

Package ‘bnstruct’

September 22, 2014

Type Package

Title Bayesian network structure learning from data with missing values

Version 1.0

Date 2012-02-13

Depends R (>= 2.10), bitops, igraph, Matrix, methods

Suggests Rgraphviz

Author Francesco Sambo

Maintainer Francesco Sambo <francesco.sambo@dei.unipd.it>

Description More about what it does (maybe more than one line)

License GPL (>=2) | file LICENSE

Encoding UTF-8

R topics documented:

add.observations<-	3
asia	4
asia_10000	5
belief.propagation	5
bn	6
BN-class	7
bn<-	8
BNDataset-class	8
boots	10
boots<-	11
bootstrap	11
build.junction.tree	12
child	13
child_NA_5000	13
cpts	14

cpts<-	15
dag	15
dag<-	16
data.file	16
data.file<-	17
discreteness	17
discreteness<-	18
em	18
get.boot	19
get.data	20
get.imputed.data	21
get.most.probable.values	21
get.raw.data	22
has.boots	23
has.data	23
has.imp.boots	24
has.imputed.data	25
has.raw.data	25
header.file	26
header.file<-	27
imp.boots	28
imp.boots<-	28
impute	29
imputed.data<-	29
InferenceEngine-class	30
jpts	31
jpts<-	32
jt.cliques	32
jt.cliques<-	33
junction.tree	33
junction.tree<-	34
knn.impute	34
layering	35
learn.params	36
learn.structure	37
marginals	38
name	39
name<-	40
node.sizes	40
node.sizes<-	41
num.boots	41
num.boots<-	42
num.items	42
num.items<-	43
num.nodes	43
num.nodes<-	44
num.variables	44
num.variables<-	45

`add.observations<-` 3

observations	45
observations<-	46
plot	47
print	47
query	48
raw.data<-	49
read.bif	49
read.dataset	50
read.dsc	51
save.to.eps	52
show	52
test.updated.bn	53
updated.bn	53
updated.bn<-	54
variables	54
variables<-	55
wpdag	56
wpdag<-	56
Index	57

<code>add.observations<-</code>	<i>add further evidence to an existing list of observations of an InferenceEngine.</i>
------------------------------------	--

Description

Add a list of observations to an `InferenceEngine` that already has observations, using a list composed by the two following vectors:

- `observed.vars` vector of observed variables;
- `observed.vals` vector of values observed for the variables in `observed.vars` in the corresponding position.

Usage

```
add.observations(x) <- value
```

```
## S4 replacement method for signature 'InferenceEngine'  
add.observations(x) <- value
```

Arguments

<code>x</code>	an InferenceEngine .
<code>value</code>	the list of observations of the InferenceEngine .

Details

In case of multiple observations of the same variable, the last observation is the one used, as the most recent.

See Also

[observations<-](#)

asia

load Asia dataset.

Description

Wrapper for a loader for the Asia dataset, with only raw data.

Usage

```
asia()
```

Details

The dataset has 10000 items, no missing data, so no imputation needs to be performed.

Value

a BNDataset containing the Child dataset.

See Also

[asia_10000](#)

Examples

```
dataset <- asia()
print(dataset)
```

asia_10000	Asia <i>dataset</i> .
------------	-----------------------

Description

The Asia dataset contains 10000 complete (no missing data, no latent variables) randomly generated items of the Asia Bayesian Network. No imputation needs to be performed, so only raw data is present.

Format

a [BNDataset](#) with raw data slow filled.

Details

The data the BNDataset object is built from is located in files `pkg_folder/extdata/asia_10000.header` and `pkg_folder/extdata/asia_10000.data`.

References

S. Lauritzen, D. Spiegelhalter. Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 50(2):157-224, 1988.

See Also

[asia](#)

<code>belief.propagation</code>	<i>perform belief propagation.</i>
---------------------------------	------------------------------------

Description

Perform belief propagation for the network of an InferenceEngine, given a set of observations when present. In the current version of bnstruct, belief propagation can be computed only over a junction tree.

Usage

```
belief.propagation(ie, net = NULL, observed.vars = NULL,
  observed.vals = NULL, return.potentials = FALSE, ...)

## S4 method for signature 'InferenceEngine'
belief.propagation(ie, net = NULL,
  observed.vars = NULL, observed.vals = NULL, return.potentials = FALSE,
  ...)
```

Arguments

ie an [InferenceEngine](#) object.
 net a [BN](#) object.
 observed.vars list of observed variables.
 observed.vals values taken by variables listed in observed.vars.
 return.potentials if TRUE only the potentials are returned, instead of the default [BN](#).
 ... potential further arguments of methods.

Value

updated [InferenceEngine](#) object.

Examples

```
## Not run:
dataset <- BNdataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
bn <- BN(dataset)
ie <- InferenceEngine(bn)
ie <- belief.propagation(ie)

observations(ie) <- list("observed.vars"=("A","G","X"), "observed.vals"=c(1,2,1))
belief.propagation(ie)

## End(Not run)
```

bn	<i>get the BN object contained in an InferenceEngine.</i>
----	---

Description

Return a network contained in an [InferenceEngine](#).

Usage

```
bn(x)

## S4 method for signature 'InferenceEngine'
bn(x)
```

Arguments

x an [InferenceEngine](#).

Value

the [BN](#) object contained in an [InferenceEngine](#).

BN-class

*BN class.***Description**

BN class.

Instantiate a [BN](#) object.**Usage**

```
## S4 method for signature 'BN'
initialize(.Object, dataset = NULL, ...)

BN(dataset = NULL, ...)
```

Arguments

<code>.Object</code>	a BN
<code>dataset</code>	a BNDataset object containing the dataset the network is built upon, if any. The remaining parameters are considered only if a starting dataset is provided.
<code>...</code>	potential further arguments of methods.

Details

The constructor may be invoked without parameters – in this case an empty network will be created, and its slots will be filled manually by the user. This is usually viable only if the user already has knowledge about the network structure.

Value

BN object.

Slots

`name`: name of the network
`num.nodes`: number of nodes in the network
`variables`: names of the variables in the network
`discreteness`: TRUE if variable is discrete, FALSE if variable is continue
`node.sizes`: if variable `i` is discrete, `node.sizes[i]` contains the cardinality of `i`, if `i` is instead discrete the value is the number of states variable `i` takes when discretized
`cpts`: list of conditional probability tables of the network
`dag`: adjacency matrix of the network
`wpdag`: weighted partially dag

Examples

```
## Not run:
net.1 <- BN()

dataset <- BNDataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
net.2 <- BN(dataset)

## End(Not run)
```

bn<-	set the original BN object contained in an InferenceEngine.
------	---

Description

Add an original network to an InferenceEngine.

Usage

```
bn(x) <- value

## S4 replacement method for signature 'InferenceEngine'
bn(x) <- value
```

Arguments

x	an InferenceEngine.
value	the BN object contained in an InferenceEngine.

BNDataset-class	BNDataset class.
-----------------	------------------

Description

Contains the all of the data that can be extracted from a given dataset: raw data, imputed data, raw and imputed data with bootstrap.

initialize a BNDataset object.

constructor for BNDataset object

Usage

```
## S4 method for signature 'BNDataset'
initialize(.Object, ...)
```

```
BNDataset(name = "", data = NULL, variables = c(), node.sizes = c(),
  discreteness = c(), ...)
```


Arguments

<code>.Object</code>	an empty BNDataset.
<code>...</code>	potential further arguments of methods.
<code>name</code>	name of the dataset.
<code>data</code>	raw data.frame.
<code>variables</code>	vector of variable names.
<code>node.sizes</code>	vector of variable cardinalities (for discrete variables) or quantization ranges (for continuous variables).
<code>discreteness</code>	a vector of elements in {c,d} for continuous and discrete variables (respectively)

Details

Dataset should be provided in the following format... (describe)

Value

a BNDataset object.
BNDataset object.

