Package 'bspcov'

May 8, 2026

Type Package

Title Bayesian Sparse Estimation of a Covariance Matrix

Version 1.0.3

Date 2025-08-14

Maintainer Kyeongwon Lee <kwlee1718@gmail.com>

Author Kwangmin Lee [aut],
Kyeongwon Lee [aut, cre],
Kyoungjae Lee [aut],
Seongil Jo [aut],
Jaeyong Lee [ctb]

Depends R (>= 4.2)

Description Bayesian estimations of a covariance matrix for multivariate normal data. Assumes that the covariance matrix is sparse or band matrix and positive-definite. Methods implemented include the beta-mixture shrinkage prior (Lee et al. (2022) <doi:10.1016/j.jmva.2022.105067>), screened beta-mixture prior (Lee et al. (2024) <doi:10.1214/24-BA1495>), and post-processed posteriors for banded and sparse covariances (Lee et al. (2023) <doi:10.1214/22-BA1333>; Lee and Lee (2023) <doi:10.1016/j.jeconom.2023.105475>). This software has been developed using funding supported by Basic Science Research Program through the National Research Foundation of Korea ('NRF') funded by the Ministry of Education ('RS-2023-00211979', 'NRF-2022R1A5A7033499', 'NRF-2020R1A4A1018207' and 'NRF-2020R1C1C1A01013338').

Imports GIGrvg, coda, progress, BayesFactor, MASS, mvnfast, matrixcalc, matrixStats, purrr, dplyr, RSpectra, Matrix, plyr, CholWishart, magrittr, future, furrr, ks, ggplot2, ggmcmc, caret, FinCovRegularization, mvtnorm, stats, patchwork, reshape2, future.apply

Suggests hdbinseg, POET, tidyquant, tidyr, timetk, quantmod

License GPL-2

LazyLoad yes

URL https://github.com/statjs/bspcov

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

LazyData true

LazyDataCompression xz

Repository CRAN

Date/Publication 2025-08-18 19:20:30 UTC

Contents

---

bandPPP *Bayesian Estimation of a Banded Covariance Matrix*

Description

Provides a post-processed posterior for Bayesian inference of a banded covariance matrix.

Usage

```
bandPPP(X, k, eps, prior = list(), nsample = 2000)
```

Arguments

X	a n $\times$ p data matrix with column mean zero.
k	a scalar value (natural number) specifying the bandwidth of covariance matrix.
eps	a small positive number decreasing to 0 with default value $(log(k))^2 * (k + log(p))/n$.
prior	a list giving the prior information. The list includes the following parameters (with default values in parentheses): A (I) giving the positive definite scale matrix for the inverse-Wishart prior, nu (p + k) giving the degree of freedom of the inverse-Wishar prior.
nsample	a scalar value giving the number of the post-processed posterior samples.

Details

Lee, Lee, and Lee (2023+) proposed a two-step procedure generating samples from the post-processed posterior for Bayesian inference of a banded covariance matrix:

- Initial posterior computing step: Generate random samples from the following initial posterior obtained by using the inverse-Wishart prior $IW_p(B_0, \nu_0)$

$$\Sigma \mid X_1, \ldots, X_n \sim IW_p(B_0 + nS_n, \nu_0 + n),$$

  where $S_n = n^{-1} \sum_{i=1}^n X_i X_i^\top$.

- Post-processing step: Post-process the samples generated from the initial samples

$$\Sigma_{(i)} := \begin{cases} B_k(\Sigma^{(i)}) + \left[\epsilon_n - \lambda_{\min}\{B_k(\Sigma^{(i)})\}\right] I_p, & \text{if } \lambda_{\min}\{B_k(\Sigma^{(i)})\} < \epsilon_n, \\ B_k(\Sigma^{(i)}), & \text{otherwise ,} \end{cases}$$

where $\Sigma^{(1)}, \ldots, \Sigma^{(N)}$ are the initial posterior samples, ϵ_n is a small positive number decreasing to 0 as $n \to \infty$, and $B_k(B)$ denotes the k-band operation given as

$$B_k(B) = \{b_{ij} I(|i - j| \leq k)\} \text{ for any } B = (b_{ij}) \in R^{p \times p}.$$

For more details, see Lee, Lee and Lee (2023+).

Value

Sigma	a nsample $\times$ p(p+1)/2 matrix including lower triangular elements of covariance matrix.
p	dimension of covariance matrix.

Author(s)

Kwangmin Lee

References

Lee, K., Lee, K., and Lee, J. (2023+), "Post-processes posteriors for banded covariances", *Bayesian Analysis*, DOI: 10.1214/22-BA1333.

See Also

cv.bandPPP estimate

Examples

```
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::bandPPP(X,2,0.01,nsample=100)
```

---

bmspcov *Bayesian Sparse Covariance Estimation*

Description

Provides a Bayesian sparse and positive definite estimate of a covariance matrix via the beta-mixture shrinkage prior.

