Package ‘netresponse’

February 29, 2024

Type Package
Title Functional Network Analysis
Version 1.62.0
Date 2023-04-03
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), BiocStyle, Rgraphviz, rmarkdown, methods, minet, mclust, reshape2
Imports ggplot2, graph, igraph, parallel, plyr, qvalue, RColorBrewer
Suggests knitr
URL https://github.com/antagomir/netresponse
BugReports https://github.com/antagomir/netresponse/issues
biocViews CellBiology, Clustering, GeneExpression, Genetics, Network, GraphAndNetwork, DifferentialExpression, Microarray, NetworkInference, Transcription
VignetteBuilder knitr
RoxygenNote 7.2.3
git_url https://git.bioconductor.org/packages/netresponse
git_branch RELEASE_3_18
git_last_commit b3dff34
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-02-28
R topics documented:

netresponse-package .......................................................... 3
add.ellipse ................................................................. 4
bic.mixture ................................................................. 5
bic.mixture.multivariate ..................................................... 6
bic.mixture.univariate ....................................................... 7
bic.select.best.mode ......................................................... 8
centerData ................................................................. 8
check.matrix ................................................................. 9
check.network ............................................................. 10
continuous.responses ......................................................... 11
detect.responses ........................................................... 12
dna ............................................................................. 14
enrichment.list.factor ....................................................... 15
enrichment.list.factor.minimal ............................................ 16
factor.responses ............................................................ 17
factor.responses.minimal ................................................... 18
filter.netw ................................................................. 19
filter.network ............................................................. 20
find.similar.features ......................................................... 21
generate.toydata ............................................................ 22
get.dat,NetResponseModel-method ....................................... 23
get.mis ....................................................................... 24
get.model.parameters ....................................................... 24
get.subnets,NetResponseModel-method ................................ 26
getqofz,NetResponseModel-method ....................................... 27
independent.models .......................................................... 28
list.responses.continuous.multi ......................................... 28
list.responses.continuous.single ......................................... 29
list.responses.factor ........................................................ 31
list.responses.factor.minimal ............................................. 32
list.significant.responses .................................................. 33
listify.groupings ............................................................ 34
mixture.model ............................................................. 34
model.stats ................................................................. 36
NetResponseModel-class .................................................... 37
order.responses ............................................................ 37
osmo ......................................................................... 39
P.r.s ....................................................................... 40
PrS ......................................................................... 40
P.rs.joint ................................................................. 41
P.rs.joint.individual ......................................................... 42
P.S ....................................................................... 42
P.s.individual ............................................................ 43
P.s.r ..................................................................... 44
P.Sr ............................................................ 44
pick.model.pairs ............................................................ 45
Description

Global modeling of transcriptional responses in interaction networks.

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


Examples

```r
## Not run:
# 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 )
## End(Not run)
```

add.ellipse

**Add ellipse to an existing plot**

Description

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

Usage

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

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>

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.

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

Value

Fitted latent class model (parameters and free energy)

Author(s)

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

Multivariate BIC mixture

Description

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

Usage

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

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.
min.modes Minimum number of modes to be checked for mixture model selection
... Further optional arguments to be passed

Value

Fitted latent class model (parameters and free energy)

Author(s)

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

References

See citation('netresponse')
Univariate BIC mixture

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.

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

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  
*Select best mode with BIC*

**Description**
Select optimal number of mixture components by adding components until the increase in objective function is below threshold.

**Usage**
```r
bic.select.best.mode(x, max.modes = 1, bic.threshold = 1, 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`  
  Optional. Minimum number of modes.

**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**
```r
centerData(X, rm.na = TRUE, meanvalue = NULL)
```
check.matrix

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(matrix(rnorm(100), 10, 10))

Description

Mostly for internal purposes. Check input matrix format.

Usage

check.matrix(datamatrix)

Arguments

datamatrix See detect.responses

Value

The datamatrix, possibly added with necessary formatting for the netresponse algorithm.

Author(s)

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

References
See citation('netresponse')

See Also
detect.responses

Examples

```r
# datamatrix <- check.matrix(datamatrix)
```

Description
Internal use to check input network and format detect.responses.

Usage