Slots

`name`: name of the dataset
`header.file`: name and location of the header file
`data.file`: name and location of the data file
`variables`: names of the variables in the network
`node.sizes`: cardinality of each variable of the network
`num.variables`: number of variables (columns) in the dataset
`discreteness`: TRUE if variable is discrete, FALSE if variable is continue
`num.items`: number of observations (rows) in the dataset
`has.rawdata`: TRUE if the dataset contains data read from a file
`has.impdata`: TRUE if the dataset contains imputed data (computed from raw data)
`raw.data`: matrix containing raw data
`imputation`: TRUE if it dataset contains imputed data
`imputed.data`: matrix containing imputed data
`has.boots`: dataset has bootstrap samples
`boots`: list of bootstrap samples
`has.imp.boots`: dataset has imputed bootstrap samples
`imp.boots`: list of imputed bootstrap samples
`num.boots`: number of bootstrap samples

Examples

```
## Not run:
# create from files
dataset <- BNDataset(name = "MyData")
dataset <- read.dataset(dataset, "file.header", "file.data")

# other way: create from raw dataset and metadata
data <- matrix(c(1:16), nrow = 4, ncol = 4)
dataset <- BNDataset(name = "MyData", data = data,
                     variables = c("a", "b", "c", "d"),
                     node.sizes = c(4,8,12,16),
                     discreteness = rep('d',4))

## End(Not run)
```

boots	<i>get list of bootstrap samples of a BNDataset.</i>
-------	--

Description

Return the list of samples computed from raw data of a dataset.

Usage

```
boots(x)

## S4 method for signature 'BNDataset'
boots(x)
```

Arguments

x a [BNDataset](#) object.

Value

the list of bootstrap samples.

See Also

[has.boots](#), [has.imp.boots](#), [imp.boots](#)

boots<-	<i>set list of bootstrap samples of a BNDataset.</i>
---------	--

Description

Add to a dataset a list of samples from raw data computed using bootstrap.

Usage

```
boots(x) <- value
```

```
## S4 replacement method for signature 'BNDataset'
boots(x) <- value
```

Arguments

x	a BNDataset object.
value	the list of bootstrap samples.

bootstrap	<i>Perform bootstrap.</i>
-----------	---------------------------

Description

Create a list of num.boots samples of the original dataset.

Usage

```
bootstrap(object, num.boots = 100, seed = 0, imputation = FALSE,
  k.impute = 10, na.string.symbol = "?", ...)
```

```
## S4 method for signature 'BNDataset'
bootstrap(object, num.boots = 100, seed = 0,
  imputation = FALSE, k.impute = 10, na.string.symbol = "?", ...)
```

Arguments

object	the BNDataset object.
num.boots	number of sampled datasets for bootstrap.
seed	random seed.
imputation	TRUE if imputation has to be performed.
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken (useful only if imputation == TRUE).

```
na.string.symbol      character that denotes NA in the dataset (useful only if imputation == TRUE).
...                  potential further arguments of methods.
```

Examples

```
## Not run:
dataset <- BNdataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
dataset <- bootstrap(dataset, num.boots = 1000)

## End(Not run)
```

build.junction.tree *build a JunctionTree.*

Description

Starting from the adjacency matrix of the directed acyclic graph of the network contained in an InferenceEngine, build a JunctionTree for the network and store it into an InferenceEngine.

Usage

```
build.junction.tree(object, ...)

## S4 method for signature 'InferenceEngine'
build.junction.tree(object, ...)
```

Arguments

```
object      an InferenceEngine object.
...         potential further arguments for methods.
```

Examples

```
## Not run:
dataset <- BNdataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
net <- BN(dataset)
eng <- InferenceEngine()
eng <- build.junction.tree(eng)

## End(Not run)
```

child	<i>load Child dataset.</i>
-------	----------------------------

Description

Wrapper for a loader for the Child raw dataset; also perform imputation.

Usage

```
child()
```

Details

The dataset has 5000 items, with random missing values (no latent variables). BNDataset object contains the raw dataset and imputed dataset, with k=10 (see [impute](#) for related explanation).

Value

a BNDataset containing the Child dataset.

See Also

[child_NA_5000](#)

Examples

```
dataset <- child()
print(dataset)
```

child_NA_5000	<i>Child dataset.</i>
---------------	-----------------------

Description

The Child dataset contains 5000 randomly generated items with missing data (no latent variables) of the Child Bayesian Network. Imputation is performed, so both raw and imputed data is present.

Format

a [BNDataset](#) with a raw and imputed data slow filled with 5000 items.

Details

The data the BNDataset object is built from is located in files `pkg_folder/extdata/extdata/Child_data_na_5000.head` and `pkg_folder/extdata/extdata/Child_data_na_5000.data`.

References

D. J. Spiegelhalter, R. G. Cowell (1992). Learning in probabilistic expert systems. In Bayesian Statistics 4 (J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith, eds.) 447-466. Clarendon Press, Oxford.

See Also

[child](#)

cpts

get the list of conditional probability tables of a [BN](#).

Description

Return the list of conditional probability tables of the variables of a [BN](#) object. Each probability table is associated to the corresponding variable, and its dimensions are named according to the variable they represent.

Usage

```
cpts(x)
```

```
## S4 method for signature 'BN'  
cpts(x)
```

Arguments

x an object.

Details

Each conditional probability table is represented as a multidimensional array. The ordering of the dimensions of each variable is not guaranteed to follow the actual conditional distribution. E.g. dimensions for conditional probability $P(C|A,B)$ can be either (C,A,B) or (A,B,C), depending on if some operations have been performed, or how the probability table has been computed. Users should not rely on dimension numbers, but should instead select the dimensions using their names.

Value

list of the conditional probability tables of the desired object.

cpts<-	<i>set the list of conditional probability tables of a network.</i>
--------	---

Description

Set the list of conditional probability tables of a [BN](#) object.

Usage

```
cpts(x) <- value

## S4 replacement method for signature 'BN'
cpts(x) <- value
```

Arguments

x	an object.
value	list of the conditional probability tables of the object.

Details

Each conditional probability table is represented as a multidimensional array. To retrieve single dimensions (e.g. to compute marginals), users should provide dimensions names.

dag	<i>get adjacency matrix of a network.</i>
-----	---

Description

Return the adjacency matrix of the directed acyclic graph representing the structure of a network.

Usage

```
dag(x)

## S4 method for signature 'BN'
dag(x)
```

Arguments

x	an object.
---	------------

Value

matrix containing the adjacency matrix of the directed acyclic graph representing the structure of the object.

dag<-	<i>set adjacency matrix of an object.</i>
-------	---

Description

Set the adjacency matrix of the directed acyclic graph representing the structure of a network.

Usage

```
dag(x) <- value

## S4 replacement method for signature 'BN'
dag(x) <- value
```

Arguments

x	an object.
value	matrix containing the adjacency matrix of the directed acyclic graph representing the structure of the object.

data.file	<i>get data file of a BNDataset.</i>
-----------	--

Description

Return the data filename of a dataset (with the path to its position, as given by the user). The data filename may contain a header in the first row, containing the list of names of the variables, in the same order as in the header file. After the header, if present, the file contains a data.frame with the observations, one item per row.

Usage

```
data.file(x)

## S4 method for signature 'BNDataset'
data.file(x)
```

Arguments

x	a BNDataset .
---	-------------------------------

Value

data filename of the dataset.

See Also

[data.file](#)

data.file<-	<i>set data file of a BNDataset.</i>
-------------	--

Description

Set the data filename of a dataset (with the path to its position, as given by the user). The data filename may contain a header in the first row, containing the list of names of the variables, in the same order as in the header file. After the header, if present, the file contains a data.frame with the observations, one item per row.

Usage

```
data.file(x) <- value
```

```
## S4 replacement method for signature 'BNDataset'
data.file(x) <- value
```

Arguments

x	a BNDataset .
value	data filename.

See Also

[header.file<-](#)

discreteness	<i>get status (discrete or continuous) of the variables of an object.</i>
--------------	---

Description

Get a vector representing the status of the variables (with their names) of a [BN](#) or [BNDataset](#). Elements of the vector are c if the variable is continue, and d if the variable is discrete.