Usage

```
bmspcov(
  X,
  Sigma,
  prior = list(),
  nsample = list(),
  nchain = 1,
  seed = NULL,
  do.parallel = FALSE,
  show_progress = TRUE
)
```

Arguments

X	a n $\times$ p data matrix with column mean zero.
Sigma	an initial guess for Sigma.
prior	a list giving the prior information. The list includes the following parameters (with default values in parentheses): a (1/2) and b (1/2) giving the shape parameters for beta distribution, lambda (1) giving the hyperparameter for the diagonal elements, tau1sq (10000/(n*p^4)) giving the hyperparameter for the shrinkage prior of covariance.
nsample	a list giving the MCMC parameters. The list includes the following integers (with default values in parentheses): burnin (1000) giving the number of MCMC samples in transition period, nmc (1000) giving the number of MCMC samples for analysis.

nchain	number of MCMC chains to run. Default is 1.
seed	random seed for reproducibility. If NULL, no seed is set. For multiple chains, each chain gets seed + i - 1.
do.parallel	logical indicating whether to run multiple chains in parallel using future.apply. Default is FALSE. When TRUE, automatically sets up a multisession plan with nchain workers if no parallel plan is already configured.
show_progress	logical indicating whether to show progress bars during MCMC sampling. Default is TRUE. Automatically set to FALSE for individual chains when running in parallel.

Details

Lee, Jo and Lee (2022) proposed the beta-mixture shrinkage prior for estimating a sparse and positive definite covariance matrix. The beta-mixture shrinkage prior for $\Sigma = (\sigma_{jk})$ is defined as

$$\pi(\Sigma) = \frac{\pi^u(\Sigma) I(\Sigma \in C_p)}{\pi^u(\Sigma \in C_p)},\ C_p = \{ \text{ all } p \times p \text{ positive definite matrices } \},$$

where $\pi^u(\cdot)$ is the unconstrained prior given by

$$\pi^u(\sigma_{jk} \mid \rho_{jk}) = N\left(\sigma_{jk} \mid 0, \frac{\rho_{jk}}{1-\rho_{jk}}\tau_1^2\right)$$

$$\pi^u(\rho_{jk}) = Beta(\rho_{jk} \mid a, b),\ \rho_{jk} = 1 - 1/(1+\phi_{jk})$$

$$\pi^u(\sigma_{jj}) = Exp(\sigma_{jj} \mid \lambda).$$

For more details, see Lee, Jo and Lee (2022).

Value

Sigma	a nmc $\times$ p(p+1)/2 matrix including lower triangular elements of covariance matrix. For multiple chains, this becomes a list of matrices.
Phi	a nmc $\times$ p(p+1)/2 matrix including shrinkage parameters corresponding to lower triangular elements of covariance matrix. For multiple chains, this becomes a list of matrices.
p	dimension of covariance matrix.
mcmctime	elapsed time for MCMC sampling. For multiple chains, this becomes a list.
nchain	number of chains used.
burnin	number of burn-in samples discarded.
nmc	number of MCMC samples retained for analysis.

Author(s)

Kyoungjae Lee, Seongil Jo, and Kyeongwon Lee

References

Lee, K., Jo, S., and Lee, J. (2022), "The beta-mixture shrinkage prior for sparse covariances with near-minimax posterior convergence rate", *Journal of Multivariate Analysis*, 192, 105067.

See Also

sbmspcov

Examples

```
set.seed(1)
n <- 20
p <- 5

# generate a sparse covariance matrix:
True.Sigma <- matrix(0, nrow = p, ncol = p)
diag(True.Sigma) <- 1
Values <- -runif(n = p*(p-1)/2, min = 0.2, max = 0.8)
nonzeroIND <- which(rbinom(n=p*(p-1)/2,1,prob=1/p)==1)
zeroIND = (1:(p*(p-1)/2))[-nonzeroIND]
Values[zeroIND] <- 0
True.Sigma[lower.tri(True.Sigma)] <- Values
True.Sigma[upper.tri(True.Sigma)] <- t(True.Sigma)[upper.tri(True.Sigma)]
if(min(eigen(True.Sigma)$values) <= 0){
  delta <- -min(eigen(True.Sigma)$values) + 1.0e-5
  True.Sigma <- True.Sigma + delta*diag(p)
}

# generate a data
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = True.Sigma)

# compute sparse, positive covariance estimator:
fout <- bspcov::bmspcov(X = X, Sigma = diag(diag(cov(X))))
post.est.m <- bspcov::estimate(fout)
sqrt(mean((post.est.m - True.Sigma)^2))
sqrt(mean((cov(X) - True.Sigma)^2))

# Multiple chains example with parallel processing:
# fout_multi <- bspcov::bmspcov(X = X, Sigma = diag(diag(cov(X))),
#                               nchain = 4, do.parallel = TRUE)
# post.est.multi <- bspcov::estimate(fout_multi)
# When do.parallel = TRUE, the function automatically sets up
# a multisession plan with nchain workers for parallel execution.
```

---

colon *colon dataset*

Description

The colon cancer dataset, which includes gene expression values from 22 colon tumor tissues and 40 non-tumor tissues.