```r
check.network(network, datamatrix, verbose = FALSE)
```

Arguments

- `network`: Input network, see detect.responses
- `datamatrix`: Input datamatrix, see detect.responses
- `verbose`: Print intermediate messages

Value

- `formatted`: Formatted network (self-links removed)
- `original`: Original network (possible in another representation format)
- `delta`: Cost function changes corresponding to the 'formatted' network.
- `nodes`: Nodes corresponding to the 'formatted' network.

Author(s)
Maintainer: Leo Lahti <leo.lahti@iki.fi>

References
See citation('netresponse')

See Also
detect.responses
Examples

# check.network(network, datamatrix, verbose = FALSE)

---

**continuous.responses  Continuous responses**

**Description**
Quantify association between modes and continuous variable

**Usage**

```r
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)

**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')

**Examples**

```r
res <- continuous.responses(annotation.vector = NULL, model = NULL)
```
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.

prior.alpha, prior.alphaKsi, prior.betaKsi

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

update.hyperparams

Logical. Indicate whether to update hyperparameters during modeling.

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.

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 neighbors. 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 information 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

```r
## Not run:
#data(toydata) # Load toy data set
#D <- toydata$emat # Response matrix (for example, gene expression)
#netw <- toydata$netw # Network

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

## End(Not run)
```

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

Usage

data(dna)
enrichment.list.factor

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

Source

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

References


Examples

data(dna)

enrichment.list.factor

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 q0fz.
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: \( \log(\frac{P(r,s)}{P(r)P(s)}) \)
verbose Follow progress by intermediate messages.

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

#

```r
enrichment.list.factor.minimal
```

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

```r
enrichment.list.factor.minimal(
  groupings,
  method,
  verbose = FALSE,
  annotation.vector,
  level
)
```
factor.responses

Arguments

- **groupings**: List of groupings. Each model should have a sample-cluster assignment matrix `qofz`.
- **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.
- **annotation.vector**: annotation vector
- **level**: level

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

```r
res <- enrichment.list.factor.minimal(groupings = NULL, method = NULL, annotation.vector = NULL, level = NULL)
```

---

**factor.responses**

**Factor responses**

**Description**

List responses for each level of the given factor
Usage

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

Arguments

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

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')

Examples

```R
res <- factor.responses(annotation.vector = NULL, groupings = NULL)
```

Description

List responses for each level of the given factor
Usage

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

Arguments

  annotation.vector  
  annotation vector with discrete factor levels, and named by the samples

  groupings  
  List of groupings. Each model should have a sample-cluster assignment matrix qofz, or a vector of cluster indices named by the samples.

  method  
  method for enrichment calculation

  min.size  
  minimum sample size for a response

  data  
  data (samples x features; or a vector in univariate case)

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’)

Examples

res <- factor.responses.minimal(annotation.vector = NULL, groupings = NULL)

filter.netw

Usage

filter.netw(network, delta, datamatrix, params)
filter.network

Arguments

- network: network
- delta: associated cost function value changes for each node merge
- datamatrix: datamatrix
- params: parameters

Details

Include only edges with the highest mutual information, calculated based on the first principal components.

Value

Filtered network

Author(s)

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

References

See citation('netresponse')
Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation('netresponse')

Examples
#

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.
Description

Generate toy data.

Usage

D <- generate.toydata()

Arguments

Dim         Dimensionality of data
Nc          Number of modes
Ns          Number of data points
sd0         Component spread
rgam.shape  Shape parameter for Gamma distribution
rgam.scale  Scale parameter for Gamma distribution

Value

Simulated data matrix (samples x features)
**get.dat,NetResponseModel-method**

**Author(s)**
Leo Lahti <leo.lahti@iki.fi>

**References**
See citation('netresponse')

**Examples**
```r
d <- generate.toydata()
```

---

**Description**

Get subnetwork data

**Usage**
```r
## S4 method for signature 'NetResponseModel'
get.dat(model, subnet.id, sample = 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.
- `sample` Define the retrieved samples

**Value**

Subnet data matrix

**Author(s)**
Leo Lahti <leo.lahti@iki.fi>

**References**
See citation('netresponse')

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

Estimate mutual information for node pairs based on the first principal components.

Usage

get.mis(datamatrix, network, delta, network.nodes, G, params)

Arguments

- datamatrix: datamatrix
- network: network
- delta: delta
- network.nodes: network.nodes
- G: G
- params: params

Value

mutual information matrix

Author(s)

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

References

See citation('netresponse')

---

Description

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

Usage

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

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


Examples

