Package 'BCDAG'

June 14, 2024

Title Bayesian Structure and Causal Learning of Gaussian Directed Graphs

Version 1.1.1

Description A collection of functions for structure learning of causal networks and estimation of joint causal effects from observational Gaussian data. Main algorithm consists of a Markov chain Monte Carlo scheme for posterior inference of causal structures, parameters and causal effects between variables.
References:
F. Castelletti and A. Mascaro (2021) <doi:10.1007/s10260-021-00579-1>,
F. Castelletti and A. Mascaro (2022) <doi:10.48550/arXiv.2201.12003>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Imports graph, graphics, gRbase, grDevices, lattice, methods, mvtnorm, Rgraphviz, stats, utils

Suggests rmarkdown, knitr, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Depends R (>= 2.10)

URL https://github.com/alesmascaro/BCDAG

BugReports https://github.com/alesmascaro/BCDAG/issues

NeedsCompilation no

Author Federico Castelletti [aut],
Alessandro Mascaro [aut, cre, cph]

Maintainer Alessandro Mascaro <alessandro.mascaro@upf.edu>

Repository CRAN

Date/Publication 2024-06-14 13:10:02 UTC

Contents

`as_graphNEL` *Transform adjacency matrix into graphNEL object*

Description

Function to transform an adjacency matrix into a graphNEL object.

Usage

```
as_graphNEL(DAG)
```

Arguments

`DAG`	Adjacency matrix of a DAG

Value

A graphNEL object

Examples

```
# Randomly generate DAG
q <- 4; w = 0.2
set.seed(123)
DAG <- rDAG(q,w)
as_graphNEL(DAG)
```

causaleffect *Compute causal effects between variables*

Description

This function computes the total joint causal effect on variable `response` consequent to an intervention on variables `targets` for a given a DAG structure and parameters `(D,L)`

Usage

```
causaleffect(targets, response, L, D)
```

Arguments

`targets`	numerical vector with labels of target nodes
`response`	numerical label of response variable
`L`	(q, q) matrix of regression-coefficient parameters
`D`	(q, q) diagonal matrix of conditional-variance parameters

Details

We assume that the joint distribution of random variables $X_1, \ldots, X_q$ is zero-mean Gaussian with covariance matrix Markov w.r.t. a Directed Acyclic Graph (DAG). In addition, the allied Structural Equation Model (SEM) representation of a Gaussian DAG-model allows to express the covariance matrix as a function of the (Cholesky) parameters `(D,L)`, collecting the conditional variances and regression coefficients of the SEM.

The total causal effect on a given variable of interest (`response`) consequent to a joint intervention on a set of variables (`targets`) is defined according to Pearl's do-calculus theory and under the Gaussian assumption can be expressed as a function of parameters `(D,L)`.

Value

The joint total causal effect, represented as a vector of same length of `targets`

Author(s)

Federico Castelletti and Alessandro Mascaro

References

J. Pearl (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge.

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

P. Nandy, M.H. Maathuis and T. Richardson (2017). Estimating the effect of joint interventions from observational data in sparse high-dimensional settings. *Annals of Statistics* 45(2), 647-674.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
# Total causal effect on node 1 of an intervention on {5,6}
causaleffect(targets = c(6,7), response = 1, L = L, D = D)
# Total causal effect on node 1 of an intervention on {5,7}
causaleffect(targets = c(5,7), response = 1, L = L, D = D)
```

---

confint.bcdagCE *Credible Intervals for bcdagCE Object*

Description

Computes credible (not confidence!) intervals for one or more target variables from objects of class bcdagCE.

Usage

```
## S3 method for class 'bcdagCE'
confint(object, parm = "all", level = 0.95, ...)
```

Arguments

object	a bcdagCE object from which credible intervals are computed.
parm	an integer vector indexing the target variables for which credible intervals are computed. If missing, all variables are considered.
level	the credible level required.
...	additional arguments.

Value

A matrix with columns giving lower and upper credible limits for each variable.

Examples