Usage

```
discreteness(x)
```

```
## S4 method for signature 'BNDataset'
discreteness(x)
```

```
## S4 method for signature 'BN'
discreteness(x)
```

Arguments

`x` an object.

Value

vector containing, for each variable of the desired object, c if the variable is continue, and d if the variable is discrete.

`discreteness<-` *set status (discrete or continuous) of the variables of an object.*

Description

Set the list of variable status for the variables in a network or a dataset.

Usage

```
discreteness(x) <- value

## S4 replacement method for signature 'BNdataset'
discreteness(x) <- value

## S4 replacement method for signature 'BN'
discreteness(x) <- value
```

Arguments

`x` an object.

`value` a vector of elements in {c,d} for continuous and discrete variables (respectively).

`em` *expectation-maximization algorithm.*

Description

Learn parameters of a network using the Expectation-Maximization algorithm.

Usage

```
em(x, dataset, threshold = 0.001, k.impute = 10, ...)

## S4 method for signature 'InferenceEngine,BNdataset'
em(x, dataset, threshold = 0.001,
   k.impute = 10, ...)
```

Arguments

x	an InferenceEngine .
dataset	observed dataset with missing values for the Bayesian Network of x.
threshold	threshold for convergence, used as stopping criterion.
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken.
...	further potential arguments for method.

Value

a list containing: an [InferenceEngine](#) with a new updated network ("InferenceEngine"), and the imputed dataset ("BNDataset").

Examples

```
## Not run:
em(x, dataset)

## End(Not run)
```

get.boot	<i>get selected element of bootstrap list.</i>
----------	--

Description

Given a [BNDataset](#), return the sample corresponding to given index.

Usage

```
get.boot(dataset, index, imputed = TRUE, ...)

## S4 method for signature 'BNDataset,numeric'
get.boot(dataset, index, imputed = TRUE, ...)
```

Arguments

dataset	a BNDataset object.
index	the index of the requested sample.
imputed	TRUE if samples from imputed dataset are to be used.
...	potential further arguments of methods (ignored).

See Also

[bootstrap](#)

Examples

```
## Not run:
dataset <- BNDataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
dataset <- bootstrap(dataset, num.boots = 1000)

for (i in 1:num.boots(dataset))
  print(get.boot(dataset, i))

## End(Not run)
```

get.data	<i>get data of a BNDataset.</i>
----------	---------------------------------

Description

Return data contained in a [BNDataset](#) object, if any. Preference is given to imputed data, if available, because the imputed dataset is (supposed to be), in general, more useful. To obtain specifically raw or imputed data, one must revert to [get.raw.data\(\)](#) and [get.imputed.data\(\)](#), respectively.

Usage

```
get.data(x)

## S4 method for signature 'BNDataset'
get.data(x)
```

Arguments

x a [BNDataset](#).

See Also

[has.data](#), [has.raw.data](#), [has.imputed.data](#), [get.raw.data](#), [get.imputed.data](#)

Examples

```
## Not run:
x <- BNDataset()
x <- read.dataset(x, "file.header", "file.data")
get.data(x) # returns raw dataset, the only one present in dataset

x <- impute(x)
get.data(x) # returns imputed dataset, since it is present now

## End(Not run)
```

get.imputed.data	<i>get imputed data of a BNDataset.</i>
------------------	---

Description

Return imputed data contained in a [BNDataset](#) object, if any.

Usage

```
get.imputed.data(x)

## S4 method for signature 'BNDataset'
get.imputed.data(x)
```

Arguments

x a [BNDataset](#).

See Also

[has.data](#), [has.raw.data](#), [has.imputed.data](#), [get.data](#), [get.raw.data](#)

get.most.probable.values	<i>compute the most probable values to be observed.</i>
--------------------------	---

Description

Return an array containing the values that each variable of the network is more likely to take, according to the CPTS. In case of ties take the first value.

Usage

```
get.most.probable.values(x, ...)
```

S4 method for signature 'BN'

```
get.most.probable.values(x, ...)
```

S4 method for signature 'InferenceEngine'

```
get.most.probable.values(x, ...)
```

Arguments

x a [BN](#) or [InferenceEngine](#) object.

... potential further arguments of methods.

Value

array containing, in each position, the most probable value for the corresponding variable.

Examples

```
## Not run:  
# try with a BN object x  
get.most.probable.values(x)  
  
# now build an InferenceEngine object  
eng <- InferenceEngine(x)  
get.most.probable.values(eng)  
  
## End(Not run)
```

get.raw.data

get raw data of a BNdataset.

Description

Return raw data contained in a [BNdataset](#) object, if any.

Usage

```
get.raw.data(x)  
  
## S4 method for signature 'BNdataset'  
get.raw.data(x)
```

Arguments

x a [BNdataset](#).

See Also

[has.data](#), [has.raw.data](#), [has.imputed.data](#), [get.data](#), [get.imputed.data](#)

has.boots	<i>check whether a BNDataset has bootstrap samples or not.</i>
-----------	--

Description

Return TRUE if the given dataset contains samples for bootstrap, FALSE otherwise.

Usage

```
has.boots(x)

## S4 method for signature 'BNDataset'
has.boots(x)
```

Arguments

x a [BNDataset](#) object.

Value

TRUE if dataset has bootstrap samples.

See Also

[has.imp.boots](#), [boots](#), [imp.boots](#)

has.data	<i>check if a BNDataset contains any data.</i>
----------	--

Description

Check whether a [BNDataset](#) object actually contains raw or imputed data.

Usage

```
has.data(x)

## S4 method for signature 'BNDataset'
has.data(x)
```

Arguments

x a [BNDataset](#).

See Also

[has.raw.data](#), [has.imputed.data](#), [get.data](#), [get.raw.data](#), [get.imputed.data](#)

Examples

```
## Not run:
x <- BNDataset()
has.data(x) # FALSE

x <- read.dataset(x, "file.header", "file.data")
has.data(x) # TRUE

## End(Not run)
```

has.imp.boots	<i>check whether a BNDataset has bootstrap samples from imputed data or not.</i>
---------------	--

Description

Return TRUE if the given dataset contains samples for bootstrap from imputed dataset, FALSE otherwise.

Usage

```
has.imp.boots(x)

## S4 method for signature 'BNDataset'
has.imp.boots(x)
```

Arguments

x a [BNDataset](#) object.

Value

TRUE if dataset has bootstrap samples from imputed data.

See Also

[has.boots](#), [boots](#), [imp.boots](#)

has.imputed.data	<i>check if a BNDataset contains imputed data.</i>
------------------	--

Description

Check whether a [BNDataset](#) object actually contains imputed data.

Usage

```
has.imputed.data(x)

## S4 method for signature 'BNDataset'
has.imputed.data(x)
```

Arguments

x a [BNDataset](#).

See Also

[has.data](#), [has.raw.data](#), [get.data](#), [get.raw.data](#), [get.imputed.data](#)

Examples

```
## Not run:
x <- BNDataset()
has.imputed.data(x) # FALSE

x <- read.dataset(x, "file.header", "file.data")
has.imputed.data(x) # FALSE, since read.dataset() actually reads raw data.

x <- impute(x)
has.imputed.data(x) # TRUE

## End(Not run)
```

has.raw.data	<i>check if a BNDataset contains raw data.</i>
--------------	--

Description

Check whether a [BNDataset](#) object actually contains raw data.

Usage

```
has.raw.data(x)

## S4 method for signature 'BNDataset'
has.raw.data(x)
```

Arguments

x a [BNDataset](#).

See Also

[has.data](#), [has.imputed.data](#), [get.data](#), [get.raw.data](#), [get.imputed.data](#)

Examples

```
## Not run:
x <- BNDataset()
has.raw.data(x) # FALSE

x <- read.dataset(x, "file.header", "file.data")
has.raw.data(x) # TRUE, since read.dataset() actually reads raw data.

## End(Not run)
```

header.file	<i>get header file of a BNDataset.</i>
-------------	--

Description

Return the header filename of a dataset (with the path to its position, as given by the user), present if the dataset has been read from a file and not manually inserted. The header file contains three rows:

1. list of names of the variables, in the same order as in the data file;
2. list of cardinalities of the variables, if discrete, or levels for quantization if continuous;
3. list of status of the variables: c for continuous variables, d for discrete ones.

