

# Package ‘bnstruct’

September 1, 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	4
belief.propagation	5
bn	6
BN-class	6
bn<-	8
BNDataset-class	8
boots	10
boots<-	10
bootstrap	11
build.junction.tree	12
child	12
child_NA_5000	13
cpts	14

cpts<-	14
dag	15
dag<-	15
data.file	16
data.file<-	16
discreteness	17
discreteness<-	17
em	18
get.boot	19
get.data	19
get.imputed.data	20
get.most.probable.values	21
get.raw.data	22
has.boots	22
has.data	23
has.imp.boots	23
has.imputed.data	24
has.raw.data	25
header.file	25
header.file<-	26
imp.boots	27
imp.boots<-	27
impute	28
imputed.data<-	28
InferenceEngine-class	29
jpts	30
jpts<-	31
jt.cliques	31
jt.cliques<-	32
junction.tree	32
junction.tree<-	33
layering	33
learn.params	34
learn.structure	35
marginals	37
name	37
name<-	38
node.sizes	38
node.sizes<-	39
num.boots	39
num.boots<-	40
num.items	40
num.items<-	41
num.nodes	41
num.nodes<-	42
num.variables	42
num.variables<-	43
observations	43

`add.observations<-` 3

<code>observations&lt;-</code>	44
<code>plot</code>	44
<code>print</code>	45
<code>query</code>	46
<code>raw.data&lt;-</code>	46
<code>read.dataset</code>	47
<code>save.to.eps</code>	48
<code>test.updated.bn</code>	49
<code>updated.bn</code>	49
<code>updated.bn&lt;-</code>	50
<code>variables</code>	50
<code>variables&lt;-</code>	51
<code>wpdag</code>	51
<code>wpdag&lt;-</code>	52

**Index** 53

---

`add.observations<-`     *add further evidence to an existing list of observations of an [InferenceEngine](#).*

---

## 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
```

## Arguments

`x`                     an [InferenceEngine](#).  
`value`                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	<i>load Asia dataset.</i>
------	---------------------------

---

**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	<i>Asia 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.

**Usage**

```
asia_10000
```

**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](#)

---

belief.propagation	<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, ...)

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

## Arguments

<code>ie</code>	an <a href="#">InferenceEngine</a> object.
<code>net</code>	a <a href="#">BN</a> object.
<code>observed.vars</code>	list of observed variables.
<code>observed.vals</code>	values taken by variables listed in <code>observed.vars</code> .
<code>return.potentials</code>	if TRUE only the potentials are returned, instead of the default <a href="#">BN</a> .
<code>...</code>	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	<i>BN class.</i>
----------	------------------

---

## Description

BN class.

Instantiate a BN object.

**Usage**

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

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

**Arguments**

<code>name</code>	name of the Bayesian Network.
<code>dataset</code>	a <a href="#">BNDataset</a> 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.
<code>.Object</code>	An object: see the Details section.

**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)
```

---

<code>bn&lt;-</code>	<i>set the original BN object contained in an <a href="#">InferenceEngine</a>.</i>
----------------------	--

---

### Description

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

### Usage

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

### Arguments

<code>x</code>	an <a href="#">InferenceEngine</a> .
<code>value</code>	the BN object contained in an <a href="#">InferenceEngine</a> .

---

<code>BNDataset-class</code>	<i>BNDataset class.</i>
------------------------------	-------------------------

---

### 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 = "", ...)
```

### Arguments

<code>.Object</code>	an empty <a href="#">BNDataset</a> .
<code>name</code>	name of the dataset.
<code>...</code>	potential further arguments of methods.

### 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:
dataset <- BNDataset()
dataset <- read.dataset(dataset, "file.header", "file.data")

## End(Not run)
```

---

boots	<i>get list of bootstrap samples of a <a href="#">BNDataset</a>.</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 <a href="#">BNDataset</a>.</i>
---------	--

---

**Description**

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

**Usage**

```
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, ...)

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

## Arguments

<code>object</code>	the <code>BNDataset</code> object.
<code>num.boots</code>	number of sampled datasets for bootstrap.
<code>seed</code>	random seed.
<code>imputation</code>	TRUE if imputation has to be performed.
<code>na.string.symbol</code>	character that denotes NA in the dataset (useful only if <code>imputation == TRUE</code> ).
<code>k.impute</code>	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken (useful only if <code>imputation == TRUE</code> ).
<code>...</code>	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	<i>build a JunctionTree.</i>
---------------------	------------------------------

---

### 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 <a href="#">InferenceEngine</a> 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	Child <i>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.

**Usage**

