Description

The R package BDgraph provides statistical tools for Bayesian structure learning in undirected graphical models. The package is implemented the recent improvements in the Bayesian graphical models literature, including Mohammadi and Wit (2015), Mohammadi et al. (2017), and Dobra and Mohammadi (2018). The computationally intensive tasks of the package are implemented in parallel using OpenMP in C++ and interfaced with R, to speed up the computations. Besides, the package contains several functions for simulation and visualization, as well as three multivariate datasets taken from the literature.

Details

The package includes the following main functions:
bdgraph  Search algorithm in graphical models
bdgraph.mp1  Search algorithm in graphical models using marginal pseudo-likelihood
bdgraph.sim  Graph data simulation
graph.sim  Graph simulation
bdgraph.ts  Search algorithm in time series graphical models
bdgraph.npn  Nonparametric transfer
compare  Graph structure comparison
plinks  Estimated posterior link probabilities
plotcoda  Convergence plot
plotroc  ROC plot
rgwish  Sampling from G-Wishart distribution
rwish  Sampling from Wishart distribution
select  Graph selection
traceplot  Trace plot of graph size

How to cite this package

Whenever using this package, please cite as


Author(s)

Reza Mohammadi
Amsterdam Business School
University of Amsterdam
Maintainer: Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

bdgraph, bdgraph.mpl, bdgraph.sim, compare, rgwish

Examples

```r
## Not run:
library( BDgraph )

# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 70, p = 6, size = 7, vis = TRUE )

# Running algorithm based on GGMs
bdgraph.obj <- bdgraph( data = data.sim, iter = 5000 )

summary( bdgraph.obj )

# To compare the result with true graph
compare( data.sim, bdgraph.obj, main = c( "True graph", "BDgraph" ), vis = TRUE )

# Running algorithm based on GGMs and marginal pseudo-likelihood
bdgraph.obj_mpl <- bdgraph.mpl( data = data.sim, iter = 5000 )

summary( bdgraph.obj_mpl )

# To compare the results of both algorithms with true graph
compare( data.sim, bdgraph.obj, bdgraph.obj_mpl,
        main = c( "True graph", "BDgraph", "BDgraph_mpl" ), vis = TRUE )

## End(Not run)
```

bdgraph

Search algorithm in graphical models

Description

As the main function of the BDgraph package, this function consists of several sampling algorithms for Bayesian model determination in undirected graphical models. To speed up the computations, the birth-death MCMC sampling algorithms are implemented in parallel using OpenMP in C++.
bdgraph

Usage

bdgraph( data, n = NULL, method = "ggm", not.cont = NULL, 
algorithm = "bdmcmc", iter = 5000, burnin = iter / 2, 
g.prior = 0.5, df.prior = 3, g.start = "empty", 
jump = NULL, save = FALSE, print = 1000, 
cores = NULL, threshold = 1e-8 )

Arguments

data
There are two options: (1) an \((n \times p)\) matrix or a \(\text{data.frame}\) corresponding to the data, (2) an \((p \times p)\) covariance matrix as \(S = X'X\) which \(X\) is the data matrix \((n\) is the sample size and \(p\) is the number of variables). It also could be an object of class "sim", from function \texttt{bdgraph.sim}. The input matrix is automatically identified by checking the symmetry.

n
The number of observations. It is needed if the "data" is a covariance matrix.

method
A character with two options "ggm" (default) and "gcgm". Option "ggm" is for Gaussian graphical models based on Gaussianity assumption. Option "gcgm" is for Gaussian copula graphical models for the data that not follow Gaussianity assumption (e.g. continuous non-Gaussian, discrete, or mixed dataset).

not.cont
For the case method = "gcgm", a vector with binary values in which 1 for not continuous variables.

algorithm
A character with two options "bdmcmc" (default) and "rjmcmc". Option "bdmcmc" is based on birth-death MCMC algorithm. Option "rjmcmc" is based on reversible jump MCMC algorithm.

iter
The number of iteration for the sampling algorithm.

burnin
The number of burn-in iteration for the sampling algorithm.

g.prior
For determining the prior distribution of each edge in the graph. There are two options: a single value between 0 and 1 (e.g. 0.5 as a noninformative prior) or an \((p \times p)\) matrix with elements between 0 and 1.

df.prior
The degree of freedom for G-Wishart distribution, \(W_G(b,D)\), which is a prior distribution of the precision matrix.

g.start
Corresponds to a starting point of the graph. It could be an \((p \times p)\) matrix, "empty" (default), or "full". Option "empty" means the initial graph is an empty graph and "full" means a full graph. It also could be an object with S3 class "bdgraph" of package \texttt{BDgraph} or the class "ssgraph" of package \texttt{ssgraph}; this option can be used to run the sampling algorithm from the last objects of previous run (see examples).

jump
It is only for the BDMCMC algorithm (algorithm = "bdmcmc"). It is for simultaneously updating multiple links at the same time to update graph in the BDMCMC algorithm.

save
Logical: if FALSE (default), the adjacency matrices are NOT saved. If TRUE, the adjacency matrices after burn-in are saved.

print
Value to see the number of iteration for the MCMC algorithm.
The number of cores to use for parallel execution. The case `cores = "all"` means all CPU cores to use for parallel execution. The default is to use "all" CPU cores of the computer.

threshold

The threshold value for the convergence of sampling algorithm from G-Wishart for the precision matrix.

Value

An object with S3 class "bdgraph" is returned:

- **p_links**: An upper triangular matrix which corresponds the estimated posterior probabilities of all possible links.
- **K_hat**: The posterior estimation of the precision matrix.

For the case "save = TRUE" is returned:

- **sample_graphs**: A vector of strings which includes the adjacency matrices of visited graphs after burn-in.
- **graph_weights**: A vector which includes the waiting times of visited graphs after burn-in.
- **all_graphs**: A vector which includes the identity of the adjacency matrices for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **all_weights**: A vector which includes the waiting times for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.