```r
# 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)
```
### Description

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

### Usage

```r
## S4 method for signature 'NetResponseModel'
get.subnets(
  model,
  get.names = TRUE,
  min.size = 2,
  max.size = Inf,
  min.responses = 2
)
```

### 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

Examples

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

getqofz,NetResponseModel-method

Sample-to-response matrix of probabilities $P(r|s)$.

Description

Retrieve $P(r|s)$ from NetResponseModel model.

Usage

```
getqofz(model, subnet.id, log = FALSE)
```

Arguments

- `model`: NetResponseModel object.
- `subnet.id`: Subnetwork to investigate.
- `log`: Output in log probabilities.

Details

Calculates probability density for each response on a given sample based on the estimated Gaussian mixture model.

Value

Samples x responses matrix. Each entry is a probability $P(r|s)$.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

```
# qofz <- getqofz(model, subnet.id, log = FALSE)
```
Description

Mainly for internal use. Provide independent models for each node.

Usage

independent.models(datamatrix, params)

Arguments

datamatrix    datamatrix
params       parameters

Value

nodes        Model for each node
C            Costs for individual models

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

res <- independent.models(datamatrix = NULL, params = NULL)

Description

Investigate association of a continuous variable and the modes given a list of groupings
list.responses.continuous.multi

Usage

list.responses.continuous.multi(
    annotation.df,
    groupings,
    method = "t-test",
    pth = Inf,
    verbose = TRUE,
    rounding = NULL
)

Arguments

annotation.df  annotation data.frame with discrete factor levels, rows named by the samples

groupings  Sample mode information. Each element corresponds to one grouping; each grouping lists samples for the modes within that grouping.

method  method for quantifying the association

pth  p-value threshold applied to adjusted p-values

verbose  verbose

rounding  rounding digits

Value

Table listing all associations between the factor levels and responses

Author(s)

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

References

See citation('netresponse')

Examples

res <- list.responses.continuous.multi(annotation.df = NULL, groupings = NULL)

list.responses.continuous.single

Investigate association of a continuous variable and the modes

Description

Investigate association of a continuous variable and the modes.
Usage

list.responses.continuous.single(
  annotation.df,
  groupings,
  method = "t-test",
  pth = Inf,
  verbose = TRUE,
  rounding = NULL,
  adjust.p = TRUE
)

Arguments

annotation.df annotation data.frame with discrete factor levels, rows named by the samples

groupings Sample mode information. Each element corresponds to one of the modes and lists the samples assignment matrix qofz. Alternatively, a vector of mode indices named by the samples can be given.

method method for quantifying the association

pth p-value threshold (for adjusted p-values)

verbose verbose

rounding rounding digits

adjust.p Adjust p-values (this will add p.adj column and remove pvalue column in the output table)

Value

Table listing all associations between the factor levels and responses

Author(s)

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

References

See citation('netresponse')

Examples

res <- list.responses.continuous.single(annotation.df = NULL, groupings = NULL)
list.responses.factor  List significant responses

Description
List significantly associated responses for all factors and levels in the given annotation matrix

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, or a vector of cluster indices named by the samples.
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

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.minimal

List factor responses (minimal)

Description
List significantly associated responses for all factors and levels in the given annotation matrix

Usage
list.responses.factor.minimal(
  annotation.df,
  groupings,
  method = "hypergeometric",
  min.size = 2,
  pth = Inf,
  verbose = TRUE,
  data = NULL,
  rounding = NULL
)

Arguments

  annotation.df  annotation data.frame with discrete factor levels, rows named by the samples
  groupings      List of groupings. Each model should have a sample-cluster assignment matrix qofz, or a vector of cluster indices named by the samples.
  method         method for enrichment calculation
  min.size       minimum sample size for a response
  pth            p-value threshold; applied to adjusted p-value
  verbose        verbose
  data           data (samples x features; or a vector in univariate case)
  rounding       rounding digits

Value
A list with two elements: Table listing all associations between the factor levels and responses; multiple p-value adjustment method

Author(s)
Contact: Leo Lahti <leo.lahti@iki.fi>

References
See citation(‘netresponse’)
list.significant.responses

Listing significant responses

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

Statistics of the significantly associated responses.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

See Also

response.enrichment

Examples

#
listify.groupings  Convert grouping info into a list; each element corresponds to a group and lists samples in that group.

Description

Convert grouping info into a list; each element corresponds to a group and lists samples in that group.

Usage

listify.groupings(groupings, verbose = FALSE)

Arguments

groupings         a list, a vector, or a samplesxmodes assignment matrix
verbatim          verbose

Value

Group list

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

res <- listify.groupings(groupings = NULL)

mixture.model     Mixture model

Description

Fit Gaussian mixture model
mixture.model

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.

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")

Examples

