Package 'birte'

October 17, 2017

Version 1.12.0

Date 2015-06-16

Title Bayesian Inference of Regulatory Influence on Expression (biRte)

Author Holger Froehlich, contributions by Benedikt Zacher

Maintainer Holger Froehlich <frohlich@bit.uni-bonn.de>

Depends R(>= 3.0.0), RcppArmadillo (>= 0.3.6.1), Rcpp

Imports MASS, limma(>= 3.22.0), glmnet, Biobase, nem, graphics, stats, utils

Reference H. Froehlich, biRte: Bayesian Inference of Context Specific Regulator Activities and Transcriptional Networks, Bioinformatics, 2015, in press.

Suggests knitr

Enhances Rgraphviz

LinkingTo RcppArmadillo, Rcpp

NeedsCompilation yes

SystemRequirements BLAS, LAPACK

Materials Please enable BLAS and LAPACK for armadillo library (see http://arma.sourceforge.net/faq.html). The file config.hpp can be found under library/RcppArmadillo/include/armadillo_bits.

Description Expression levels of mRNA molecules are regulated by different processes, comprising inhibition or activation by transcription factors and post-transcriptional degradation by microRNAs. biRte uses regulatory networks of TFs, miRNAs and possibly other factors, together with mRNA, miRNA and other available expression data to predict the relative influence of a regulator on the expression of its target genes. Inference is done in a Bayesian modeling framework using Markov-Chain-Monte-Carlo. A special feature is the possibility for follow-up network reverse engineering between active regulators.

License GPL (>= 2)

biocViews Microarray, Sequencing, GeneExpression, Transcription, Network, Bayesian, Regression, NetworkInference

LazyLoad yes

VignetteBuilder knitr

R topics documented:

`birteFitRidge` *Fit ridge regression model given a defined set of active regulators.*

Description

Given a most likely configuration of active regulators identified by biRte, this method fits a conventional ridge regression model to explain gene expression. This function is required, if one would like to use MAP based prediction of gene expression instead of Bayesian predictions (see `birtePredict`). To fit the ridge regression model the R-package `ridge` is employed, which provides an efficient tuning of the regularization hyperparameter.

Usage

```
birteFitRidge(model, mRNA.train, ref.cond=1)
```

Arguments

`model`	output of `birteRun`
`mRNA.train`	vector of gene expression values
`ref.cond`	condition to consider

Details

In order to make predictions with the fitted ridge regression model (`birtePredict`) store it into a slot "fit.ridge" of the object returned by `birteRun` and `birteLimma`, respectively.

Value

an object of class "cv.glmnet" (see `cv.glmnet`)

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
sim = simulateData(affinities2)
limmamRNA = limmaAnalysis(sim$dat.mRNA, design=NULL, "treated - control")

# burnin and sampling size is much too small in reality
result = birteLimma(dat.mRNA=sim$dat.mRNA, data.regulators=NULL,
limmamRNA=limmamRNA,
affinities=affinities2, niter=100, nburnin=100, thin=2)

fit.ridge = birteFitRidge(result, sim$dat.mRNA[,1])
```

`birtePredict` *Prediction of gene expression via biRte.*

Description

Given a biRte model, this function makes posterior inference about gene expression data.

Usage

```
birtePredict(model, test.genes, method=c("Bayes", "MAP"), knock.out=NULL)
```

Arguments

`model` output of `birteRun`

`test.genes` Set of gene IDs. Gene IDs should be contained into the defined regulator-target gene network. Note that expression data is generally not required to be available for these genes.

`method` Bayes: estimate expectation of posterior predictive distribution. MAP: Use previously fitted ridge regression model (`birteFitRidge`), which has to be stored into a slot "fit.ridge".

`knock.out` optionally: A character vector of those regulators, which should be removed from the network before making predictions, hence simulating a knock.out

Value

#conditions x #replicates matrix containing data.frame objects with

`gene` gene, for which predictions are made

`mean` expected expression

`sd` SD of predictions (only for method="Bayes")

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
sim = simulateData(affinities2)
limmamRNA = limmaAnalysis(sim$dat.mRNA, design=NULL, "treated - control")

# burnin and sampling size is much too small in reality
result = birteLimma(dat.mRNA=sim$dat.mRNA, data.regulators=NULL,
limmamRNA=limmamRNA,
affinities=affinities2, niter=100, nburnin=100, thin=2)

est = birtePredict(result, rownames(sim$dat.mRNA))
```

