

Package ‘BayesNetBP’

February 16, 2018

Type Package

Title Bayesian Network Belief Propagation

Version 1.3.0

Date 2018-02-15

Author Han Yu

Maintainer Han Yu <hyu9@buffalo.edu>

Depends R (>= 3.3.0), stats, graphics, utils

Imports Rgraphviz, gRbase, igraph, RColorBrewer, fields, doBy, qtlnet,
methods, qtl, graph, bnlearn

Suggests shiny, googleVis

URL <https://github.com/hyu-ub/BayesNetBP>

BugReports <https://github.com/hyu-ub/BayesNetBP/issues>

LazyData true

Description Belief propagation methods in Bayesian Networks to propagate evidence through the network. The implementation of these methods are based on the article: Cowell, RG (2005). Local Propagation in Conditional Gaussian Bayesian Networks <<http://www.jmlr.org/papers/volume6/cowell05a/>>.

License GPL-2

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-16 15:22:08 UTC

R topics documented:

AbsorbEvidence	2
bn_to_graphNEL	3
chest	4
ClusterTree-class	4

ClusterTreeCompile	5
ComputeKLDs	6
ElimTreeInitialize	7
emission	8
FactorQuery	9
GetValue	10
Initializer	11
liver	12
LocalModelCompile	12
Marginals	13
PlotCGBN	15
PlotMarginals	16
PlotTree	17
Propagate	17
runBayesNetApp	18
Sampler	19
SummaryMarginals	20
toytree	21
yeast	22

Index	23
--------------	-----------

AbsorbEvidence	<i>Absorb evidence into the model</i>
----------------	---------------------------------------

Description

Absorb evidence into the model

Usage

AbsorbEvidence(tree, vars, values)

Arguments

tree	a ClusterTree object
vars	a vector of the names of observed variables
values	a list of observed values of the variables. Aside from a single value, The element of the list can also be a vector of likelihood values

Details

Absorb multiple types and pieces of evidences into a [ClusterTree](#) object. The discrete compartment of the [ClusterTree](#) will be automatically propagated after evidence absorption, so that the object will be ready for making queries and absorbing additional evidence.

Value

`ClusterTree` object with the evidence absorbed

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

Lauritzen, S. L., & Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 157-224.

Examples

```
data(liver)
tree.init.p <- Initializer(dag=liver$dag, data=liver$data,
                          node.class=liver$node.class,
                          propagate = TRUE)
tree.post <- AbsorbEvidence(tree.init.p, c("Nr1i3", "chr1_42.65"), list(1,"1"))
```

bn_to_graphNEL	<i>Convert a bn object to graphNEL object</i>
----------------	---

Description

Convert a bn object to graphNEL object while removing isolated nodes

Usage

```
bn_to_graphNEL(graph_bn)
```

Arguments

`graph_bn` a bn object of Bayesian network

Value

a graphNEL object

Author(s)

Han Yu

chest	<i>A simulated data from the Chest Clinic example</i>
-------	---

Description

Simulated data from the Chest Clinic example (also known as the Asia example) from Lauritzen and Spiegelhalter, 1988.

Usage

```
data(chest)
```

Format

The data set chest contains two objects:

data a data.frame object of 10000 observations and 8 discrete variables.

dag a graphNEL object specifying the network structure.

References

Lauritzen and Spiegelhalter (1988) Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (with Discussion). J. Roy. Stat. Soc. 50, p. 157-224.

Dethlefsen, C., & Hojsgaard, S. (2005). A common platform for graphical models in R: The gRbase package. Journal of Statistical Software, 14(17), 1-12.

ClusterTree-class	<i>An S4 class of the cluster tree.</i>
-------------------	---

Description

The ClusterTree object is the computational object for belief propagation.

Slots

cluster A vector storing the name of clusters in the cluster tree.

node A vector storing the name of nodes in the Bayesian network.

graph A list of two graphNEL objects: \$dag stores the graph of Bayesian network, \$tree stores the graph of the cluster tree.

member A named list of the node cluster membership.

parent A named vector indicating the parent node of a given cluster in the cluster tree.

cluster.class A named vector of logical values indicating whether a cluster is continuous or discrete.

