Package 'LassoHiDFastGibbs'

January 18, 2026

Type Package

Title Fast High-Dimensional Gibbs Samplers for Bayesian Lasso Regression

Version 0.1.0

Description Provides fast and scalable Gibbs sampling algorithms for Bayesian Lasso regression model in high-dimensional settings. The package implements efficient partially collapsed and nested Gibbs samplers for Bayesian Lasso, with a focus on computational efficiency when the number of predictors is large relative to the sample size.

License GPL-3

Encoding UTF-8

LazyData true

SystemRequirements C++17

Imports Rcpp

LinkingTo Rcpp, RcppArmadillo, RcppEigen, RcppNumerical, RcppClock

RoxygenNote 7.3.2

Suggests posterior

URL https://github.com/MJDavoudabadi/LassoHiDFastGibbs

BugReports https://github.com/MJDavoudabadi/LassoHiDFastGibbs/issues

Contents

`blasso_gibbs_2block_bl` *Bayesian lasso Gibbs sampler: 2-block (beta–lambda2) variant*

Description

Implements a two-block Gibbs sampler for the Bayesian lasso regression model in which the regression coefficients are updated jointly with the global shrinkage parameter λ^2 in one block, while the noise variance and local shrinkage parameters are updated conditionally in separate steps.

Usage

```
blasso_gibbs_2block_bl(
  vy,
  mX,
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  va_init = NULL,
  verbose = max(1L, floor(nsamples/5)),
  lower = 1e-12,
  upper = 5000
)
```

Arguments

`vy`	Numeric response vector of length n.
`mX`	Numeric design matrix of dimension n x p.
`a, b`	Hyperparameters for the inverse-gamma prior on sigma^2.
`u, v`	Hyperparameters for the prior on lambda^2.
`nsamples`	Integer number of MCMC iterations.
`lambda_init`	Initial value for lambda.
`sigma2_init`	Initial value for sigma^2.
`va_init`	Optional initial values for local shrinkage parameters (length p).
`verbose`	Print progress every `verbose` iterations (0 = silent).
`lower, upper`	Bounds used by the slice sampler for lambda^2.

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 30; p <- 6
X <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
out <- blasso_gibbs_2block_bl(
  vy = y, mX = X,
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  verbose = 0
)
str(out)
```

`blasso_gibbs_2block_bs`

Bayesian lasso Gibbs sampler: 2-block (beta–sigma2) variant

Description

Implements a two-block Gibbs sampler for the Bayesian lasso regression model in which the regression coefficients are updated jointly with the noise variance σ^2 in one block, while the global shrinkage parameter and local shrinkage parameters are updated conditionally in separate steps.

Usage

```
blasso_gibbs_2block_bs(
  vy,
  mX,
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  verbose = max(1L, floor(nsamples/5))
)
```

Arguments

`vy`	Numeric response vector of length n.
`mX`	Numeric design matrix of dimension n x p.
`a, b`	Hyperparameters for the inverse-gamma prior on sigma^2.
`u, v`	Hyperparameters for the prior on lambda^2.
`nsamples`	Integer number of MCMC iterations.
`lambda_init`	Initial value for lambda.
`sigma2_init`	Initial value for sigma^2.
`verbose`	Print progress every `verbose` iterations (0 = silent).

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 30; p <- 6
X <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
out <- blasso_gibbs_2block_bs(
  vy = y, mX = X,
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  verbose = 0
)
str(out)
```

---

blasso_pcg_lambda2_va *Bayesian lasso PCG sampler: lambda2 collapsed over local scales*

Description

Lasso-specific Partially-collapsed Gibbs (PCG) variant with the local scales (va) collapsed in the λ^2 update.

Usage

```
blasso_pcg_lambda2_va(
  vy,
  mX,
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  verbose = max(1L, floor(as.integer(nsamples)/10))
)
```

Arguments

vy	Numeric response vector of length n.
mX	Numeric design matrix of dimension n x p.
a, b	Hyperparameters for the inverse-gamma prior on σ^2.

u, v	Hyperparameters for the prior on λ^2.
nsamples	Number of MCMC iterations.
lambda_init	Initial value for λ.
sigma2_init	Initial value for σ^2.
verbose	Print progress every verbose iterations (0 = silent).