```
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
```

```
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = w,
                     fast = TRUE, save.memory = FALSE, verbose = FALSE)
out_ce <- get_causaleffect(out_mcmc, targets = c(4,6), response = 1)
confint(out_ce, c(4,6), 0.95)
```

`get_causaleffect` *Estimate total causal effects from the MCMC output*

Description

This function provides causal effect estimates from the output of `learn_DAG`

Usage

```
get_causaleffect(learnDAG_output, targets, response, verbose = TRUE)
```

Arguments

`learnDAG_output`	object of class `bcdag`
`targets`	numerical $(p, 1)$ vector with labels of target nodes
`response`	numerical label of response variable
`verbose`	if `TRUE`, progress bar of MCMC sampling is displayed

Details

Output of `learn_dag` function consists of S draws from the joint posterior of DAGs and DAG-parameters in a zero-mean Gaussian DAG-model; see the documentation of `learn_DAG` for more details.

The total causal effect on a given variable of interest (`response`) consequent to a joint intervention on a set of variables (`targets`) is defined according to Pearl's do-calculus theory and under the Gaussian assumption can be expressed as a function of parameters `(D,L)`, representing a (Cholesky) reparameterization of the covariance matrix.

Specifically, to each intervened variable a causal effect coefficient is associated and the posterior distribution of the latter can be recovered from posterior draws of the DAG parameters returned by `learn_DAG`. For each coefficient a sample of size S from its posterior is available. If required, the only Bayesian Model Average (BMA) estimate (obtained as the sample mean of the S draws) can be returned by setting `BMA = TRUE`.

Notice that, whenever implemented with `collapse = FALSE`, `learn_DAG` returns the marginal posterior distribution of DAGs only. In this case, `get_causaleffect` preliminarly performs posterior inference of DAG parameters by drawing samples from the posterior of `(D,L)`.

Print, summary and plot methods are available for this function. `print` returns the values of the prior hyperparameters used in the learnDAG function. `summary` returns, for each causal effect parameter, the marginal posterior mean and quantiles for different α levels, and posterior probabilities of negative, null and positive causal effects. `plot` provides graphical summaries (boxplot and histogram of the distribution) for the posterior of each causal effect parameter.

Value

An S3 object of class `bcdagCE` containing S draws from the joint posterior distribution of the p causal effect coefficients, organized into an (S, p) matrix, posterior means and credible intervals (under different $(1-\alpha)$ levels) for each causal effect coefficient, and marginal posterior probabilities of positive, null and negative causal effects.

Author(s)

Federico Castelletti and Alessandro Mascaro

References

J. Pearl (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge.

F. Castelletti and A. Mascaro (2021) Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

P. Nandy, M.H. Maathuis and T. Richardson (2017). Estimating the effect of joint interventions from observational data in sparse high-dimensional settings. *Annals of Statistics* 45(2), 647-674.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = w,
                     fast = TRUE, save.memory = FALSE)
head(out_mcmc$Graphs)
head(out_mcmc$L)
head(out_mcmc$D)
# Compute the BMA estimate of coefficients representing
# the causal effect on node 1 of an intervention on {3,4}
out_causal = get_causaleffect(learnDAG_output = out_mcmc, targets = c(3,4), response = 1)$post_mean

# Methods
print(out_causal)
```

```
summary(out_causal)
plot(out_causal)
```

`get_diagnostics` *MCMC diagnostics*

Description

This function provides diagnostics of convergence for the MCMC output of `learn_DAG` function.

Usage

```
get_diagnostics(learnDAG_output, ask = TRUE, nodes = integer(0))
```

Arguments

`learnDAG_output`	object of class `bcdag`
`ask`	Boolean argument passed to par() for visualization;
`nodes`	Numerical vector indicating those nodes for which we want to compute the posterior probability of edge inclusion;

Details

Function `learn_DAG` implements a Markov Chain Monte Carlo (MCMC) algorithm for structure learning and posterior inference of Gaussian DAGs. Output of the algorithm is a collection of S DAG structures (represented as (q, q) adjacency matrices) and DAG parameters (D, L) approximately drawn from the joint posterior. In addition, if `learn_DAG` is implemented with `collapse = TRUE`, the only approximate marginal posterior of DAGs (represented by the collection of S DAG structures) is returned; see the documentation of `learn_DAG` for more details.