---

birteRun *Main interface for Bayesian Inference of Regulatory Influence on Expression (biRte).*

Description

The function birteRun estimates regulator activities from gene expression data plus a given regulator-target gene network via MCMC sampling. The function assumes experimental data to be of one of the following formats: i) expression matrix (#genes x #samples) ii) mRNA log fold changes

In the first case it is now also possible to estimate regulator activity for each individual sample. In addition one can estimate condition specific regulator activity, if there are exactly two conditions.

In addition to miRNA, mRNA and TF expression data, biRte also allows for integrating other data types (e.g. CNV data - data type 'other'). Note that in such a case also the corresponding regulator-target gene relationships have to be defined.

birteLimma is a convenience function, which allows to directly pass results from a previous limma analysis (see `limmaAnalysis`) to birteRun. When working with relative expression levels (log fold changes) birteLimma allows to deal with arbitrary complex statistical designs, including e.g. time as a covariate.

Usage

```
birteRun(dat.mRNA, mRNA.Sigma=NULL, nrep.mRNA=c(5, 5),
    df.mRNA=sum(nrep.mRNA)-2,
    data.regulators=NULL, sigma.regulators=NULL,
    nrep.regulators=NULL, diff.regulators=NULL,
   init.regulators=NULL, theta.regulators=NULL,
    reg.interactions=FALSE, affinities, use.affinities=FALSE,
    niter=100000, nburnin=100000, thin=50,
    potential_swaps=NULL, only_switches=FALSE, only.diff.TFs=TRUE,
    explain.LFC=TRUE,
    single.sample=FALSE, single.sample.estimator=c("mpost", "MAP"),
    model=c("no-plug-in", "all-plug-in"))

birteLimma(dat.mRNA=NULL, limmamRNA,
   data.regulators=NULL, limma.regulators=NULL,
    fdr.regulators=NULL, lfc.regulators=NULL,
init.regulators=NULL, theta.regulators=NULL,
    reg.interactions=FALSE, affinities, use.affinities=FALSE,
   niter=100000, nburnin=100000, thin=50,
    potential_swaps=NULL, only_switches=FALSE, only.diff.TFs=TRUE,
    explain.LFC=TRUE,
    single.sample=FALSE, single.sample.estimator=c("mpost", "MAP"),
    model=c("no-plug-in", "all-plug-in"))
```

Arguments

`dat.mRNA` mRNA expression data matrix. IMPORTANT: Replicates must be ordered according to nrep.mRNA

`mRNA.Sigma` gene expression variances (array data). IMPORTANT: Names have to match the row names in dat.mRNA. If mRNA.Sigma = NULL, birte.run tries to deduce variances from a limma analysis (see `limmaAnalysis`)

`nrep.mRNA` number of replicates per condition.

`df.mRNA` residual degrees of freedom of linear model for mRNA data.

`data.regulators`
list with at most 3 components (miRNA, TF, other). Each component contains one data matrix. IMPORTANT: Samples in data matrices have to be grouped according to conditions. That means first there are all samples from the first condition, then those from the second condition, etc.

`sigma.regulators`
list with at most 3 components (miRNA, TF, other). Each component contains one named vector of expression variances (array data) or dispersion parameters (RNAseq data). IMPORTANT: Names have to fit to the row names of the data matrix.

