Package 'bnstruct'

September 30, 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<-` *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

## S4 replacement method for signature 'InferenceEngine'
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 *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 *dataset.*

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

belief.propagation *perform belief propagation.*

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 *get the BN object contained in an InferenceEngine.*

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 definition.*

Description

BN class definition.

Instantiate a BN object.

Usage

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

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

Arguments

.Object	a BN
dataset	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.
...	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

scoring.func: scoring function used in structure learning (when performed)

struct.algo: algorithm used in structure learning (when performed)

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

.Object	an empty BNDataset.
...	potential further arguments of methods.
name	name of the dataset.
data	raw data.frame.
variables	vector of variable names.
node.sizes	vector of variable cardinalities (for discrete variables) or quantization ranges (for continuous variables).
discreteness	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 *get list of bootstrap samples of a* BNDataset.

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<- *set list of bootstrap samples of a* BNDataset.

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 *Perform bootstrap.*

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 *load* Child *dataset.*

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 *dataset.*

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.heade
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. 0. 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<- *set the list of conditional probability tables of a network.*

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 *get adjacency matrix of a network.*

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<- *set adjacency matrix of an object.*

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 *get data file of a* BNDataset.

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<- *set data file of a* BNDataset.

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 *get status (discrete or continuous) of the variables of an object.*

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 contaning, 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 *get selected element of bootstrap list.*

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` *get data of a BNDataset.*

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 *get imputed data of a BNDataset.*

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

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 *check whether a BNDataset has bootstrap samples or not.*

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 *check if a BNDataset contains any data.*

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` *check whether a* `BNDataset` *has bootstrap samples from imputed data or not.*

Description

Return `TRUE` if the given dataset contains samples for bootstrap from inputed 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 *check if a BNDataset contains impited data.*

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 *check if a BNDataset contains raw data.*

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 *get header file of a BNDataset.*

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

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

Arguments

x	a BNDataset.
value	header filename.

See Also

data.file<-

imp.boots *get list of bootstrap samples from imputed data of a* BNDataset.

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<- *set list of bootstrap samples from imputed data of a* BNDataset.

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 *Impute a BNDataset raw data with missing values.*

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<- *add imputed data.*

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: • observed.vars:vector of observed variables; • observed.vals:vector of values observed for the variables in observed.vars in the corresponding position.

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 *get the list of joint probability tables compiled by an* `InferenceEngine`.

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<- *set the list of joint probability tables compiled by an InferenceEngine.*

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 *get the list of cliques of the junction tree of an InferenceEngine.*

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

## 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.

---

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<- *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

data	a data frame
k	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.
cat.var	vector containing the indices of the variables to be considered as categorical. Default: all variables.
to.impute	vector indicating which rows of the dataset are to be imputed. Default: impute all rows.
using	vector indicating which rows of the dataset are to be used to search for neighbours. Default: use all rows.

Value

imputed data frame.

---

layering *return the layering of the nodes.*

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

x	a BN or InferenceEngine object.
updated.bn	TRUE if x is an InferenceEngine 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, 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` *learn the structure of a network.*

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

`bn`	a BN object.
`dataset`	a BNDataset.
`algo`	the algorithm to use. Currently, one among `sm` (Silander-Myllymaki) and `mmhc` (Max-Min Hill Climbing, default).
`scoring.func`	the scoring function to use. Currently, one among `BDeu`, `AIC`, `BIC`.
`alpha`	confidence threshold (only for `mmhc`).
`ess`	Equivalent Sample Size value.
`bootstrap`	`TRUE` to use bootstrap samples.
`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`	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 *get name of an object.*

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<- *set name of an object.*

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 *get size of the variables of an object.*

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 contaning the size of each variable of the desired object.

node.sizes<- *set the size of variables of an object.*

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 contaning the size of each variable of the object.

num.boots *get number of bootstrap samples of a* BNDataset.

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<- *set number of bootstrap samples of a* BNDataset.

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 *get number of items of a* BNDataset.

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<- *set number of items of a BNDataset.*

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 *get number of nodes of an object.*

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<- *set number of nodes of an object.*

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 *get number of variables of a BNDataset.*

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.

Value

number of variables of the desired dataset.

See Also

num.nodes

---

num.variables<- *set number of variables of a* BNDataset.

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

x	a BNDataset object.
value	number of variables of the dataset.

---

observations *get the list of observations of an* InferenceEngine.

Description

Return the list of observations added to an InferenceEngine.