Author(s)

Reza Mohammadi &lt;a.mohammadi@uva.nl&gt; and Ernst Wit

References


See Also

*bdgraph.mpl, bdgraph.sim, summary.bdgraph, compare*
Examples

```r
## Not run:
# --- Example 1
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 20, p = 6, size = 7, vis = TRUE )
bdgraph.obj <- bdgraph( data = data.sim, iter = 1000 )
summary( bdgraph.obj )

# To compare our result with true graph
compare( data.sim, bdgraph.obj, main = c( "True graph", "BDgraph" ) )

# Running algorithm with starting points from previous run
bdgraph.obj2 <- bdgraph( data = data.sim, g.start = bdgraph.obj )

compare( data.sim, bdgraph.obj, bdgraph.obj2,
       main = c( "True graph", "Frist run", "Second run" ) )

# --- Example 2
# Generating mixed data from a 'scale-free' graph
data.sim <- bdgraph.sim( n = 50, p = 6, type = "mixed", graph = "scale-free", vis = TRUE )
bddgraph.obj <- bdgraph( data = data.sim, method = "gcm", iter = 10000 )

summary( bdgraph.obj )

compare( data.sim, bdgraph.obj )

## End(Not run)
```

bdgraph.mpl

Search algorithm in graphical models using marginal pseudo-likelihood

Description

This function consists of several sampling algorithms for Bayesian model determination in undirected graphical models based on marginal pseudo-likelihood. To speed up the computations, the birth-death MCMC sampling algorithms are implemented in parallel using OpenMP in C++.

Usage

```r
bdgraph.mpl( data, n = NULL, method = "gcm", transfer = TRUE,
        algorithm = "bdmcmc", iter = 5000, burnin = iter / 2,
        g.prior = 0.5, g.start = "empty",
        jump = NULL, alpha = 0.5, save = FALSE,
        print = 1000, cores = NULL, operator = "or" )
```
Arguments

- **data**
  - There are two options: (1) an \((n \times p)\) matrix or a data.frame corresponding to the data, (2) an \((p \times p)\) covariance matrix as \(S = X'X\) which \(X\) is the data matrix \((n\) is the sample size and \(p\) is the number of variables). It also could be an object of class "sim", from function `bdgraph.sim`. The input matrix is automatically identified by checking the symmetry.

- **n**
  - The number of observations. It is needed if the "data" is a covariance matrix.

- **method**
  - A character with two options "ggm" (default), "dgm" and "dgm-binary". Option "ggm" is for Gaussian graphical models based on Gaussianity assumption. Option "dgm" is for discrete graphical models for the data that are discrete. Option "dgm-binary" is for discrete graphical models for the data that are binary.

- **transfer**
  - For only discrete data which method = "dgm" or method = "dgm-binary".

- **algorithm**
  - A character with two options "bdmcmc" (default) and "rjmcmc". Option "bdmcmc" is based on birth-death MCMC algorithm. Option "rjmcmc" is based on reversible jump MCMC algorithm.

- **iter**
  - The number of iteration for the sampling algorithm.

- **burnin**
  - The number of burn-in iteration for the sampling algorithm.

- **g.prior**
  - For determining the prior distribution of each edge in the graph. There are two options: a single value between 0 and 1 (e.g. 0.5 as a noninformative prior) or an \((p \times p)\) matrix with elements between 0 and 1.

- **g.start**
  - Corresponds to a starting point of the graph. It could be an \((p \times p)\) matrix, "empty" (default), or "full". Option "empty" means the initial graph is an empty graph and "full" means a full graph. It also could be an object with S3 class "bdgraph" of package BDgraph or the class "ssgraph" of package ssgraph; this option can be used to run the sampling algorithm from the last objects of previous run (see examples).

- **jump**
  - It is only for the BDMCMC algorithm (algorithm = "bdmcmc"). It is for simultaneously updating multiple links at the same time to update graph in the BDMCMC algorithm.

- **alpha**
  - Value of the hyper parameter of Dirichlet, which is a prior distribution.

- **save**
  - Logical: if FALSE (default), the adjacency matrices are NOT saved. If TRUE, the adjacency matrices after burn-in are saved.

- **print**
  - Value to see the number of iteration for the MCMC algorithm.

- **cores**
  - The number of cores to use for parallel execution. The case cores= "all" means all CPU cores to use for parallel execution. The default is to use all CPU cores of the computer.

- **operator**
  - A character with two options "or" (default) and "and". It is for hill-climbing algorithm.

Value

An object with S3 class "bdgraph" is returned:

- **p_links**
  - An upper triangular matrix which corresponds the estimated posterior probabilities of all possible links.
For the case "save = TRUE" is returned:

- **sample_graphs**: A vector of strings which includes the adjacency matrices of visited graphs after burn-in.
- **graph_weights**: A vector which includes the waiting times of visited graphs after burn-in.
- **all_graphs**: A vector which includes the identity of the adjacency matrices for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.
- **all_weights**: A vector which includes the waiting times for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.

**Author(s)**

Reza Mohammadi (<a.mohammadi@uva.nl>, Adrian Dobra, and Johan Pensar)

**References**


**See Also**

`bdgraph`, `bdgraph.sim`, `summary.bdgraph`, `compare`

**Examples**

```r
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 70, p = 5, size = 7, vis = TRUE )

bdgraph.obj <- bdgraph.mpl( data = data.sim, iter = 500 )

summary( bdgraph.obj )

# To compare the result with true graph
compare( data.sim, bdgraph.obj, main = c( "True graph", "BDgraph" ) )
```
bdgraph.npn  
Nonparametric transfer  

Description  
Transfers non-Gaussian data to Gaussian.

Usage  
bdgraph.npn( data, npp = "shrinkage", npp.thresh = NULL )

Arguments  
data  
An \((n \times p)\) matrix or a data.frame corresponding to the data \((n\) is the sample size and \(p\) is the number of variables).  
npp  
A character with three options "shrinkage" (default), "truncation", and "skeptic". Option "shrinkage" is for the shrunken transformation, option "truncation" is for the truncated transformation and option "skeptic" is for the non-paranormal skeptic transformation. For more details see references.  
npp.thresh  
The truncation threshold; it is only for the truncated transformation (npp= "truncation"). The default value is \(1/(4n^{1/4}\sqrt{n}\log(n))\).

Value  
data  
An \((n\times p)\) matrix of transferred data, if npp = "shrinkage" or "truncation", and a non-paranormal correlation \((p \times p)\) matrix, if npp = "skeptic".