Diagnostics of convergence for the MCMC output are conducted by monitoring across MCMC iterations: (1) the number of edges in the DAGs; (2) the posterior probability of edge inclusion for each possible edge $u- > v$. With regard to (1), a traceplot of the number of edges in the DAGs visited by the MCMC chain at each step $s = 1, ..., S$ is first provided as the output of the function. The absence of trends in the plot can provide information on a genuine convergence of the MCMC chain. In addition, the traceplot of the average number of edges in the DAGs visited up to time s, for $s = 1, ..., S$, is also returned. The convergence of the curve around a "stable" average size generally suggests good convergence of the algorithm. With regard to (2), for each edge $u- > v$, the posterior probability at time s, for $s = 1, ..., S$, can be estimated as as the proportion of DAGs visited by the MCMC up to time s which contain the directed edge $u- > v$. Output is organized in q plots (one for each node $v = 1, ..., q$), each summarizing the posterior probabilities of edges $u- > v$, $u = 1, ..., q$. If the number of nodes is larger than 30 the traceplot of a random sample of 30 nodes is returned.

Value

A collection of plots summarizing the behavior of the number of edges and the posterior probabilities of edge inclusion computed from the MCMC output.

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

F. Castelletti (2020). Bayesian model selection of Gaussian Directed Acyclic Graph structures. *International Statistical Review* 88 752-775.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC for posterior inference of DAGs only (collapse = TRUE)
out_mcmc = learn_DAG(S = 5000, burn = 1000, a = q, U = diag(1,q)/n, data = X, w = 0.1,
                                    fast = TRUE, save.memory = FALSE, collapse = TRUE)
# Produce diagnostic plots
get_diagnostics(out_mcmc)
```

`get_edgeprobs` *Compute posterior probabilities of edge inclusion from the MCMC output*

Description

This function computes the posterior probability of inclusion for each edge $u- > v$ given the MCMC output of `learn_DAG`;

Usage

```
get_edgeprobs(learnDAG_output)
```

Arguments

`learnDAG_output`
: object of class `bcdag`

Details

Output of learn_dag function consists of S draws from the joint posterior of DAGs and DAG-parameters in a zero-mean Gaussian DAG-model; see the documentation of learn_DAG for more details.

The posterior probability of inclusion of $u- > v$ is estimated as the frequency of DAGs visited by the MCMC which contain the directed edge $u- > v$. Posterior probabilities are collected in a (q, q) matrix with (u, v)-element representing the estimated posterior probability of edge $u- > v$.

Value

A (q, q) matrix with posterior probabilities of edge inclusion

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
# Generate observations from a Gaussian DAG-model
n = 200
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (Set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = 0.1,
                     fast = TRUE, save.memory = FALSE)
# Compute posterior probabilities of edge inclusion
get_edgeprobs(out_mcmc)
```

---

get_MAPdag *Compute the maximum a posteriori DAG model from the MCMC output*

Description

This function computes the maximum a posteriori DAG model estimate (MAP) from the MCMC output of learn_DAG

Usage

```
get_MAPdag(learnDAG_output)
```

Arguments

learnDAG_output	object of class bcdag

Details

Output of learn_dag function consists of S draws from the joint posterior of DAGs and DAG-parameters in a zero-mean Gaussian DAG-model; see the documentation of learn_DAG for more details.

The Maximum A Posteriori (MAP) model estimate is defined as the DAG visited by the MCMC with the highest associated posterior probability. Each DAG posterior probability is estimated as the frequency of visits of the DAG in the MCMC chain. The MAP estimate is represented through its (q, q) adjacency matrix, with (u, v)-element equal to one whenever the MAP contains $u- > v$, zero otherwise.