`nrep.regulators`
list with at most 3 components (miRNA, TF, other). Each component contains the number of replicates per condition

`diff.regulators`
list with at most 3 components (miRNA, TF, other). Each component is a character vector with differentially expressed regulators. Has to be subset of row names of the data matrix

`init.regulators`
list with at most 3 components (miRNA, TF, other). Each component is matrix of #conditions x length(affinities[[regulator type]]): initial states for regulators. In case this matrix is not provided (i.e. NULL) initial states are assumed to be 0. IMPORTANT: column names have to match names(affinities[[regulator type]])

`theta.regulators`
list with at most 3 components (miRNA, TF, other). If single numbers are provided, each component contains the expected fraction of active regulators. If vectors are provided, each vector entry corresponds to the individual probability of a specific regulator to be active. Accordingly, vectors should be named in agreement with the regulator-target gene network. If affinities$other corresponds to interaction terms between regulators, theta.regulators can also be provided as a #regulators x #regulators matrix.

`reg.interactions`
If TRUE, entries of affinities$other are interpreted as interaction terms between regulators.

`affinities` Regulator-target gene interactions. This is a list with at most three components (TF, miRNA, other). Each of these lists again contains a weighted adjacency list representation. See `humanNetworkSimul` for an example. IMPORTANT: gene names used in this network have to match with row names of dat.mRNA. Moreover, regulator names have to fit to row names of the corresponding data matrices. It is assumed that in presence of regulators of type "other", there exist also miRNA regulators. If you want to circumvent this behavior (e.g. you want to use CNVs, but *no* miRNAs), store these regulators in the slot "miRNA".

`use.affinities` Should weights given in the bipartite regulator-target gene graph given a specific meaning? If yes, it is assumed that weights correspond to quantitative influences of regulators on their targets.

`niter` Number of MCMC iterations (AFTER burnin).

`nburnin` Number of MCMC iterations UNTIL burnin is assumed to be finished.

`thin` Thinning of Markov chain: only use every thin's sample for posterior computation.

`potential_swaps`
Pre-computed potential swaps (OPTIONAL, see get_potential_swaps).

`only_switches` Should only switches be performed?

`only.diff.TFs` Should, in case of TF expression data, only the information for differentially expressed TFs be considered? Note that this makes fewer assumption about the relation of mRNA and protein expression data, but typically leads to less conservative results (i.e. more TFs predicted to be active).

`limmamRNA` results of limma analysis for mRNA data according to `limmaAnalysis`

`limma.regulators`
list with at most 3 components (miRNA, TF, other). Each component contains the results of a limma analysis for regulator data according to `limmaAnalysis`

`lfc.regulators` list with at most 3 components (miRNA, TF, other). Each component contains the log fold change cutoff for differential expression. It is assumed to be 0, if not provided.

`fdr.regulators` list with at most 3 components (miRNA, TF, other). Each component contains the FDR cutoff for differential expression (DEFAULT: 0.05).

`explain.LFC` If yes, biRte tries to explain mRNA log fold changes rather than expression levels itself.

`single.sample` If yes, biRte tries to explain mRNA data for each individual sample. The output is a #samples x #regulators matrix.

`single.sample.estimator`
Which type of estimate for regulator activity should be provided? mpost = marginal posterior activity probability for each regulator; MAP = most likely regulator configuration found during MCMC sampling

`model` If "no-plug-in", for marginal log likelihoods are considered for regulator specific expression data. Otherwise, (posterior) variance estimates are used directly.

Value

If single.sample is FALSE, the function returns a list containing the following entries:

`post` #regulators x #conditions matrix containing the marginal probability for each regulator to influence mRNA expression.

`map` #regulators x #conditions matrix containing the regulator configuration with highest joint probability.

`coef` matrix of #conditions x #replicates: Each entry is itself a matrix (embedded into a list) of #coefficients x #effective samples. The data contains the expected regression coefficients.

`log_lik_trace` (Marginal) log-likelihood trace of MCMC sampling.