Author(s)  
Reza Mohammadi <a.mohammadi@uva.nl>

References  

See Also  
bdgraph.sim, bdgraph, bdgraph.mpl
Examples

## Not run:

```r
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 6, p = 4, size = 4 )data <- ( data.sim $ data - 3 ) ^ 4

# Transfer the data by truncation
bdgraph.nnpn( data, nnpn = "truncation" )

# Transfer the data by shrunken
bdgraph.nnpn( data, nnpn = "shrunken" )

# Transfer the data by skeptic
bdgraph.nnpn( data, nnpn = "skeptic" )

## End(Not run)
```

bdgraph.sim

### Description

Simulating multivariate distributions with different types of underlying graph structures, including "random", "cluster", "scale-free", "hub", "fixed", "circle", "AR(1)", "AR(2)", "star", and "lattice". Based on the underlying graph structure, it generates four different types of datasets, including multivariate Gaussian, non-Gaussian, discrete, or mixed data. This function can be used also for only simulating graphs by option `n=0`, as a default.

### Usage

```r
bdgraph.sim( p = 10, graph = "random", n = 0, type = "Gaussian", prob = 0.2,
             size = NULL, mean = 0, class = NULL, cut = 4, b = 3,
             D = diag( p ), K = NULL, sigma = NULL, vis = FALSE )
```

### Arguments

- **p**: The number of variables (nodes).
- **graph**: The graph structure with option "random", "cluster", "scale-free", "hub", "fixed", "circle", "AR(1)", "AR(2)", "star", and "lattice". It also could be an adjacency matrix corresponding to a graph structure (an upper triangular matrix in which $g_{ij} = 1$ if there is a link between notes $i$ and $j$, otherwise $g_{ij} = 0$).
- **n**: The number of samples required. Note that for the case $n = 0$, only graph is generated.
The type of data with four options: "Gaussian" (default), "non-Gaussian", "discrete", "mixed", and "binary". For option "Gaussian", data are generated from multivariate normal distribution. For option "non-Gaussian", data are transferred multivariate normal distribution to continuous multivariate non-Gaussian distribution. For option "discrete", data are transferred from multivariate normal distribution to discrete multivariate distribution. For option "mixed", data are transferred from multivariate normal distribution to mixture of 'count', 'ordinal', 'non-Gaussian', 'binary' and 'Gaussian', respectively. For option "binary", data are generated directly from the joint distribution, in this case \( p \) must be less than 17.

- **prob**: If graph="random", it is the probability that a pair of nodes has a link.
- **size**: The number of links in the true graph (graph size).
- **mean**: A vector specifies the mean of the variables.
- **class**: If graph="cluster", it is the number of classes.
- **cut**: If type="discrete", it is the number of categories for simulating discrete data.
- **b**: The degree of freedom for G-Wishart distribution, \( W_G(b,D) \).
- **D**: The positive definite \((p \times p)\) "scale" matrix for G-Wishart distribution, \( W_G(b, D) \). The default is an identity matrix.
- **K**: If graph="fixed", it is a positive-definite symmetric matrix specifies as a true precision matrix.
- **sigma**: If graph="fixed", it is a positive-definite symmetric matrix specifies as a true covariance matrix.
- **vis**: Visualize the true graph structure.

**Value**

An object with S3 class "sim" is returned:

- **data**: Generated data as an \((n \times p)\) matrix.
- **sigma**: The covariance matrix of the generated data.
- **K**: The precision matrix of the generated data.
- **G**: The adjacency matrix corresponding to the true graph structure.

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

**References**


See Also

`graph.sim`, `bdgraph`, `bdgraph.mpl`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( p = 10, n = 50, prob = 0.3, vis = TRUE )
print( data.sim )

# Generating multivariate normal data from a 'hub' graph
data.sim <- bdgraph.sim( p = 6, n = 3, graph = "hub", vis = FALSE )
round( data.sim $ data, 2 )

# Generating mixed data from a 'hub' graph
data.sim <- bdgraph.sim( p = 8, n = 10, graph = "hub", type = "mixed" )
round( data.sim $ data, 2 )

# Generating only a 'scale-free' graph (with no data)
graph.sim <- bdgraph.sim( p = 8, graph = "scale-free" )
plot( graph.sim )
graph.sim $ G

## End(Not run)
```

Description

This function is for Bayesian model determination in time series graphical models, based on birth-death MCMC method.

Usage

```r
bdgraph.ts( data, NLength = NULL, n, iter = 1000, burnin = iter / 2,
g.prior = 0.5, df.prior = rep( 3, Nlength ), g.start = "empty",
save = FALSE, print = 500, cores = NULL )
```
Arguments

- **data**: The aggregate periodogram $P_k$, which is arranged as a large $px(N \cdot length \cdot p)$ matrix $[P_1, P_2, ..., P_{N \cdot length}]$.

- **nlength**: The length of the time series.

- **n**: The number of observations.

- **iter**: The number of iteration for the sampling algorithm.

- **burnin**: The number of burn-in iteration for the sampling algorithm.

- **g.prior**: For determining the prior distribution of each edge in the graph. There are two options: a single value between 0 and 1 (e.g. 0.5 as a noninformative prior) or an $(p \times p)$ matrix with elements between 0 and 1.

- **df.prior**: The degree of freedom for complex G-Wishart distribution, $CW_G(b, D)$, which is a prior distribution of the precision matrix in each frequency.

- **g.start**: Corresponds to a starting point of the graph. It could be "empty" (default) and "full". Option "empty" means the initial graph is an empty graph and "full" means a full graph. It also could be an object with S3 class "bdgraph"; with this option we could run the sampling algorithm from the last objects of previous run (see examples).

- **save**: Logical: if FALSE (default), the adjacency matrices are NOT saved. If TRUE, the adjacency matrices after burn-in are saved.

- **print**: Value to see the number of iteration for the MCMC algorithm.

- **cores**: The number of cores to use for parallel execution. The case cores="all" means all CPU cores to use for parallel execution. The default is to use "all" CPU cores of the computer.

Value

An object with S3 class "bdgraph" is returned:

- **p_links**: An upper triangular matrix which corresponds the estimated posterior probabilities of all possible links.

- **K_hat**: The posterior estimation of the precision matrix.

For the case "save = TRUE" is returned:

- **sample_graphs**: A vector of strings which includes the adjacency matrices of visited graphs after burn-in.

- **graph_weights**: A vector which includes the waiting times of visited graphs after burn-in.

- **all_graphs**: A vector which includes the identity of the adjacency matrices for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.

- **all_weights**: A vector which includes the waiting times for all iterations after burn-in. It is needed for monitoring the convergence of the BD-MCMC algorithm.

- **status**: An integer to indicate the iteration where the algorithm exits, since if the sum of all rates is 0 at some iteration, the graph at this iteration is regarded as the real graph. It is 0 if the algorithm doesn’t exit.
Author(s)
Lang Liu, Nicholas Foti, Alex Tank and Reza Mohammadi <a.mohammadi@uva.nl>

References

See Also
bdgraph, bdgraph.mpl, bdgraph.sim, summary.bdgraph, compare

Examples
```r
## Not run:
# Generating time series data
Nlength = 100; N = 150; p = 6; b = 3; threshold = 1e-8;