Usage

```
header.file(x)

## S4 method for signature 'BNDataset'
header.file(x)
```

Arguments

x a [BNDataset](#).

Value

header filename of the dataset.

See Also

[data.file](#)

header.file<-	<i>set header file of a BNDataset.</i>
---------------	--

Description

Set the header filename of a dataset (with the path to its position, as given by the user). The header file has to contain three rows:

1. list of names of the variables, in the same order as in the data file;
2. list of cardinalities of the variables, if discrete, or levels for quantization if continuous;
3. list of status of the variables: c for continuous variables, d for discrete ones.

Further rows are ignored.

Usage

```
header.file(x) <- value
```

```
## S4 replacement method for signature 'BNDataset'  
header.file(x) <- value
```

Arguments

x	a BNDataset .
value	header filename.

See Also

[data.file<-](#)

imp.boots	<i>get list of bootstrap samples from imputed data of a BNDataset.</i>
-----------	--

Description

Return the list of samples computed from raw data of a dataset.

Usage

```
imp.boots(x)

## S4 method for signature 'BNDataset'
imp.boots(x)
```

Arguments

x a [BNDataset](#) object.

Value

the list of bootstrap samples from imputed data.

See Also

[has.boots](#), [has.imp.boots](#), [boots](#)

imp.boots<-	<i>set list of bootstrap samples from imputed data of a BNDataset.</i>
-------------	--

Description

Add to a dataset a list of samples from imputed data computed using bootstrap.

Usage

```
imp.boots(x) <- value

## S4 replacement method for signature 'BNDataset'
imp.boots(x) <- value
```

Arguments

x a [BNDataset](#) object.
value the list of bootstrap samples from imputed data.

impute	<i>Impute a BNDataset raw data with missing values.</i>
--------	---

Description

Impute a [BNDataset](#) raw data with missing values.

Usage

```
impute(object, k.impute = 10)

## S4 method for signature 'BNDataset'
impute(object, k.impute = 10)
```

Arguments

object	the BNDataset object.
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken.
...	potential further arguments of methods.

Examples

```
## Not run:
dataset <- BNDataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
dataset <- impute(dataset)

## End(Not run)
```

imputed.data<-	<i>add imputed data.</i>
----------------	--------------------------

Description

Insert imputed data in a [BNDataset](#) object.

Usage

```
imputed.data(x) <- value

## S4 replacement method for signature 'BNDataset'
imputed.data(x) <- value
```

Arguments

x	a BNDataset .
value	a matrix of integers containing a dataset.

Details

Users are encouraged to not use this method whenever possible, in favour of [read.dataset](#) with flag `imputation = TRUE`.

See Also

[has.data](#), [has.imputed.data](#), [get.data](#), [read.dataset](#)

InferenceEngine-class *InferenceEngine class.*

Description

InferenceEngine class.

Constructor method of [InferenceEngine](#) class.

constructor for [InferenceEngine](#) object

Usage

```
## S4 method for signature 'InferenceEngine'
initialize(.Object, ...)
```

```
InferenceEngine(bn = NULL, observations = NULL, ...)
```

Arguments

.Object	an empty InferenceEngine object.
...	potential further arguments of methods.
bn	a BN object.
observations	a list of observations composed by the two following vectors: <ul style="list-style-type: none"> • <code>observed.vars</code>:vector of observed variables; • <code>observed.vals</code>:vector of values observed for the variables in <code>observed.vars</code> in the corresponding position.

Value

an InferenceEngine object.

InferenceEngine object.

Details

Each joint probability table is represented as a multidimensional array. To retrieve single dimensions (e.g. to compute marginals), users should not rely on dimension numbers, but should instead select the dimensions using their names.

Value

the list of joint probability tables compiled by the [InferenceEngine](#).

jpts<-	<i>set the list of joint probability tables compiled by an InferenceEngine.</i>
--------	---

Description

Add a list of joint probability tables for the cliques of the junction tree.

Usage

```
jpts(x) <- value

## S4 replacement method for signature 'InferenceEngine'
jpts(x) <- value
```

Arguments

x	an InferenceEngine .
value	the list of joint probability tables compiled by the InferenceEngine .

Details

Each joint probability table is represented as a multidimensional array. To retrieve single dimensions (e.g. to compute marginals), users should provide dimension names.

jt.cliques	<i>get the list of cliques of the junction tree of an InferenceEngine.</i>
------------	--

Description

Return the list of cliques containing the variables associated to each node of a junction tree.

Usage

```
jt.cliques(x)

## S4 method for signature 'InferenceEngine'
jt.cliques(x)
```


`jt.cliques<-`

33

Arguments

`x` an [InferenceEngine](#).

Value

the list of cliques of the junction tree contained in the [InferenceEngine](#).

<code>jt.cliques<-</code>	<i>set the list of cliques of the junction tree of an InferenceEngine.</i>
------------------------------	--

Description

Add to the [InferenceEngine](#) a list containing the cliques of variables composing the nodes of the junction tree.

Usage

```
jt.cliques(x) <- value

## S4 replacement method for signature 'InferenceEngine'
jt.cliques(x) <- value
```

Arguments

`x` an [InferenceEngine](#).
`value` the list of cliques of the junction tree contained in the [InferenceEngine](#).

<code>junction.tree</code>	<i>get the junction tree of an InferenceEngine.</i>
----------------------------	---

Description

Return the adjacency matrix representing the junction tree computed for a network.

Usage

```
junction.tree(x)

## S4 method for signature 'InferenceEngine'
junction.tree(x)
```

Arguments

`x` an [InferenceEngine](#).

Details

Rows and columns are named after the (variables in the) cliques that each node of the junction tree represent.

Value

the junction tree contained in the [InferenceEngine](#).

See Also

[build.junction.tree](#)

junction.tree<-	set the junction tree of an InferenceEngine .
-----------------	---

Description

Set the adjacency matrix of the junction tree computed for a network.

Usage

```
junction.tree(x) <- value

## S4 replacement method for signature 'InferenceEngine'
junction.tree(x) <- value
```

Arguments

x	an InferenceEngine .
value	the junction tree to be inserted in the InferenceEngine .

knn.impute	Perform imputation of a data frame using k-NN.
------------	--

Description

Perform imputation of missing data in a data frame using the k-Nearest Neighbour algorithm. For discrete variables we use the mode, for continuous variables the median value is instead taken.

Usage

```
knn.impute(data, k = 10, cat.var = 1:ncol(data), to.impute = 1:nrow(data),
  using = 1:nrow(data))
```

Arguments

<code>data</code>	a data frame
<code>k</code>	number of neighbours to be used; for categorical variables the mode of the neighbours is used, for continuous variables the median value is used instead. Default: 10.
<code>cat.var</code>	vector containing the indices of the variables to be considered as categorical. Default: all variables.
<code>to.impute</code>	vector indicating which rows of the dataset are to be imputed. Default: impute all rows.
<code>using</code>	vector indicating which rows of the dataset are to be used to search for neighbours. Default: use all rows.

Value

imputed data frame.

<code>layering</code>	<i>return the layering of the nodes.</i>
-----------------------	--

Description

Compute the topological ordering of the nodes of a network, in order to divide the network in layers.

Usage

```
layering(x, updated.bn = TRUE, ...)

## S4 method for signature 'BN'
layering(x, updated.bn = TRUE, ...)

## S4 method for signature 'InferenceEngine'
layering(x, updated.bn = TRUE, ...)
```

Arguments

<code>x</code>	a BN or InferenceEngine object.
<code>updated.bn</code>	TRUE if <code>x</code> is an <code>InferenceEngine</code> and the updated network is chosen (kept only for compatibility with other methods).
<code>...</code>	potential further arguments for methods.

Value

a vector containing layers the nodes can be divided into.