`eff_sample_size` effective sample size after burnin and thinning

`contains.interactions` TRUE, if affinities$other corresponds to interaction terms between regulators, FALSE otherwise

`explain.LFC` TRUE, if model explains mRNA log fold change, FALSE otherwise

`nburnin` number of burnin iterations - as provided as an argument

`affinities` original regulator-target gene network

`C_cnt` number of conditions

`design` design matrix effectively used for model training

`param` estimated parameters for mRNA precision (i.e. inverse variance) distribution

If single.sample is TRUE the output is a #samples x #regulator matrix.

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
sim = simulateData(affinities2)
limmamRNA = limmaAnalysis(sim$dat.mRNA, design=NULL, "treated - control")
limmamiRNA = limmaAnalysis(sim$dat.miRNA, design=NULL, "treated - control")
limmaTF = limmaAnalysis(sim$dat.TF, design=NULL, "treated - control")

# burnin and sampling size is much too small in reality
result = birteLimma(dat.mRNA=sim$dat.mRNA,
data.regulators=list(miRNA=sim$dat.miRNA, TF=sim$dat.TF),
limmamRNA=limmamRNA, limma.regulators=list(miRNA=limmamiRNA, TF=limmaTF),
affinities=affinities2, niter=100, nburnin=100, thin=2)
plotConvergence(result)
pred = birtePredict(result, rownames(sim$dat.mRNA))
MSE.Bayes = mean((pred[[1]][[1]]$mean - limmamRNA$pvalue.tab[rownames(sim$dat.mRNA),"logFC"])^2)
MSE.Bayes


# real data
library(Biobase)
data(EColiOxygen)
# prepare network
affinities = list(TF=sapply(names(EColiNetwork$TF), function(tf){
w = rep(1, length(EColiNetwork$TF[[tf]]));
names(w)= EColiNetwork$TF[[tf]]; w}))
# prepare data
mydat = exprs(EColiOxygen)
colnames(mydat) = make.names(paste(pData(EColiOxygen)$GenotypeVariation,
pData(EColiOxygen)$GrowthProtocol, sep="."))
```

```
mydat = cbind(mydat[,colnames(mydat) =="wild.type.aerobic"],
exprs(EColiOxygen)[,colnames(mydat) == "wild.type.anaerobic"])

# more realistic sampling
## Not run:
result = birteRun(dat.mRNA=mydat,
nrep.mRNA=c(3,4), affinities=affinities, niter=10000, nburnin=10000)
plotConvergence(result)

## End(Not run)
```

EColiNetwork *Example TF-target graph from Regulon DB.*

Description

This list contains the TF-target graph used in the vignette.

Usage

```
EColiNetwork
```

Format

A list containing the target gene sets of 160 TFs.

Value

A list containing the target gene sets of 160 TFs.

Source

This TF-target graph was taken from (R. Castelo and A. Roverato, 2009). It is a pre-filtered version of RegulonDB 6.1.

References

R. Castelo and A. Roverato. Reverse engineering molecular regulatory networks from microarray data with qp-graphs. *J Comput Biol*, 16(2):213227, Feb 2009.

`EColiOxygen` *Example data set from E. Coli to sample TF activities.*

Description

This data set gives expression values for three experiments of the E. Coli K12 strain under aerobic and three experiments under anaerobic growth. It is used in the vignette to illustrate application of birta to TFs only.

Usage

```
EColiOxygen
```

Format

ExpressionSet

Value

ExpressionSet

Source

The original data comes from (Covert et al., 2004) The normalized data set used here is taken from the qpgraph package by R. Castelo and A. Roverato.

References

M. W. Covert, E. M. Knight, J. L. Reed, M. J. Herrgard, and B. O. Palsson. Integrating high-throughput and computational data elucidates bacterial networks. *Nature*, 429(6987):9296,May 2004.

`estimateNetwork` *Estimate network between active regulators using Nested Effects Models (NEMs).*

Description

Given a biRte model, this function makes posterior inference about possible upstream-downstream relationships between active regulators. This is done based on observed differential expression of putative target genes. The idea is that regulator A acts upstream of regulator B, if differentially expressed targets of B are a subset of those of A.

Usage

