Package ‘bayeslm’

June 18, 2018

Type Package
Title Efficient Sampling for Gaussian Linear Regression with Arbitrary Priors
Version 0.8.0
Date 2018-6-17
Author P. Richard Hahn, Jingyu He, Hedibert Lopes
Maintainer Jingyu He <jingyu.he@chicagobooth.edu>
License LGPL (>= 2)
Imports Rcpp (>= 0.12.7), stats, graphics, grDevices, coda, RcppParallel
SystemRequirements GNU make
Depends R (>= 2.10)
URL http://jingyuhe.com/software.html
NeedsCompilation yes
LinkingTo Rcpp, RcppArmadillo, RcppParallel
Repository CRAN
Date/Publication 2018-06-18 17:57:57 UTC

R topics documented:

bayeslm-package .............................................. 2
bayeslm ......................................................... 2
hs_gibbs ........................................................ 6
plot.MCMC ...................................................... 7
predict.bayeslm.fit ........................................... 8
summary.bayeslm.fit .......................................... 9
summary.MCMC ................................................ 10

Index 12
bayeslm-package

Efficient sampling for Gaussian linear regression with arbitrary priors

Description

The elliptical slice sampler for Bayesian linear regression with shrinkage priors such as horseshoe, Laplace prior, ridge prior.

Author(s)

P. Richard Hahn, Jingyu He and Hedibert Lopes

Maintainer: Jingyu He <jingyu.he@chicagobooth.edu>

References


See Also

bayeslm

Usage

```r
## Default S3 method:
bayeslm(Y = FALSE, X = FALSE, prior = "horseshoe", penalize = NULL,
block_vec = NULL, sigma = NULL, s2 = 1, kap2 = 1, N = 20000L, burnin = 0L,
thinning = 1L, vglobal = 1, sampling_vglobal = TRUE, verb = FALSE, icept = TRUE,
standardize = TRUE, singular = FALSE, scale_sigma_prior = TRUE, prior_mean = NULL,
prob_vec = NULL, cc = NULL, ...)

## S3 method for class 'formula'
bayeslm(formula, data = list(), Y = FALSE, X = FALSE,
prior = "horseshoe", penalize = NULL, block_vec = NULL, sigma = NULL,
s2 = 1, kap2 = 1, N = 20000L, burnin = 0L, thinning = 1L, vglobal = 1,
sampling_vglobal = TRUE, verb = FALSE, standardize = TRUE, singular = FALSE,
scale_sigma_prior = TRUE, prior_mean = NULL,
prob_vec = NULL, cc = NULL, ...)
```
Arguments

- **formula**: formula of the model to fit.
- **data**: an optional data frame containing the variables in the model. By default the variables are taken from the environment which `bayeslm` is called from.
- **Y** data.frame, matrix, or vector of inputs Y. Response variable.
- **X** data.frame, matrix, or vector of inputs X. Regressors.
- **prior**: Indicating shrinkage prior to use. "horseshoe" for approximate horseshoe prior (default), "laplace" for laplace prior, "ridge" for ridge prior, "sharkfin" for "sharkfin" prior and "nonlocal" for nonlocal prior.
- **block_vec**: A vector indicating number of regressors in each block. Sum of all entries should be the same as number of regressors. The default value is block_vec = rep(1, p), put every regressor in its own block (slice-within-Gibbs sampler).
- **penalize**: A vector indicating shrink regressors or not. It's length should be the same as number of regressors. 1 indicates shrink corresponding coefficient, 0 indicates no shrinkage. The default value is rep(1, p), shrink all coefficients.
- **sigma**: Initial value of residual standard error. The default value is half of standard error of Y.
- **s2, kap2**: Parameter of prior over sigma, an inverse gamma prior with rate s2 and shape s2.
- **N**: Number of posterior samples (after burn-in).
- **burnin**: Number of burn-in samples. If burnin > 0, the function will draw N + burnin samples and return the last N samples only.
- **thinning**: Number of thinnings. thinning = 1 means no thinning.
- **vglobal**: Initial value of global shrinkage parameter. Default value is 1.
- **sampling_vglobal**: Bool, if TRUE, sampling the global shrinkage parameter by random walk Metropolis Hastings on log scale, otherwise always stay at the initial value vglobal.
- **verb**: Bool, if TRUE, print out sampling progress.
- **icept**: Bool, if the inputs are matrix X and Y, and icept = TRUE, the function will estimate intercept. Default value is TRUE. If the input is formula Y ~ X, option icept is useless, control intercept by formular Y ~ X or Y ~ X - 1.
- **standardize**: Bool, if TRUE, standardize X and Y before sampling.
- **singular**: Bool, if TRUE, take it as a rank-deficient case such as n < p or X'X is singular. See section 2.3.2 of the paper for details.
- **scale_sigma_prior**: Bool, if TRUE, the prior of regression coefficient $\beta$ is scaled by residual standard error $\sigma$.
- **prior_mean**: vector, specify prior mean of nonlocal prior for each regressor. It should have length p (no intercept) or p + 1 (intercept). The default value is 1.5 for all regressors.
- **prob_vec**: vector, specify prior mean of sharkfin prior for each regressor. It should have length p (no intercept) or p + 1 (intercept). The default value is 0.25 for all regressors.
cc Only works when singular == TRUE, precision parameter of ridge adjustment. It should be a vector with length SpS. If it is NULL, it will be set as rep(10, p).