`node.class` A named vector of logical values indicating whether a node is continuous or discrete.
`assignment` A named list indicating the assignment of discrete nodes discrete clusters.
`propagated` A logical value indicating whether the discrete compartment has been propagated.
`cpt` A named list of the conditional probability tables.
`jpt` A named list of the joint distribution tables.
`lppotential` A named list of the linear predictor potentials assigned to each cluster in the lppotential slots.
`postbag` A named list of the linear predictor potentials assigned to each cluster in the postbag slots.
`activeflag` A named vector of logical values indicating whether a continuous cluster is active.
`absorbed.variables` A vector of characters indicating variables observed with hard evidence.
`absorbed.values` A list indicating the values of the variables observed with hard evidence.
`absorbed.soft.variables` A vector of characters indicating variables observed with soft or likelihood evidence.
`absorbed.soft.values` A list of the likelihoods of the soft or likelihood evidence.

ClusterTreeCompile *Compile the cluster tree*

Description

Get the cluster sets and strong semi-elimination tree from the Bayesian network

Usage

```
ClusterTreeCompile(dag, node.class)
```

Arguments

<code>dag</code>	a graphNEL object of the Bayesian network
<code>node.class</code>	a named vector of logical values, TRUE if node is discrete, FALSE if otherwise

Details

This function forms the cluster sets and the semi-elimination tree graph from the Bayesian network. The procedures include acquiring the elimination order, moralization, triangulation, obtaining cluster sets, forming strong elimination tree and strong semi-elimination tree. The cluster sets and the semi-elimination tree are required to initialize the cluster tree.

Value

`tree.graph` a graphNEL object of semi-elimination tree.
`dag` a graphNEL object of original Bayesian network.
`cluster.sets` a list of members of each cluster.
`node.class` a named vector of logical values, TRUE if node is discrete, FALSE if otherwise
`elimination.order` a vector of node names sorted by the elimination order.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also[ElimTreeInitialize](#)**Examples**

```
data(liver)
cst <- ClusterTreeCompile(dag=liver$dag, node.class=liver$node.class)
```

 ComputeKLDs

Compute signed and symmetric Kullback-Leibler divergence

Description

Compute signed and symmetric Kullback-Leibler divergence of variables over a spectrum of evidence

Usage

```
ComputeKLDs(tree, var0, vars, seq, pbar = TRUE, method = "gaussian")
```

Arguments

tree	a ClusterTree object
var0	the variable to have evidence absorbed
vars	the variables to have divergence computed
seq	a vector of numeric values as the evidences
pbar	logical(1) whether to show progress bar
method	method for divergence computation: gaussian for Gaussian approximation, for Monte Carlo integration

Details

Compute signed and symmetric Kullback-Leibler divergence of variables over a spectrum of evidence. The signed and symmetric Kullback-Leibler divergence is also known as Jeffery's signed information (JSI) for continuous variables.

Value

a data.frame of the divergence

Author(s)

Han Yu

Examples

```
## Not run:
data(liver)
tree.init.p <- Initializer(dag=liver$dag, data=liver$data,
                          node.class=liver$node.class,
                          propagate = TRUE)
klds <- ComputeKLDs(tree=tree.init.p, var0="Nr1i3",
                   vars=setdiff(tree.init.p@node, "Nr1i3"),
                   seq=seq(-3,3,0.5))

head(klds)

## End(Not run)
```

ElimTreeInitialize *Initialize the elimination tree*

Description

Initialize the elimination tree with the local models

Usage

```
ElimTreeInitialize(tree, dag, model, node.sets, node.class)
```

Arguments

tree	a graphNEL object of the elimination tree
dag	a graphNEL object of the Bayesian network
model	a list of local models built from LocalModelCompile function
node.sets	a list of cluster sets obtained from ClusterTreeCompile function
node.class	a named vector of logical values, TRUE if node is discrete, FALSE if otherwise

Details

Initialize the elimination tree with the local models

Value

[ClusterTree](#) object with the local models incorporated

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also

The functions [ClusterTreeCompile](#) and [LocalModelCompile](#) provide necessary objects to obtain [ClusterTree](#) object by initializing the elimination tree through this function.