I = diag( p )
A = 0.5 * matrix( rbinom( p * p, 1, 0.2 ), p, p )

A[ lower.tri( A ) ] = 0
diag( A ) = 0.5

G = matrix( 0, p, p )
K = matrix( 0, p, p * Nlength )

lambda = seq( 0, Nlength - 1, 1 ) * 2 * base::pi / Nlength

K0 = matrix( 0, p, p * Nlength )
K_times = matrix( 1, p, p )

for( k in 1 : Nlength )
{ # Compute K0
  K0[ , ( k * p - p + 1 ) : ( k * p ) ] = I + t( A ) %*% A +
  complex( 1, cos( -lambda[ k ] ), sin( -lambda[ k ] ) ) * A +
  complex( 1, cos( lambda[ k ] ), sin( lambda[ k ] ) ) * t( A )

  K_times = K_times * ( K0[ , ( k * p - p + 1 ) : ( k * p ) ] ! = 0 )

  diag( K[ , ( k * p - p + 1 ) : ( k * p ) ] ) = 1
}
Churn data set

Description

The data can be downloaded from IBM Sample Data Sets. Customer churn occurs when customers stop doing business with a company, also known as customer attrition. The data set contains 3333 rows (customers) and 20 columns (features). The "Churn" column is our target which indicate whether customer churned (left the company) or not.
Usage

data(churn)

Format

The churn dataset, as a data frame, contains 3333 rows (customers) and 20 columns (variables/features). The 20 variables are:

- **State**: Categorical, for the 50 states and the District of Columbia.
- **Account.Length**: count, how long account has been active.
- **Area.Code**: Categorical.
- **Int.1.Plan**: Categorical, yes or no, international plan.
- **VMail.Plan**: Categorical, yes or no, voice mail plan.
- **VMail.Message**: Count, number of voice mail messages.
- **Day.Mins**: Continuous, minutes customer used service during the day.
- **Day.Calls**: Count, total number of calls during the day.
- **Day.Charge**: Continuous, total charge during the day.
- **Eve.Mins**: Continuous, minutes customer used service during the evening.
- **Eve.Calls**: Count, total number of calls during the evening.
- **Eve.Charge**: Continuous, total charge during the evening.
- **Night.Mins**: Continuous, minutes customer used service during the night.
- **Night.Calls**: Count, total number of calls during the night.
- **Night.Charge**: Continuous, total charge during the night.
- **Intl.Mins**: Continuous, minutes customer used service to make international calls.
- **Intl.Calls**: Count, total number of international calls.
- **Intl.Charge**: Continuous, total international charge.
- **CustServ.Calls**: Count, number of calls to customer service.
- **Churn**: Categorical, True or False. Indicator of whether the customer has left the company (True or False).

References


Examples

data(churn)

summary(churn)
**Graph structure comparison**

**Description**

This function provides several measures to assess the performance of the graphical structure learning.

**Usage**

```r
compare( simNobj, bdgraphNobj, bdgraphNobj2 = NULL, bdgraphNobj3 = NULL,
    main = NULL, vis = FALSE )
```

**Arguments**

- `sim.obj`: An object with S3 class "sim" from function `bdgraph.sim`. It also can be the adjacency matrix corresponding to the true graph structure.
- `bdgraph.obj`: An object with S3 class "bdgraph" from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`. It also can be an object of S3 class "select", from the function `huge.select` of R package `huge`. It also can be an adjacency matrix corresponding to an estimated graph.
- `bdgraph.obj2`, `bdgraph.obj3`: An object with S3 class "bdgraph" from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`. It also can be an object of S3 class "select", from the function `huge.select` of R package `huge`. It also can be an adjacency matrix corresponding to an estimated graph. It is for comparing two different approaches.
- `main`: A character vector giving the names for the result table.
- `vis`: Visualize the true graph and estimated graph structures.

**Value**

- **True positive**: The number of correctly estimated links.
- **True negative**: The number of true non-existing links which is correctly estimated.
- **False positive**: The number of links which they are not in the true graph, but are incorrectly estimated.
- **False negative**: The number of links which they are in the true graph, but are not estimated.
- **F1-score**: A weighted average of the "positive predictive" and "true positive rate". The F1-score value reaches its best value at 1 and worst score at 0.
- **Specificity**: The Specificity value reaches its best value at 1 and worst score at 0.
- **Sensitivity**: The Sensitivity value reaches its best value at 1 and worst score at 0.
- **MCC**: The Matthews Correlation Coefficients (MCC) value reaches its best value at 1 and worst score at 0.
geneExpression

Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>, Antonio Abbruzzo, and Ivan Vujacic

References

See Also
bdgraph, bdgraph.mpl, bdgraph.sim, plotroc

Examples
```
# Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 7, vis = TRUE )

# Running sampling algorithm based on GGMs
sample.ggm <- bdgraph( data = data.sim, method = "ggm", iter = 10000 )

# Comparing the results
compare( data.sim, sample.ggm, main = c("True", "GGM"), vis = TRUE )

# Running sampling algorithm based on GCGMs
sample.gcgm <- bdgraph( data = data.sim, method = "gcgm", iter = 10000 )

# Comparing GGM and GCGM methods
compare( data.sim, sample.ggm, sample.gcgm, main = c("True", "GGM", "GCGM"), vis = TRUE )
```

```
# End(Not run)
```

geneExpression

*Human gene expression dataset*

Description

The dataset contains human gene expression of 100 transcripts (with unique Illumina TargetID) measured on 60 unrelated individuals.
Usage

data( geneExpression )

Format

The format is a matrix with 60 rows (number of individuals) and 100 column (number of transcripts).

Source

The genotypes of those 60 unrelated individuals are available from the Sanger Institute website at ftp://ftp.sanger.ac.uk/pub/genevar

Examples

data( geneExpression )
dim( geneExpression )
head( geneExpression )

---

gnorm

Normalizing constant for G-Wishart

Description

Calculates log of the normalizing constant of G-Wishart distribution based on the Monte Carlo method, developed by Atay-Kayis and Massam (2005).