Value

The (q, q) adjacency matrix of the maximum a posteriori DAG model

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

G. Garcia-Donato and M.A. Martinez-Beneito (2013). On sampling strategies in Bayesian variable selection problems with large model spaces. *Journal of the American Statistical Association* 108 340-352.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
# Generate observations from a Gaussian DAG-model
n = 200
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (Set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = 0.1,
```

```
                    fast = TRUE, save.memory = FALSE)
# Produce the MAP DAG estimate
get_MAPdag(out_mcmc)
```

`get_MPMdag` *Compute the median probability DAG model from the MCMC output*

Description

This function computes the Median Probability DAG Model estimate (MPM) from the MCMC output of `learn_DAG`

Usage

```
get_MPMdag(learnDAG_output)
```

Arguments

`learnDAG_output`
: object of class `bcdag`

Details

Output of `learn_dag` function consists of S draws from the joint posterior of DAGs and DAG-parameters in a zero-mean Gaussian DAG-model; see the documentation of `learn_DAG` for more details.

The Median Probability DAG Model estimate (MPM) is obtained by including all edges whose posterior probability exceeds 0.5. The posterior probability of inclusion of $u- > v$ is estimated as the frequency of DAGs visited by the MCMC which contain the directed edge $u- > v$; see also function `get_edgeprobs` and the corresponding documentation.

Value

The (q, q) adjacency matrix of the median probability DAG model

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication

M.M. Barbieri and J.O. Berger (2004). Optimal predictive model selection. *The Annals of Statistics* 32 870-897

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
# Generate observations from a Gaussian DAG-model
n = 200
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (Set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = 0.1,
                     fast = TRUE, save.memory = FALSE)
# Produce the MPM DAG estimate
get_MPMdag(out_mcmc)
```

---

get_neighboringDAGs *Enumerate all neighbors of a DAG*

Description

This functions takes any DAG with q nodes as input and returns all the neighboring DAGs, i.e. all those DAGs that can be reached by the addition, removal or reversal of an edge.

Usage

```
get_neighboringDAGs(DAG)
```

Arguments

DAG	Adjacency matrix of a DAG

Value

The (q, q, K) array containing all neighboring DAGs, with K being the total number of neighbors

Examples

```
# Randomly generate a DAG
q <- 4; w <- 0.2
set.seed(123)
DAG <- rDAG(q,w)
# Get neighbors
neighbors <- get_neighboringDAGs(DAG)
neighbors
```

learn_DAG *MCMC scheme for Gaussian DAG posterior inference*

Description

This function implements a Markov Chain Monte Carlo (MCMC) algorithm for structure learning of Gaussian DAGs and posterior inference of DAG model parameters

Usage

```
learn_DAG(
  S,
  burn,
  data,
  a,
  U,
  w,
  fast = FALSE,
  save.memory = FALSE,
  collapse = FALSE,
  verbose = TRUE
)
```

Arguments

S	integer final number of MCMC draws from the posterior of DAGs and parameters
burn	integer initial number of burn-in iterations, needed by the MCMC chain to reach its stationary distribution and not included in the final output
data	(n, q) data matrix
a	common shape hyperparameter of the compatible DAG-Wishart prior, $a > q-1$
U	position hyperparameter of the compatible DAG-Wishart prior, a (q, q) s.p.d. matrix
w	edge inclusion probability hyperparameter of the DAG prior in $[0, 1]$
fast	boolean, if TRUE an approximate proposal for the MCMC moves is implemented
save.memory	boolean, if TRUE MCMC draws are stored as strings, instead of arrays
collapse	boolean, if TRUE only structure learning of DAGs is performed
verbose	If TRUE, progress bars are displayed

Details

Consider a collection of random variables $X_1, \ldots, X_q$ whose distribution is zero-mean multivariate Gaussian with covariance matrix Markov w.r.t. a Directed Acyclic Graph (DAG). Assuming the underlying DAG is unknown (model uncertainty), a Bayesian method for posterior inference on the

joint space of DAG structures and parameters can be implemented. The proposed method assigns a prior on each DAG structure through independent Bernoulli distributions, $Ber(w)$, on the 0-1 elements of the DAG adjacency matrix. Conditionally on a given DAG, a prior on DAG parameters (D, L) (representing a Cholesky-type reparameterization of the covariance matrix) is assigned through a compatible DAG-Wishart prior; see also function `rDAGWishart` for more details.