Examples

```
data(liver)
cst <- ClusterTreeCompile(dag=liver$dag, node.class=liver$node.class)
models <- LocalModelCompile(data=liver$data, dag=liver$dag, node.class=liver$node.class)
tree.init <- ElimTreeInitialize(tree=cst$tree.graph,
                               dag=cst$dag,
                               model=models,
                               node.sets=cst$cluster.sets,
                               node.class=cst$node.class)
```

emission

A ClusterTree Example of Emission Model

Description

A propagated [ClusterTree](#) object named emission. This model contains nine variables, including three discrete: Filter State (Fs), Waste Type (W), Burning Regimen (B) and six continuous variables: Metals in Waste (Min), Metals Emission (Mout), Filter Efficiency (E), Dust Emission (D), CO2 Concentration in Emission (C), Light Penetrability (L).

Usage

```
data(emission)
```

Format

The data set contains a propagated [ClusterTree](#) object emission ready for evidence absorption and making queries.

References

Lauritzen, Steffen L., and Frank Jensen. Stable local computation with conditional Gaussian distributions. *Statistics and Computing* 11.2 (2001): 191-203.

Description

Obtain the joint, marginal, and conditional distributions of discrete variables

Usage

```
FactorQuery(tree, vars = c(), mode = c("joint", "conditional", "list"))
```

Arguments

tree	a ClusterTree object
vars	the variables to be queried
mode	type of desired distribution

Details

Query the joint distribution of any combination of discrete variables when mode is "joint", or conditional distribution of a discrete variable. The mode "list" return a list of variable combinations, such that joint distributions of any subset of them are ready for extraction. Queries outside this list are also supported but may take longer computing time. This function will also return marginal distribution if only one variable is queried.

Value

data.frame object specifying a joint or conditional distribution.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

Examples

```
data(chest)
dag <- chest$dag
node.class <- rep(TRUE, length(dag@nodes))
names(node.class) <- dag@nodes
tree.init.p <- Initializer(dag=dag, data=chest$data,
                          node.class=node.class,
                          propagate=TRUE)
```

```
# joint distribution
FactorQuery(tree=tree.init.p, vars=c("tub", "xray", "dysp", "asia"), mode="joint")

# conditional distribution
FactorQuery(tree=tree.init.p, vars=c("xray"), mode="conditional")
```

GetValue

Possible values of a discrete variable

Description

Obtain all the possible values of a discrete variable.

Usage

```
GetValue(tree, var, message = TRUE)
```

Arguments

tree	a ClusterTree object
var	the variables to be queried
message	type of desired distribution

Value

a vector of the possible values of discrete variable. If the variable is continuous, the returned value will be NULL.

Author(s)

Han Yu

Examples

```
data(toytree)
GetValue(toytree, "HDL")
```

Initializer	<i>Initialize a ClusterTree object</i>
-------------	--

Description

Initialize a ClusterTree object

Usage

```
Initializer(dag, data, node.class, propagate = TRUE)
```

Arguments

dag	a graphNEL object of the Bayesian network
data	a data.frame object
node.class	a named vector of logical values, TRUE if node is discrete, FALSE if otherwise
propagate	logical TRUE if the discrete part of the ClusterTree to be propagated

Details

A wrapper function to initialize a [ClusterTree](#) object. It combines the functions of [ClusterTreeCompile](#), [LocalModelCompile](#), [ElimTreeInitialize](#) and [Propagate](#), thus initialize the [ClusterTree](#) object in a single step.

Value

[ClusterTree](#) object

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also

[ClusterTreeCompile](#), [LocalModelCompile](#), [ElimTreeInitialize](#), [Propagate](#)

Examples

```
data(liver)
tree.init.p <- Initializer(dag=liver$dag, data=liver$data,
                          node.class=liver$node.class,
                          propagate = TRUE)
```

liver

Mus Musculus HDL QTL data from Leduc et. al. (2012)

Description

Liver QTL data was obtained from a F2 inner-cross between inbred MRL/MpJ and SM/J strains of mice.

Usage

```
data(liver)
```

Format

The data set liver contains three objects: the data, a learned Bayesian network structure and vector specifying node type. The fields are described as follows:

`data` a `data.frame` object that contains 280 samples (rows) and 15 variables: genotype data (genotype states at 5 SNP markers) and phenotype data (HDL levels and normalized expression values of 10 genes). Three of these phenotypes are dichotomized, including Cyp2b10, Spgl1 and HDL. Genotypes and dichotomized phenotypes are of class `factor` and continuous phenotypes are of class `numeric`.

`dag` a `graphNEL` object, which is the network structure learned by `qtlnet` package.

`node.class` a named vector of `logical` values indicating whether each node is discrete.