```r
res <- mixture.model(NULL)
```

---

model.stats | model.stats

Description

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

Usage

```r
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


Examples

```
# 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

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

```r
res <- order.responses(models = NULL, sample = NULL)
# - for given sample/s (factor level),
# order responses (across all subnets) by association strength
# (enrichment score); overrepresentation
# order.responses(model, sample, method = 'hypergeometric')
```

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

Usage

```r
data(osmo)
```

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

Source

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

References


Examples

data(osmo)

P.r.s

Description: Probability of mode given a sample (a data vector)
Mainly for internal use; documentation will be provided later. Tools
for calculating densities with Gaussian mixture models.

Description

Arguments: @param dat features x samples data matrix for mixture modeling @param pars Gaussian
mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd
(covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector
giving weight for each component. @param log Logical. Return densities in log domain. @param
scaling Try to avoid floating errors. To be improved later.

Usage

P.r.s(dat, pars, log = TRUE, scaling = 0)

Details

Returns: @return Probability density

Author(s)

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

References

See citation("netresponse")

P.rS

Description: Probability of mode given multiple samples (ie. data ma-
trix) Mainly for internal use; documentation will be provided later.
Tools for calculating densities with Gaussian mixture models.

Description

Arguments: @param dat features x samples data matrix for mixture modeling @param pars Gaussian
mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd
(covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector
giving weight for each component. @param log Logical. Return densities in log domain.
Usage

    P.rs(joint)(dat, pars, log = TRUE)

Details

Returns: @return Probability density

Author(s)

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

References

See citation("netresponse")

---

**Description**

Arguments: @param dat features x samples data matrix for mixture modeling @param pars Gaussian mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd (covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector giving weight for each component. @param log Logical. Return densities in log domain.

Usage

    P.rs.joint(dat, pars, log = TRUE)

Details

Returns: @return Probability density

Author(s)

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

References

See citation("netresponse")
P.rs.joint.individual  

Description: Joint probability density for mode and sample Mainly for internal use; documentation will be provided later. Tools for calculating densities with Gaussian mixture models.

Arguments: 
- @param dat features x samples data matrix for mixture modeling
- @param pars Gaussian mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd (covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector giving weight for each component.
- @param log Logical. Return densities in log domain.

Usage

```r
P.rs.joint.individual(dat, pars, log = TRUE)
```

Details

Returns: @return Probability density

Author(s)

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

References

See citation("netresponse")

P.S

Description: Probability density for sample Mainly for internal use; documentation will be provided later. Tools for calculating densities with Gaussian mixture models.

Arguments: 
- @param dat features x samples data matrix for mixture modeling
- @param pars Gaussian mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd (covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector giving weight for each component.
- @param log Logical. Return densities in log domain.

Usage

```r
P.S(dat, pars, log = TRUE)
```

Details

Returns: @return Probability density
**P.s.individual**

**Description:** Probability density for individual sample. Mainly for internal use; documentation will be provided later. Tools for calculating densities with Gaussian mixture models.

**Arguments:**
- `dat`: features x samples data matrix for mixture modeling
- `pars`: Gaussian mixture model parameters (diagonal covariances); list with elements `mu` (mean vectors), `sd` (covariance diagonals), `w` (weights). The `mu` and `sd` are component x features matrices, `w` is vector giving weight for each component.
- `log`: Logical. Return densities in log domain.

**Usage**