Posterior inference on the joint space of DAGs and DAG parameters is carried out through a Partial Analytic Structure (PAS) algorithm. Two steps are iteratively performed for $s = 1, 2, ...$: (1) update of the DAG through a Metropolis Hastings (MH) scheme; (2) sampling from the posterior distribution of the (updated DAG) parameters. In step (1) the update of the (current) DAG is performed by drawing a new (direct successor) DAG from a suitable proposal distribution. The proposed DAG is obtained by applying a local move (insertion, deletion or edge reversal) to the current DAG and is accepted with probability given by the MH acceptance rate. The latter requires to evaluate the proposal distribution at both the current and proposed DAGs, which in turn involves the enumeration of all DAGs that can be obtained from local moves from respectively the current and proposed DAG. Because the ratio of the two proposals is approximately equal to one, and the approximation becomes as precise as q grows, a faster strategy implementing such an approximation is provided with `fast = TRUE`. The latter choice is especially recommended for moderate-to-large number of nodes q.

Output of the algorithm is a collection of S DAG structures (represented as (q, q) adjacency matrices) and DAG parameters (D, L) approximately drawn from the joint posterior. The various outputs are organized in (q, q, S) arrays; see also the example below. If the target is DAG learning only, a collapsed sampler implementing the only step (1) of the MCMC scheme can be obtained by setting `collapse = TRUE`. In this case, the algorithm outputs a collection of S DAG structures only. See also functions `get_edgeprobs`, `get_MAPdag`, `get_MPMdag` for posterior summaries of the MCMC output.

Print, summary and plot methods are available for this function. `print` provides information about the MCMC output and the values of the input prior hyperparameters. `summary` returns, besides the previous information, a (q, q) matrix collecting the marginal posterior probabilities of edge inclusion. `plot` returns the estimated Median Probability DAG Model (MPM), a (q, q) heat map with estimated marginal posterior probabilities of edge inclusion, and a barplot summarizing the distribution of the size of DAGs visited by the MCMC.

Value

An S3 object of class `bcdag` containing S draws from the posterior of DAGs and (if `collapse = FALSE`) of DAG parameters D and L. If `save.memory = FALSE`, these are stored in three arrays of dimension (q, q, S). Otherwise, they are stored as strings.

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

F. Castelletti and A. Mascaro (2022). BCDAG: An R package for Bayesian structural and Causal learning of Gaussian DAGs. *arXiv pre-print*, url: https://arxiv.org/abs/2201.12003

F. Castelletti (2020). Bayesian model selection of Gaussian Directed Acyclic Graph structures. *International Statistical Review* 88 752-775.

Examples

```
# Randomly generate a DAG and the DAG-parameters
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
# Generate observations from a Gaussian DAG-model
n = 200
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)

## Set S = 5000 and burn = 1000 for better results

# [1] Run the MCMC for posterior inference of DAGs and parameters (collapse = FALSE)
out_mcmc = learn_DAG(S = 50, burn = 10, a = q, U = diag(1,q)/n, data = X, w = 0.1,
                     fast = FALSE, save.memory = FALSE, collapse = FALSE)
# [2] Run the MCMC for posterior inference of DAGs only (collapse = TRUE)
out_mcmc_collapse = learn_DAG(S = 50, burn = 10, a = q, U = diag(1,q)/n, data = X, w = 0.1,
                              fast = FALSE, save.memory = FALSE, collapse = TRUE)
# [3] Run the MCMC for posterior inference of DAGs only with approximate proposal
# distribution (fast = TRUE)
# out_mcmc_collapse_fast = learn_DAG(S = 50, burn = 10, a = q, U = diag(1,q)/n, data = X, w = 0.1,
#                                    fast = FALSE, save.memory = FALSE, collapse = TRUE)
# Compute posterior probabilities of edge inclusion and Median Probability DAG Model
# from the MCMC outputs [2] and [3]
get_edgeprobs(out_mcmc_collapse)
# get_edgeprobs(out_mcmc_collapse_fast)
get_MPMdag(out_mcmc_collapse)
# get_MPMdag(out_mcmc_collapse_fast)

# Methods
print(out_mcmc)
summary(out_mcmc)
plot(out_mcmc)
```

