

# Package: spexvb (via r-universe)

June 17, 2026

**Type** Package

**Title** Parameter Expanded Variational Bayes for High-Dimensional Linear Regression

**Version** 0.1.0

**Description** Implements a parameter expanded variational Bayes algorithm for linear regression models with high-dimensional variable selection. The methodology utilizes spike-and-slab priors to perform simultaneous estimation and selection. Details can be found in Olejua et al. (2024) [doi:10.21203/rs.3.rs-7208847/v1](https://doi.org/10.21203/rs.3.rs-7208847/v1).

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Imports** Rcpp, glmnet, caret, foreach

**LinkingTo** Rcpp, RcppArmadillo

**Suggests** testthat (>= 3.0.0), roxygen2, knitr, rmarkdown, doParallel

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Peter Olejua [aut, cre] (ORCID: <https://orcid.org/0000-0002-2478-0908>), Alexander McLain [aut]

**Maintainer** Peter Olejua <polejua@email.sc.edu>

**Repository** <https://cran.r-universe.dev>

**Date/Publication** 2026-02-17 21:40:02 UTC

**RemoteUrl** <https://github.com/cran/spexvb>

**RemoteRef** HEAD

**RemoteSha** afc6dc85143796b6ef76d34a29d241461bd9ea67

## Contents

cv.spexvb . . . . .	2
cv.spexvb.fit . . . . .	4
get.initials . . . . .	6
get.initials.logistic . . . . .	7
hspvb . . . . .	9
spexvb . . . . .	10
spexvb.logistic . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

cv.spexvb	<i>Cross-validation for Sparse Parameter Expanded Variational Bayes (spexvb)</i>
-----------	--

---

## Description

Performs k-fold cross-validation for the spexvb model, allowing for evaluation of model performance across different tau\_alpha values.

## Usage

```
cv.spexvb(
  k = 5,
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  tau_e = NULL,
  update_order = NULL,
  mu_alpha = 1,
  tau_alpha = c(0, 10^(3:7)),
  tau_b = 400,
  standardize = TRUE,
  intercept = TRUE,
  max_iter = 100L,
  tol = 1e-05,
  seed = 12376,
  verbose = TRUE,
  parallel = TRUE
)
```

**Arguments**

<code>k</code>	Integer, the number of folds for cross-validation. Must be greater than 2.
<code>X</code>	A design matrix.
<code>Y</code>	A response vector.
<code>mu_0</code>	Initial variational mean (posterior expectation of $\beta_j \mid s_j = 1$ ). If NULL, initialized automatically.
<code>omega_0</code>	Initial variational probability (posterior expectation of $s_j$ ). If NULL, initialized automatically.
<code>c_pi_0</code>	Prior parameter for pi (beta distribution shape1). If NULL, initialized automatically.
<code>d_pi_0</code>	Prior parameter for pi (beta distribution shape2). If NULL, initialized automatically.
<code>tau_e</code>	Initial precision of errors. If NULL, initialized automatically.
<code>update_order</code>	A numeric vector specifying the order of updates for coefficients. If NULL, initialized automatically.
<code>mu_alpha</code>	Mean for the prior on alpha (expansion parameter).
<code>tau_alpha</code>	A numeric vector of tau_alpha values to cross-validate over. Must have at least two values.
<code>tau_b</code>	Initial precision for $\beta_j$ (when $s_j = 1$ ).
<code>standardize</code>	Logical. Center Y, and center and scale X. Default is TRUE.
<code>intercept</code>	Logical. Whether to include an intercept. Default is TRUE. After the model is fit on the centered and scaled data, the final coefficients are "unscaled" to put them back on the original scale of your data. The intercept is then calculated separately using the means and the final coefficients.
<code>max_iter</code>	Maximum number of outer loop iterations for each spexvb fit.
<code>tol</code>	Convergence tolerance for each spexvb fit.
<code>seed</code>	Seed for reproducibility of data splitting and glmnet initials.
<code>verbose</code>	Logical, if TRUE, prints progress messages during cross-validation.
<code>parallel</code>	Logical, if TRUE, search in parallel.

**Details**

This function performs k-fold cross-validation to find the optimal tau\_alpha for the spexvb model. It iterates through different tau\_alpha values, trains the model on training folds, and evaluates performance on the held-out test fold. To leverage parallel processing, ensure a parallel backend (e.g., from doParallel or doSNOW packages) is registered using registerDoParallel() or similar before calling this function.