```
child_NA_5000
```

**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	<i>get the list of conditional probability tables of a <a href="#">BN</a>.</i>
------	--

---

### 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
```

### 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
```

**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 <a href="#">BNDataset</a>.</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 <a href="#">BNDataset</a>.</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
```

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

---

**Description**

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

**Usage**

```
discreteness(x) <- value
```

**Arguments**

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

---

em	<i>expectation-maximization algorithm.</i>
----	--

---

## Description

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

## Usage

```
em(x, dataset, ...)

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

## Arguments

x	an <a href="#">InferenceEngine</a> .
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, ...)

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

### Arguments

dataset	a <a href="#">BNDataset</a> 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`*compute the most probable values to be observed.*

---

## 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

<code>x</code>	a <a href="#">BN</a> or <a href="#">InferenceEngine</a> object.
<code>...</code>	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)
```

---

<code>get.raw.data</code>	<i>get raw data of a <a href="#">BNDataset</a>.</i>
---------------------------	---

---

### 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](#)

---

<code>has.boots</code>	<i>check whether a <a href="#">BNDataset</a> 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 <a href="#">BNDataset</a> 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 <a href="#">BNDataset</a> 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 [BNDDataset](#) 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 BNDDataset contains imputed data.</i>
------------------	---

---

**Description**

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

**Usage**

```
has.imputed.data(x)

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

**Arguments**

x                      a [BNDDataset](#).

**See Also**

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

**Examples**

```
## Not run:
x <- BNDDataset()
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 <a href="#">BNDataset</a> 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 <a href="#">BNDataset</a>.</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<-                      *set header file of a [BNDataset](#).*

---

**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
```

**Arguments**

x                      a [BNDataset](#).  
value                      header filename.

**See Also**

[data.file<-](#)

---

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

---

**Description**

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

**Usage**

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

**Arguments**

x                    a [BNDataset](#) object.

value                the list of bootstrap samples from imputed data.

---

impute	<i>Impute a <a href="#">BNDataset</a> raw data with missing values.</i>
--------	---

---

### Description

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

### Usage

```
impute(object, ...)
```

## S4 method for signature 'BNDataset'

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

### Arguments

object	the <a href="#">BNDataset</a> 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
```

### Arguments

x	a <a href="#">BNDataset</a> .
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.

`bn` a `BN` object.

`observations` 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.

`...` potential further arguments of methods.

**Value**

an `InferenceEngine` object.

`InferenceEngine` object.

**Slots**

junction.tree: junction tree adjacency matrix.  
 num.nodes: number of nodes in the junction tree.  
 cliques: list of cliques composing the nodes of the junction tree.  
 triangulated.graph: adjacency matrix of the original triangulated graph.  
 jpts: inferred joint probability tables.  
 bn: original Bayesian Network (as object of class [BN](#)) as provided by the user, or learnt from a dataset. NULL if missing.  
 updated.bn: Bayesian Network (as object of class [BN](#)) as modified by a belief propagation computation. In particular, it will have different conditional probability tables with respect to its original version. NULL if missing.  
 observed.vars: list of observed variables, by name or number.  
 observed.vals: list of observed values for the corresponding variables in observed.vars.

**Examples**

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

obs <- list(c("A", "G", "X"), c(1, 2, 1))
eng.2 <- InferenceEngine(bn, obs)

## End(Not run)
```

---

jpts	<i>get the list of joint probability tables compiled by an <a href="#">InferenceEngine</a>.</i>
------	---

---

**Description**

Return the list of joint probability tables for the cliques of the junction tree obtained after belief propagation has been performed.

**Usage**

```
jpts(x)

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

**Arguments**

x                    an [InferenceEngine](#).

**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 <a href="#">InferenceEngine</a>.</i>
--------	---

---

**Description**

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

**Usage**

```
jpts(x) <- value
```

**Arguments**

x	an <a href="#">InferenceEngine</a> .
value	the list of joint probability tables compiled by the <a href="#">InferenceEngine</a> .