Format

data.frame

Source

http://genomics-pubs.princeton.edu/oncology/affydata/.

Examples

```
data("colon")
head(colon)
```

`cv.bandPPP` *CV for Bayesian Estimation of a Banded Covariance Matrix*

Description

Performs leave-one-out cross-validation (LOOCV) to calculate the predictive log-likelihood for a post-processed posterior of a banded covariance matrix and selects the optimal parameters.

Usage

```
cv.bandPPP(X, kvec, epsvec, prior = list(), nsample = 2000, ncores = 2)
```

Arguments

`X`	a n $\times$ p data matrix with column mean zero.
`kvec`	a vector of natural numbers specifying the bandwidth of covariance matrix.
`epsvec`	a vector of small positive numbers decreasing to 0.
`prior`	a list giving the prior information. The list includes the following parameters (with default values in parentheses): `A (I)` giving the positive definite scale matrix for the inverse-Wishart prior, `nu (p + k)` giving the degree of freedom of the inverse-Wishar prior.
`nsample`	a scalar value giving the number of the post-processed posterior samples.
`ncores`	a scalar value giving the number of CPU cores.

Details

The predictive log-likelihood for each k and ϵ_n is estimated as follows:

$$\sum_{i=1}^{n} \log S^{-1} \sum_{s=1}^{S} p(X_i \mid B_k^{(\epsilon_n)}(\Sigma_{i,s})),$$

where X_i is the ith observation, $\Sigma_{i,s}$ is the sth posterior sample based on $(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$, and $B_k^{(\epsilon_n)}$ represents the banding post-processing function. For more details, see (3) in Lee, Lee and Lee (2023+).

Value

elpd — a M $\times$ 3 dataframe having the expected log predictive density (ELPD) for each k and eps, where M = length(kvec) * length(epsvec).

Author(s)

Kwangmin Lee

References

Lee, K., Lee, K., and Lee, J. (2023+), "Post-processes posteriors for banded covariances", *Bayesian Analysis*, DOI: 10.1214/22-BA1333.

Gelman, A., Hwang, J., and Vehtari, A. (2014). "Understanding predictive information criteria for Bayesian models." *Statistics and computing*, 24(6), 997-1016.

See Also

bandPPP

Examples

```
Sigma0 <- diag(1,50)
X <- mvtnorm::rmvnorm(25,sigma = Sigma0)
kvec <- 1:2
epsvec <- c(0.01,0.05)
res <- bspcov::cv.bandPPP(X,kvec,epsvec,nsample=10,ncores=4)
plot(res)
```

---

cv.thresPPP *CV for Bayesian Estimation of a Sparse Covariance Matrix*

Description

Performs cross-validation to estimate spectral norm error for a post-processed posterior of a sparse covariance matrix.

Usage

```
cv.thresPPP(
  X,
  thresvec,
  epsvec,
  prior = NULL,
  thresfun = "hard",
  nsample = 2000,
```

```
  ncores = 2
)
```

Arguments

X	a n $\times$ p data matrix with column mean zero.
thresvec	a vector of real numbers specifying the parameter of the threshold function.
epsvec	a vector of small positive numbers decreasing to 0.
prior	a list giving the prior information. The list includes the following parameters (with default values in parentheses): A (I) giving the positive definite scale matrix for the inverse-Wishart prior, nu (p + k) giving the degree of freedom of the inverse-Wishar prior.
thresfun	a string to specify the type of threshold function. fun ('hard') giving the thresholding function ('hard' or 'soft') for the thresholding PPP procedure.
nsample	a scalar value giving the number of the post-processed posterior samples.
ncores	a scalar value giving the number of CPU cores.

Details

Given a set of train data and validation data, the spectral norm error for each γ and ϵ_n is estimated as follows:

$$||\hat{\Sigma}(\gamma, \epsilon_n)^{(train)} - S^{(val)}||_2$$

where $\hat{\Sigma}(\gamma, \epsilon_n)^{(train)}$ is the estimate for the covariance based on the train data and $S^{(val)}$ is the sample covariance matrix based on the validation data. The spectral norm error is estimated by the 10-fold cross-validation. For more details, see the first paragraph on page 9 in Lee and Lee (2023).

Value

CVdf	a M $\times$ 3 dataframe having the estimated spectral norm error for each thres and eps, where M = length(thresvec) * length(epsvec)

Author(s)

Kwangmin Lee

References

Lee, K. and Lee, J. (2023), "Post-processes posteriors for sparse covariances", *Journal of Econometrics*, 236(3), 105475.