---

leukemia *Protein levels for 68 diagnosed AML patients of subtype M2*

Description

A dataset containing the protein expression levels of 18 proteins for 68 AML patients of subtype M2 (according to French-American-British (FAB) classification system). The 18 proteins selected are known to be involved in apoptosis and cell cycle regulation according to the KEGG database (Kanehisa et al. 2012). This is a subset of the dataset presented in Kornblau et al. (2009).

Usage

```
leukemia
```

Format

A data frame with 68 rows and 18 variables:

AKT AKT protein, expression level

AKT.p308 AKT.p308 protein, expression level

AKT.p473 AKT.p473 protein, expression level

BAD BAD protein, expression level

BAD.p112 BAD.p112 protein, expression level

BAD.p136 BAD.p136 protein, expression level

BAD.p155 BAD.p155 protein, expression level

BAX BAX protein, expression level

BCL2 BCL2 protein, expression level

BCLXL BCLXL protein, expression level

CCND1 CCND1 protein, expression level

GSK3 GSK3 protein, expression level

GSK3.p GSK3.p protein, expression level

MYC MYC protein, expression level

PTEN PTEN protein, expression level

PTEN.p PTEN.p protein, expression level

TP53 TP53 protein, expression level

XIAP XIAP protein, expression level ...

Source

http://bioinformatics.mdanderson.org/Supplements/Kornblau-AML-RPPA/aml-rppa.xls

References

Kornblau, S. M., Tibes, R., Qiu, Y. H., Chen, W., Kantarjian, H. M., Andreeff, M., ... & Mills, G. B. (2009). Functional proteomic profiling of AML predicts response and survival. Blood, The Journal of the American Society of Hematology, 113(1), 154-164.

Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., & Tanabe, M. (2012). KEGG for integration and interpretation of large-scale molecular data sets. Nucleic acids research, 40(D1), D109-D114.

plot.bcdag *bcdag object plot*

Description

This method returns summary plots of the output of learn_DAG().

Usage

```
## S3 method for class 'bcdag'
plot(x, ..., ask = TRUE)
```

Arguments

x	a bcdag object for which a plot is desired
...	additional arguments affecting the summary produced
ask	Boolean argument passed to par() for visualization;

Value

Plot of the Median Probability DAG, a heatmap of the probabilities of edge inclusion and an histogram of the sizes of graphs visited by learn_DAG().

Examples

```
n <- 1000
q <- 4
DAG <- matrix(c(0,1,1,0,0,0,0,0,0,0,0,0,0,1,1,0), nrow = q)

L <- DAG
L[L != 0] <- runif(q, 0.2, 1)
diag(L) <- c(1,1,1,1)
D <- diag(1, q)
Sigma <- t(solve(L))%*%D%*%solve(L)

a <- 6
g <- 1/1000
U <- g*diag(1,q)
w = 0.2

set.seed(1)
X <- mvtnorm::rmvnorm(n, sigma = Sigma)

out <- learn_DAG(1000, 0, X, a, U, w, fast = TRUE, collapse = TRUE, save.memory = FALSE)
plot(out)
```

plot.bcdagCE *bcdagCE object plot*

Description

This method returns summary plots of the output of `get_causaleffect()`.