**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 <a href="#">InferenceEngine</a>.</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)
```

**Arguments**

x                      an [InferenceEngine](#).

**Value**

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

---

```
jt.cliques<-
```

*set the list of cliques of the junction tree of an [InferenceEngine](#).*

---

**Description**

Add to the InferenceEngine a list containing the cliques of variables composing the nodes of the junction tree.

**Usage**

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

**Arguments**

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

---

```
junction.tree
```

*get the junction tree of an [InferenceEngine](#).*

---

**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<-	<i>set the junction tree of an <a href="#">InferenceEngine</a>.</i>
-----------------	---

---

**Description**

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

**Usage**

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

**Arguments**

x	an <a href="#">InferenceEngine</a> .
value	the junction tree to be inserted in the <a href="#">InferenceEngine</a> .

---

layering	<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, ...)

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

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

**Arguments**

x	a <a href="#">BN</a> or <a href="#">InferenceEngine</a> object.
updated.bn	TRUE if x is an <a href="#">InferenceEngine</a> and the updated network is chosen (kept only for compatibility with other methods).
...	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, ...)
```

## S4 method for signature 'BN,BNDataset'

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

**Arguments**

bn	a <a href="#">BN</a> object.
dataset	a <a href="#">BNDataset</a> 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, ...)

## 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 <a href="#">BN</a> object.
<code>dataset</code>	a <a href="#">BNdataset</a> .
<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>num.boots</code>	number of bootstrap samples to generate, if needed.

layering	vector containing the layers each node belongs to (only for sm).
max.fanin.layers	matrix of available parents in each layer (only for sm).
max.fanin	maximum number of parents for each node (only for sm).
cont.nodes	use an empty vector.
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).
imputation	TRUE if imputation is needed; if bootstrap=TRUE, imputed samples will be also used.
na.string.symbol	symbol for NA values (missing data).
k.impute	number of neighbours to be used; for discrete variables we use mode, for continuous variables the median value is instead taken.
seed	random seed.
...	potential further arguments of methods.

### 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	<i>compute the list of inferred marginals of a BN.</i>
-----------	--

---

### 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 <a href="#">InferenceEngine</a>
...	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
```

**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
```

**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 <a href="#">BNDataset</a>.</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 <a href="#">BNDataset</a> object.
---	-------------------------------------

**Value**

the number of bootstrap samples.

---

num.boots<-	<i>set number of bootstrap samples of a <a href="#">BNDataset</a>.</i>
-------------	--

---

### Description

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

### Usage

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

### Arguments

x	a <a href="#">BNDataset</a> object.
value	the number of bootstrap samples.

---

num.items	<i>get number of items of a <a href="#">BNDataset</a>.</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 <a href="#">BNDataset</a> object.
---	-------------------------------------

### Value

number of items of the desired dataset.



---

num.items<-	<i>set number of items of a <a href="#">BNDataset</a>.</i>
-------------	--

---

**Description**

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

**Usage**

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

**Arguments**

x	a <a href="#">BNDataset</a> 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
```

### Arguments

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

---

num.variables	<i>get number of variables of a <a href="#">BNDataset</a>.</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 <a href="#">BNDataset</a> object.
---	-------------------------------------

### Value

number of variables of the desired dataset.

### See Also

[num.nodes](#)

---

num.variables<-	<i>set number of variables of a <a href="#">BNDataset</a>.</i>
-----------------	--

---

**Description**

Set the number of variables observed in a dataset.

**Usage**

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

**Arguments**

x	a <a href="#">BNDataset</a> object.
value	number of variables of the dataset.

---

observations	<i>get the list of observations of an <a href="#">InferenceEngine</a>.</i>
--------------	--

---

**Description**

Return the list of observations added to an InferenceEngine.

**Usage**

```
observations(x)

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

**Arguments**

x	an <a href="#">InferenceEngine</a> .
---	--------------------------------------

**Details**

Output is a list in the following format:

- observed.varsvector of observed variables;
- observed.valsvector of values observed for the variables in observed.vars in the corresponding position.

**Value**

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

---

<code>observations&lt;-</code>	<i>set the list of observations of an <a href="#">InferenceEngine</a>.</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
```

### Arguments

<code>x</code>	an <a href="#">InferenceEngine</a> .
<code>value</code>	the list of observations of the <a href="#">InferenceEngine</a> .

### 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<-](#)

---

<code>plot</code>	<i>plot a <a href="#">BN</a> as a picture.</i>
-------------------	--

---

### Description

Plot the network as a picture to default output.

### Usage

```
## S4 method for signature '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.  
**use.node.names** TRUE if node names have to be printed. If FALSE, number 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.  
**...** potential further arguments of methods.