See Also

thresPPP

Examples

```
Sigma0 <- diag(1,50)
X <- mvtnorm::rmvnorm(25,sigma = Sigma0)
thresvec <- c(0.01,0.1)
epsvec <- c(0.01,0.1)
res <- bspcov::cv.thresPPP(X,thresvec,epsvec,nsample=100)
plot(res)
```

`estimate` *Point-estimate of posterior distribution*

Description

Compute the point estimate (mean) to describe posterior distribution. For multiple chains, combines all chains to compute a more robust estimate.

Usage

```
estimate(object, ...)

## S3 method for class 'bspcov'
estimate(object, ...)
```

Arguments

`object`	an object from **bandPPP**, **bmspcov**, **sbmspcov**, and **thresPPP**.
`...`	additional arguments for estimate.

Value

`Sigma`	the point estimate (mean) of covariance matrix. For multiple chains, uses combined samples from all chains.

Author(s)

Seongil Jo and Kyeongwon Lee

See Also

plot.postmean.bspcov

Examples

```
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::bandPPP(X,2,0.01,nsample=100)
est <- bspcov::estimate(res)
```

`plot.bspcov` *Plot Diagnostics of Posterior Samples and Cross-Validation*

Description

Provides a trace plot of posterior samples and a plot of a learning curve for cross-validation. For multiple chains, each chain is displayed in a different color for convergence assessment.

Usage

```
## S3 method for class 'bspcov'
plot(x, ..., cols, rows)
```

Arguments

`x`	an object from **bmspcov**, **sbmspcov**, **cv.bandPPP**, and **cv.thresPPP**.
`...`	additional arguments for ggplot2.
`cols`	a scalar or a vector including specific column indices for the trace plot.
`rows`	a scalar or a vector including specific row indices greater than or equal to columns indices for the trace plot.

Value

`plot`	a plot for diagnostics of posterior samples **x**. For multiple chains, returns colored trace plots with each chain in a different color.

Author(s)

Seongil Jo and Kyeongwon Lee

Examples

```
set.seed(1)
n <- 100
p <- 20

# generate a sparse covariance matrix:
```

```
True.Sigma <- matrix(0, nrow = p, ncol = p)
diag(True.Sigma) <- 1
Values <- -runif(n = p*(p-1)/2, min = 0.2, max = 0.8)
nonzeroIND <- which(rbinom(n=p*(p-1)/2,1,prob=1/p)==1)
zeroIND = (1:(p*(p-1)/2))[-nonzeroIND]
Values[zeroIND] <- 0
True.Sigma[lower.tri(True.Sigma)] <- Values
True.Sigma[upper.tri(True.Sigma)] <- t(True.Sigma)[upper.tri(True.Sigma)]
if(min(eigen(True.Sigma)$values) <= 0){
  delta <- -min(eigen(True.Sigma)$values) + 1.0e-5
  True.Sigma <- True.Sigma + delta*diag(p)
}

# generate a data
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = True.Sigma)

# compute sparse, positive covariance estimator:
fout <- bspcov::sbmspcov(X = X, Sigma = diag(diag(cov(X))))
plot(fout, cols = c(1, 3, 4), rows = c(1, 3, 4))
plot(fout, cols = 1, rows = 1:3)

# Cross-Validation for Banded Structure
Sigma0 <- diag(1,50)
X <- mvtnorm::rmvnorm(25,sigma = Sigma0)
kvec <- 1:2
epsvec <- c(0.01,0.05)
res <- bspcov::cv.bandPPP(X,kvec,epsvec,nsample=10,ncores=4)
plot(res)
```

`plot.postmean.bspcov` *Plot method for postmean.bspcov objects*

Description

Create heatmap visualization for posterior mean estimate of sparse covariance matrix. Provides flexible visualization options with customizable aesthetics and labeling.

Usage

```
## S3 method for class 'postmean.bspcov'
plot(
  x,
  title = NULL,
  color_limits = NULL,
  color_low = "black",
  color_high = "white",
  base_size = 14,
```

```
  legend_position = "bottom",
  x_label = "Variable",
  y_label = "Variable",
  show_values = FALSE,
  ...
)
```

Arguments

x	an object of class postmean.bspcov from estimate() function.
title	character string for plot title. If NULL, auto-generated title is used.
color_limits	optional vector of length 2 specifying color scale limits. If NULL, computed from data.
color_low	color for low values in heatmap. Default is "black".
color_high	color for high values in heatmap. Default is "white".
base_size	base font size for plot theme. Default is 14.
legend_position	position of legend. Default is "bottom".
x_label	label for x-axis. Default is "Variable".
y_label	label for y-axis. Default is "Variable".
show_values	logical indicating whether to display values on tiles. Default is FALSE.
...	additional arguments passed to plotting functions.

Value

A ggplot object showing heatmap visualization of the posterior mean covariance matrix.