**Value**

A list containing cross-validation results:

<code>ordered_tau_alpha</code>	The sorted vector of tau_alpha values used.
--------------------------------	---

epe\_test\_k      A matrix of prediction errors (MSE) for each fold (rows) and each tau\_alpha (columns).

CVE              Cross-Validation Error (mean MSE) for each tau\_alpha.

tau\_alpha\_opt    The tau\_alpha value that minimizes the CVE.

### Examples

```
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
Y <- X[,1] * 2 + rnorm(n)

# Run cross-validation only (returns errors and optimal tau_alpha)
cv_res <- cv.spexvb(k = 3, X = X, Y = Y)

# Inspect the optimal tau_alpha
print(cv_res$tau_alpha_opt)
```

---

cv.spexvb.fit

*Cross-validation and Final Model Fitting for SPEVXB*

---

### Description

This function performs k-fold cross-validation to determine the optimal tau\_alpha parameter for the spexvb model, and then fits a final spexvb model to the full dataset using this optimal tau\_alpha. Initial values for the final model are also derived from the full dataset.

### Usage

```
cv.spexvb.fit(
  k = 5,
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  tau_e = NULL,
  update_order = NULL,
  mu_alpha = 1,
  tau_alpha = c(0, 10^(3:7)),
  tau_b = 400,
  standardize = TRUE,
  intercept = TRUE,
  max_iter = 100L,
```

```

    tol = 1e-05,
    seed = 12376,
    verbose = TRUE,
    parallel = TRUE
  )

```

## Arguments

<code>k</code>	Integer, the number of folds to use for cross-validation. Must be greater than 2.
<code>X</code>	A design matrix.
<code>Y</code>	A response vector.
<code>mu_0</code>	Initial variational mean (posterior expectation of $\beta_j \mid s_j = 1$ ). If NULL, initialized automatically by <code>get.initials</code> .
<code>omega_0</code>	Initial variational probability (posterior expectation of $s_j$ ). If NULL, initialized automatically by <code>get.initials</code> .
<code>c_pi_0</code>	Prior parameter for pi (beta distribution shape1). If NULL, initialized automatically by <code>get.initials</code> .
<code>d_pi_0</code>	Prior parameter for pi (beta distribution shape2). If NULL, initialized automatically by <code>get.initials</code> .
<code>tau_e</code>	Initial precision of errors. If NULL, initialized automatically by <code>get.initials</code> .
<code>update_order</code>	A numeric vector specifying the order of updates for coefficients. If NULL, initialized automatically by <code>get.initials</code> .
<code>mu_alpha</code>	Mean for the prior on alpha (expansion parameter).
<code>tau_alpha</code>	A numeric vector of <code>tau_alpha</code> values to cross-validate over. Must have at least two values.
<code>tau_b</code>	Initial precision for $\beta_j$ (when $s_j = 1$ ).
<code>standardize</code>	Logical. Center Y, and center and scale X. Default is TRUE.
<code>intercept</code>	Logical. Whether to include an intercept. Default is TRUE. After the model is fit on the centered and scaled data, the final coefficients are "unscaled" to put them back on the original scale of your data. The intercept is then calculated separately using the means and the final coefficients.
<code>max_iter</code>	Maximum number of outer loop iterations for both CV fits and the final fit.
<code>tol</code>	Convergence tolerance for both CV fits and the final fit.
<code>seed</code>	Seed for reproducibility of data splitting and <code>glmnet</code> initials.
<code>verbose</code>	Logical, if TRUE, prints progress messages during cross-validation.
<code>parallel</code>	Logical, if TRUE, search in parallel.

## Details

This function orchestrates the cross-validation process and the final model fit. It first gets initial values for the full dataset, then uses `cv.spexvb` to find the `tau_alpha` that minimizes cross-validation error, and finally calls `spexvb` on the complete dataset with the chosen `tau_alpha`.

**Value**

The final fitted spexvb model, which is a list containing the approximate posterior parameters and convergence information for the full dataset using the optimal tau\_alpha determined by cross-validation.

**See Also**

[cv.spexvb](#), [spexvb](#)

**Examples**

```
# Generate simple synthetic data
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
Y <- X[,1] * 2 + rnorm(n)

# Run cross-validation and fit final model
# (Setting k=3 to keep it quick for the example)
fit <- cv.spexvb.fit(k = 3, X = X, Y = Y)
```

---

get.initials

*Get initial values for spexvb*

---

**Description**

This function initializes parameters for the spexvb model.