References

Leduc MS, Blair RH, Verdugo RA, Tsaih SW, Walsh K, Churchill GA, Paigen B.(2012). "Using bioinformatics and systems genetics to dissect HDL-cholesterol genetics in an MRL/MpJ x SM/J intercross." *J Lipid Res.*, 6, 1163-75.

LocalModelCompile*Model compilation*

Description

Compile the local models

Usage

```
LocalModelCompile(data, dag = NULL, node.class = NULL)
```

Arguments

<code>data</code>	a <code>data.frame</code> object or a <code>qtlnet</code> object
<code>dag</code>	NULL if <code>data</code> is <code>qtlnet</code> object, or a <code>graphNEL</code> object of conditional Gaussian Bayesian network if <code>data</code> is <code>data.frame</code> .
<code>node.class</code>	NULL if <code>data</code> is <code>qtlnet</code> object, or a vector of logical values named by node names, TRUE for discrete, FALSE for continuous variables if <code>data</code> is <code>data.frame</code> .

Details

This function compiles the local models, including the conditional probability tables for discrete variables, and linear predictor potentials for continuous variables.

Value

`pots` a list of discrete potentials (conditional probability tables) for each discrete variable.
`bags` a list of sets of continuous potentials (lppotentials), each set for a continuous variables.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also

[ElimTreeInitialize](#)

Examples

```
data(liver)
models <- LocalModelCompile(data=liver$data, dag=liver$dag, node.class=liver$node.class)
```

Marginals

Obtain marginal distributions

Description

Get the marginal distributions of multiple variables

Usage

```
Marginals(tree, vars)
```

Arguments

tree a [ClusterTree](#) object
 vars a vector of variables for query of marginal distributions

Details

Get the marginal distributions of multiple variables. The function `Marginals` returns a list of marginal distributions. The marginal distribution of a discrete variable is a named vector of probabilities. Meanwhile, the marginal distributions of continuous variables in a CG-BN model are mixtures of Gaussian distributions. To fully represent this information, the marginal of a continuous variable is represented by a `data.frame` with three columns to specify parameters for each Gaussian distribution in the mixture, which are

mean the mean value of a Gaussian distribution.
 sd the standard deviation of a Gaussian distribution.
 n the number of Gaussian mixtures

Value

marginals a list of marginal distributions
 types a named vector indicating the types of the variables whose marginals are queried: TRUE for discrete, FALSE for continuous.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also

[PlotMarginals](#) for visualization of the marginal distributions, [SummaryMarginals](#) for summarization of the marginal distributions of continuous variables.

Examples

```
data(liver)
tree.init.p <- Initializer(dag=liver$dag, data=liver$data,
                          node.class=liver$node.class,
                          propagate = TRUE)
tree.post <- AbsorbEvidence(tree.init.p, c("Nr1i3", "chr1_42.65"), list(1,"1"))
marg <- Marginals(tree.post, c("HDL", "Ppap2a"))
marg$marginals$HDL
head(marg$marginals$Ppap2a)
```

`PlotCGBN`*Plot the Bayesian network*

Description

Plot and compare two Bayesian networks with different evidence(s) absorbed and propagated.

Usage

```
PlotCGBN(tree.1, tree.2, fontsize = NULL, pbar = FALSE, plotting = TRUE)
```

Arguments

<code>tree.1</code>	a ClusterTree
<code>tree.2</code>	a ClusterTree
<code>fontsize</code>	font size for the node labels
<code>pbar</code>	logical(1) whether to show progress bar
<code>plotting</code>	logical(1) whether to output plot

Details

Network visualization of the node-specific differences between Bayesian Networks with the same topology, but evidence that has been absorbed and propagated. The change of marginal distribution of each node is measured by signed and symmetric Kullback-Leibler divergence. The sign indicates the direction of change, with `tree.1` considered as the baseline. The magnitude of the change is reflected by the value. Nodes that are white are d-separated from the evidence.

Value

a plot of Bayesian network
a vector of signed symmetric Kullback-Leibler divergence

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

Examples

```
## Not run:
data(toytree)
tree.post <- AbsorbEvidence(toytree, c("Nr1i3"), list(1))
PlotCGBN(tree.1=toytree, tree.2=tree.post)

## End(Not run)
```

PlotMarginals

Plot the marginal distributions

Description

Plot the marginal distributions.