Examples

```
## Not run:
dataset <- BNdataset(name="MyDataset")
dataset <- read.dataset(dataset, "file.header", "file.data")
x <- BN(dataset)
layering(x)
eng <- InferenceEngine(x)
layering(x, updated.bn=TRUE)

## End(Not run)
```

learn.params

learn the parameters of a [BN](#).

Description

Learn the parameters of a [BN](#) object according to a [BNdataset](#) using MAP (Maximum A Posteriori) estimation.

Usage

```
learn.params(bn, dataset, ess = 1, ...)

## S4 method for signature 'BN,BNdataset'
learn.params(bn, dataset, ess = 1)
```

Arguments

bn	a BN object.
dataset	a BNdataset object.
ess	Equivalent Sample Size value.
...	potential further arguments of methods.

Value

new [BN](#) object with conditional probabilities.

Examples

```
## Not run:
## first create a BN and learn its structure from a dataset
dataset <- BNdataset(name = "MyDataset")
dataset <- read.dataset(dataset, "file.header", "file.data")
bn <- BN()
bn <- learn.structure(bn, dataset)
bn <- learn.params(bn, dataset, ess=1)

## End(Not run)
```

learn.structure	<i>learn the structure of a network.</i>
-----------------	--

Description

Learn the structure (the directed acyclic graph) of a [BN](#) object according to a [BNDataset](#). Currently, two algorithms are supported (can be specified using the `algo` option): `'sm'`, the Silander-Myllymaki exact algorithm, and `'mmhc'`, the Max-Min Hill-Climbing heuristic algorithm (default). Three scoring functions are also provided: `'BDeu'`, the Bayesian-Dirichlet equivalent uniform score, `'AIC'`, the Akaike Information criterion, and `'BIC'`, the Bayesian Information criterion.

Usage

```
learn.structure(bn, dataset, algo = "mmhc", scoring.func = "BDeu",
  alpha = 0.05, ess = 1, bootstrap = FALSE, layering = c(),
  max.fanin.layers = NULL, max.fanin = num.variables(dataset),
  cont.nodes = c(), raw.data = FALSE, num.boots = 100,
  imputation = TRUE, k.impute = 10, na.string.symbol = "?", seed = 0,
  ...)

## S4 method for signature 'BN,BNDataset'
learn.structure(bn, dataset, algo = "mmhc",
  scoring.func = "BDeu", alpha = 0.05, ess = 1, bootstrap = FALSE,
  layering = c(), max.fanin.layers = NULL,
  max.fanin = num.variables(dataset), cont.nodes = c(), raw.data = FALSE,
  num.boots = 100, imputation = TRUE, k.impute = 10,
  na.string.symbol = "?", seed = 0)
```

Arguments

<code>bn</code>	a BN object.
<code>dataset</code>	a BNDataset .
<code>algo</code>	the algorithm to use. Currently, one among <code>sm</code> (Silander-Myllymaki) and <code>mmhc</code> (Max-Min Hill Climbing, default).
<code>scoring.func</code>	the scoring function to use. Currently, one among <code>BDeu</code> , <code>AIC</code> , <code>BIC</code> .
<code>alpha</code>	confidence threshold (only for <code>mmhc</code>).
<code>ess</code>	Equivalent Sample Size value.
<code>bootstrap</code>	<code>TRUE</code> to use bootstrap samples.
<code>layering</code>	vector containing the layers each node belongs to (only for <code>sm</code>).
<code>max.fanin.layers</code>	matrix of available parents in each layer (only for <code>sm</code>).
<code>max.fanin</code>	maximum number of parents for each node (only for <code>sm</code>).
<code>cont.nodes</code>	vector containing the index of continuous variables.

raw.data	TRUE to learn the structure from the raw dataset. Default is to use imputed dataset (if available, otherwise the raw dataset will be used anyway).
num.boots	number of bootstrap samples to generate, if needed.
imputation	TRUE if imputation is needed; if bootstrap=TRUE, imputed samples will be also used.
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken.
na.string.symbol	symbol for NA values (missing data).
seed	random seed.
...	potential further arguments for method.

Details

The Silander-Myllymaki algorithm can take a very long time, and it is not feasible for networks of more than 20-30 nodes. It is strongly recommended that valid layering, `max.fanin.layers` and `max.fanin` parameters are passed to the method if `algo = 'sm'` is given as parameter to the method.

Value

new [BN](#) object with DAG.

Examples

```
## Not run:
dataset <- BNdataset(name = "MyDataset")
dataset <- read.dataset(dataset, "file.header", "file.data")
bn <- BN()
# use MMHC
bn <- learn.structure(bn, dataset, alpha=0.05, ess=1, bootstrap=FALSE)

# now use Silander-Myllymaki
layers <- layering(bn)
mfl <- as.matrix(read.table(header=F,
text='0 1 1 1 1 0 1 1 1 1 0 0 8 7 7 0 0 0 14 6 0 0 0 0 19'))
bn <- learn.structure(bn, dataset, algo='sm', max.fanin=3, cont.nodes=c(),
                     layering=layers, max.fanin.layers=mfl, raw.data=FALSE)

## End(Not run)
```

marginals

compute the list of inferred marginals of a BN.

Description

Given an [InferenceEngine](#), it returns a list containing the marginals for the variables in the network, according to the propagated beliefs.

Usage

```
marginals(x, ...)

## S4 method for signature 'InferenceEngine'
marginals(x, ...)
```

Arguments

x an [InferenceEngine](#)

... potential further arguments of methods.

Value

a list containing the marginals of each variable, as probability tables.

Examples

```
## Not run:
eng <- InferenceEngine(net)
marginals(eng)

## End(Not run)
```

name	<i>get name of an object.</i>
------	-------------------------------

Description

Return the name of an object, of class [BN](#) or [BNDataset](#).

Usage

```
name(x)

## S4 method for signature 'BNDataset'
name(x)

## S4 method for signature 'BN'
name(x)
```

Arguments

x an object.

Value

name of the object.

name<-	<i>set name of an object.</i>
--------	-------------------------------

Description

Set the name slot of an object of type [BN](#) or [BNDataset](#).

Usage

```
name(x) <- value

## S4 replacement method for signature 'BNDataset'
name(x) <- value

## S4 replacement method for signature 'BN'
name(x) <- value
```

Arguments

x	an object.
value	the new name of the object.

node.sizes	<i>get size of the variables of an object.</i>
------------	--

Description

Return a list containing the size of the variables of an object. It is the actual cardinality of discrete variables, and the cardinality of the discretized variable for continuous variables.

Usage

```
node.sizes(x)

## S4 method for signature 'BNDataset'
node.sizes(x)

## S4 method for signature 'BN'
node.sizes(x)
```

Arguments

x	an object.
---	------------

Value

vector containing the size of each variable of the desired object.

node.sizes<-	<i>set the size of variables of an object.</i>
--------------	--

Description

Set the size of the variables of a BN or BNDataset object. It represents the actual cardinality of discrete variables, and the cardinality of the discretized variable for continuous variables.

Usage

```
node.sizes(x) <- value

## S4 replacement method for signature 'BNDataset'
node.sizes(x) <- value

## S4 replacement method for signature 'BN'
node.sizes(x) <- value
```

Arguments

x	an object.
value	vector containing the size of each variable of the object.

num.boots	<i>get number of bootstrap samples of a BNDataset.</i>
-----------	--

Description

Return the number of bootstrap samples computed from a dataset.

Usage

```
num.boots(x)

## S4 method for signature 'BNDataset'
num.boots(x)
```

Arguments

x	a BNDataset object.
---	-------------------------------------

Value

the number of bootstrap samples.

num.boots<-	<i>set number of bootstrap samples of a BNDataset.</i>
-------------	--

Description

Set the length of the list of samples of a dataset computed using bootstrap.

Usage

```
num.boots(x) <- value

## S4 replacement method for signature 'BNDataset'
num.boots(x) <- value
```

Arguments

x	a BNDataset object.
value	the number of bootstrap samples.

num.items	<i>get number of items of a BNDataset.</i>
-----------	--

Description

Return the number of items in a dataset, that is, the number of rows in its data slot.