Usage

gnorm( adj.g, b = 3, D = diag( ncol( adj.g ) ), iter = 100 )

Arguments

adj.g  The adjacency matrix corresponding to the graph structure. It is an upper triangular matrix in which \( a_{ij} = 1 \) if there is a link between notes \( i \) and \( j \), otherwise \( a_{ij} = 0 \).

b  The degree of freedom for G-Wishart distribution, \( W_G(b, D) \).

D  The positive definite \((p \times p)\) "scale" matrix for G-Wishart distribution, \( W_G(b, D) \). The default is an identity matrix.

iter  The number of iteration for the Monte Carlo approximation.

Details

Log of the normalizing constant approximation using Monte Carlo method for a G-Wishart distribution, \( K \sim W_G(b, D) \), with density:

\[
Pr(K) = \frac{1}{I(b, D)} |K|^{(b-2)/2} \exp \left\{-\frac{1}{2} \text{trace}(K \times D)\right\}.
\]
Value

Log of the normalizing constant of G-Wishart distribution.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

rgwish, rwish

Examples

```r
## Not run:
# adj.g: adjacency matrix of graph with 3 nodes and 2 links
adj.g <- matrix( c( 0, 0, 1, 
                   0, 0, 1, 
                   0, 0, 0 ), 3, 3, byrow = TRUE )

gnorm( adj.g, b = 3, D = diag( 3 ) )
## End(Not run)
```

Description

Simulating undirected graph structures, including "random", "cluster", "scale-free", "hub", "circle", and "lattice".

Usage

```r
graph.sim( p = 10, graph = "random", prob = 0.2, size = NULL, class = NULL, vis = FALSE )
```
Arguments

- **p**: The number of variables (nodes).
- **graph**: The undirected graph structure with option "random", "cluster", "scale-free", "hub", "circle", and "lattice". It also could be an adjacency matrix corresponding to a graph structure (an upper triangular matrix in which $g_{ij} = 1$ if there is a link between notes $i$ and $j$, otherwise $g_{ij} = 0$).
- **prob**: If graph="random", it is the probability that a pair of nodes has a link.
- **size**: The number of links in the true graph (graph size).
- **class**: If graph="cluster", it is the number of classes.
- **vis**: Visualize the true graph structure.

Value

- **G**: The adjacency matrix corresponding to the simulated graph structure, as an object with S3 class "graph".

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

- `bdgraph.sim`, `bdgraph`, `bdgraph.mpl`

Examples

```r
# Generating a 'hub' graph
adj <- graph.sim( p = 8, graph = "scale-free" )

plot( adj )

adj
```
pgraph

Posterior probabilities of the graphs

Description

Provides the estimated posterior probabilities for the most likely graphs or a specific graph.

Usage

pgraph( bdgraph.obj, number.g = 4, adj = NULL )

Arguments

bdgraph.obj An object of S3 class “bdgraph”, from function bdgraph.
number.g The number of graphs with the highest posterior probabilities to be shown. This option is ignored if ‘adj’ is specified.
adj An adjacency matrix corresponding to a graph structure. It is an upper triangular matrix in which $a_{ij} = 1$ if there is a link between notes $i$ and $j$, otherwise $a_{ij} = 0$. It also can be an object of S3 class “sim”, from function bdgraph.sim.

Value

selected_g The graphs with the highest posterior probabilities.
prob_g A vector of the posterior probabilities of the graphs corresponding to ‘selected_g’.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

bdgraph, bdgraph.mpl
**Examples**

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 6, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim, save = TRUE )

# Estimated posterior probability of the true graph
pgraph( bdgraph.obj, adj = data.sim )

# Estimated posterior probability of first and second graphs with highest probabilities
pgraph( bdgraph.obj, number.g = 2 )

## End(Not run)
```

---

**plinks**

*Estimated posterior link probabilities*

**Description**

Provides the estimated posterior link probabilities for all possible links in the graph.

**Usage**

```r
plinks( bdgraph.obj, round = 2, burnin = NULL )
```

**Arguments**

- `bdgraph.obj` An object of S3 class "bdgraph", from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`.
- `round` A value for rounding all probabilities to the specified number of decimal places.
- `burnin` The number of burn-in iteration to scape.

**Value**

- `p_links` An upper triangular matrix which corresponds the estimated posterior probabilities for all possible links.

**Author(s)**

Reza Mohammadi `<a.mohammadi@uva.nl>` and Ernst Wit
References


See Also

`bdgraph`, `bdgraph.mpl`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'circle' graph
data.sim <- bdgraph.sim( n = 70, p = 6, graph = "circle", vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim, iter = 10000 )
plinks( bdgraph.obj, round = 2 )
## End(Not run)
```

plot.bdgraph

*Plot function for S3 class "bdgraph"

Description

Visualizes structure of the selected graphs which could be a graph with links for which their estimated posterior probabilities are greater than 0.5 or graph with the highest posterior probability.

Usage

```r
## S3 method for class 'bdgraph'
plot( x, cut = 0.5, number.g = NULL, layout = layout.circle, ... )
```
Arguments

- **x**: An object of S3 class "bdgraph", from function `bdgraph`.
- **cut**: Threshold for including the links in the selected graph based on the estimated posterior probabilities of the links; See the examples.
- **number.g**: The number of graphs with the highest probabilities. This option works for the case running function `bdgraph()` with option `save = TRUE`; See the examples.
- **layout**: The vertex placement algorithm which is according to R package `igraph`.
- **...**: System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

- `bdgraph`, `bdgraph.mpl`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 7, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim )
plot( bdgraph.obj )

bdgraph.obj <- bdgraph( data = data.sim, save = TRUE )
plot( bdgraph.obj, number.g = 4 )
plot( bdgraph.obj, cut = 0.4 )

## End(Not run)
```
plot.graph

Plot function for S3 class "graph"

Description

Visualizes structure of the graph.

Usage

```r
## S3 method for class 'graph'
plot( x, main = NULL, layout = layout.circle, ... )
```

Arguments

- `x` An object of S3 class "graph", from function `graph.sim`.
- `main` Graphical parameter (see plot).
- `layout` The vertex placement algorithm which is according to R package `igraph`.
- `...` System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

`graph.sim`, `bdgraph.sim`
Examples

```r
# Generating a 'random' graph
adj <- graph.sim( p = 10, graph = "random" )
plot( adj )
adj
```

---

**plot.sim**  
*Plot function for S3 class "sim"*

---

**Description**

Visualizes structure of the true graph.

**Usage**