**Usage**

```
get.initials(
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  tau_e = NULL,
  update_order = NULL,
  seed = 12376
)
```

**Arguments**

X	A design matrix.
Y	A response vector.
mu_0	Initial mean.
omega_0	Initial omega.
c_pi_0	Initial c_pi.
d_pi_0	Initial d_pi.
tau_e	Initial tau_e.
update_order	Initial update order.
seed	Seed for reproducibility.

**Details****Generate Initial Values for Variational Inference in Sparse Regression**

This helper function estimates initial values for variational parameters such as regression coefficients ( $\mu$ ), spike probabilities ( $\omega$ ), and hyperparameters like  $\tau_e$ ,  $c_{\pi}$ , and  $d_{\pi}$  using LASSO and Ridge regression fits.

**Value**

A list of initialized parameters.

**Examples**

```
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
Y <- X[,1] * 2 + rnorm(n)

initials <- get.initials(X, Y)

# Check estimated noise precision (tau_e)
print(initials$tau_e)
```

---

get.initials.logistic *Get initial values for spexvb*

---

**Description**

This function initializes parameters for the spexvb model.

**Usage**

```
get.initials.logistic(
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  update_order = NULL,
  seed = 12376
)
```

**Arguments**

X	A design matrix.
Y	A response vector.
mu_0	Initial mean.
omega_0	Initial omega.
c_pi_0	Initial c_pi.
d_pi_0	Initial d_pi.
update_order	Initial update order.
seed	Seed for reproducibility.

**Details**

Generate Initial Values for Variational Inference in Sparse Logistic Regression

This helper function estimates initial values for variational parameters such as regression coefficients ( $\mu$ ), spike probabilities ( $\omega$ ), and hyperparameters like  $c_{\pi}$ , and  $d_{\pi}$  using LASSO regression.

**Value**

A list of initialized parameters.

**Examples**

```
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
# Generate binary response
Y <- rbinom(n, 1, plogis(X[,1]))

initials <- get.initials.logistic(X, Y)

# View the initial mu (posterior means)
head(initials$mu_0)
```

---

hspvb *Hierarchical Spike-and-Slab Variational Bayes (HSPVB) for High-Dimensional Linear Regression*

---

### Description

Fits a sparse linear regression model using variational inference with a Gamma prior on the slab precision ( $\tau_b$ ). The model uses spike-and-slab priors where the slab scale is adaptively learned.

### Usage

```
hspvb(
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  a_prior_tau_b = 0.1,
  b_prior_tau_b = 1,
  tau_e = NULL,
  update_order = NULL,
  standardize = TRUE,
  intercept = TRUE,
  max_iter = 1000,
  tol = 1e-05,
  seed = 12376
)
```

### Arguments

X	A numeric matrix. The design matrix (n observations $\times$ p predictors).
Y	A numeric vector. The response vector of length n.
mu_0	Optional numeric vector. Initial variational means for regression coefficients.
omega_0	Optional numeric vector. Initial spike probabilities.
c_pi_0	Optional numeric. Prior Beta(a, b) parameter a for the spike probability $\pi$ .
d_pi_0	Optional numeric. Prior Beta(a, b) parameter b for the spike probability $\pi$ .
a_prior_tau_b	Optional numeric. Gamma prior shape parameter (a) for the slab precision $\tau_b$ . Default is 1.
b_prior_tau_b	Optional numeric. Gamma prior rate parameter (b) for the slab precision $\tau_b$ . Default is 0.01.
tau_e	Optional numeric. Known or estimated error precision $\tau_e$ .
update_order	Optional integer vector. The coordinate update order (0-indexed for C++).
standardize	Logical. Center Y, and center and scale X. Default is TRUE.

intercept	Logical. Whether to include an intercept. Default is TRUE.
max_iter	Maximum number of iterations for the variational update. Default is 1000.
tol	Convergence threshold for entropy change. Default is 1e-5.
seed	Integer seed for initialization, passed to <code>get.initials</code> . Default is 12376.

### Details

This function acts as a wrapper for the C++ implementation of the variational Bayes algorithm with a hierarchical Gamma prior on the slab precision,  $\tau_b$ .

### Value

A list with posterior summaries including estimated coefficients (`mu`), inclusion probabilities (`omega`), final expected slab precision (`tau_b`), intercept (if applicable), convergence status, etc.

### Examples

```
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
Y <- X[,1] * 2 + rnorm(n)
result <- hspvb(X = X, Y = Y)
```

---

spexvb	<i>Parameter Expanded Variational Bayes for Well-Calibrated High-Dimensional Linear Regression with Spike-and-Slab Priors</i>
--------	---

---

### Description

Fits a sparse linear regression model using variational inference with an alpha expansion step. The model uses spike-and-slab priors.