```
estimateNetwork(model, thresh=0.1, select=c("marginal", "MAP"), method="pairwise",
de.genes, bootstrap=0, typeII=0.1)
```

Arguments

`model`	biRte model
`thresh`	cutoff for marginal posterior probabilities
`select`	"marginal": select regulators based on marginal posterior probabilities; "MAP": select regulators based on MAP configuration
`method`	algorithm used for NEM based network inference, see `nem`
`de.genes`	set of differentially expressed genes
`bootstrap`	optional: number of bootstrap replicates to draw (non-parameteric bootstrap)
`typeII`	assumed type-II error rate

Value

nem-model

Author(s)

Holger Froehlich

Examples

```
# see vignette
```

`getPotentialSwaps` *Calculate swap partner for TF-/miRNA-target graph. This function is usually only called internally, but may be used to speed up repetitive calls to `birteRun`.*

Description

Calculates for TF-/miRNA-target graph all potential swap partner.

Usage

```
getPotentialSwaps(genesets, perc.overlap.cutoff=0.8, integer.id=TRUE)
```

Arguments

`genesets`	Each entry corresponds to a regulator (miRNA, TF, other) and contains its target genes.
`perc.overlap.cutoff`	Percentage cutoff of minimal overlap between two miRNAs or TFs to be possible swap partner.
`integer.id`	If TRUE, the swap partner are not output as characters, but as integer indices.

Value

A list object, where each element corresponds to one regulator.

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
genesets = c(sapply(affinities2$TF, names),
sapply(affinities2$miRNA, names), sapply(affinities2$other, names))
swaps = getPotentialSwaps(genesets)
```

---

humanNetworkSimul *Subset of regulator-target gene network for human*

Description

The human regulatory network was constructed as follows: For miRNA target gene prediction we used MiRmap (Vejnar and Zdobnov, 2012) and converted the reported scores into z-scores. P-values were then calculated based on the observed approximate normality of z-scores under the null hypothesis and corrected for multiple testing via the FDR method under dependencies of tests (Benjamini and Yekutieli, 2001). In consequence we arrived at 356 miRNAs regulating between 1 and 318 target genes (median: 5).

A TF-target gene network was compiled by computing TF binding affinities to promoter sequences of all human genes according to the TRAP model (Roider et al., 2007) via the author's R implementation. Upstream sequences of genes were retrieved here from the ENSEMBL database via biomaRt (Durinck et al., 2009). We assumed that promoter sequences were located in the range 0 - 2Kbp upstream to the transcription start site of a gene. 556 TRANSFAC (Wingender, 2008; public version) as well as 130 JASPAR (Bryne et al., 2008; accessed Dec. 2011) TFBS matrices were used. As significant we considered those TFBS, for which a FDR < 5% was reported (Benjamini-Yekutieli method).

In a subsequent step from each significant TFBS we extracted the set of TFs, for which the corresponding TFBS matrix had been defined by parsing the original TRANSFAC and JASPAR files, respectively. Ambiguities were corrected via manual curation with the help of the commercial MetaCore software. In consequence we arrived at a TF-target gene network of 344 TFs regulating between 1 and 19,392 target genes (median: 155).

For simulation purposes we randomly selected subset of 999 human genes together with the set of corresponding regulators. After network simplification (`simplify`) we arrived at 13 miRNA and 111 TF clusters.

Usage

```
humanNetworkSimul
```

Format

affinities2 TF and miRNA target gene network. For miRNAs scores from z-scores derived from MiRmap are reported. For TFs -log10(p-values).

Details

- affinities2
  - TFTF-target gene graph – list object
  - miRNAmiRNA-target gene graph – list object

  Both, the TF-target and the miRNA target gene graphs are given in form of a weighted adjacency list representation.