Usage

```
PlotMarginals(marginals, groups = NULL)
```

Arguments

marginals	the marginal distributions returned by Marginals for plotting
groups	names of the marginals to be shown on plots

Details

Plot the marginal distributions. Marginals of discrete variables are plotted as bar plots, while those of continuous variables as density plots.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

See Also

[Marginals](#)

Examples

```
data(toytree)
marg <- Marginals(toytree, c("Neu1", "Nr1i3", "chr1_42.65", "Spg11"))
PlotMarginals(marginals=marg, groups=NULL)
```

PlotTree

Plot the cluster tree

Description

Plot the structure of a [ClusterTree](#) object

Usage

```
PlotTree(tree, color = "gray90")
```

Arguments

tree	a ClusterTree object
color	nodes color

Details

Plot the structure of `clustertree` object, with the nodes labeled by corresponding elimination node. The circles represent continuous clusters, while the boxes represent discrete clusters.

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

Examples

```
data(toytree)
PlotTree(toytree)
```

Propagate

Propagate the cluster tree

Description

This function propagates the discrete compartment of a [ClusterTree](#) object.

Usage

```
Propagate(tree)
```

Arguments

tree an initialized `ClusterTree` object

Details

The discrete compartment must be propagated to get the joint distributions of discrete variables in each discrete clusters. A `ClusterTree` object must be propagated before absorbing evidence and making queries.

Value

a `ClusterTree` object

Examples

```
data(liver)
tree.init <- Initializer(dag=liver$dag, data=liver$data,
                        node.class=liver$node.class,
                        propagate = FALSE)
tree.init@propagated
tree.init.p <- Propagate(tree.init)
tree.init.p@propagated
```

runBayesNetApp

Launch the BayesNetBP Shiny App

Description

Launch the BayesNetBP Shiny App

Usage

```
runBayesNetApp(launch.browser = TRUE)
```

Arguments

launch.browser logical(1) whether launch the App in browser

Details

The function runBayesNetApp launches the Shiny App accompanied with this package. The app loads the toytree example by default and allows users to load customized `ClusterTree` object. In order to use this feature, a `ClusterTree` object should be built, propagated and named `tree.init.p`, and then saved as a `.RDATA` file. This file can be read in by the app.

The console of BayesNetBP Shiny App comprises three panels. The first part controls the model loading and network layouts. It also allows user to subset the network to facilitate visualization. The Expand function can trace the ancestors, descendants, or both, of a selected node in a stepwise manner. The expanded nodes will be colored orange. By clicking Add to list, the expanded nodes will be selected and be purple. The user can continue selecting other nodes by using Expand and Add to list functions at this stage. After selecting desired node sets, the user can subset the graph by the Subset function. The nodes in subsetted graph retain all properties before subsetting, including their colors and divergence.

The second panel is used for absorption of fixed and hard evidences. The users can add multiple pieces of evidence to a list and absorb them into the model simultaneously. The nodes with evidence absorbed will be colored green when the absorption is complete. Marginals of the nodes can be queried as density or bar plots by node types. If a set of evidence has been absorbed, the marginals both before and after absorption will be returned to facilitate comparison. To query the marginals, the user can select the node of interest in the graph, and then click Plot Marginals. The Shift in Marginals function computes the signed and symmetric Kullback-Liebler divergence for all applicable nodes in the network, and colors the nodes in a similar manner as the function [PlotCGBN](#).

The function for systematic assessment of variable marginal shifts is provided in the third panel. It allows user to specify which node to absorb the spectrum of evidence in a menu, and to select whose divergence to be calculated by firstly selecting the node on the graph and then clicking Add to Plot List. Alternatively, the user can use Add All function to select all applicable nodes into the plotting list. The result is visualized in an interactive plot.

Author(s)

Han Yu

Examples

```
## Not run:
# load or install required packages to run App
library("shiny")
library("googleVis")
library("devtools")
devtools::install_github("cytoscape/r-cytoscape.js")
# run the App in browser
runBayesNetApp(launch.browser=TRUE)

## End(Not run)
```

Description

Sampling from the joint distribution of all applicable nodes in the Bayesian network.

Usage

```
Sampler(tree, n)
```

Arguments

tree a [ClusterTree](#) object
n a integer number of observations to generate

Value

a dataframe of generated data

Author(s)

Han Yu

References

Cowell, R. G. (2005). Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6(Sep), 1517-1550.