Usage

```
observations(x)

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

Arguments

x	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`.

`observations<-` *set the list of observations of an* `InferenceEngine`.

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

`x`	an `InferenceEngine`.
`value`	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 *plot a BN as a picture.*

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 *print an object to* stdout.

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 *query BN given observations*

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<- *add raw data.*

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 *Read a network from a .bif file.*

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 in the file.

Value

a BN object.

`read.dataset` *Read a dataset from file.*

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 *Read a network from a* .dsc *file.*

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 in the file.

Value

a BN object.

read.net *Read a network from a* .net *file.*

Description

Read a network described in a .net-formatted file, and build a BN object.

Usage

```
read.net(x)

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

Arguments

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

Details

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

Value

a BN object.

sample.dataset *sample a* BNDataset *from a network of an inference engine.*

Description

sample a BNDataset from a network of an inference engine.

Usage

```
sample.dataset(x, n = 100)

## S4 method for signature 'BN'
sample.dataset(x, n = 100)

## S4 method for signature 'InferenceEngine'
sample.dataset(x, n = 100)
```

Arguments

x	a BN or InferenceEngine object.
n	number of items to sample.

Value

a BNDataset

`sample.row` *sample a row vector of values for a network.*

Description

sample a row vector of values for a network.

Usage

```
sample.row(x)

## S4 method for signature 'BN'
sample.row(x)

## S4 method for signature 'InferenceEngine'
sample.row(x)
```

Arguments

x	a BN or InferenceEngine object.

Value

a vector of values.

`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)
```

---

scoring.func *Read the scoring function used to learn the structure of a network.*

Description

Read the scoring function used in the learn.structure method. Outcome is meaningful only if the structure of a network has been learnt.

Usage

```
scoring.func(x)

## S4 method for signature 'BN'
scoring.func(x)
```

Arguments

x	the BN object.

Value

the scoring function used.

---

scoring.func<- *Set the scoring function used to learn the structure of a network.*

Description

Set the scoring function used in the learn.structure method.

Usage

```
scoring.func(x) <- value

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

Arguments

x	the BN object.
value	the scoring function used.

Value

updated BN.

---

sem *Structural Expectation-Maximization algorithm.*

Description

Learn struvture and parameters of a network with the Structural EM algorithm.

Usage

```
sem(x, dataset, struct.threshold = 10, param.threshold = 0.001,
  k.impute = 10, 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, na.string.symbol = "?", seed = 0,
  ...)

## S4 method for signature 'InferenceEngine,BNDataset'
sem(x, dataset, struct.threshold = 10,
  param.threshold = 0.001, k.impute = 10, 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, na.string.symbol = "?", seed = 0,
  ...)
```

Arguments

x	an InferenceEngine
dataset	observed dataset with missing values for the Bayesian Network of x.
struct.threshold	threshold for convergence of the structure learning step, used as stopping criterion.
param.threshold	threshold for convergence of the parameter learning step, 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.

algo	the algorithm to use. Currently, one among sm (Silander-Myllymaki) and mmhc (Max-Min Hill Climbing, default).
scoring.func	the scoring function to use. Currently, one among BDeu, AIC, BIC.
alpha	confidence threshold (only for mmhc).
ess	Equivalent Sample Size value.
bootstrap	TRUE to use bootstrap samples.
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	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.
na.string.symbol	symbol for NA values (missing data).
seed	random seed.
...	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:
sem(x, dataset)

## 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.

`struct.algo` *Read the algorithm used to learn the structure of a network.*

Description

Read the algorithm used in the `learn.structure` method. Outcome is meaningful only if the structure of a network has been learnt.

Usage

```
struct.algo(x)

## S4 method for signature 'BN'
struct.algo(x)
```

Arguments

x	the BN object.

Value

the structure learning algorithm used.

`struct.algo<-` *Set the algorithm used to learn the structure of a network.*

Description

Set the algorithm used in the `learn.structure` method.

Usage

```
struct.algo(x) <- value

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

Arguments

x	the BN object.
value	the scoring function used.

Value

updated BN.

test.updated.bn *check if an updated BN is present in an InferenceEngine.*

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 *get the updated BN object contained in an InferenceEngine.*

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

## S4 replacement method for signature 'InferenceEngine'
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)
```

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 *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 contaning the WPDAG of the object.

wpdag<- *set WPDAG of the object.*

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 contaning the WPDAG of the object.

write.dsc *Write a network saving it in a* .dsc *file.*

Description

Write a network on disk, saving it in a .dsc-formatted file.

Usage

```
write.dsc(x, path = "./")

## S4 method for signature 'BN'
write.dsc(x, path = "./")
```

Arguments

x	the BN object.
path	the relative or absolute path of the directory of the created file.

Index