... optional parameters to be passed to the low level function bayeslm.default.

Details
For details of the approach, please see Hahn, He and Lopes (2017)

Value

- **loops** A vector of number of elliptical slice sampler loops for each posterior sample.
- **sigma** A vector of posterior samples of residual standard error.
- **vglobal** A vector of posterior samples of the global shrinkage parameter.
- **beta** A matrix of posterior samples of coefficients.
- **fitted.values** Fitted values of the regression model. Take posterior mean of coefficients with 20% burnin samples.
- **residuals** Residuals of the regression model, equals y - fitted.values.

Note
horseshoe is essentially call function bayeslm with prior = "horseshoe". Same for sharkfin, ridge, blasso, nonlocal.

Author(s)
Jingyu He <jingyu.he@chicagobooth.edu>

References

Examples

```r
p = 20
n = 100
kappa = 1.25
beta_true = c(c(1,2,3),rnorm(p-3,0,0.01))
sig_true = kappa*sqrt(sum(beta_true^2))

x = matrix(rnorm(p*n),n,p)
y = x %*% beta_true + sig_true * rnorm(n)

x = as.matrix(x)
y = as.matrix(y)
```
data = data.frame(x = x, y = y)

block_vec = rep(1, p) # slice-within-Gibbs sampler, put every coefficient in its own block

fitOLS = lm(y~x-1)

# call the function using formulas
fita = bayeslm(y ~ x, prior = 'horseshoe',
block_vec = block_vec, N = 10000, burnin = 2000)
# summary the results
summary(fita)
summary(fita$beta)

# put the first two coefficients in one elliptical sampling block
block_vec2 = c(2, rep(1, p-2))
fitb = bayeslm(y ~ x, data = data, prior = 'horseshoe',
block_vec = block_vec2, N = 10000, burnin = 2000)

# comparing several different priors
fit1 = bayeslm(y,x,prior = 'horseshoe', 1cept = FALSE,
block_vec = block_vec, N = 10000, burnin=2000)
beta_est1 = colMeans(fit1$beta)

fit2 = bayeslm(y,x,prior = 'laplace', 1cept = FALSE,
block_vec = block_vec, N = 10000, burnin=2000)
beta_est2 = colMeans(fit2$beta)

fit3 = bayeslm(y,x,prior = 'ridge', 1cept = FALSE,
block_vec = block_vec, N = 10000, burnin=2000)
beta_est3 = colMeans(fit3$beta)

fit4 = bayeslm(y,x,prior = 'sharkfin', 1cept = FALSE,
block_vec = block_vec, N = 10000, burnin=2000)
beta_est4 = colMeans(fit4$beta)

fit5 = bayeslm(y,x,prior = 'nonlocal', 1cept = FALSE,
block_vec = block_vec, N = 10000, burnin=2000)
beta_est5 = colMeans(fit5$beta)

plot(NULL,xlim=range(beta_true),ylim=range(beta_true),
  xlab = "beta true", ylab = "estimation")
points(beta_true,beta_est1,pch=20)
points(beta_true,fitOLS$coef,col='red')
points(beta_true,beta_est2,pch=20,col='cyan')
points(beta_true,beta_est3,pch=20,col='orange')
points(beta_true,beta_est4,pch=20,col='pink')
points(beta_true,beta_est5,pch=20,col='lightgreen')

legend("topleft", c("OLS", "horseshoe", "laplace", "ridge", "sharkfin", "nonlocal"), col = c("red", "black", "cyan", "orange", "pink", "lightgreen"), pch = rep(1, 6))
hs_gibbs

Gibbs sampler of horseshoe regression

Description
Standard Gibbs sampler of horseshoe regression.