Examples

```
data(toytree)  
Sampler(tree = toytree, n = 10)
```

SummaryMarginals

Summary a continuous marginal distribution

Description

This function summary the marginal distributions of continuous variables by outputting the mean, standard deviation, and number of subpopulations

Usage

```
SummaryMarginals(marginals)
```

Arguments

marginals the marginal distributions obtained from [Marginals](#) function

Value

a `data.frame` object containing information about the marginal distributions for continuous variables. The marginal distributions of continuous variables in a CG-BN model are mixtures of Gaussian distributions. Therefore, besides the mean and standard deviation, the object has an additional column to specify the number of Gaussian mixtures.

`mean` the mean value of a Gaussian distribution.

`sd` the standard deviation of a Gaussian distribution.

`n` the number of Gaussian distributions in the mixture.

See Also

[Marginals](#)

Examples

```
data(liver)
tree.init.p <- Initializer(dag=liver$dag, data=liver$data,
                          node.class=liver$node.class,
                          propagate = TRUE)
marg <- Marginals(tree.init.p, c("HDL", "Ppap2a", "Neu1"))
SummaryMarginals(marginals=marg)
```

toytree

A ClusterTree Example of Liver Model

Description

A propagated [ClusterTree](#) object named `toytree`, obtained from [liver](#) QTL data.

Usage

```
data(toytree)
```

Format

The data set contains a propagated [ClusterTree](#) object `toytree`, which is ready for evidence absorption and making queries.

yeast

Saccharomyces Cerevisiae eQTL data from Kruglak et. al. (2005)

Description

eQTL data from 112 F1 segregants from a cross between BY4716 and RM11-1a strains of *Saccharomyces Cerevisiae*.

Usage

`data(yeast)`

Format

The data set yeast is a data frame of 112 observations of 50 variables: genotype data (genotype states at 12 SNP markers) and phenotype data (normalized and discretized expression values of 38 genes). Both genotypes and phenotypes are of class factor.

Details

The yeast dataset is a subset of the widely studied yeast expression dataset comprising of 112 F1 segregants from a cross between BY4716 and RM11-1a strains of *Saccharomyces Cerevisiae*. The original dataset consists of expression values reported as $\log_2(\text{sample}/\text{BY reference})$ for 6216 genes. The data can be accessed in Gene Expression Omnibus (GEO) by accession number (GSE1990). After linkage analysis and filtering based on location and significance of QTL, a final set of 38 genes and their corresponding 12 SNP markers were identified and included in the yeast dataset. The gene expression values are discretized around the median and have two states, 1 (above or equal to median) and -1 (below median). There are two genotype states: 1 or 2. Thus the final dataset is a data frame of 112 observations (genotype) of 12 variables (SNP markers) and normalized gene expression of 38 variables (genes).

References

Brem RB, Kruglyak L. The landscape of genetic complexity across 5,700 gene expression traits in yeast. Proc Natl Acad Sci U S A 2005 Feb 1;102(5):1572-7.

Brem RB, Storey JD, Whittle J, Kruglyak L. Genetic interactions between polymorphisms that affect gene expression in yeast. Nature 2005 Aug 4;436(7051):701-3.

Index

AbsorbEvidence, [2](#)

bn_to_graphNEL, [3](#)

chest, [4](#)

ClusterTree, [2](#), [3](#), [6–11](#), [14](#), [15](#), [17](#), [18](#), [20](#), [21](#)

ClusterTree-class, [4](#)

ClusterTreeCompile, [5](#), [7](#), [8](#), [11](#)

ComputeKLDs, [6](#)

ElimTreeInitialize, [6](#), [7](#), [11](#), [13](#)

emission, [8](#)

FactorQuery, [9](#)

GetValue, [10](#)

Initializer, [11](#)

liver, [12](#), [21](#)

LocalModelCompile, [7](#), [8](#), [11](#), [12](#)

Marginals, [13](#), [16](#), [20](#), [21](#)

PlotCGBN, [15](#), [19](#)

PlotMarginals, [14](#), [16](#)

PlotTree, [17](#)

Propagate, [11](#), [17](#)

runBayesNetApp, [18](#)

Sampler, [19](#)

SummaryMarginals, [14](#), [20](#)

toytrees, [21](#)

yeast, [22](#)