Usage

```
num.items(x)

## S4 method for signature 'BNDataset'
num.items(x)
```

Arguments

x	a BNDataset object.
---	-------------------------------------

Value

number of items of the desired dataset.

num.items<-	<i>set number of items of a BNDataset.</i>
-------------	--

Description

Set the number of observed items (rows) in a dataset.

Usage

```
num.items(x) <- value
```

```
## S4 replacement method for signature 'BNDataset'
num.items(x) <- value
```

Arguments

x	a BNDataset object.
value	number of items of the desired dataset.

num.nodes	<i>get number of nodes of an object.</i>
-----------	--

Description

Return the name of an object, of class [BN](#) or [InferenceEngine](#).

Usage

```
num.nodes(x)
```

```
## S4 method for signature 'BN'
num.nodes(x)
```

```
## S4 method for signature 'InferenceEngine'
num.nodes(x)
```

Arguments

x	an object.
---	------------

Value

number of nodes of the desired object.

num.nodes<-	<i>set number of nodes of an object.</i>
-------------	--

Description

Set the number of nodes of an object of type [BN](#) (number of nodes of the network) or [InferenceEngine](#) (where parameter contains the number of nodes of the junction tree).

Usage

```
num.nodes(x) <- value

## S4 replacement method for signature 'BN'
num.nodes(x) <- value

## S4 replacement method for signature 'InferenceEngine'
num.nodes(x) <- value
```

Arguments

x	an object.
value	the number of nodes in the object.

num.variables	<i>get number of variables of a BNDataset.</i>
---------------	--

Description

Return the number of the variables contained in a dataset. This value corresponds to the value of [num.nodes](#) of a network built upon the same dataset.

Usage

```
num.variables(x)

## S4 method for signature 'BNDataset'
num.variables(x)

## S4 method for signature 'BNDataset'
num.variables(x)
```

Arguments

x	a BNDataset object.
---	-------------------------------------

`num.variables<-`

45

Value

number of variables of the desired dataset.

See Also

[num.nodes](#)

<code>num.variables<-</code>	<i>set number of variables of a BNDataset.</i>
---------------------------------	--

Description

Set the number of variables observed in a dataset.

Usage

```
num.variables(x) <- value

## S4 replacement method for signature 'BNDataset'
num.variables(x) <- value
```

Arguments

<code>x</code>	a BNDataset object.
<code>value</code>	number of variables of the dataset.

<code>observations</code>	<i>get the list of observations of an InferenceEngine.</i>
---------------------------	--

Description

Return the list of observations added to an [InferenceEngine](#).

Usage

```
observations(x)

## S4 method for signature 'InferenceEngine'
observations(x)
```

Arguments

<code>x</code>	an InferenceEngine .
----------------	--------------------------------------

Details

Output is a list in the following format:

- `observed.vars` vector of observed variables;
- `observed.vals` vector of values observed for the variables in `observed.vars` in the corresponding position.

Value

the list of observations of the [InferenceEngine](#).

<code>observations<-</code>	<i>set the list of observations of an InferenceEngine.</i>
--------------------------------	--

Description

Add a list of observations to an [InferenceEngine](#), using a list of observations composed by the two following vectors:

- `observed.vars` vector of observed variables;
- `observed.vals` vector of values observed for the variables in `observed.vars` in the corresponding position.

Usage

```
observations(x) <- value
```

```
## S4 replacement method for signature 'InferenceEngine'
observations(x) <- value
```

Arguments

<code>x</code>	an InferenceEngine .
<code>value</code>	the list of observations of the InferenceEngine .

Details

Replace previous list of observations, if present. In order to add evidence, and not just replace it, one must use the [add.observations<-](#) method.

In case of multiple observations of the same variable, the last observation is the one used, as the most recent.

See Also

[add.observations<-](#)

plot	<i>plot a BN as a picture.</i>
------	--------------------------------

Description

plot a BN as a picture.

Usage

```
## S3 method for class 'BN'
plot(x, ..., use.node.names = TRUE, frac = 0.2,
      max.weight = max(dag(x)), node.col = rep("white", ncol(dag(x))),
      plot.wpdag = FALSE)
```

Arguments

x	a BN object.
...	potential further arguments for methods.
use.node.names	TRUE if node names have to be printed. If FALSE, numbers are used instead.
frac	fraction
max.weight	max.weight
node.col	list of (R) colors for the nodes.
plot.wpdag	if TRUE plot the network according to the WPDAG computed using bootstrap instead of the DAG.

print	<i>print an object to stdout.</i>
-------	-----------------------------------

Description

print an object to stdout.

Usage

```
print(x, ...)

## S4 method for signature 'BNdataset'
print(x, show.raw.data = FALSE,
      show.imputed.data = FALSE, ...)

## S4 method for signature 'BN'
print(x, ...)

## S4 method for signature 'InferenceEngine'
print(x, engine = "jt", ...)
```

Arguments

x an object.
... potential other arguments.
show.raw.data when x is a [BNDataset](#), print also raw dataset, if available.
show.imputed.data when x is a [BNDataset](#), print also imputed dataset, if available.
engine when x is an [InferenceEngine](#), specify the inference engine to be shown. Currently only engine = 'jt' is supported.

query	<i>query BN given observations</i>
-------	------------------------------------

Description

query BN given observations

Usage

```
query(x, observed.vars = c(), observed.vals = c(), ...)
```

```
## S4 method for signature 'BN'
query(x, observed.vars, observed.vals)
```

Arguments

x a BN.
observed.vars vector of observed variables.
observed.vals vector of observed values for corresponding variables in observed.vars.
... potential further arguments for method.

Value

most probable values given observations

raw.data<-	<i>add raw data.</i>
------------	----------------------

Description

Insert raw data in a [BNDataset](#) object.

Usage

```
raw.data(x) <- value
```

```
## S4 replacement method for signature 'BNDataset'
raw.data(x) <- value
```

Arguments

x	a BNDataset .
value	a matrix of integers containing a dataset.

Details

Users are encouraged to not use this method whenever possible, in favour of [read.dataset](#).

See Also

[has.data](#), [has.raw.data](#), [get.data](#), [read.dataset](#)

read.bif	<i>Read a network from a .bif file.</i>
----------	---

Description

Read a network described in a .bif-formatted file, and build a [BN](#) object.

Usage

```
read.bif(x)
```

```
## S4 method for signature 'character'
read.bif(x)
```

Arguments

x	the .bif file, with absolute/relative position.
---	---

Details

The method relies on a coherent ordering of variable values and parameters.

Value

a [BN](#) object.

read.dataset	<i>Read a dataset from file.</i>
--------------	----------------------------------

Description

File has to be in format (describe...)

Usage

```
read.dataset(object, header.file, data.file, imputation = FALSE,
  header.flag = FALSE, na.string.symbol = "?", sep.symbol = "",
  k.impute = 10, bootstrap = FALSE, num.boots = 100, seed = 0,
  starts.from = 0, ...)
```

```
## S4 method for signature 'BNDataset,character,character'
read.dataset(object, header.file,
  data.file, imputation = FALSE, header.flag = FALSE,
  na.string.symbol = "?", sep.symbol = "", k.impute = 10,
  bootstrap = FALSE, num.boots = 100, seed = 0, starts.from = 0, ...)
```

Arguments

object	the BNDataset object.
header.file	the header file.
data.file	the data file.
imputation	TRUE if imputation has to be performed.
header.flag	TRUE if the first row of dataset file is an header (e.g. it contains the variable names).
na.string.symbol	character that denotes NA in the dataset.
sep.symbol	separator among values in the dataset.
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken (useful only if imputation == TRUE).
bootstrap	TRUE if bootstrap has to be performed; prepares a list of datasets sampled from the original one.
num.boots	number of sampled datasets for bootstrap (useful only if bootstrap == TRUE).

seed random seed (useful only if bootstrap == TRUE).
 starts.from starting value for entries in the dataset (observed values, default is 0).
 ... potential further arguments of methods.