**Examples**

```
## Not run:
plot(x, use.node.names=TRUE, frac=0.2, max.weight=1,
      node.col=c("cyan"), plot.wpdag=FALSE)

## End(Not run)
```

---

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.  
**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 <a href="#">InferenceEngine</a> , specify the inference engine to be shown. Currently only engine = 'jt' is supported.
...	potential other arguments.

---

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

---

**Description**

query BN given observations

**Usage**

```
query(x, ...)  
  
## 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

---

<i>raw.data&lt;-</i>	<i>add raw data.</i>
----------------------	----------------------

---

**Description**

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

**Usage**

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

---

## Description

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

## Usage

```
read.dataset(object, header.file, data.file, ...)

## 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, ...)
```

## Arguments

object	the <a href="#">BNDataset</a> object.
header.file	the header file.
data.file	the data file.
na.string.symbol	character that denotes NA in the dataset.
sep.symbol	separator among values in the dataset.
header.flag	TRUE if the first row of dataset file is an header (e.g. it contains the variable names).
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).
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).
...	potential further arguments of methods.

### Examples

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

## End(Not run)
```

---

save.to.eps	save a <a href="#">BN</a> 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 <a href="#">BN</a> 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)
```



---

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<-                    set the updated BN object contained in an InferenceEngine.
```

---

**Description**

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

**Usage**

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

**Arguments**

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

---

```
variables                    get variables of an object.
```

---

**Description**

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

**Usage**

```
variables(x)

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

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

`variables<-`

51

### 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
```