Author(s)

Seongil Jo, Kyeongwon Lee

See Also

estimate, plot.bspcov, plot.quantile.bspcov

Examples

```
# Example with simulated data
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::thresPPP(X, eps=0.01, thres=list(value=0.5,fun='hard'), nsample=100)
est <- bspcov::estimate(res)

# Basic plot
plot(est)
```

```
# Plot with custom color scheme
plot(est, color_low = "blue", color_high = "red")
```

`plot.quantile.bspcov` *Plot method for quantile.bspcov objects*

Description

Create visualization plots for posterior quantiles of covariance matrices. Produces heatmap visualizations showing uncertainty across different quantile levels.

Usage

```
## S3 method for class 'quantile.bspcov'
plot(
  x,
  type = "heatmap",
  titles = NULL,
  ncol = 3,
  color_limits = NULL,
  color_low = "black",
  color_high = "white",
  base_size = 14,
  legend_position = "bottom",
  x_label = "Variable",
  y_label = "Variable",
  width = NULL,
  height = 6,
  ...
)
```

Arguments

`x`	an object of class `quantile.bspcov` from `quantile()` function.
`type`	character string specifying plot type. Options are "heatmap" (default), "uncertainty", or "comparison".
`titles`	character vector of titles for each quantile plot. If NULL, auto-generated titles are used.
`ncol`	number of columns in the plot layout. Default is 3.
`color_limits`	optional vector of length 2 specifying color scale limits. If NULL, computed from data.
`color_low`	color for low values in heatmap. Default is "black".
`color_high`	color for high values in heatmap. Default is "white".

base_size	base font size for plot theme. Default is 14.
legend_position	position of legend. Default is "bottom".
x_label	label for x-axis. Default is "Variable".
y_label	label for y-axis. Default is "Variable".
width	plot width when saving. Default is calculated based on number of quantiles.
height	plot height when saving. Default is 6.
...	additional arguments passed to plotting functions.

Value

A ggplot object (single quantile) or patchwork object (multiple quantiles) showing heatmap visualizations.

Author(s)

Kyeongwon Lee

See Also

quantile, plot.bspcov, plot.postmean.bspcov

Examples

```
# Example with simulated data
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::bandPPP(X, 2, 0.01, nsample = 100)
quant <- quantile(res)

# Plot uncertainty visualization
plot(quant)

# Plot with custom titles and labels
plot(quant, titles = c("Lower Bound", "Median", "Upper Bound"),
     x_label = "Variable", y_label = "Variable")
```

proc_colon *Preprocess Colon Gene Expression Data*

Description

The proc_colon function preprocesses colon gene expression data by:

1. Log transforming the raw counts.
2. Performing two-sample t-tests for each gene between normal and tumor samples.
3. Selecting the top 50 genes by absolute t-statistic.
4. Returning the filtered expression matrix and sample indices/groups.

Usage

```
proc_colon(colon, tissues)
```

Arguments

colon	A numeric matrix of raw colon gene expression values (genes × samples). Rows are genes; columns are samples.
tissues	A numeric vector indicating tissue type per sample: positive for normal, negative for tumor.

Value

A list with components:

X A numeric matrix (samples x 50 genes) of selected, log-transformed expression values.

normal_idx Integer indices of normal-tissue columns in the original data.

tumor_idx Integer indices of tumor-tissue columns in the original data.

group Integer vector of length ncol(colon), with 1 = normal, 2 = tumor.

Examples

```
data("colon")
data("tissues")
set.seed(1234)
colon_data <- proc_colon(colon, tissues)
X <- colon_data$X

foo <- bmspcov(X, Sigma = cov(X))
sigmah <- estimate(foo)
```

proc_SP500 *Preprocess SP500 data*

Description

The proc_SP500 function preprocesses the SP500 stock data by calculating monthly returns for selected sectors and generating idiosyncratic errors.

Usage

```
proc_SP500(SP500data, sectors)
```

Arguments

SP500data	A data frame containing SP500 stock data with columns including: **symbol** Stock symbol. **date** Date of the stock data. **adjusted** Adjusted closing price of the stock. **sector** Sector of the stock.
sectors	A character vector specifying the sectors to include in the analysis.

Details

1. Calculates monthly returns for each stock in the specified sectors
2. Estimates the number of factors via hdbinseg::get.factor.model(ic="ah")
3. Uses POET::POET() to extract factor loadings/factors and form idiosyncratic errors

Value

A list with components:

Uhat	A matrix of idiosyncratic errors.
Khat	Estimated number of factors.
factorparthat	Estimated factor returns.
sectornames	Sector for each column of Uhat.

Examples

```
data("SP500")
set.seed(1234)
sectors <- c("Energy", "Financials", "Materials",
   "Real Estate", "Utilities", "Information Technology")

Uhat <- proc_SP500(SP500, sectors)$Uhat
PPPres <- thresPPP(Uhat, eps = 0, thres = list(value = 0.0020, fun = "hard"), nsample = 100)
postmean <- estimate(PPPres)
```

```
diag(postmean) <- 0 # hide color for diagonal
plot(postmean)
```

---

quantile.bspcov *Quantiles of posterior distribution*

Description

Compute quantiles to describe posterior distribution. For multiple chains, combines all chains to compute more robust quantiles.