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)

out <- blasso_pcg_lambda2_va(
  vy = y, mX = X,
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  verbose = 0
)

summary(out$vlambda2)
```

---

blasso_pcg_sigma2_va *Bayesian lasso PCG sampler: sigma2 collapsed over local scales*

Description

Lasso-specific Partially-collapsed Gibbs (PCG) variant with the local scales (va) collapsed in the σ^2 update.

Usage

```
blasso_pcg_sigma2_va(
  vy,
  mX,
  a,
  b,
  u,
  v,
  nsamples,
```

```
  lambda_init = 1,
  sigma2_init = 1,
  va_init = NULL,
  verbose = max(1L, floor(as.integer(nsamples)/10)),
  lower = 1e-12,
  upper = 5000
)
```

Arguments

vy	Numeric response vector of length n.
mX	Numeric design matrix of dimension n x p.
a, b	Hyperparameters for the inverse-gamma prior on σ^2.
u, v	Hyperparameters for the prior on λ^2.
nsamples	Number of MCMC iterations.
lambda_init	Initial value for λ.
sigma2_init	Initial value for σ^2.
va_init	Optional initial local-scale vector (length p). If NULL, defaults to rep(1, ncol(mX)).
verbose	Print progress every verbose iterations (0 = silent).
lower, upper	Bounds used by the slice sampler.

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)

out <- blasso_pcg_sigma2_va(
  vy = y, mX = X,
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  va_init = rep(1, p),
  verbose = 0
)

summary(out$vsigma2)
```

LassoHiDFastGibbs *Fast High-Dimensional Gibbs Samplers for Bayesian Lasso Regression*

Description

Provides fast and scalable Gibbs sampling algorithms for Bayesian Lasso regression model in high-dimensional settings. The package implements efficient partially collapsed and nested Gibbs samplers for Bayesian Lasso, with a focus on computational efficiency when the number of predictors is large relative to the sample size.

Author(s)

Maintainer: Mohammad Javad Davoudabadi <mohammad.davoudabadi@qut.edu.au> [copyright holder]

Authors:

- John Ormerod <john.ormerod@sydney.edu.au> (ORCID)
- Garth Tarr <garth.tarr@gmail.com> (ORCID)
- Samuel Mueller <samuel.mueller@sydney.edu.au> (ORCID)

See Also

Useful links:

- https://github.com/MJDavoudabadi/LassoHiDFastGibbs
- Report bugs at https://github.com/MJDavoudabadi/LassoHiDFastGibbs/issues

normalize *Normalize Response and Covariates*

Description

This function centers and (optionally) scales the response vector and each column of the design matrix using the population variance. It is used to prepare data for Bayesian Lasso regression.

Usage

```
normalize(y, X, scale = TRUE)
```

Arguments

y	A numeric response vector.
X	A numeric matrix or data frame of covariates (design matrix).
scale	Logical; if TRUE, variables are scaled to have unit population variance (default is TRUE).

Value

A list with the following elements:

- `vy`: Normalized response vector.
- `mX`: Normalized design matrix.
- `mu.y`: Mean of the response vector.
- `sigma2.y`: Population variance of the response vector.
- `mu.x`: Vector of column means of `X`.
- `sigma2.x`: Vector of population variances for columns of `X`.

Examples

```
set.seed(1)
X <- matrix(rnorm(100 * 10), 100, 10)
beta <- c(2, -3, rep(0, 8))
y <- as.vector(X %*% beta + rnorm(100))
norm_result <- normalize(y, X)
```

`penalized_nested_Gibbs` *Penalized nested Gibbs sampler for Bayesian linear regression*

Description

Runs the nested Gibbs sampler for a Gaussian linear model $y = X\beta + \epsilon$ with either a lasso or horseshoe penalty (shrinkage prior) on β. The algorithm supports both $n \geq p$ and $p > n$ regimes.