### Arguments

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

---

`wpdag`                      *get the WPDAG of an object.*

---

### 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
```

**Arguments**

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

# Index

## \*Topic **datasets**

- asia\_10000, [4](#)
- child\_NA\_5000, [13](#)
- add.observations<-, [3](#)
- add.observations<-, InferenceEngine-method  
(add.observations<-), [3](#)
- asia, [4](#), [5](#)
- asia\_10000, [4](#), [4](#)
- belief.propagation, [5](#)
- belief.propagation, InferenceEngine  
(belief.propagation), [5](#)
- belief.propagation, InferenceEngine-method  
(belief.propagation), [5](#)
- BN, [5](#), [6](#), [8](#), [14](#), [17](#), [21](#), [29](#), [30](#), [33–38](#), [41](#), [42](#),  
[44](#), [45](#), [48–51](#)
- BN (BN-class), [6](#)
- bn, [6](#)
- BN, BN-class (BN-class), [6](#)
- bn, InferenceEngine (bn), [6](#)
- bn, InferenceEngine-method (bn), [6](#)
- BN-class, [6](#)
- bn<-, [8](#)
- bn<-, InferenceEngine-method (bn<-), [8](#)
- BNDataset, [4](#), [7](#), [8](#), [10](#), [11](#), [13](#), [16](#), [17](#), [19](#), [20](#),  
[22–28](#), [34](#), [35](#), [37–43](#), [45–47](#), [50](#), [51](#)
- BNDataset (BNDataset-class), [8](#)
- BNDataset, BNDataset-class  
(BNDataset-class), [8](#)
- BNDataset-class, [8](#)
- boots, [10](#), [22](#), [24](#), [27](#)
- boots, BNDataset (boots), [10](#)
- boots, BNDataset-method (boots), [10](#)
- boots<-, [10](#)
- boots<-, BNDataset-method (boots<-), [10](#)
- bootstrap, [11](#), [19](#)
- bootstrap, BNDataset (bootstrap), [11](#)
- bootstrap, BNDataset-method (bootstrap),  
[11](#)
- build.junction.tree, [12](#), [33](#)
- build.junction.tree, InferenceEngine  
(build.junction.tree), [12](#)
- build.junction.tree, InferenceEngine-method  
(build.junction.tree), [12](#)
- child, [12](#), [13](#)
- child\_NA\_5000, [13](#), [13](#)
- cpts, [14](#)
- cpts, BN (cpts), [14](#)
- cpts, BN-method (cpts), [14](#)
- cpts<-, [14](#)
- cpts<-, BN-method (cpts<-), [14](#)
- dag, [15](#)
- dag, BN (dag), [15](#)
- dag, BN-method (dag), [15](#)
- dag<-, [15](#)
- dag<-, BN-method (dag<-), [15](#)
- data.file, [16](#), [16](#), [26](#)
- data.file, BNDataset (data.file), [16](#)
- data.file, BNDataset-method (data.file),  
[16](#)
- data.file<-, [16](#)
- data.file<-, BNDataset-method  
(data.file<-), [16](#)
- discreteness, [17](#)
- discreteness, BN (discreteness), [17](#)
- discreteness, BN-method (discreteness),  
[17](#)
- discreteness, BNDataset (discreteness),  
[17](#)
- discreteness, BNDataset-method  
(discreteness), [17](#)
- discreteness<-, [17](#)
- discreteness<-, BN-method  
(discreteness<-), [17](#)
- discreteness<-, BNDataset-method  
(discreteness<-), [17](#)
- 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, 19, 20, 22–25, 29, 47
- get.data, BNDataset (get.data), 19
- get.data, BNDataset-method (get.data), 19
- get.imputed.data, 19, 20, 20, 22–25
- get.imputed.data, BNDataset (get.imputed.data), 20
- get.imputed.data, BNDataset-method (get.imputed.data), 20
- 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, 19, 20, 22, 23–25
- get.raw.data, BNDataset (get.raw.data), 22
- get.raw.data, BNDataset-method (get.raw.data), 22
- has.boots, 10, 22, 24, 27
- has.boots, BNDataset (has.boots), 22
- has.boots, BNDataset-method (has.boots), 22
- has.data, 20, 22, 23, 24, 25, 29, 47
- has.data, BNDataset (has.data), 23
- has.data, BNDataset-method (has.data), 23
- has.imp.boots, 10, 22, 23, 27
- has.imp.boots, BNDataset (has.imp.boots), 23
- has.imp.boots, BNDataset-method (has.imp.boots), 23
- has.imputed.data, 20, 22, 23, 24, 25, 29
- has.imputed.data, BNDataset (has.imputed.data), 24
- has.imputed.data, BNDataset-method (has.imputed.data), 24
- has.raw.data, 20, 22–24, 25, 47
- has.raw.data, BNDataset (has.raw.data), 25
- has.raw.data, BNDataset-method (has.raw.data), 25
- header.file, 25
- header.file, BNDataset (header.file), 25
- header.file, BNDataset-method (header.file), 25
- header.file<-, 26
- header.file<-, BNDataset-method (header.file<-), 26
- imp.boots, 10, 22, 24, 27
- imp.boots, BNDataset (imp.boots), 27
- imp.boots, BNDataset-method (imp.boots), 27
- imp.boots<-, 27
- imp.boots<-, BNDataset-method (imp.boots<-), 27
- impute, 12, 28
- impute, BNDataset (impute), 28
- impute, BNDataset-method (impute), 28
- imputed.data<-, 28
- imputed.data<-, BNDataset-method (imputed.data<-), 28
- InferenceEngine, 3, 5, 6, 8, 12, 18, 21, 29–33, 37, 41–44, 46, 49, 50
- InferenceEngine (InferenceEngine-class), 29
- InferenceEngine, InferenceEngine-class (InferenceEngine-class), 29
- InferenceEngine-class, 29
- initialize, BN-method (BN-class), 6