Usage
hs_gibbs(Y, X, nsamps, a, b, scale_sigma_prior)

Arguments
Y  Response of regression.
X  Matrix of regressors.
nsamps  Number of posterior samples.
a  Parameter of inverse Gamma prior on \( \sigma \).
b  Parameter of inverse Gamma prior on \( \sigma \).
scale_sigma_prior  Bool, if TRUE, use prior scaled by \( \sigma \).

Details
This function implements standard Gibbs sampler of horseshoe regression. The prior is \( y \mid \beta, \sigma^2, X \sim MVN(X\beta, \sigma^2 I) \beta_i \mid \tau, \lambda_i, \sigma \sim N(0, \lambda_i^2 \tau^2 \sigma^2) \sigma^2 \sim IG(a, b) \tau \sim C^+(0, 1) \lambda_i \sim C^+(0, 1) \)

Author(s)
Jingyu He
plot.MCMC

See Also

summary.mcmc

Examples

```r
x = matrix(rnorm(1000), 100, 10)
y = x + rnorm(100)
fit = hs_gibbs(y, x, 1000, 1, 1, TRUE)
summary(fit)
```

---

**Description**

plot.MCMC is an S3 method to plot empirical distribution of posterior draws. The input is a MCMC matrix.

**Usage**

```r
## S3 method for class 'MCMC'
plot(x, names = NULL, burnin = trunc(.1 * nrow(x)), tvalues, TRACEPLOT = TRUE, DEN = TRUE, INT = TRUE,
     CHECK_NDRAWS = TRUE, ...)
```

**Arguments**

- `x`: A MCMC class matrix of posterior draws, such as bayeslm$beta.
- `names`: an optional character vector of names for the columns of `x`.
- `burnin`: Number of draws to burn-in (default value is 0.1 * nrow(X)).
- `tvalues`: vector of true values.
- `TRACEPLOT`: logical, TRUE provide sequence plots of draws and acfs (default: TRUE)
- `DEN`: logical, TRUE use density scale on histograms (default: TRUE)
- `INT`: logical, TRUE put various intervals and points on graph (default: TRUE)
- `CHECK_NDRAWS`: logical, TRUE check that there are at least 100 draws (default: TRUE)
- `...`: optional arguments for generic function.

**Details**

This function is modified from package bayesm by Peter Rossi. It plots summary of posterior draws.

**Author(s)**

Peter Rossi, Anderson School, UCLA, <perissichi@gmail.com>.
predict.bayeslm.fit

See Also

summary.bayeslm.fit

Examples

```r
x = matrix(rnorm(1000), 100, 10)
y = x + rnorm(10) + rnorm(100)
fit = bayeslm(y ~ x)
plot(fit$beta)
```

---

predict.bayeslm.fit  Predict new data

Description

predict.bayeslm.fit is an S3 method to predict response on new data.

Usage

```r
## S3 method for class 'bayeslm.fit'
predict(object, data, burnin, X, ...)
```

Arguments

- `object`: object is output of `bayeslm` function.
- `data`: A data frame or list of new data to predict.
- `burnin`: number of draws to burn-in (default value is `0.1 * nrow(X)`).
- `X`: If call `bayeslm` with matrices input `x` and `y` but not formula, pass new matrix to predict here. See example for details.
- `...`: optional arguments for generic function.

Details

Make prediction on new data set, users are allowed to adjust number of burn-in samples.

