$netw   # Network between the features
  model <- toydata$model # Pre-calculated NetResponseModel obtained with
                         # model <- detect.responses(D, netw)
```

---

vdp.mixt *vdp.mixt*

Description

Accelerated variational Dirichlet process Gaussian mixture.

Usage

```
vdp.mixt(dat, prior.alpha = 1, prior.alphaKsi = 0.01,
  prior.betaKsi = 0.01, do.sort = TRUE,
  threshold = 1e-05, initial.K = 1, ite = Inf,
  implicit.noise = 0, c.max = 10, speedup = TRUE,
  min.size = 5)
```

Arguments

dat	Data matrix (samples x features).
prior.alpha,prior.alphaKsi,prior.betaKsi	Prior parameters for Gaussian mixture model (normal-inverse-Gamma prior). alpha tunes the mean; alphaKsi and betaKsi are the shape and scale parameters of the inverse Gamma function, respectively.

do.sort	When true, qOFz will be sorted in decreasing fashion by component size, based on colSums(qOFz). The qOFz matrix describes the sample-component assignments in the mixture model.
threshold	Defines the minimal free energy improvement that stops the algorithm: used to define convergence limit.
initial.K	Initial number of mixture components.
ite	Defines maximum number of iterations on posterior update (updatePosterior). Increasing this can potentially lead to more accurate results, but computation may take longer.
implicit.noise	Adds implicit noise; used by vdp.mk.log.lambda.so and vdp.mk.hp.posterior.so. By adding noise (positive values), one can avoid overfitting to local optima in some cases, if this happens to be a problem.
c.max	Maximum number of candidates to consider in find.best.splitting. During mixture model calculations new mixture components can be created until this upper limit has been reached. Defines the level of truncation for a truncated stick-breaking process.
speedup	When learning the number of components, each component is splitted based on its first PCA component. To speed up, approximate by using only subset of data to calculate PCA.
min.size	Minimum size for a component required for potential splitting during mixture estimation.

Details

Implementation of the Accelerated variational Dirichlet process Gaussian mixture model algorithm by Kenichi Kurihara et al., 2007.

ALGORITHM SUMMARY This code implements Gaussian mixture models with diagonal covariance matrices. The following greedy iterative approach is taken in order to obtain the number of mixture models and their corresponding parameters:

1. Start from one cluster, $T = 1$. 2. Select a number of candidate clusters according to their values of "Nc" = \sum_n=1^N q_z_n (z_n = c) (larger is better). 3. For each of the candidate clusters, c: 3a. Split c into two clusters, c1 and c2, through the bisector of its principal component. Initialise the responsibilities q_z_n(z_n = c_1) and q_z_n(z_n = c_2). 3b. Update only the parameters of c1 and c2 using the observations that belonged to c, and determine the new value for the free energy, FT+1. 3c. Reassign cluster labels so that cluster 1 corresponds to the largest cluster, cluster 2 to the second largest, and so on. 4. Select the split that lead to the maximal reduction of free energy, FT+1. 5. Update the posterior using the newly split data. 6. If FT - FT+1 < \epsilon then halt, else set T := T +1 and go to step 2.

The loop is implemented in the function greedy(...)

Value

prior	Prior parameters of the vdp-gm model (qofz: priors on observation lables; Mu: centroids; S2: variance).
posterior	Posterior estimates for the model parameters and statistics.

weights	Mixture proportions, or weights, for the Gaussian mixture components.
centroids	Centroids of the mixture components.
sds	Standard deviations for the mixture model components (posterior modes of the covariance diagonals square root). Calculated as sqrt(invgam.scale/(invgam.shape + 1)).
qOFz	Sample-to-cluster assigments (soft probabilistic associations).
Nc	Component sizes
invgam.shape	Shape parameter (alpha) of the inverse Gamma distribution
invgam.scale	Scale parameter (beta) of the inverse Gamma distribution
Nparams	Number of model parameters
K	Number of components in the mixture model
opts	Model parameters that were used.
free.energy	Free energy of the model.

Note

This implementation is based on the Variational Dirichlet Process Gaussian Mixture Model implementation, Copyright (C) 2007 Kenichi Kurihara (all rights reserved) and the Agglomerative Independent Variable Group Analysis package (in Matlab): Copyright (C) 2001-2007 Esa Alhoniemi, Antti Honkela, Krista Lagus, Jeremias Seppa, Harri Valpola, and Paul Wagner.

Author(s)

Maintainer: Leo Lahti <leo.lahti@iki.fi>

References

Kenichi Kurihara, Max Welling and Nikos Vlassis: Accelerated Variational Dirichlet Process Mixtures. In B. Sch\"olkopf and J. Platt and T. Hoffman (eds.), Advances in Neural Information Processing Systems 19, 761–768. MIT Press, Cambridge, MA 2007.

Examples

```
set.seed(123)

  # Generate toy data with two Gaussian components
  dat <- rbind(array(rnorm(400), dim = c(200,2)) + 5,
               array(rnorm(400), dim = c(200,2)))

  # Infinite Gaussian mixture model with
  # Variational Dirichlet Process approximation
  mixt <- vdp.mixt( dat )

  # Centroids of the detected Gaussian components
  mixt$posterior$centroids

  # Hard mixture component assignments for the samples
  apply(mixt$posterior$qOFz, 1, which.max)
```

`write.netresponse.results` *Write NetResponse results summary into a file.*

Description

Write NetResponse results summary into a file.

Usage

```
write.netresponse.results(x, subnet.ids = NULL, filename)
```

Arguments

`x`	NetResponseModel
`subnet.ids`	List of subnet ids to consider. By default, all subnets.
`filename`	Output file name.

Details

Experimental version.

Value

Used for side effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation("netresponse")

Examples

```
#
```

Index