Usage

```
## S3 method for class 'bspcov'
quantile(x, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

x	an object from **bandPPP**, **bmspcov**, **sbmspcov**, and **thresPPP**.
probs	numeric vector of probabilities with values in [0,1]. Default is c(0.025, 0.5, 0.975).
...	additional arguments for quantile.

Value

quantiles	a list containing quantile matrices for each probability level. For multiple chains, uses combined samples from all chains.

Author(s)

Kyeongwon Lee

See Also

estimate, plot.postmean.bspcov

Examples

```
# Example with simulated data
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::bandPPP(X, 2, 0.01, nsample=100)
quant <- quantile(res)

# Get 95% credible intervals
```

```
quant <- quantile(res, probs = c(0.025, 0.975))
```

---

save_quantile_plot *Save quantile plot to file*

Description

Convenience function to save quantile plots with appropriate dimensions.

Usage

```
save_quantile_plot(x, filename, width = NULL, height = 6, ...)
```

Arguments

x	an object of class quantile.bspcov.
filename	filename to save the plot.
width	plot width. If NULL, calculated based on number of quantiles.
height	plot height. Default is 6.
...	additional arguments passed to plot.quantile.bspcov and ggsave.

Examples

```
# Example with simulated data
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::bandPPP(X, 2, 0.01, nsample = 100)
quant <- quantile(res)
```

---

sbmspcov *Bayesian Sparse Covariance Estimation using Sure Screening*

Description

Provides a Bayesian sparse and positive definite estimate of a covariance matrix via screened beta-mixture prior.

Usage

```
sbmspcov(
  X,
  Sigma,
  cutoff = list(),
  prior = list(),
  nsample = list(),
  nchain = 1,
  seed = NULL,
  do.parallel = FALSE,
  show_progress = TRUE
)
```

Arguments

X	a n $\times$ p data matrix with column mean zero.
Sigma	an initial guess for Sigma.
cutoff	a list giving the information for the threshold. The list includes the following parameters (with default values in parentheses): method ('FNR') giving the method for determining the threshold value (method='FNR' uses the false negative rate (FNR)-based approach, method='corr' chooses the threshold value by sample correlations), rho a lower bound of meaningfully large correlations whose the defaults values are 0.25 and 0.2 for method = 'FNR' and method = 'corr', respectively. Note. If method='corr', rho is used as the threshold value. FNR (0.05) giving the prespecified FNR level when method = 'FNR'. nsimdata (1000) giving the number of simulated datasets for calculating Jeffreys' default Bayes factors when method = 'FNR'.
prior	a list giving the prior information. The list includes the following parameters (with default values in parentheses): a (1/2) and b (1/2) giving the shape parameters for beta distribution, lambda (1) giving the hyperparameter for the diagonal elements, tau1sq (log(p)/(p^2*n)) giving the hyperparameter for the shrinkage prior of covariance.
nsample	a list giving the MCMC parameters. The list includes the following integers (with default values in parentheses): burnin (1000) giving the number of MCMC samples in transition period, nmc (1000) giving the number of MCMC samples for analysis.
nchain	number of MCMC chains to run. Default is 1.
seed	random seed for reproducibility. If NULL, no seed is set. For multiple chains, each chain gets seed + i - 1.
do.parallel	logical indicating whether to run multiple chains in parallel using future.apply. Default is FALSE. When TRUE, automatically sets up a multisession plan with nchain workers if no parallel plan is already configured.
show_progress	logical indicating whether to show progress bars during MCMC sampling. Default is TRUE. Automatically set to FALSE for individual chains when running in parallel.

Details

Lee, Jo, Lee, and Lee (2023+) proposed the screened beta-mixture shrinkage prior for estimating a sparse and positive definite covariance matrix. The screened beta-mixture shrinkage prior for $\Sigma = (\sigma_{jk})$ is defined as

$$\pi(\Sigma) = \frac{\pi^u(\Sigma) I(\Sigma \in C_p)}{\pi^u(\Sigma \in C_p)},\ C_p = \{ \text{ all } p \times p \text{ positive definite matrices } \},$$

where $\pi^u(\cdot)$ is the unconstrained prior given by

$$\pi^u(\sigma_{jk} \mid \psi_{jk}) = N\left(\sigma_{jk} \mid 0, \frac{\psi_{jk}}{1-\psi_{jk}}\tau_1^2\right),\ \psi_{jk} = 1 - 1/(1+\phi_{jk})$$

$$\pi^u(\psi_{jk}) = Beta(\psi_{jk} \mid a, b) \text{ for } (j,k) \in S_r(\hat{R})$$

$$\pi^u(\sigma_{jj}) = Exp(\sigma_{jj} \mid \lambda),$$

where $S_r(\hat{R}) = \{(j,k) : 1 < j < k \leq p, |\hat{\rho}_{jk}| > r\}$, $\hat{\rho}_{jk}$ are sample correlations, and r is a threshold given by user.

For more details, see Lee, Jo, Lee and Lee (2022+).

Value

`Sigma`	a nmc $\times$ p(p+1)/2 matrix including lower triangular elements of covariance matrix. For multiple chains, this becomes a list of matrices.
`p`	dimension of covariance matrix.
`Phi`	a nmc $\times$ p(p+1)/2 matrix including shrinkage parameters corresponding to lower triangular elements of covariance matrix. For multiple chains, this becomes a list of matrices.
`INDzero`	a list including indices of off-diagonal elements screened by sure screening. For multiple chains, this becomes a list of lists.
`cutoff`	the cutoff value specified by FNR-approach. For multiple chains, this becomes a list.
`mcmctime`	elapsed time for MCMC sampling. For multiple chains, this becomes a list.
`nchain`	number of chains used.
`burnin`	number of burn-in samples discarded.
`nmc`	number of MCMC samples retained for analysis.

Author(s)

Kyoungjae Lee, Seongil Jo, and Kyeongwon Lee

References

Lee, K., Jo, S., Lee, K., and Lee, J. (2023+), "Scalable and optimal Bayesian inference for sparse covariance matrices via screened beta-mixture prior", arXiv:2206.12773.

See Also

bmspcov

Examples