Examples

```
## Not run:
dataset <- BNDataset()
dataset <- read.dataset(dataset, header="file.header", dataset="file.data")

## End(Not run)
```

read.dsc	<i>Read a network from a .dsc file.</i>
----------	---

Description

Read a network described in a .dsc-formatted file, and build a [BN](#) object.

Usage

```
read.dsc(x)

## S4 method for signature 'character'
read.dsc(x)
```

Arguments

x the .dsc file, with absolute/relative position.

Details

The method relies on a coherent ordering of variable values and parameters.

Value

a [BN](#) object.

save.to.eps	save a BN picture as .eps file.
-------------	---

Description

Save an image of a Bayesian Network as an .eps file.

Usage

```
save.to.eps(x, filename)

## S4 method for signature 'BN,character'
save.to.eps(x, filename)
```

Arguments

x	a BN object
filename	name (with path, if needed) of the file to be created

See Also

[plot](#)

Examples

```
## Not run:
save.to.eps(x, "out.eps")

## End(Not run)
```

show	Show method for objects.
------	--------------------------

Description

The show method allows to provide a custom aspect for the output that is generated when the name of an instance is gives as command in an R session.

Usage

```
show(object)
```

Arguments

object	an object.
--------	------------

test.updated.bn	<i>check if an updated BN is present in an InferenceEngine.</i>
-----------------	---

Description

Check if an InferenceEngine actually contains an updated network, in order to provide the chance of a fallback and use the original network if no belief propagation has been performed.

Usage

```
test.updated.bn(x)

## S4 method for signature 'InferenceEngine'
test.updated.bn(x)
```

Arguments

x an InferenceEngine.

Value

TRUE if an updated network is contained in the InferenceEngine, FALSE otherwise.

Examples

```
## Not run:
dataset <- BNdataset()
dataset <- read.dataset(dataset, "file.header", "file.data")
bn <- BN(dataset)
ie <- InferenceEngine(bn)
test.updated.bn(ie) # FALSE

observations(ie) <- list("observed.vars"=("A","G","X"), "observed.vals"=c(1,2,1))
ie <- belief.propagation(ie)
test.updated.bn(ie) # TRUE

## End(Not run)
```

updated.bn	<i>get the updated BN object contained in an InferenceEngine.</i>
------------	---

Description

Return an updated network contained in an InferenceEngine.

Usage

```
updated.bn(x)

## S4 method for signature 'InferenceEngine'
updated.bn(x)
```

Arguments

x an [InferenceEngine](#).

Value

the updated [BN](#) object contained in an [InferenceEngine](#).

updated.bn<-	<i>set the updated BN object contained in an InferenceEngine.</i>
--------------	---

Description

Add an updated network to an [InferenceEngine](#).

Usage

```
updated.bn(x) <- value

## S4 replacement method for signature 'InferenceEngine'
updated.bn(x) <- value
```

Arguments

x an [InferenceEngine](#).
value the updated [BN](#) object contained in an [InferenceEngine](#).

variables	<i>get variables of an object.</i>
-----------	------------------------------------

Description

Get the list of variables (with their names) of a [BN](#) or [BNDataset](#).

`variables<-`

55

Usage

```
variables(x)

## S4 method for signature 'BNdataset'
variables(x)

## S4 method for signature 'BN'
variables(x)
```

Arguments

`x` an object.

Value

vector of the variables names of the desired object.

`variables<-` *set variables of an object.*

Description

Set the list of variable names in a [BN](#) or [BNdataset](#) object.

Usage

```
variables(x) <- value

## S4 replacement method for signature 'BNdataset'
variables(x) <- value

## S4 replacement method for signature 'BN'
variables(x) <- value
```

Arguments

`x` an object.

`value` vector containing the variable names of the object. Overwrites `num.nodes` slot if non-matching.

wpdag	<i>get the WPDAG of an object.</i>
-------	------------------------------------

Description

Return the weighted partially directed acyclic graph of a network, when available (e.g. when bootstrap on dataset is performed).

Usage

```
wpdag(x)

## S4 method for signature 'BN'
wpdag(x)
```

Arguments

x an object.

Value

matrix containing the WPDAG of the object.

wpdag<-	<i>set WPDAG of the object.</i>
---------	---------------------------------

Description

Set the weighted partially directed acyclic graph of a network (e.g. in case bootstrap on dataset is performed).

Usage

```
wpdag(x) <- value

## S4 replacement method for signature 'BN'
wpdag(x) <- value
```

Arguments

x an object.
value matrix containing the WPDAG of the object.

Index

add.observations<-, 3
add.observations<-, InferenceEngine-method
 (add.observations<-), 3
asia, 4, 5
asia_10000, 4, 5

belief.propagation, 5
belief.propagation, InferenceEngine
 (belief.propagation), 5
belief.propagation, InferenceEngine-method
 (belief.propagation), 5
BN, 6–8, 14, 15, 17, 21, 30, 31, 35–40, 43, 44,
 47, 49–55
BN (BN-class), 7
bn, 6
BN, BN-class (BN-class), 7
bn, InferenceEngine (bn), 6
bn, InferenceEngine-method (bn), 6
BN-class, 7
bn<-, 8
bn<-, InferenceEngine-method (bn<-), 8
BNdataset, 5, 7, 8, 10, 11, 13, 16, 17, 19–30,
 36, 37, 39–45, 48–50, 54, 55
BNdataset (BNdataset-class), 8
BNdataset, BNdataset-class
 (BNdataset-class), 8
BNdataset-class, 8
boots, 10, 23, 24, 28
boots, BNdataset (boots), 10
boots, BNdataset-method (boots), 10
boots<-, 11
boots<-, BNdataset-method (boots<-), 11
bootstrap, 11, 19
bootstrap, BNdataset (bootstrap), 11
bootstrap, BNdataset-method (bootstrap),
 11
build.junction.tree, 12, 34
build.junction.tree, InferenceEngine
 (build.junction.tree), 12

build.junction.tree, InferenceEngine-method
 (build.junction.tree), 12

child, 13, 14
child_NA_5000, 13, 13
cpts, 14
cpts, BN (cpts), 14
cpts, BN-method (cpts), 14
cpts<-, 15
cpts<-, BN-method (cpts<-), 15

dag, 15
dag, BN (dag), 15
dag, BN-method (dag), 15
dag<-, 16
dag<-, BN-method (dag<-), 16
data.file, 16, 16, 27
data.file, BNdataset (data.file), 16
data.file, BNdataset-method (data.file),
 16
data.file<-, 17
data.file<-, BNdataset-method
 (data.file<-), 17
discreteness, 17
discreteness, BN (discreteness), 17
discreteness, BN-method (discreteness),
 17
discreteness, BNdataset (discreteness),
 17
discreteness, BNdataset-method
 (discreteness), 17
discreteness<-, 18
discreteness<-, BN-method
 (discreteness<-), 18
discreteness<-, BNdataset-method
 (discreteness<-), 18