Usage

```
penalized_nested_Gibbs(
  vy,
  mX,
  penalty_type = c("lasso", "horseshoe"),
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  va_init = NULL,
  verbose = max(1L, floor(as.integer(nsamples)/10)),
  lower = 1e-12,
  upper = 5000,
  s_beta = 1L,
  s_siglam = 1L
)
```

Arguments

vy	Numeric response vector of length n.
mX	Numeric design matrix of dimension $n \times p$.
penalty_type	Character string: either "lasso" or "horseshoe".
a, b	Hyperparameters for the inverse-gamma prior on σ^2.
u, v	Hyperparameters for the prior on λ^2.
nsamples	Integer number of outer MCMC iterations.
lambda_init	Initial value for λ.
va_init	Optional initial values for the local shrinkage parameters (length p). If NULL, a vector of ones is used.
verbose	Print progress every verbose iterations (0 = silent).
lower, upper	Bounds for the slice sampler used for λ^2.
s_beta	Integer: number of inner updates of β per outer iteration.
s_siglam	Integer: number of inner updates of (σ^2, λ^2) per β update.

Value

A list with components:

mBeta Matrix of sampled β draws (rows correspond to stored draws).

vsigma2 Vector of sampled σ^2 draws.

vlambda2 Vector of sampled λ^2 draws.

lm_penalized_nested_gibbs().

Examples

```
set.seed(1)
n <- 50; p <- 10
X <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)

out <- penalized_nested_Gibbs(
  vy = y, mX = X,
  penalty_type = "lasso",
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1,
  va_init = NULL,
  verbose = 0,
  lower = 1e-12,
  upper = 5000,
  s_beta = 1,
  s_siglam = 1
)
str(out)
```

`penalized_pcg_beta_sigma2` *Penalized PCG sampler: beta block, lambda2 collapsed over sigma2*

Description

Partially-collapsed Gibbs (PCG) sampler variant that updates β in a dedicated block and samples λ^2 using a collapsed step over σ^2.

Usage

```
penalized_pcg_beta_sigma2(
  vy,
  mX,
  penalty_type = c("lasso", "horseshoe"),
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  verbose = max(1L, floor(as.integer(nsamples)/10))
)
```

Arguments

`vy`	Numeric response vector of length n.
`mX`	Numeric design matrix of dimension n x p.
`penalty_type`	Character string: `"lasso"` or `"horseshoe"`.
`a, b`	Hyperparameters for the inverse-gamma prior on σ^2.
`u, v`	Hyperparameters for the prior on λ^2.
`nsamples`	Number of MCMC iterations.
`lambda_init`	Initial value for λ.
`sigma2_init`	Initial value for σ^2.
`verbose`	Print progress every `verbose` iterations (0 = silent).

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)

out <- penalized_pcg_beta_sigma2(
  vy = y, mX = X, penalty_type = "horseshoe",
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  verbose = 0
)

summary(out$mBeta)
```

---

penalized_pcg_lambda2_sigma2

Penalized PCG sampler: lambda2 collapsed over sigma2

Description

Partially-collapsed Gibbs (PCG) sampler for a Gaussian linear model with a shrinkage prior/penalty on regression coefficients. This variant samples λ^2 using a collapsed step over σ^2 (see implementation).

Usage

```
penalized_pcg_lambda2_sigma2(
  vy,
  mX,
  penalty_type = c("lasso", "horseshoe"),
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  verbose = max(1L, floor(as.integer(nsamples)/10))
)
```

Arguments

vy	Numeric response vector of length n.
mX	Numeric design matrix of dimension n x p.
penalty_type	Character string: "lasso" or "horseshoe".
a, b	Hyperparameters for the inverse-gamma prior on σ^2.
u, v	Hyperparameters for the prior on λ^2.

nsamples	Number of MCMC iterations.
lambda_init	Initial value for λ.
sigma2_init	Initial value for σ^2.
verbose	Print progress every verbose iterations (0 = silent).

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)
out <- penalized_pcg_lambda2_sigma2(
  vy = y, mX = X, penalty_type = "lasso",
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200, lambda_init = 1, sigma2_init = 1,
  verbose = 0
)
str(out)
```

---

penalized_pcg_sigma2_beta

Penalized PCG sampler: sigma2 collapsed over beta

Description

Partially-collapsed Gibbs (PCG) sampler variant that samples σ^2 using a collapsed step over β. Requires initial values for local scales va_init.