```r
## S3 method for class 'sim'
plot( x, main = NULL, layout = layout.circle, ... )
```

**Arguments**

- `x`  
  An object of S3 class "sim", from function `bdgraph.sim`.
- `main`  
  Graphical parameter (see plot).
- `layout`  
  The vertex placement algorithm which is according to R package `igraph`.
- `...`  
  System reserved (no specific usage).

**Author(s)**

Reza Mohammadi `<a.mohammadi@uva.nl>` and Ernst Wit

**References**


See Also

`graphNsim`, `bdgraph.sim`

Examples

```r
# Generating synthetic multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 10, p = 15 )
plot( data.sim )
```

Description

Visualizes the cumulative occupancy fractions of all possible links in the graph. It can be used for monitoring the convergence of the sampling algorithms, BDMCMC and RJMCMC.

Usage

```r
plotcoda( bdgraph.obj, thin = NULL, control = TRUE, main = NULL, ... )
```

Arguments

- **bdgraph.obj**: An object of S3 class "bdgraph", from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`.
- **thin**: An option for getting fast result for a cumulative plot according to part of the iteration.
- **control**: Logical: if TRUE (default) and the number of nodes is greater than 15, then 100 links randomly is selected for visualization.
- **main**: Graphical parameter (see plot).
- **...**: System reserved (no specific usage).

Details

Note that a spending time for this function depends on the number of nodes.
For fast result, you can choose bigger value for the 'thin' option.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit
References


See Also

`bdgraph`, `bdgraph.mpl`, `traceplot`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'circle' graph
data.sim <- bdgraph.sim( n = 50, p = 6, graph = "circle", vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim, iter = 10000, burnin = 0, save = TRUE )
plotcoda( bdgraph.obj )
## End(Not run)
```

---

**plotroc**  

*ROC plot*

Description

Draws the ROC curve according to the true graph structure for object of S3 class "bdgraph", from function `bdgraph`.

Usage

```r
plotroc( sim.obj, bdgraph.obj, bdgraph.obj2 = NULL, bdgraph.obj3 = NULL,  
         bdgraph.obj4 = NULL, cut = 20, smooth = FALSE, label = TRUE,  
         main = "ROC Curve" )
```
Arguments

- **sim.obj**: An object of S3 class "sim", from function `bdgraph.sim`. It also can be the adjacency matrix corresponding to the true graph structure in which $a_{ij} = 1$ if there is a link between notes $i$ and $j$, otherwise $a_{ij} = 0$.

- **bdgraph.obj**: An object of S3 class "bdgraph", from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`. It also can be an object of S3 class "select", from the function `huge.select` of R package `huge`. It also can be an upper triangular matrix corresponding to the estimated posterior probabilities for all possible links.

- **bdgraph.obj2, bdgraph.obj3, bdgraph.obj4**: An object of S3 class "bdgraph", from function `bdgraph`. It also can be an object of S3 class "ssgraph", from the function `ssgraph` of R package `ssgraph`. It also can be an object of S3 class "select", from the function `huge.select` of R package `huge`. It also can be an upper triangular matrix corresponding to the estimated posterior probabilities for all possible links. It is for comparing two different approaches.

- **cut**: Number of cut points.

- **smooth**: Logical: for smoothing the ROC curve.

- **label**: Logical: for adding legend to the ROC plot.

- **main**: An overall title for the plot.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

- `bdgraph`, `bdgraph.mpl`, `compare`
Examples

## Not run:

```r
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 30, p = 6, size = 7, vis = TRUE )

# Running sampling algorithm
bdgraph.obj <- bdgraph( data = data.sim, iter = 10000 )
# Comparing the results
plotroc( data.sim, bdgraph.obj )

# To compare the results based on CGGMs approach
bdgraph.obj2 <- bdgraph( data = data.sim, method = "gcgm", iter = 10000 )
# Comparing the results
plotroc( data.sim, bdgraph.obj, bdgraph.obj2, label = FALSE )
legend( "bottomright", c( "GGMs", "GCGMs" ), lty = c( 1,2 ), col = c( "black", "red" ) )

## End(Not run)
```

---

### print.bdgraph

**Print function for S3 class "bdgraph"**

**Description**

Prints the information about the selected graph which could be a graph with links for which their estimated posterior probabilities are greater than 0.5 or graph with the highest posterior probability. It provides adjacency matrix, size and posterior probability of the selected graph.

**Usage**

```r
## S3 method for class 'bdgraph'
print( x, round = 2, ... )
```

**Arguments**

- **x**: An object of S3 class "bdgraph", from function `bdgraph`.
- **round**: A value to round the probabilities to the specified number of decimal places.
- **...**: System reserved (no specific usage).

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit
References


See Also

`bdgraph`, `bdgraph.mpl`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim )

print( bdgraph.obj )

## End(Not run)
```

Description

Prints the information about the type of data, the sample size, the graph type, the number of nodes, number of links and sparsity of the true graph.

Usage

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

Arguments

- `x` An object of S3 class "sim", from function `bdgraph.sim`.
- `...` System reserved (no specific usage).
Author(s)
Reza Mohammadi <a.mohammadi@uva.nl>, Adrian Dobra, and Ernst Wit

References

See Also
graph.sim, bdgraph.sim

Examples
```r
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 20, p = 10, vis = TRUE )
print( data.sim )
```

Description
The dataset consist of 6 discrete variables as the potential risk factors of coronary heart disease. The data collected from 1841 men employed of a car factory in Czechoslovakia (Reinis et al. 1981).

Usage
data( reinis )

Format
The format is a matrix with 1841 rows (number of individuals) and 6 column (number of variables).
rgcwish

References


Examples

```r
data( reinis )
summary( reinis )
```

---

Sampling from complex G-Wishart distribution

Description

Generates random matrices, distributed according to the complex G-Wishart distribution with parameters \( b \) and \( D \), \( CW_G(b, D) \).

Usage