```
P.s.individual(dat, pars, log = TRUE)
```

**Details**

Returns: @return Probability density

**Author(s)**

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

**References**

See citation("netresponse")
P.s.r

Description: Probability density for sample given mode. Mainly for internal use; documentation will be provided later. Tools for calculating densities with Gaussian mixture models.

Arguments:
@param dat features x samples data matrix for mixture modeling
@param pars Gaussian mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd (covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector giving weight for each component.
@param log Logical. Return densities in log domain.

Usage
P.s.r(dat, pars, log = TRUE)

Details
Returns: @return Probability density

Author(s)
Contact: Leo Lahti <leo.lahti@iki.fi>

References
See citation("netresponse")

P.Sr

Description: Probability density for sample group given mode. Mainly for internal use; documentation will be provided later. Tools for calculating densities with Gaussian mixture models.

Arguments:
@param dat features x samples data matrix for mixture modeling
@param pars Gaussian mixture model parameters (diagonal covariances); list with elements mu (mean vectors), sd (covariance diagonals), w (weights). The mu and sd are component x features matrices, w is vector giving weight for each component.
@param log Logical. Return densities in log domain.

Usage
P.Sr(dat, pars, log = TRUE)

Details
Returns: @return Probability density
Author(s)

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

References

See citation("netresponse")

---

**pick.model.pairs**  
*Pick model pairs*

**Description**

Mainly for internal use. Calculate joint model for each node pair

**Usage**

`pick.model.pairs(network, network.nodes, node.models, datamatrix, params)`

**Arguments**

- `network`  
- `network.nodes`  
- `node.models`  
- `datamatrix`  
- `params`  

**Value**

- `model.pairs`  
- `delta`  

**Author(s)**

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

**References**

See citation('netresponse')

**Examples**

```
### pick.model.parameters

**Description**

Pick model parameters

**Usage**

```r
pick.model.parameters(m, nodes)
```

**Arguments**

- `m`: vdp.mixt output
- `nodes`: node names for naming purposes

**Value**

Model parameters

**Author(s)**

Leo Lahti <leo.lahti@iki.fi>

**References**

See citation('netresponse')

**Examples**

```r
pick.model.parameters(m = NULL, nodes = NULL)
```

### PlotMixture

**Description**

Plot mixtures.

**Usage**

```r
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

Value

Used for its side-effects

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse') for citation details.

Examples

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

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

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

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

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

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
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)
PlotMixtureUnivariate  Plot univariate mixtures

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

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

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
cluster.assignments Vector of cluster indices, indicating cluster for each data point
...        Further arguments for plot function.

Value
Used for its side-effects
plotPCA

Author(s)
Leo Lahti <leo.lahti@iki.fi>

References
See citation('netresponse') for citation details.

Examples

# plotMixtureUnivariate(dat, means, sds, ws)

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, npoints = NULL, ...)

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.
npoints Argument to the ellipse function
... 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.
plot_associations

Description

Plot association strength between user-defined category labels and responses in a selected subnetwork. Associations are shown in terms -log10(p) enrichment values for the annotation categories for the responses within the specified subnetwork. No correction for multiple testing.

Usage

plot_associations(
  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_

Value

Used for side effect (plotting).

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse').
plot_data

See Also

plot_responses

Examples

#

plot_data  Plot observed data.

Description

Plotting tool for measurement data. Produces boxplot for each feature in each annotation category for the selected subnetwork.

Usage

plot_data(x, subnet.id, labels, ...)

Arguments

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

Value

ggplot2 plot object

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

See Also

plot_responses

Examples

#
**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**

`plot_expression(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_matrix

Visualize a matrix with one or two-way color scale.

Description

Fast investigation of matrix objects; standard visualization choices are made automatically; fast and easy-to-use but does not necessarily provide optimal visualization.

Usage

plot_matrix(
  mat,  
  type = "twoway",  
  midpoint = 0,  
  palette = NULL,  
  colors = NULL,  
  col.breaks = NULL,  
  interval = 0.1,  
  plot_axes = "both",  
  row.tick = 1,  
  col.tick = 1,  
  cex.xlab = 0.9,  
  cex.ylab = 0.9,  
  xlab = NULL,  
  ylab = NULL,  
  limit.trunc = 0,  
  mar = c(5, 4, 4, 2),  
  ...  
)

Arguments

mat matrix  

String. Specifies visualization type. Options: ‘oneway' (color scale ranges from white to dark red; the color can be changed if needed); ‘twoway' (color scale ranges from dark blue through white to dark red; colors can be changed if needed)

midpoint middle point for the color plot: smaller values are shown with blue, larger are shown with red in type = 'twoway'

palette Optional. Color palette.

colors Optional. Colors.

col.breaks breakpoints for the color palette

interval interval for palette color switches

plot_axes String. Indicates whether to plot x-axis ('x'), y-axis ('y'), or both ('both').

row.tick interval for plotting row axis texts
plot_response

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>col.tick</td>
<td>interval for plotting column axis texts</td>
</tr>
<tr>
<td>cex.xlab</td>
<td>use this to specify distinct font size for the x axis</td>
</tr>
<tr>
<td>cex.ylab</td>
<td>use this to specify distinct font size for the y axis</td>
</tr>
<tr>
<td>xlab</td>
<td>optional x axis labels</td>
</tr>
<tr>
<td>ylab</td>
<td>optional y axis labels</td>
</tr>
<tr>
<td>limit.trunc</td>
<td>color scale limit breakpoint</td>
</tr>
<tr>
<td>mar</td>
<td>image margins</td>
</tr>
<tr>
<td>...</td>
<td>optional parameters to be passed to function 'image', see help(image) for further details</td>
</tr>
</tbody>
</table>

Value

A list with the color palette (colors), color breakpoints (breaks), and palette function (palette.function)

Author(s)

Leo Lahti <microbiome-admin@googlegroups.com>

References

See citation('microbiome')

Examples

