Package ‘ssgraph’

May 9, 2022

Title Bayesian Graph Structure Learning using Spike-and-Slab Priors

Version 1.13

Description Bayesian estimation for undirected graphical models using spike-and-slab priors. The package handles continuous, discrete, and mixed data.

URL https://www.uva.nl/profile/a.mohammadi

Depends BDgraph (>= 2.58)

Suggests skimr, knitr, rmarkdown

VignetteBuilder knitr

License GPL (>= 2)

Repository CRAN

NeedsCompilation yes

Author Reza Mohammadi [aut, cre] (<https://orcid.org/0000-0001-9538-0648>)

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

Encoding UTF-8

Date/Publication 2022-05-09 08:50:06 UTC

R topics documented:

ssgraph-package *Bayesian Graphical Estimation using Spike-and-Slab Priors*

Description

The R package **ssgraph** is for Bayesian estimation of graphical models by using spike-and-slab priors. The package is implemented the recent improvements in the Bayesian graphical models' literature, including Wang (2015). To speed up the computations, the computationally intensive tasks of the package are implemented in C++ in parallel using **OpenMP**.

How to cite this package

To cite **ssgraph** in publications use:

```
Mohammadi R. (2020). ssgraph: Bayesian Graphical Estimation using
Spike-and-Slab Priors, R package version 1.11,
https://cran.r-project.org/package=ssgraph
```

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References

Wang, H. (2015). Scaling it up: Stochastic search structure learning in graphical models, *Bayesian Analysis*, 10(2):351-377

George, E. I. and McCulloch, R. E. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881-889

Griffin, J. E. and Brown, P. J. (2010) Inference with normal-gamma prior distributions in regression problems. *Bayesian Analysis*, 5(1):171-188

Mohammadi, A. et al (2017). Bayesian modelling of Dupuytren disease by using Gaussian copula graphical models, *Journal of the Royal Statistical Society: Series C*, 66(3):629-645

Mohammadi, R. and Wit, E. C. (2019). **BDgraph**: An R Package for Bayesian Structure Learning in Graphical Models, *Journal of Statistical Software*, 89(3):1-30

Mohammadi, A. and Wit, E. C. (2015). Bayesian Structure Learning in Sparse Gaussian Graphical Models, *Bayesian Analysis*, 10(1):109-138

Examples

```
## Not run:

library( ssgraph )

# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 100, p = 8, size = 10, vis = TRUE )
```

```
# Running algorithm based on GGMs
ssgraph.obj <- ssgraph( data = data.sim, iter = 5000, save = TRUE )

summary( ssgraph.obj )

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

## End(Not run)
```

`plot.ssgraph` *Plot function for* `S3` *class* `"ssgraph"`

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

```
## S3 method for class 'ssgraph'
plot( x, cut = 0.5, ... )
```

Arguments

`x`	An object of `S3` class `"ssgraph"`, from function `ssgraph`.
`cut`	Threshold for including the links in the selected graph based on the estimated posterior probabilities of the links; See the examples.
`...`	System reserved (no specific usage).

Author(s)

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

References

Mohammadi, R. and Wit, E. C. (2019). **BDgraph**: An `R` Package for Bayesian Structure Learning in Graphical Models, *Journal of Statistical Software*, 89(3):1-30

Mohammadi, A. and Wit, E. C. (2015). Bayesian Structure Learning in Sparse Gaussian Graphical Models, *Bayesian Analysis*, 10(1):109-138

Mohammadi, A. et al (2017). Bayesian modelling of Dupuytren disease by using Gaussian copula graphical models, *Journal of the Royal Statistical Society: Series C*, 66(3):629-645

See Also

`ssgraph`

Examples

```
## Not run:
# Generating multivariate normal data from a 'scale-free' graph
data.sim <- bdgraph.sim( n = 60, p = 7, graph = "scale-free", vis = TRUE )

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

plot( ssgraph.obj )

plot( ssgraph.obj, cut = 0.3 )

## End(Not run)
```

---

print.ssgraph *Print function for* S3 *class* "ssgraph"

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

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

Arguments

x	An object of S3 class "ssgraph", from function ssgraph.
...	System reserved (no specific usage).