```r
rgcwish( n = 1, adj.g = NULL, b = 3, D = NULL )
```

Arguments

- **n** The number of samples required.
- **adj.g** The adjacency matrix corresponding to the graph structure which can be non-decomposable or decomposable. It should be an upper triangular matrix in which \( a_{ij} = 1 \) if there is a link between notes \( i \) and \( j \), otherwise \( a_{ij} = 0 \). adj.g could be an object of class "graph", from function `graph.sim`. It also could be an object of class "sim", from function `bdgraph.sim`. It also could be an object of class "bdgraph", from functions `bdgraph.mpl` or `bdgraph`.
- **b** The degree of freedom for complex G-Wishart distribution, \( CW_G(b, D) \).
- **D** The positive definite \((p \times p)\) "scale" matrix for complex G-Wishart distribution, \( CW_G(b, D) \). The default is an identity matrix.

Details

Sampling from the complex G-Wishart distribution, \( K \sim CW_G(b, D) \), with density:

\[
Pr(K) \propto |K|^b \exp\left\{-\text{trace}(K \times D)\right\},
\]

which \( b > 2 \) is the degree of freedom and \( D \) is a symmetric positive definite matrix.
Value

A numeric array, say A, of dimension \((p \times p \times n)\), where each \(A[i, , i]\) is a positive definite matrix, a realization of the complex G-Wishart distribution, \(CW_G(b, D)\).

Author(s)

Lang Liu, Nicholas Foti, Alex Tank and Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also

rgwish, rwish

Examples

```r
## Not run:
# Generating a 'circle' graph as a non-decomposable graph
adj.g <- graph.sim( p = 5, graph = "circle" )
adj.g  # adjacency of graph with 5 nodes

sample <- rgcwish( n = 3, adj.g = adj.g, b = 3, D = diag(5) )
round( sample, 2 )

## End(Not run)
```

rgwish  

Sampling from G-Wishart distribution

Description

Generates random matrices, distributed according to the G-Wishart distribution with parameters \(b\) and \(D\), \(W_G(b, D)\) with respect to the graph structure \(G\). Note this function works for both non-decomposable and decomposable graphs.

Usage

```r
rgwish( n = 1, adj.g = NULL, b = 3, D = NULL, threshold = 1e-8 )
```
Arguments

n The number of samples required.
adj.g The adjacency matrix corresponding to the graph structure which can be non-decomposable or decomposable. It should be an upper triangular matrix in which \( a_{ij} = 1 \) if there is a link between notes \( i \) and \( j \), otherwise \( a_{ij} = 0 \). adj.g could be an object of class "graph", from function graph.sim. It also could be an object of class "sim", from function bdgraph.sim. It also could be an object of class "bdgraph", from functions bdgraph.mp1 or bdgraph.
b The degree of freedom for G-Wishart distribution, \( W_G(b, D) \).
D The positive definite \((p \times p)\) "scale" matrix for G-Wishart distribution, \( W_G(b, D) \). The default is an identity matrix.
threshold The threshold value for the convergence of sampling algorithm from G-Wishart.

Details

Sampling from G-Wishart distribution, \( K \sim W_G(b, D) \), with density:

\[
Pr(K) \propto |K|^{(b-2)/2} \exp \left\{ -\frac{1}{2} \text{trace}(K \times D) \right\},
\]

which \( b > 2 \) is the degree of freedom and \( D \) is a symmetric positive definite matrix.

Value

A numeric array, say \( A \), of dimension \((p \times p \times n)\), where each \( A[, , i] \) is a positive definite matrix, a realization of the G-Wishart distribution, \( W_G(b, D) \). Note, for the case \( n = 1 \), the output is a matrix.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References


See Also
gnorm, rwish, rgcwish
Examples

# Generating a 'circle' graph as a non-decomposable graph
adj.g <- graph.sim( p = 5, graph = "circle" )
adj.g # adjacency of graph with 5 nodes

sample <- rgwish( n = 1, adj.g = adj.g, b = 3, D = diag( 5 ) )
round( sample, 2 )

sample <- rgwish( n = 5, adj.g = adj.g )
round( sample, 2 )


rmvnorm

Generate data from the multivariate Normal distribution

Description

Random generation function from the multivariate Normal distribution with mean equal to \( \text{mean} \)
and covariance matrix \( \text{sigma} \).

Usage

\[
\text{rmvnorm}( n = 10, \text{mean} = \text{rep}( 0, \text{length} = \text{ncol}( \text{sigma} ) ), \\
\text{sigma} = \text{diag}( \text{length}( \text{mean} ) ) )
\]

Arguments

\n
- **n** Number of observations.
- **mean** Mean vector, default is \( \text{rep}(0, \text{length} = \text{ncol}(\text{sigma})) \).
- **sigma** positive definite covariance matrix, default is \( \text{diag}(\text{length}(\text{mean})) \).

Value

A numeric matrix with rows equal to \( n \) and columns equal to \( \text{length}(\text{mean}) \).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

See Also

bdgraph.sim, rwish, rgwish
Examples

```r
mean <- c(5, 20)
sigma <- matrix(c(4, 2, 2, 5), 2, 2) # covariance matrix

sample <- rmvnorm(n = 500, mean = mean, sigma = sigma)
plot(sample)
```

**rwish**

*Sampling from Wishart distribution*

**Description**

Generates random matrices, distributed according to the Wishart distribution with parameters $b$ and $D$, $W(b, D)$.

**Usage**

```r
rwish(n = 1, p = 2, b = 3, D = diag(p))
```

**Arguments**

- `n`: The number of samples required.
- `p`: The number of variables (nodes).
- `b`: The degree of freedom for Wishart distribution, $W(b, D)$.
- `D`: The positive definite ($p \times p$) "scale" matrix for Wishart distribution, $W(b, D)$. The default is an identity matrix.

**Details**

Sampling from Wishart distribution, $K \sim W(b, D)$, with density:

$$Pr(K) \propto |K|^{(b-2)/2} \exp\left\{-\frac{1}{2}\text{trace}(K \times D)\right\},$$

which $b > 2$ is the degree of freedom and $D$ is a symmetric positive definite matrix.

**Value**

A numeric array, say $A$, of dimension $(p \times p \times n)$, where each $A[i, , i]$ is a positive definite matrix, a realization of the Wishart distribution $W(b, D)$. Note, for the case $n = 1$, the output is a matrix.

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl>
References


See Also
gnorm, rgwish, rgcwish

Examples

```r
sample <- rwish( n = 3, p = 5, b = 3, D = diag( 5 ) )
round( sample, 2 )
```

---

**select**

*Graph selection*

**Description**

Provides the selected graph which, based on input, could be a graph with links for which their estimated posterior probabilities are greater than 0.5 (default) or a graph with the highest posterior probability; see examples.

**Usage**