```r
mat <- rbind(c(1,2,3,4,5), c(1, 3, 1), c(4,2,2))
plot_matrix(mat, 'twoway', midpoint = 3)
```

Description

Plot a specific transcriptional response for a given subnetwork. TRUE, colors = TRUE, plot_type = 'twopi', ...

Usage

```r
plot_response(
  x,
  mynet,
  mybreaks,
  mypalette,
  plot_names = TRUE,
  colors = TRUE,
  plot_type = "twopi",
  ...
)
```
plot_responses

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


Examples

```r
#tmp <- plot_response(model, mynet,
#  maintext = paste('Subnetwork', subnet.id))
```

plot_responses

Description

Plot the detected transcriptional responses for a given subnetwork. `plot_mode = 'network', xaxis = TRUE, yaxis = TRUE, plot_type = 'twopi', mar = c(5, 4, 4, 2), horiz = TRUE, datamatrix = NULL, scale = FALSE, ...`

Usage

```r
plot_responses(
  x,
  subnet.id,
  nc = 3,
  plot_names = TRUE,
  plot_mode = "network",
  xaxis = TRUE,
)```
plot_responses

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 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)

Description

Plot the color scale used in visualization.

Usage

plot_scale(
  x,
  y,
  m = NULL,
  cex.axis = 1.5,
  label.step = 2,
  interval = 0.1,
  two.sided = TRUE,
  label.start = NULL,
  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.
Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

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

Description

Plot the given subnetwork.

Usage

```r
plot_subnet(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

Description

Function to read network files.

Usage

read.sif(sif.file, format = 'graphNEL', directed = FALSE, header = TRUE, sep = '	', ...)

Arguments

sif.file Name of network file in SIF format.
format Output format: igraph 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 igraph or graphNEL format.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

Examples

#net <- read.sif('network.sif')
remove.negative.edges  remove.negative.edges

Description

Remove edges where the connected nodes correlate negatively

Usage

remove.negative.edges(network, delta, datamatrix)

Arguments

network  network
delta    associated cost function value changes for each node merge
datamatrix  datamatrix

Value

Filtered network

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

#

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(
  total.samples,
  response.samples,
  annotated.samples,
  method = "hypergeometric"
)
**Arguments**

- `total.samples`  All samples in the data
- `response.samples`  Samples in the investigated subset
- `annotated.samples`  Samples at the investigated annotation level for enrichment calculation
- `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**

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

---

**Description**

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

**Usage**

```r
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:

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


Examples

# 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)
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


Examples

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')

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

Examples

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

Description
Split q of z.
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

```r
## S3 method for class 'qofz'
split(qOFz, c, new.c, dat, speedup = TRUE, min.size = 4)
```

Arguments
- `qOFz`: qOFz
- `c`: c
- `new.c`: new.c
- `dat`: dat
- `speedup`: speedup
- `min.size`: min.size

Details
INPUT: data, qOFz, hp_posterior, hp_prior, opts
OUTPUT: list(new.qOFz, new.c); * new.qOFz: posterior over labels including the split clusters. * new.c: index of the newly created cluster. DESCRIPTION: Implements the VDP algorithm step 3a.