Value

see above

Note

The object 'affinities2' may be extended by one further component 'other' containing additional regulatory information (e.g. CNVs, predefined interaction terms between regulators).

References

Vejnar, C. E. and Zdobnov, E. M. (2012). Mirmap: comprehensive prediction of microrna target repression strength. Nucleic Acids Res, 40(22):11673-11683.

Roider, H. G., Kanhere, A., Manke, T., and Vingron, M. (2007). Predicting transcription factor affinities to dna from a biophysical model. Bioinformatics, 23(2):134-141.

`limmaAnalysis` *Simple limma analysis on expression data with one contrast.*

Description

This is a convenience function, which can be used to perform a limma analysis with one contrast. Subsequently results may be passed to `birteLimma`

Usage

```
limmaAnalysis(dat, design=NULL, contrast)
```

Arguments

`dat`	A matrix or ExpressionSet containing the expression values.
`design`	A design matrix. If no design matrix is provided, the method tries to infer a design matrix by taking column names as a factor. That means the design matrix is constructed according to ~0 + factor(colnames(dat).
`contrast`	Contrast for the linear model. Currently only one contrast is supported. If more contrasts are desired separate biRte analyses should be run for each of them.

Value

Returns a list containing the following entries:

`pvalue.tab`	Containing the result of the topTable function from limma.
`lm.fit`	Linear fit to the model, i.e. output of function 'eBayes'.
`design`	The design used.
`contrast`	The contrasts used.

Author(s)

Holger Froehlich

References

G. K. Smyth. Limma : Linear Models for Microarray Data. *Bioinformatics*, (2005):397-420.

Examples

```
# see birte.run
```

---

plotConvergence *Plot the marginal log-likelihood of the model along MCMC samples (after thinning).*

Description

Plot the marginal log-likelihood of the model along MCMC samples (after thinning).

Usage

```
plotConvergence(res, title="")
```

Arguments

`res`	The output of birte.run (a list object).
`title`	Optional title of the plot.

Value

none

Author(s)

Holger Froehlich

See Also

`birteRun`

Examples

```
# see birteRun
```

proposeInteractions *Propose possible regulator-regulator interactions that could be worthwhile to be tested into the biRte model.*

Description

The purpose is to identify interesting interactions terms for our model. Assuming the design matrix to be binary, the interaction term between two variables X1 and X2 is 1, if X1=1 AND X2=1 (zero, otherwise). The function computes the relative overlap between all pairs of variables X1 and X2 via the Tanimoto-Jaccard index. Variable pairs with cutoff.lower < Tanimoto-Jaccard < cutoff.upper are considered interesting candidates.

Usage

```
proposeInteractions(affinities, cutoff.lower=0.1, cutoff.upper=0.8)
```

Arguments

`affinities`	original regulator-target gene network
`cutoff.lower`	lower range for Tanimoto-Jaccard index
`cutoff.upper`	upper range for Tanimoto-Jaccard index

Value

A list of the same format as affinities$TF and affinites$miRNA: Each list entry corresponds to the intersection of two target gene sets.

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
affinities.int = proposeInteractions(affinities2)
```

simplify *Simplify regulator-target gene network via clustering.*

Description

Several regulators (specifically TFs) might have extremely overlapping target genes. In order to identify clusters of highly similar regulators (mainly TFs) we implemented a network simplification algorithm in biRte: We construct the biadjacency matrix of the complete bipartite regulator target-gene graph and then calculate a single linkage clustering of regulators based on the Tanimoto-Jaccard similarity of their target genes. The dendrogram is cut at a defined height (default: 0.1) to idenfity resulting groups. The algorithm is meant to simplify the inference of active regulators, because the resulting regulator clusters have more dissimilar target gene profiles.

Usage