```
set.seed(1)
n <- 20
p <- 5

# generate a sparse covariance matrix:
True.Sigma <- matrix(0, nrow = p, ncol = p)
diag(True.Sigma) <- 1
Values <- -runif(n = p*(p-1)/2, min = 0.2, max = 0.8)
nonzeroIND <- which(rbinom(n=p*(p-1)/2,1,prob=1/p)==1)
zeroIND = (1:(p*(p-1)/2))[-nonzeroIND]
Values[zeroIND] <- 0
True.Sigma[lower.tri(True.Sigma)] <- Values
True.Sigma[upper.tri(True.Sigma)] <- t(True.Sigma)[upper.tri(True.Sigma)]
if(min(eigen(True.Sigma)$values) <= 0){
  delta <- -min(eigen(True.Sigma)$values) + 1.0e-5
  True.Sigma <- True.Sigma + delta*diag(p)
}

# generate a data
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = True.Sigma)

# compute sparse, positive covariance estimator:
fout <- bspcov::sbmspcov(X = X, Sigma = diag(diag(cov(X))))
post.est.m <- bspcov::estimate(fout)
sqrt(mean((post.est.m - True.Sigma)^2))
sqrt(mean((cov(X) - True.Sigma)^2))

# Multiple chains example with parallel processing:
# fout_multi <- bspcov::sbmspcov(X = X, Sigma = diag(diag(cov(X))),
#                                nchain = 4, seed = 123, do.parallel = TRUE)
# post.est.multi <- bspcov::estimate(fout_multi)
# summary(fout_multi, cols = 1, rows = 1:3)  # Shows MCMC diagnostics
# plot(fout_multi, cols = 1, rows = 1:3)     # Shows colored trace plots
# When do.parallel = TRUE, the function automatically sets up
# a multisession plan with nchain workers for parallel execution.
```

SP500 *SP500 dataset*

Description

The S&P 500 dataset from State Street Global Advisors with the collection period from Jan 2013 to Nov 2023.

Format

list

Source

State Street Global Advisors.

Examples

```
data("SP500")
names(SP500)
```

---

summary.bspcov *Summary of Posterior Distribution*

Description

Provides the summary statistics for posterior samples of covariance matrix.

Usage

```
## S3 method for class 'bspcov'
summary(object, cols, rows, quantiles = c(0.025, 0.25, 0.5, 0.75, 0.975), ...)
```

Arguments

`object`	an object from **bandPPP**, **bmspcov**, **sbmspcov**, and **thresPPP**.
`cols`	a scalar or a vector including specific column indices.
`rows`	a scalar or a vector including specific row indices greater than or equal to columns indices.
`quantiles`	a numeric vector of quantiles to compute. Default is c(0.025, 0.25, 0.5, 0.75, 0.975).
`...`	additional arguments for the summary function.

Value

`summary`	a table of summary statistics including empirical mean, standard deviation, and quantiles for posterior samples. For multiple chains, also includes effective sample size (n_eff) and R-hat diagnostics.

Note

If both `cols` and `rows` are vectors, they must have the same length.

Author(s)

Seongil Jo and Kyeongwon Lee

Examples