Usage

```
penalized_pcg_sigma2_beta(
  vy,
  mX,
  penalty_type = c("lasso", "horseshoe"),
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
```

```
  va_init = NULL,
  verbose = max(1L, floor(as.integer(nsamples)/10)),
  lower = 1e-12,
  upper = 5000
)
```

Arguments

vy	Numeric response vector of length n.
mX	Numeric design matrix of dimension n x p.
penalty_type	Character string: "lasso" or "horseshoe".
a, b	Hyperparameters for the inverse-gamma prior on σ^2.
u, v	Hyperparameters for the prior on λ^2.
nsamples	Number of MCMC iterations.
lambda_init	Initial value for λ.
sigma2_init	Initial value for σ^2.
va_init	Optional initial local-scale vector (length p). If NULL, defaults to rep(1, ncol(mX)).
verbose	Print progress every verbose iterations (0 = silent).
lower, upper	Bounds used by the slice sampler.

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)

out <- penalized_pcg_sigma2_beta(
  vy = y, mX = X, penalty_type = "horseshoe",
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  va_init = rep(1, p),
  verbose = 0
)

summary(out$vsigma2)
```

`penalized_pcg_sigma2_lambda2` *Penalized PCG sampler: sigma2 collapsed over lambda2*

Description

Partially-collapsed Gibbs (PCG) sampler variant that samples σ^2 using a collapsed step over λ^2 (see implementation). Requires initial values for local scales `va_init`; if omitted, it is set to a vector of ones.

Usage

```
penalized_pcg_sigma2_lambda2(
  vy,
  mX,
  penalty_type = c("lasso", "horseshoe"),
  a,
  b,
  u,
  v,
  nsamples,
  lambda_init = 1,
  sigma2_init = 1,
  va_init = NULL,
  verbose = max(1L, floor(as.integer(nsamples)/10)),
  lower = 1e-12,
  upper = 5000
)
```

Arguments

`vy`	Numeric response vector of length n.
`mX`	Numeric design matrix of dimension n x p.
`penalty_type`	Character string: `"lasso"` or `"horseshoe"`.
`a`, `b`	Hyperparameters for the inverse-gamma prior on σ^2.
`u`, `v`	Hyperparameters for the prior on λ^2.
`nsamples`	Number of MCMC iterations.
`lambda_init`	Initial value for λ.
`sigma2_init`	Initial value for σ^2.
`va_init`	Optional initial local-scale vector (length p). If `NULL`, defaults to `rep(1, ncol(mX))`.
`verbose`	Print progress every `verbose` iterations (0 = silent).
`lower`, `upper`	Bounds used by the slice sampler.

Value

A list with components:

mBeta Matrix of beta draws (nsamples x p).

vsigma2 Vector of sigma^2 draws (length nsamples).

vlambda2 Vector of lambda^2 draws (length nsamples).

Examples

```
set.seed(1)
n <- 40; p <- 6
X <- matrix(rnorm(n * p), n, p)
beta <- c(1.2, 2, -1, 0.5, 0.75, 2.5)
y <- X %*% beta + rnorm(n)

out <- penalized_pcg_sigma2_lambda2(
  vy = y, mX = X, penalty_type = "lasso",
  a = 1, b = 1, u = 1, v = 1,
  nsamples = 200,
  lambda_init = 1, sigma2_init = 1,
  va_init = rep(1, p),
  verbose = 0
)

str(out)
```

Index