```
simplify(affinities, cutoff=0.9)
```

Arguments

`affinities`	original regulator-target gene network
`cutoff`	cut dendrogram at height 1 - cutoff (i.e. similarity cutoff)

Value

clustered / simplified network

Author(s)

Holger Froehlich

Examples

```
# artificial data
data(humanNetworkSimul)
affinities2 = simplify(affinities2)
```

`simulateData` *Simulate expression data.*

Description

The function draws expression data from a multivariate normal distribution with block structured co-variance matrix. First, data is drawn for a control condition (no active regulators). Then data is generated for the situation that a certain fraction of regulators is turned ’on’ (treatment condition). Regulator activity states are sampled from a Bernoulli distribution.

Usage

```
simulateData(affinities, nrep = 5, miRNAExpressions = TRUE, fn.targets = 0.1,
fp.targets = 0.2, exp.nTF = 5, exp.nmiR = 5, exp.interact = 5)
```

Arguments

`affinities`	regulator-target gene network (see `humanNetworkSimul`)
`nrep`	number of replicates per condition
`miRNAExpressions`	Should miRNA expression data be simulated?
`fn.targets`	fraction of false negative target predictions (i.e. missing edges per regulator in the bipartite regulator-gene graph)
`fp.targets`	fraction of false positive target predictions
`exp.nTF`	expected number of active TFs
`exp.nmiR`	expected number of active miRNAs
`exp.interact`	expected number of active interaction terms

Details

If active interaction terms should be simulated, a set of possible interaction terms has to be defined in `affinities$other`.

Value

`dat.mRNA`	mRNA data – active regulators are expected to induce a log FC of 1
`dat.miRNA`	miRNA data – active miRNAs are expected to show a log FC of 1
`dat.TF`	TF expression data – active miRNAs are expected to show a log FC of 0.5
`miRNAstates`	simulated miRNA activities in treatment condition
`TFstates`	simulated TF activities in treatment condition
`inter.states`	simulated regulator interaction activities in treatment condition

Author(s)

Holger Froehlich

Examples

```
data(humanNetworkSimul)
sim = simulateData(affinities2)
```

`suggestThreshold` *Automatically suggest suitable threshold for marginal regulator activities.*

Description

The algorithm fits a mixture of a beta(1,beta) and beta(alpha,1) distribution to observed marginal regulator activities. Based on this mixture a cutoff is chosen such that the expected false positive rate is below a defined threshold.

Usage

```
suggestThreshold(prob, fpr=0.001)
```

Arguments

`prob`	marginal probability obtained from `birteRun`
`fpr`	threshold for accepted false positive rate

Value

a cutoff for marginal activity probabilities

Author(s)

Holger Froehlich

References

Froehlich, H. and Klau, G. (2013). Reconstructing Consensus Bayesian Network Structures with Application to Learning Molecular Interaction Networks. In: Beissbarth, T., Kollmar, M., Leha, A., Morgenstern, B., Schultz, A.-K., Waack, S., and Wingender, E., editors, Proc. German Conference on Bioinformatics, Open Access Series in Informatics, pages 46 - 55. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Dagstuhl Publishing, Germany.

Examples

```
freq = 0.2*rbeta(100, 1, 10) + 0.8*rbeta(100, 5, 1)
thresh = suggestThreshold(freq)
```

TFexpr *Transcription factor expression values for the aerobic-anaerobic growth experiment.*

Description

This data set gives expression values for the 160 TF of the TF-target graph EColiNetwork used in the vignette.

Usage

```
TFexpr
```

Format

ExpressionSet. Rownames in the assayData correspond to entries in TF-target graph.

Value

ExpressionSet. Rownames in the assayData correspond to entries in TF-target graph.

Source

See EColiOxygen and EColiNetwork (see reference for details).

References

M. W. Covert, E. M. Knight, J. L. Reed, M. J. Herrgard, and B. O. Palsson. Integrating high-throughput and computational data elucidates bacterial networks. *Nature*, 429(6987):92-96, May 2004.

Index