em, 18
em, InferenceEngine, BNdataset (em), 18

- em, InferenceEngine, BNDataset-method (em), 18
- get.boot, 19
- get.boot, BNDataset (get.boot), 19
- get.boot, BNDataset, numeric-method (get.boot), 19
- get.data, 20, 21–23, 25, 26, 30, 49
- get.data, BNDataset (get.data), 20
- get.data, BNDataset-method (get.data), 20
- get.imputed.data, 20, 21, 22, 23, 25, 26
- get.imputed.data, BNDataset (get.imputed.data), 21
- get.imputed.data, BNDataset-method (get.imputed.data), 21
- get.most.probable.values, 21
- get.most.probable.values, BN (get.most.probable.values), 21
- get.most.probable.values, BN-method (get.most.probable.values), 21
- get.most.probable.values, InferenceEngine (get.most.probable.values), 21
- get.most.probable.values, InferenceEngine-method (get.most.probable.values), 21
- get.raw.data, 20, 21, 22, 23, 25, 26
- get.raw.data, BNDataset (get.raw.data), 22
- get.raw.data, BNDataset-method (get.raw.data), 22
- has.boots, 10, 23, 24, 28
- has.boots, BNDataset (has.boots), 23
- has.boots, BNDataset-method (has.boots), 23
- has.data, 20–22, 23, 25, 26, 30, 49
- has.data, BNDataset (has.data), 23
- has.data, BNDataset-method (has.data), 23
- has.imp.boots, 10, 23, 24, 28
- has.imp.boots, BNDataset (has.imp.boots), 24
- has.imp.boots, BNDataset-method (has.imp.boots), 24
- has.imputed.data, 20–23, 25, 26, 30
- has.imputed.data, BNDataset (has.imputed.data), 25
- has.imputed.data, BNDataset-method (has.imputed.data), 25
- has.raw.data, 20–23, 25, 25, 49
- has.raw.data, BNDataset (has.raw.data), 25
- has.raw.data, BNDataset-method (has.raw.data), 25
- header.file, 26
- header.file, BNDataset (header.file), 26
- header.file, BNDataset-method (header.file), 26
- header.file<-, 27
- header.file<-, BNDataset-method (header.file<-), 27
- imp.boots, 10, 23, 24, 28
- imp.boots, BNDataset (imp.boots), 28
- imp.boots, BNDataset-method (imp.boots), 28
- imp.boots<-, 28
- imp.boots<-, BNDataset-method (imp.boots<-), 28
- impute, 13, 29
- impute, BNDataset (impute), 29
- impute, BNDataset-method (impute), 29
- imputed.data<-, 29
- imputed.data<-, BNDataset-method (imputed.data<-), 29
- InferenceEngine, 3, 6, 8, 12, 19, 21, 30–35, 38, 39, 43–46, 48, 53, 54
- InferenceEngine (InferenceEngine-class), 30
- InferenceEngine, InferenceEngine-class (InferenceEngine-class), 30
- InferenceEngine-class, 30
- initialize, BN-method (BN-class), 7
- initialize, BNDataset-method (BNDataset-class), 8
- initialize, InferenceEngine-method (InferenceEngine-class), 30
- jpts, 31
- jpts, InferenceEngine (jpts), 31
- jpts, InferenceEngine-method (jpts), 31
- jpts<-, 32
- jpts<-, InferenceEngine-method (jpts<-), 32
- jt.cliques, 32
- jt.cliques, InferenceEngine (jt.cliques), 32
- jt.cliques, InferenceEngine-method (jt.cliques), 32

jt.cliques<-, 33
 jt.cliques<-, InferenceEngine-method
 (jt.cliques<-), 33
 junction.tree, 33
 junction.tree, InferenceEngine
 (junction.tree), 33
 junction.tree, InferenceEngine-method
 (junction.tree), 33
 junction.tree<-, 34
 junction.tree<-, InferenceEngine-method
 (junction.tree<-), 34

 knn.impute, 34

 layering, 35
 layering, BN (layering), 35
 layering, BN-method (layering), 35
 layering, InferenceEngine (layering), 35
 layering, InferenceEngine-method
 (layering), 35
 learn.params, 36
 learn.params, BN, BNDataset
 (learn.params), 36
 learn.params, BN, BNDataset-method
 (learn.params), 36
 learn.structure, 37
 learn.structure, BN, BNDataset
 (learn.structure), 37
 learn.structure, BN, BNDataset-method
 (learn.structure), 37

 marginals, 38
 marginals, InferenceEngine (marginals),
 38
 marginals, InferenceEngine-method
 (marginals), 38

 name, 39
 name, BN (name), 39
 name, BN-method (name), 39
 name, BNDataset (name), 39
 name, BNDataset-method (name), 39
 name<-, 40
 name<-, BN-method (name<-), 40
 name<-, BNDataset-method (name<-), 40
 node.sizes, 40
 node.sizes, BN (node.sizes), 40
 node.sizes, BN-method (node.sizes), 40
 node.sizes, BNDataset (node.sizes), 40

 node.sizes, BNDataset-method
 (node.sizes), 40
 node.sizes<-, 41
 node.sizes<-, BN-method (node.sizes<-),
 41
 node.sizes<-, BNDataset-method
 (node.sizes<-), 41
 num.boots, 41
 num.boots, BNDataset (num.boots), 41
 num.boots, BNDataset-method (num.boots),
 41
 num.boots<-, 42
 num.boots<-, BNDataset-method
 (num.boots<-), 42
 num.items, 42
 num.items, BNDataset (num.items), 42
 num.items, BNDataset-method (num.items),
 42
 num.items<-, 43
 num.items<-, BNDataset-method
 (num.items<-), 43
 num.nodes, 43, 44, 45
 num.nodes, BN (num.nodes), 43
 num.nodes, BN-method (num.nodes), 43
 num.nodes, InferenceEngine (num.nodes),
 43
 num.nodes, InferenceEngine-method
 (num.nodes), 43
 num.nodes<-, 44
 num.nodes<-, BN-method (num.nodes<-), 44
 num.nodes<-, InferenceEngine-method
 (num.nodes<-), 44
 num.variables, 44
 num.variables, BNDataset
 (num.variables), 44
 num.variables, BNDataset-method
 (num.variables), 44
 num.variables<-, 45
 num.variables<-, BNDataset-method
 (num.variables<-), 45

 observations, 45
 observations, InferenceEngine
 (observations), 45
 observations, InferenceEngine-method
 (observations), 45
 observations<-, 46
 observations<-, InferenceEngine-method
 (observations<-), 46

plot, [47](#), [52](#)
 plot, BN (plot), [47](#)
 plot.BN (plot), [47](#)
 plot.BN, BN (plot), [47](#)
 print, [47](#)
 print, BN (print), [47](#)
 print, BN-method (print), [47](#)
 print, BNdataset (print), [47](#)
 print, BNdataset-method (print), [47](#)
 print, InferenceEngine (print), [47](#)
 print, InferenceEngine-method (print), [47](#)

 query, [48](#)
 query, BN (query), [48](#)
 query, BN-method (query), [48](#)

 raw.data<-, [49](#)
 raw.data<-, BNdataset-method
 (raw.data<-), [49](#)
 read.bif, [49](#)
 read.bif, character (read.bif), [49](#)
 read.bif, character-method (read.bif), [49](#)
 read.dataset, [30](#), [49](#), [50](#)
 read.dataset, BNdataset, character, character
 (read.dataset), [50](#)
 read.dataset, BNdataset, character, character-method
 (read.dataset), [50](#)
 read.dsc, [51](#)
 read.dsc, character (read.dsc), [51](#)
 read.dsc, character-method (read.dsc), [51](#)

 save.to.eps, [52](#)
 save.to.eps, BN, character (save.to.eps),
 [52](#)
 save.to.eps, BN, character-method
 (save.to.eps), [52](#)
 show, [52](#)
 show, AllTheClasses-method (show), [52](#)
 show, BN-method (show), [52](#)
 show, BNdataset-method (show), [52](#)
 show, InferenceEngine-method (show), [52](#)

 test.updated.bn, [53](#)
 test.updated.bn, InferenceEngine
 (test.updated.bn), [53](#)
 test.updated.bn, InferenceEngine-method
 (test.updated.bn), [53](#)

 updated.bn, [53](#)
 updated.bn, InferenceEngine
 (updated.bn), [53](#)
 updated.bn, InferenceEngine-method
 (updated.bn), [53](#)
 updated.bn<-, [54](#)
 updated.bn<-, InferenceEngine-method
 (updated.bn<-), [54](#)

 variables, [54](#)
 variables, BN (variables), [54](#)
 variables, BN-method (variables), [54](#)
 variables, BNdataset (variables), [54](#)
 variables, BNdataset-method (variables),
 [54](#)
 variables<-, [55](#)
 variables<-, BN-method (variables<-), [55](#)
 variables<-, BNdataset-method
 (variables<-), [55](#)

 wpdag, [56](#)
 wpdag, BN (wpdag), [56](#)
 wpdag, BN-method (wpdag), [56](#)
 wpdag<-, [56](#)
 wpdag<-, BN-method (wpdag<-), [56](#)