```r
select( bdgraph.obj, cut = NULL, vis = FALSE )
```

**Arguments**

- `bdgraph.obj`: An object of S3 class "bdgraph", from function `bdgraph`. It also can be an object of S3 class "ssgraph", from function `ssgraph` from R package `ssgraph`. It also can be an \((p \times p)\) matrix in which each element of the matrix response to the weight of the links.
- `cut`: Threshold for including the links in the selected graph based on the estimated posterior probabilities of the links; see the examples.
- `vis`: Visualize the selected graph structure.

**Value**

- \(G\): An adjacency matrix corresponding to the selected graph.
Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

bdgraph, bdgraph.mpl

Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 7, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim )

select( bdgraph.obj )

bdgraph.obj <- bdgraph( data = data.sim, save = TRUE )

select( bdgraph.obj )

select( bdgraph.obj, cut = 0.5, vis = TRUE )

## End(Not run)
```

summary.bdgraph

*Summary function for S3 class "bdgraph"*

Description

Provides a summary of the results for function *bdgraph.*
Usage

```r
## S3 method for class 'bdgraph'
summary(object, round = 2, vis = TRUE, ...)
```

Arguments

- `object`: An object of S3 class "bdgraph", from function `bdgraph`.
- `round`: A value for rounding all probabilities to the specified number of decimal places.
- `vis`: Visualize the results.
- `...`: System reserved (no specific usage).

Value

- `selected_g`: The adjacency matrix corresponding to the selected graph which has the highest posterior probability.
- `p_links`: An upper triangular matrix corresponding to the posterior probabilities of all possible links.
- `K_hat`: The estimated precision matrix.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

`bdgraph`, `bdgraph.mpl`

Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 7, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim )
```
surveyData

Labor force survey data

Description

The survey dataset concerns 1002 males in the U.S labor force, described by Hoff (2007). The seven observed variables which have been measured on various scales are as follow: the income (income), degree (degree), the number of children (children), parents income (pincome), parents degree (pdegree), number of parents children (pchildren), and age (age).

Usage

data( surveyData )

Format

The format is a matrix with 1002 rows (number of individuals) and 7 column (number of variables).

References


Examples

```r
data( surveyData )
summary( surveyData )
```
traceplot

Description

Trace plot for graph size for the objects of S3 class "bdgraph", from function bdgraph. It is a tool for monitoring the convergence of the sampling algorithms, BDMCMC and RJMCMC.

Usage

traceplot ( bdgraph.obj, acf = FALSE, pacf = FALSE, main = NULL, ... )

Arguments

bdgraph.obj An object of S3 class "bdgraph", from function bdgraph. It also can be an object of S3 class "ssgraph", from the function ssgraph of R package ssgraph.
acf Visualize the autocorrelation functions for graph size.
pacf Visualize the partial autocorrelations for graph size.
main Graphical parameter (see plot).
... System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl> and Ernst Wit

References


See Also

plotcoda, bdgraph, bdgraph.mpl
Examples

```r
## Not run:
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 50, p = 6, size = 7, vis = TRUE )

bdgraph.obj <- bdgraph( data = data.sim, iter = 10000, burnin = 0, save = TRUE )

traceplot( bdgraph.obj )

traceplot( bdgraph.obj, acf = TRUE, pacf = TRUE )

## End(Not run)
```

### transfer

---

#### transfer for discrete data

**Description**

Transfers discrete data, by counting the duplicated rows.

**Usage**

```r
transfer( r_data )
```

**Arguments**

- `r_data` An \((n \times p)\) matrix or a data.frame corresponding to the data \((n\) is the sample size and \(p\) is the number of variables).

**Value**

- `data` An \((n \times p + 1)\) matrix of transferred data, in which the last column is the frequency of duplicated rows.

**Author(s)**

Reza Mohammadi <a.mohammadi@uva.nl> and Adrian Dobra

**References**


**See Also**

`bdgraph.mpl`, `bdgraph.sim`
Examples

# Generating multivariate binary data from a 'random' graph
data.sim <- bdgraph.sim( n = 12, p = 4, size = 4, type = "discrete", cut = 2 )
r_data <- data.sim $ data
r_data

# Transfer the data
transfer( r_data )
Index

++Topic **Package, BDgraph, Bayesian Structure Learning, Gaussian Graphical Models, Gaussian Copula Graphical Models, BDMCMC, G-Wishart, Marginal Pseudo-likelihood**

BDgraph-package, 2
++Topic **datasets**
churn, 16
geneExpression, 19
reinis, 34
surveyData, 43
calc_joint_dist (BDgraph-package), 2
compute_measures (BDgraph-package), 2
compute_tp_fp (BDgraph-package), 2
detect_cores (BDgraph-package), 2
generate_clique_factors (BDgraph-package), 2
global_hc (BDgraph-package), 2
global_hc_binary (BDgraph-package), 2
hill_climb_mpl (BDgraph-package), 2
hill_climb_mpl_binary (BDgraph-package), 2
local_mb_hc (BDgraph-package), 2
local_mb_hc_binary (BDgraph-package), 2
log_mpl_binary (BDgraph-package), 2
log_mpl_discrete (BDgraph-package), 2
sample_ug (BDgraph-package), 2

BDgraph, 5, 8
BDgraph (BDgraph-package), 2
bdgraph, 4, 4, 9, 10, 13, 15, 18, 19, 22–26, 29–33, 35, 37, 40–42, 44
BDgraph-package, 2
bdgraph.mpl, 4, 6, 7, 10, 13, 15, 19, 22, 23, 25, 26, 30, 31, 33, 35, 37, 41, 42, 44, 45

bdgraph.npn, 10
bdgraph.sim, 4–6, 8–10, 11, 15, 18, 19, 22, 23, 27–29, 31, 33–35, 37, 38, 45
bdgraph.ts, 13
churn, 16
compare, 4, 6, 9, 15, 18, 31
geneExpression, 19
gnorm, 20, 37, 40
graph.sim, 13, 21, 27, 29, 34, 35, 37
huge, 18, 31
huge.select, 18, 31

igraph, 26–28
plinks, 24
plot.bdgraph, 25
plot.graph, 27
plot.coda, 29, 44
plotroc, 19, 30
print.bdgraph, 32
print.sim, 33
reinis, 34
rgcwish, 35, 37, 40
rgwish, 4, 21, 36, 38, 40
rmvnorm, 38
rwish, 21, 36–38, 39
select, 40
ssgraph, 5, 8, 18, 24, 29, 31, 40, 44
summary.bdgraph, 6, 9, 15, 41
surveyData, 43

traceplot, 30, 44
transfer, 45