```
# Example with simulated data
set.seed(1)
n <- 20
p <- 5

# generate a sparse covariance matrix:
True.Sigma <- matrix(0, nrow = p, ncol = p)
diag(True.Sigma) <- 1
Values <- -runif(n = p*(p-1)/2, min = 0.2, max = 0.8)
nonzeroIND <- which(rbinom(n=p*(p-1)/2,1,prob=1/p)==1)
zeroIND = (1:(p*(p-1)/2))[-nonzeroIND]
Values[zeroIND] <- 0
True.Sigma[lower.tri(True.Sigma)] <- Values
True.Sigma[upper.tri(True.Sigma)] <- t(True.Sigma)[upper.tri(True.Sigma)]
if(min(eigen(True.Sigma)$values) <= 0){
  delta <- -min(eigen(True.Sigma)$values) + 1.0e-5
  True.Sigma <- True.Sigma + delta*diag(p)
}

# generate a data
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = True.Sigma)

# compute sparse, positive covariance estimator:
fout <- bspcov::sbmspcov(X = X, Sigma = diag(diag(cov(X))))
summary(fout, cols = c(1, 3, 4), rows = c(1, 3, 4))
summary(fout, cols = 1, rows = 1:p)

# Custom quantiles:
summary(fout, cols = 1, rows = 1:3, quantiles = c(0.05, 0.5, 0.95))
```

---

thresPPP *Bayesian Estimation of a Sparse Covariance Matrix*

Description

Provides a post-processed posterior (PPP) for Bayesian inference of a sparse covariance matrix.

Usage

```
thresPPP(X, eps, thres = list(), prior = list(), nsample = 2000)
```

Arguments

`X`	a n $\times$ p data matrix with column mean zero.
`eps`	a small positive number decreasing to 0.
`thres`	a list giving the information for thresholding PPP procedure. The list includes the following parameters (with default values in parentheses): `value (0.1)` giving the positive real number for the thresholding PPP procedure, `fun ('hard')` giving the thresholding function ('hard' or 'soft') for the thresholding PPP procedure.
`prior`	a list giving the prior information. The list includes the following parameters (with default values in parentheses): `A (I)` giving the positive definite scale matrix for the inverse-Wishart prior, `nu (p + 1)` giving the degree of freedom of the inverse-Wishar prior.
`nsample`	a scalar value giving the number of the post-processed posterior samples.

Details

Lee and Lee (2023) proposed a two-step procedure generating samples from the post-processed posterior for Bayesian inference of a sparse covariance matrix:

- Initial posterior computing step: Generate random samples from the following initial posterior obtained by using the inverse-Wishart prior $IW_p(B_0, \nu_0)$

$$\Sigma \mid X_1, \ldots, X_n \sim IW_p(B_0 + nS_n, \nu_0 + n),$$

  where $S_n = n^{-1} \sum_{i=1}^n X_i X_i^\top$.

- Post-processing step: Post-process the samples generated from the initial samples

$$\Sigma_{(i)} := \begin{cases} H_{\gamma_n}(\Sigma^{(i)}) + \left[\epsilon_n - \lambda_{\min}\{H_{\gamma_n}(\Sigma^{(i)})\}\right] I_p, & \text{if } \lambda_{\min}\{H_{\gamma_n}(\Sigma^{(i)})\} < \epsilon_n, \\ H_{\gamma_n}(\Sigma^{(i)}), & \text{otherwise ,} \end{cases}$$

where $\Sigma^{(1)}, \ldots, \Sigma^{(N)}$ are the initial posterior samples, ϵ_n is a positive real number, and $H_{\gamma_n}(\Sigma)$ denotes the generalized threshodling operator given as

$$(H_{\gamma_n}(\Sigma))_{ij} = \begin{cases} \sigma_{ij}, & \text{if } i = j, \\ h_{\gamma_n}(\sigma_{ij}), & \text{if } i \neq j, \end{cases}$$

where σ_{ij} is the (i, j) element of Σ and $h_{\gamma_n}(\cdot)$ is a generalized thresholding function.

For more details, see Lee and Lee (2023).

Value

`Sigma`	a nsample $\times$ p(p+1)/2 matrix including lower triangular elements of covariance matrix.
`p`	dimension of covariance matrix.

Author(s)

Kwangmin Lee

References

Lee, K. and Lee, J. (2023), "Post-processes posteriors for sparse covariances", *Journal of Econometrics*.

See Also

cv.thresPPP

Examples

```
n <- 25
p <- 50
Sigma0 <- diag(1, p)
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma0)
res <- bspcov::thresPPP(X, eps=0.01, thres=list(value=0.5,fun='hard'), nsample=100)
est <- bspcov::estimate(res)
```

---

tissues *tissues dataset*

Description

The tissues data of colon cancer dataset, which includes gene expression values from 22 colon tumor tissues and 40 non-tumor tissues.

Format

numeric

Source

http://genomics-pubs.princeton.edu/oncology/affydata/.

Examples

```
data("tissues")
head(tissues)
```

Index