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References

Mohammadi, R. and Wit, E. C. (2019). **BDgraph**: An R Package for Bayesian Structure Learning in Graphical Models, *Journal of Statistical Software*, 89(3):1-30

Mohammadi, A. and Wit, E. C. (2015). Bayesian Structure Learning in Sparse Gaussian Graphical Models, *Bayesian Analysis*, 10(1):109-138

Mohammadi, A. et al (2017). Bayesian modelling of Dupuytren disease by using Gaussian copula graphical models, *Journal of the Royal Statistical Society: Series C*, 66(3):629-645

See Also

ssgraph

Examples

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

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

print( ssgraph.obj )

## End(Not run)
```

---

ssgraph *Algorithm for graphical models using spike-and-slab priors*

Description

This function has a sampling algorithm for Bayesian model determination in undirected graphical models, based on spike-and-slab priors.

Usage

```
ssgraph( data, n = NULL, method = "ggm", not.cont = NULL,
         iter = 5000, burnin = iter / 2, var1 = 4e-04,
         var2 = 1, lambda = 1, g.prior = 0.5, g.start = "full",
         sig.start = NULL, save = FALSE, print = 1000, cores = NULL )
```

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 the `bdgraph.sim` function of `R` package `BDgraph`. 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 indicates not continuous variables.
`iter`	The number of iteration for the sampling algorithm.
`burnin`	The number of burn-in iteration for the sampling algorithm.
`var1`	Value for the variance of the the prior of precision matrix for the places that there is no link in the graph.
`var2`	Value for the variance of the the prior of precision matrix for the places that there is link in the graph.

lambda	Value for the parameter of diagonal element of the prior of precision matrix.
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 "ssgraph" of package ssgraph or "bdgraph" of package BDgraph; this option can be used to run the sampling algorithm from the last objects of previous run (see examples).
sig.start	Corresponds to a starting point of the covariance matrix. It must be positive definite matrix.
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 default is to use 2 CPU cores of the computer. The case cores="all" means all CPU cores to use for parallel execution.

Value

An object with S3 class "ssgraph" 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 also 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 counted numbers 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 MCMC sampling 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 MCMC sampling algorithm.

Author(s)

Reza Mohammadi <a.mohammadi@uva.nl>

References

Wang, H. (2015). Scaling it up: Stochastic search structure learning in graphical models, *Bayesian Analysis*, 10(2):351-377

George, E. I. and McCulloch, R. E. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881-889

Griffin, J. E. and Brown, P. J. (2010) Inference with normal-gamma prior distributions in regression problems. *Bayesian Analysis*, 5(1):171-188

Mohammadi, A. et al (2017). Bayesian modelling of Dupuytren disease by using Gaussian copula graphical models, *Journal of the Royal Statistical Society: Series C*, 66(3):629-645

Mohammadi, R. and Wit, E. C. (2019). **BDgraph**: An R Package for Bayesian Structure Learning in Graphical Models, *Journal of Statistical Software*, 89(3):1-30

Mohammadi, A. and Wit, E. C. (2015). Bayesian Structure Learning in Sparse Gaussian Graphical Models, *Bayesian Analysis*, 10(1):109-138

See Also

bdgraph, bdgraph.mpl, summary.ssgraph, compare

Examples

```
# Generating multivariate normal data from a 'random' graph
data.sim <- bdgraph.sim( n = 80, p = 7, prob = 0.5, vis = TRUE )

# Running algorithm based on GGMs
ssgraph.obj <- ssgraph( data = data.sim, iter = 1000 )

summary( ssgraph.obj )

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

## Not run:

# Running algorithm with starting points from previous run
ssgraph.obj2 <- ssgraph( data = data.sim, iter=5000, g.start = ssgraph.obj )

compare( data.sim, ssgraph.obj, ssgraph.obj2, vis = TRUE,
         main = c( "Target", "Frist run", "Second run" ) )

## End(Not run)
```

---

summary.ssgraph   *Summary function for* S3 *class* "ssgraph"

Description

Provides a summary of the results for function ssgraph.

Usage

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

Arguments

`object`	An object of S3 class "ssgraph", from function ssgraph.
`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>

References

Mohammadi, R. and Wit, E. C. (2019). **BDgraph**: An R Package for Bayesian Structure Learning in Graphical Models, *Journal of Statistical Software*, 89(3):1-30

Mohammadi, A. and Wit, E. C. (2015). Bayesian Structure Learning in Sparse Gaussian Graphical Models, *Bayesian Analysis*, 10(1):109-138

Mohammadi, A. et al (2017). Bayesian modelling of Dupuytren disease by using Gaussian copula graphical models, *Journal of the Royal Statistical Society: Series C*, 66(3):629-645

See Also

ssgraph

Examples

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

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

summary( ssgraph.obj )

## End(Not run)
```

Index