Value
object Component must have at least min.size samples to be splitted.'
**Description**

Toy data for NetResponse examples.

**Usage**

data(toydata)

**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**


**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)

**Description**

Mainly for internal use. Calculate joint model for given node pair and update delta accordingly.

**Usage**

update.model.pair(datamatrix, delta, network, edge, network.nodes, G, params, model.nodes, model.pairs)
Arguments

- datamatrix
- delta
- network
- edge
- network.nodes
- G
- params
- model.nodes
- model.pairs

Value

- model.pairs
- delta

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

References
See citation('netresponse')

Examples

#

Description

Accelerated variational Dirichlet process Gaussian mixture.
vdp.mixt

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 split 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, $F_{T+1}$. 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, $F_{T+1}$. 5. Update the posterior using the newly split data. 6. If $F_T - F_{T+1} < \epsilon$ then halt, else set $T := T + 1$ and go to step 2.

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

Value

<table>
<thead>
<tr>
<th>prior</th>
<th>Prior parameters of the vdp-gm model ($qOfz$: priors on observation lables; Mu: centroids; S2: variance).</th>
</tr>
</thead>
<tbody>
<tr>
<td>posterior</td>
<td>Posterior estimates for the model parameters and statistics.</td>
</tr>
<tr>
<td>weights</td>
<td>Mixture proportions, or weights, for the Gaussian mixture components.</td>
</tr>
<tr>
<td>centroids</td>
<td>Centroids of the mixture components.</td>
</tr>
<tr>
<td>sds</td>
<td>Standard deviations for the mixture model components (posterior modes of the covariance diagonals square root). Calculated as $\sqrt{\text{invgam.scale}/(\text{invgam.shape} + 1)}$.</td>
</tr>
<tr>
<td>qOFz</td>
<td>Sample-to-cluster assignments (soft probabilistic associations).</td>
</tr>
<tr>
<td>Nc</td>
<td>Component sizes</td>
</tr>
<tr>
<td>invgam.shape</td>
<td>Shape parameter (alpha) of the inverse Gamma distribution</td>
</tr>
<tr>
<td>invgam.scale</td>
<td>Scale parameter (beta) of the inverse Gamma distribution</td>
</tr>
<tr>
<td>Nparams</td>
<td>Number of model parameters</td>
</tr>
<tr>
<td>K</td>
<td>Number of components in the mixture model</td>
</tr>
<tr>
<td>opts</td>
<td>Model parameters that were used.</td>
</tr>
<tr>
<td>free.energy</td>
<td>Free energy of the model.</td>
</tr>
</tbody>
</table>

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.
vectorize.groupings

Description

Convert grouping info into a vector; each element corresponds to a group and lists samples in that group.

Usage

vectorize.groupings(groupings, verbose = FALSE)

Arguments

groupings a list, a vector, or a samplesxmodes assignment matrix
verbose verbose

References

write.netresponse.results

Value

Indicator vector

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')

Examples

#

write.netresponse.results

Write NetResponse results summary into a file.

Description

Experimental version.

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.

Value

Used for side effects.