Usage

```
## S3 method for class 'bcdagCE'
plot(x, ..., which_ce = integer(0))
```

Arguments

x	a bcdagCE object for which a plot is desired
...	additional arguments affecting the summary produced
which_ce	specifies the list of nodes for which you intend to generate a boxplot and a histogram

Value

Boxplot and histogram of the posterior distribution of the causal effects computed using get_causaleffect().

Examples

```
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = w,
                     fast = TRUE, save.memory = FALSE, verbose = FALSE)
out_ce <- get_causaleffect(out_mcmc, targets = c(4,6), response = 1)
plot(out_ce)
```

print.bcdag *bcdag object print*

Description

This method returns a summary of the input given to `learn_DAG()` to produce the bcdag object.

Usage

```
## S3 method for class 'bcdag'
print(x, ...)
```

Arguments

x	a bcdag object for which a summary is desired
...	additional arguments affecting the summary produced

Value

A printed message listing the inputs given to learn_DAG.

Examples

```
n <- 1000
q <- 4
DAG <- matrix(c(0,1,1,0,0,0,0,0,0,0,0,0,1,1,0), nrow = q)

L <- DAG
L[L != 0] <- runif(q, 0.2, 1)
diag(L) <- c(1,1,1,1)
D <- diag(1, q)
Sigma <- t(solve(L))%*%D%*%solve(L)

a <- 6
g <- 1/1000
U <- g*diag(1,q)
w = 0.2

set.seed(1)
X <- mvtnorm::rmvnorm(n, sigma = Sigma)

out <- learn_DAG(1000, 0, X, a, U, w, fast = TRUE, collapse = TRUE, save.memory = FALSE)
print(out)
```

print.bcdagCE *bcdagCE object print*

Description

This method returns a summary of the inputs given to learn_DAG() and get_causaleffect() to obtain the bcdagCE object.

Usage

```
## S3 method for class 'bcdagCE'
print(x, ...)
```

Arguments

x	a bcdagCE object for which a summary is desired
...	additional arguments affecting the summary produced

Value

A printed message listing the inputs given to learn_DAG and get_causaleffect.

Examples

```
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = w,
                     fast = TRUE, save.memory = FALSE, verbose = FALSE)
out_ce <- get_causaleffect(out_mcmc, targets = c(4,6), response = 1)
print(out_ce)
```

rDAG *Generate a Directed Acyclic Graph (DAG) randomly*

Description

This function randomly generates a Directed Acyclic Graph (DAG) with q nodes and probability of edge inclusion w.

Usage

```
rDAG(q, w)
```

Arguments

q	number of nodes
w	probability of edge inclusion in $[0, 1]$

Details

The $0-1$ adjacency matrix of the DAG is generated by drawing each element in the lower triangular part in $0, 1$ with probability $1 - w, w$. Accordingly, the DAG has lower-triangular adjacency matrix and nodes are numerically labeled according to a topological ordering implying $u > v$ whenever $u- > v$.

Value

DAG (q, q) adjacency matrix of the generated DAG

Examples

```
# Randomly generate a DAG on q = 8 nodes with probability of edge inclusion w = 0.2
q = 8
w = 0.2
set.seed(123)
rDAG(q = q, w = w)
```

rDAGWishart *Random samples from a compatible DAG-Wishart distribution*

Description

This function implements a direct sampling from a compatible DAG-Wishart distribution with parameters a and U.

Usage

```
rDAGWishart(n, DAG, a, U)
```

Arguments

n	number of samples
DAG	(q, q) adjacency matrix of the DAG
a	common shape hyperparameter of the compatible DAG-Wishart, $a > q - 1$
U	position hyperparameter of the compatible DAG-Wishart, a (q, q) s.p.d. matrix

Details

Assume the joint distribution of random variables $X_1, \ldots, X_q$ is zero-mean Gaussian with covariance matrix Markov w.r.t. a Directed Acyclic Graph (DAG). The allied Structural Equation Model (SEM) representation of a Gaussian DAG-model allows to express the covariance matrix as a function of the (Cholesky) parameters (D, L), collecting the regression coefficients and conditional variances of the SEM.