Author(s)

Jingyu He
Examples

```r
x = matrix(rnorm(1000), 100, 10)
y = x %*% rnorm(10) + rnorm(100)
data = list(x = x, y = y)

# Train the model with formula input
fit1 = bayeslm(y ~ x, data = data)
# predict
pred1 = predict(fit1, data)

# Train the model with matrices input
fit2 = bayeslm(Y = y, X = x)
pred2 = predict(fit2, X = x)
```

summary.bayeslm.fit

Summarize fitted object of `bayeslm`

Description

`summary.bayeslm.fit` is an S3 method to summarize returned object of function `bayeslm`. The input should be `bayeslm` object.

Usage

```r
## S3 method for class 'bayeslm.fit'
summary(object, names=NULL, burnin=NULL, quantiles=FALSE, trailer=TRUE, ...)
```

Arguments

- `object` object is a fitted object, returned by function `bayeslm`.
- `names` an optional character vector of names for all the coefficients.
- `burnin` number of draws to burn-in (if it is NULL, will set default value as 0.2*`nrow(object$beta)`)
- `quantiles` logical for should quantiles be displayed (def: FALSE)
- `trailer` logical for should a trailer be displayed (def: TRUE)
- `...` optional arguments for generic function

Details

This function summarize returned object of function `bayeslm`. It prints mean, std Dev, effective sample size (computed by function `effectiveSize` in package coda) coefficients posterior samples. If `quantiles=TRUE`, quantiles of marginal distributions in the columns of `X` are displayed.

The function also returns significance level, defined by whether the symmetric posterior quantile-based credible interval excludes zero. For example, a regression coefficient with one * has 0.025 quantile and 0.975 quantile with the same sign. Similarly, '***' denotes 0.0005 and 0.9995, '**' denotes 0.005 and 0.995, '*' denotes 0.025 and 0.975, '.' denotes 0.05 and 0.95 quantiles with the same sign.
**Author(s)**

Jingyu He

**See Also**

`summary.mcmc`

**Examples**

```r
x = matrix(rnorm(1000), 100, 10)
y = x + rnorm(10) + rnorm(100)
fit = bayeslm(y ~ x)
summary(fit)
```

---

**Description**

`summary.mcmc` is an S3 method to summarize posterior draws of the model. The input should be a matrix of draws.

**Usage**

```r
## S3 method for class 'MCMC'
summary(object, names, burnin = trunc(.1 * nrow(X)), quantiles = FALSE, trailer = TRUE, ...)
```

**Arguments**

- `object`: object is a matrix of draws, usually an object of class MCMC. It’s same as `X`.
- `names`: an optional character vector of names for the columns of `X`.
- `burnin`: number of draws to burn-in (default value is `0.1 * nrow(X)`).
- `quantiles`: logical for should quantiles be displayed (def: FALSE).
- `trailer`: logical for should a trailer be displayed (def: TRUE).
- `...`: optional arguments for generic function.

**Details**

This function is modified from package bayesm by Peter Rossi. It summarize object MCMC. Mean, Std Dev, effective sample size (computed by function `effectiveSize` in package coda) are displayed. If quantiles=TRUE, quantiles of marginal distributions in the columns of `X` are displayed.

The function also returns significance level, defined by whether the symmetric posterior quantile-based credible interval excludes zero. For example, a regression coefficient with one * has 0.025 quantile and 0.975 quantile with the same sign. Similarly, '***' denotes 0.0005 and 0.9995, '**' denotes 0.005 and 0.995, '*' denotes 0.025 and 0.975, '.' denotes 0.05 and 0.95 quantiles with the same sign.
Author(s)

Peter Rossi, Anderson School, UCLA, <perossichi@gmail.com>.

See Also

summary.bayeslm.fit

Examples

```r
x = matrix(rnorm(1000), 100, 10)
y = x %*% rnorm(10) + rnorm(100)
fit=bayeslm(y~x)
summary(fit$beta)
```
Index

*Topic **linear regression**
  bayeslm, 2
*Topic **package**
  bayeslm-package, 2
*Topic **summary**
  summary.MCMC, 10
*Topic **summary**
  plot.MCMC, 7
  summary.bayeslm.fit, 9
*Topic **univar**
  hs_gibbs, 6
  predict.bayeslm.fit, 8

bayeslm, 2
bayeslm-package, 2

hs_gibbs, 6

plot.MCMC, 7
predict.bayeslm.fit, 8

summary.bayeslm.fit, 8, 9, 11
summary.MCMC, 10
summary.mcmc, 7, 10