Author(s)

Leo Lahti <leo.lahti@iki.fi>

References

See citation('netresponse')
Index

* classes
  NetResponseModel-class, 37
* datasets
  dna, 14
  osmo, 39
* internal
  check.matrix, 9
  filter.netw, 19
  filter.network, 20
  generate.toydata, 22
  getqofz.NetResponseModel-method, 27
  independent.models, 28
  P.r.s, 40
  P.rs, 40
  P.rs.joint, 41
  P.rs.joint.individual, 42
  P.S, 42
  P.s.individual, 43
  P.s.r, 44
  P.Sr, 44
  pick.model.pairs, 45
  pick.model.parameters, 46
  remove.negative.edges, 62
  update.model.pair, 67
* iteration
  detect.responses, 12
  vdp.mixt, 68
* maths
  centerData, 8
* methods
  detect.responses, 12
  vdp.mixt, 68
* misc
  toydata, 67
* package
  netresponse-package, 3
* utilities
  add.ellipse, 4
  bic.mixture, 5
  bic.mixture.multivariate, 6
  bic.mixture.univariate, 7
  centerData, 8
  continuous.responses, 11
  enrichment.list.factor, 15
  enrichment.list.factor.minimal, 16
  factor.responses, 17
  factor.responses.minimal, 18
  find.similar.features, 21
  get.dat.NetResponseModel-method, 23
  get.model.parameters, 24
  get.subnets.NetResponseModel-method, 26
  getqofz.NetResponseModel-method, 27
  list.responses.continuous.multi, 28
  list.responses.continuous.single, 29
  list.responses.factor, 31
  list.responses.factor.minimal, 32
  list.significant.responses, 33
  listify.groupings, 34
  mixture.model, 34
  model.stats, 36
  order.responses, 37
  P.r.s, 40
  P.rS, 40
  P.rs.joint, 41
  P.rs.joint.individual, 42
  P.S, 42
  P.s.individual, 43
  P.s.r, 44
  P.Sr, 44
  plot_associations, 52
  plot_data, 53
  plot_expression, 54
plot_matrix, 55
plot_response, 56
plot_responses, 57
plot_scale, 59
plot_subnet, 60
PlotMixture, 46
PlotMixtureBivariate, 47
PlotMixtureMultivariate, 48
PlotMixtureUnivariate, 50
plotPCA, 51
read.sif, 61
response.enrichment, 62
response2sample, 63
sample2response, 64
set.breaks, 65
vectorize.groupings, 71
write.netresponse.results, 72
[[.,NetResponseModel-method
 (NetResponseModel-class), 37

add.ellipse, 4

bic.mixture, 5
bic.mixture.multivariate, 6
bic.mixture.univariate, 7
bic.select.best.mode, 8

centerData, 8
check.matrix, 9
check.network, 10
continuous.responses, 11

detect.responses, 12, 37
dna, 14

enrichment.list.factor, 15
enrichment.list.factor.minimal, 16

factor.responses, 17
factor.responses.minimal, 18
filter.netw, 19
filter.network, 20
find.similar.features, 21

generate.toydata, 22
generate.toydata, 22
get.dat
 (get.dat,NetResponseModel-method), 23
get.dat,NetResponseModel-method, 23
get.mis, 24

get.model.parameters, 24
get.subnets
 (get.subnets,NetResponseModel-method), 26
get.subnets,NetResponseModel-method, 26
getqofz
 (getqofz,NetResponseModel-method), 27
getqofz,NetResponseModel-method, 27

independent.models, 28

list.responses.continuous.multi, 28
list.responses.continuous.single, 29
list.responses.factor, 31
list.responses.factor.minimal, 32
list.significant.responses, 33
listify.groupings, 34

mixture.model, 34
model.stats, 36

netresponse (netresponse-package), 3
netresponse-package, 3
NetResponseModel-class, 37

order.responses, 37
osmo, 39

P.r.s, 40
P.rs, 40
P.rs.joint, 41
P.rs.joint.individual, 42
P.S, 42
P.s.individual, 43
P.s.r, 44
P.Sr, 44
pick.model.pairs, 45
pick.model.parameters, 46
plot_associations, 52
plot_data, 53
plot_expression, 54
plot_matrix, 55
plot_response, 56
plot_responses, 57
plot_scale, 54, 58, 59
plot_subnet, 60
PlotMixture, 46
PlotMixtureBivariate, 47
INDEX

PlotMixtureMultivariate, 48
PlotMixtureUnivariate, 50
plotPCA, 51

read.sif, 61
remove.negative.edges, 62
response.enrichment, 62
response2sample, 63

sample2response, 64
set.breaks, 65
show, NetResponseModel-method
   (NetResponseModel-class), 37
split.qofz, 66

toydata, 67

update.model.pair, 67

vdp.mixt, 68
vectorize.groupings, 71

write.netresponse.results, 72