The DAG-Wishart distribution (Cao et. al, 2019) with shape hyperparameter $a = (a_1, ..., a_q)$ and position hyperparameter U (a s.p.d. (q, q) matrix) provides a conjugate prior for parameters (D, L). In addition, to guarantee compatibility among Markov equivalent DAGs (same marginal likelihood), the default choice (here implemented) $a_j = a + |pa(j)| - q + 1$ $(a > q - 1)$, with $|pa(j)|$ the number of parents of node j in the DAG, was introduced by Peluso and Consonni (2020).

Value

A list of two elements: a (q, q, n) array collecting n sampled matrices L and a (q, q, n) array collecting n sampled matrices D

Author(s)

Federico Castelletti and Alessandro Mascaro

References

F. Castelletti and A. Mascaro (2021). Structural learning and estimation of joint causal effects among network-dependent variables. *Statistical Methods and Applications*, Advance publication.

X. Cao, K. Khare and M. Ghosh (2019). Posterior graph selection and estimation consistency for high-dimensional Bayesian DAG models. *The Annals of Statistics* 47 319-348.

S. Peluso and G. Consonni (2020). Compatible priors for model selection of high-dimensional Gaussian DAGs. *Electronic Journal of Statistics* 14(2) 4110 - 4132.

Examples

```
# Randomly generate a DAG on q = 8 nodes with probability of edge inclusion w = 0.2
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
```

```
# Draw from a compatible DAG-Wishart distribution with parameters a = q and U = diag(1,q)
outDL = rDAGWishart(n = 5, DAG = DAG, a = q, U = diag(1, q))
outDL
```

---

summary.bcdag *bcdag object summaries*

Description

This method produces summaries of the input and output of the function `learn_DAG()`.

Usage

```
## S3 method for class 'bcdag'
summary(object, ...)
```

Arguments

`object`	a bcdag object for which a summary is desired
`...`	additional arguments affecting the summary produced

Value

A printed message listing the inputs given to learn_DAG and the estimated posterior probabilities of edge inclusion.

Examples

```
n <- 1000
q <- 4
DAG <- matrix(c(0,1,1,0,0,0,0,0,0,0,0,0,0,1,1,0), nrow = q)

L <- DAG
L[L != 0] <- runif(q, 0.2, 1)
diag(L) <- c(1,1,1,1)
D <- diag(1, q)
Sigma <- t(solve(L))%*%D%*%solve(L)

a <- 6
g <- 1/1000
U <- g*diag(1,q)
w = 0.2

set.seed(1)
X <- mvtnorm::rmvnorm(n, sigma = Sigma)

out <- learn_DAG(1000, 0, X, a, U, w, fast = TRUE, collapse = TRUE, save.memory = FALSE)
summary(out)
```

summary.bcdagCE *bcdagCE object summary*

Description

This method produces summaries of the input and output of the function `get_causaleffect()`.

Usage

```
## S3 method for class 'bcdagCE'
summary(object, ...)
```

Arguments

`object`	a `bcdagCE` object for which a summary is desired
`...`	additional arguments affecting the summary produced

Value

A printed message listing the inputs given to learn_DAG() and get_causaleffect() and summary statistics of the posterior distribution.

Examples

```
q = 8
w = 0.2
set.seed(123)
DAG = rDAG(q = q, w = w)
outDL = rDAGWishart(n = 1, DAG = DAG, a = q, U = diag(1, q))
L = outDL$L; D = outDL$D
Sigma = solve(t(L))%*%D%*%solve(L)
n = 200
# Generate observations from a Gaussian DAG-model
X = mvtnorm::rmvnorm(n = n, sigma = Sigma)
# Run the MCMC (set S = 5000 and burn = 1000 for better results)
out_mcmc = learn_DAG(S = 500, burn = 100, a = q, U = diag(1,q)/n, data = X, w = w,
                     fast = TRUE, save.memory = FALSE, verbose = FALSE)
out_ce <- get_causaleffect(out_mcmc, targets = c(4,6), response = 1)
# summary(out_ce)
```

Index