- initialize, BNDataset-method (BNDataset-class), 8
- initialize, InferenceEngine-method (InferenceEngine-class), 29
- jpts, 30
- jpts, InferenceEngine (jpts), 30
- jpts, InferenceEngine-method (jpts), 30
- jpts<-, 31
- jpts<-, InferenceEngine-method (jpts<-), 31
- jt.cliques, 31
- jt.cliques, InferenceEngine (jt.cliques), 31
- jt.cliques, InferenceEngine-method (jt.cliques), 31

jt.cliques<-, 32  
 jt.cliques<-, InferenceEngine-method  
     (jt.cliques<-), 32  
 junction.tree, 32  
 junction.tree, InferenceEngine  
     (junction.tree), 32  
 junction.tree, InferenceEngine-method  
     (junction.tree), 32  
 junction.tree<-, 33  
 junction.tree<-, InferenceEngine-method  
     (junction.tree<-), 33  
  
 layering, 33  
 layering, BN (layering), 33  
 layering, BN-method (layering), 33  
 layering, InferenceEngine (layering), 33  
 layering, InferenceEngine-method  
     (layering), 33  
 learn.params, 34  
 learn.params, BN, BNDataset  
     (learn.params), 34  
 learn.params, BN, BNDataset-method  
     (learn.params), 34  
 learn.structure, 35  
 learn.structure, BN, BNDataset  
     (learn.structure), 35  
 learn.structure, BN, BNDataset-method  
     (learn.structure), 35  
  
 marginals, 37  
 marginals, InferenceEngine (marginals),  
     37  
 marginals, InferenceEngine-method  
     (marginals), 37  
  
 name, 37  
 name, BN (name), 37  
 name, BN-method (name), 37  
 name, BNDataset (name), 37  
 name, BNDataset-method (name), 37  
 name<-, 38  
 name<-, BN-method (name<-), 38  
 name<-, BNDataset-method (name<-), 38  
 node.sizes, 38  
 node.sizes, BN (node.sizes), 38  
 node.sizes, BN-method (node.sizes), 38  
 node.sizes, BNDataset (node.sizes), 38  
 node.sizes, BNDataset-method  
     (node.sizes), 38  
  
 node.sizes<-, 39  
 node.sizes<-, BN-method (node.sizes<-),  
     39  
 node.sizes<-, BNDataset-method  
     (node.sizes<-), 39  
 num.boots, 39  
 num.boots, BNDataset (num.boots), 39  
 num.boots, BNDataset-method (num.boots),  
     39  
 num.boots<-, 40  
 num.boots<-, BNDataset-method  
     (num.boots<-), 40  
 num.items, 40  
 num.items, BNDataset (num.items), 40  
 num.items, BNDataset-method (num.items),  
     40  
 num.items<-, 41  
 num.items<-, BNDataset-method  
     (num.items<-), 41  
 num.nodes, 41, 42  
 num.nodes, BN (num.nodes), 41  
 num.nodes, BN-method (num.nodes), 41  
 num.nodes, InferenceEngine (num.nodes),  
     41  
 num.nodes, InferenceEngine-method  
     (num.nodes), 41  
 num.nodes<-, 42  
 num.nodes<-, BN-method (num.nodes<-), 42  
 num.nodes<-, InferenceEngine-method  
     (num.nodes<-), 42  
 num.variables, 42  
 num.variables, BNDataset  
     (num.variables), 42  
 num.variables, BNDataset-method  
     (num.variables), 42  
 num.variables<-, 43  
 num.variables<-, BNDataset-method  
     (num.variables<-), 43  
  
 observations, 43  
 observations, InferenceEngine  
     (observations), 43  
 observations, InferenceEngine-method  
     (observations), 43  
 observations<-, 44  
 observations<-, InferenceEngine-method  
     (observations<-), 44  
  
 plot, 44, 48

plot, BN (plot), 44  
 plot, BN-method (plot), 44  
 print, 45  
 print, BN (print), 45  
 print, BN-method (print), 45  
 print, BNdataset (print), 45  
 print, BNdataset-method (print), 45  
 print, InferenceEngine (print), 45  
 print, InferenceEngine-method (print), 45  
  
 query, 46  
 query, BN (query), 46  
 query, BN-method (query), 46  
  
 raw.data<-, 46  
 raw.data<-, BNdataset-method  
     (raw.data<-), 46  
 read.dataset, 29, 47, 47  
 read.dataset, BNdataset, character, character  
     (read.dataset), 47  
 read.dataset, BNdataset, character, character-method  
     (read.dataset), 47  
  
 save.to.eps, 48  
 save.to.eps, BN, character (save.to.eps),  
     48  
 save.to.eps, BN, character-method  
     (save.to.eps), 48  
  
 test.updated.bn, 49  
 test.updated.bn, InferenceEngine  
     (test.updated.bn), 49  
 test.updated.bn, InferenceEngine-method  
     (test.updated.bn), 49  
  
 updated.bn, 49  
 updated.bn, InferenceEngine  
     (updated.bn), 49  
 updated.bn, InferenceEngine-method  
     (updated.bn), 49  
 updated.bn<-, 50  
 updated.bn<-, InferenceEngine-method  
     (updated.bn<-), 50  
  
 variables, 50  
 variables, BN (variables), 50  
 variables, BN-method (variables), 50  
 variables, BNdataset (variables), 50  
 variables, BNdataset-method (variables),  
     50  
  
 variables<-, 51  
 variables<-, BN-method (variables<-), 51  
 variables<-, BNdataset-method  
     (variables<-), 51  
  
 wpdag, 51  
 wpdag, BN (wpdag), 51  
 wpdag, BN-method (wpdag), 51  
 wpdag<-, 52  
 wpdag<-, BN-method (wpdag<-), 52