### Usage

```
spexvb(
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  tau_e = NULL,
  update_order = NULL,
  mu_alpha = 1,
  tau_alpha = 1000,
  tau_b = 400,
```

```

    standardize = TRUE,
    intercept = TRUE,
    max_iter = 1000,
    tol = 1e-05,
    seed = 12376
  )

```

### Arguments

<code>X</code>	A numeric matrix. The design matrix ( $n$ observations $\times$ $p$ predictors).
<code>Y</code>	A numeric vector. The response vector of length $n$ .
<code>mu_0</code>	Optional numeric vector. Initial variational means for regression coefficients.
<code>omega_0</code>	Optional numeric vector. Initial spike probabilities.
<code>c_pi_0</code>	Optional numeric. Prior Beta( $a$ , $b$ ) parameter $a$ for the spike probability.
<code>d_pi_0</code>	Optional numeric. Prior Beta( $a$ , $b$ ) parameter $b$ for the spike probability.
<code>tau_e</code>	Optional numeric. Known or estimated error precision.
<code>update_order</code>	Optional integer vector. The coordinate update order (0-indexed for C++).
<code>mu_alpha</code>	Prior mean for alpha. Default is 1.
<code>tau_alpha</code>	Prior precision for alpha. Default is 1000.
<code>tau_b</code>	Slab prior precision. Default is 400.
<code>standardize</code>	Logical. Center $Y$ , and center and scale $X$ . Default is TRUE.
<code>intercept</code>	Logical. Whether to include an intercept. Default is TRUE. After the model is fit on the centered and scaled data, the final coefficients are "unscaled" to put them back on the original scale of your data. The intercept is then calculated separately using the means and the final coefficients.
<code>max_iter</code>	Maximum number of iterations for the variational update. Default is 1000.
<code>tol</code>	Convergence threshold for entropy and alpha change. Default is $1e-5$ .
<code>seed</code>	Integer seed for cross-validation in glmnet. Default is 12376.

### Details

This function acts as a wrapper for a C++ implementations of the SPEVXB algorithm.

### Value

A list with posterior summaries including estimated coefficients ( $\mu$ ), inclusion probabilities ( $\omega$ ), intercept (if applicable), alpha path, convergence status, etc.

### Examples

```

n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
Y <- X[,1] * 2 + rnorm(n)
result <- spexvb(X = X, Y = Y)

```

---

 spexvb.logistic

 Parameter Expanded Variational Bayes for Sparse Logistic Regression
 

---

## Description

Fits a sparse logistic regression model using variational inference with an alpha expansion step.

## Usage

```
spexvb.logistic(
  X,
  Y,
  mu_0 = NULL,
  omega_0 = NULL,
  c_pi_0 = NULL,
  d_pi_0 = NULL,
  update_order = NULL,
  mu_alpha = 1,
  tau_alpha = 10,
  tau_b = 1,
  max_iter = 300,
  tol = 1e-04
)
```

## Arguments

X	A numeric matrix. The design matrix (n observations $\times$ p predictors).
Y	A numeric vector. The response vector (0/1).
mu_0	Optional numeric vector. Initial variational means.
omega_0	Optional numeric vector. Initial spike probabilities.
c_pi_0	Optional numeric. Prior Beta(a, b) parameter a.
d_pi_0	Optional numeric. Prior Beta(a, b) parameter b.
update_order	Optional integer vector. Coordinate update order (0-indexed).
mu_alpha	Prior mean for alpha. Default is 1.
tau_alpha	Prior precision for alpha. Default is 10.
tau_b	Slab prior precision. Default is 1.
max_iter	Maximum iterations. Default is 300.
tol	Convergence tolerance. Default is 1e-4.

## Value

A list with posterior summaries including estimated coefficients (mu), inclusion probabilities (omega), final expected slab precision (tau\_b), intercept (if applicable), convergence status, etc.

**Examples**

```
n <- 50
p <- 100
X <- matrix(rnorm(n * p), n, p)
# Generate binary response
Y <- rbinom(n, 1, plogis(X[,1] * 2))

fit <- spexvb.logistic(X, Y)

# Check convergence
print(fit$converged)
```

# Index

`cv.spexvb`, 2, 6

`cv.spexvb.fit`, 4

`get.initials`, 6

`get.initials.logistic`, 7

`hspvb`, 9

`spexvb`, 6, 10

`spexvb.logistic`, 12