Package ‘mBvs’

August 28, 2018

Type Package

Title Bayesian Variable Selection Methods for Multivariate Data

Version 1.3

Date 2018-8-27

Author Kyu Ha Lee, Mahlet G. Tadesse, Brent A. Coull, Jacqueline R. Starr

Maintainer Kyu Ha Lee <`klee@forsyth.org`>

Description Bayesian variable selection methods for data with multivariate responses and multiple covariates. The package contains implementations of multivariate Bayesian variable selection methods for continuous data and zero-inflated count data.

License GPL (>= 2)

Depends R (>= 3.3.0)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-08-28 07:12:16 UTC

R topics documented:

`mBvs-package` *Bayesian Variable Selection Methods for Multivariate Data*

Description

Bayesian variable selection methods for data with multivariate responses and multiple covariates. The package contains implementations of multivariate Bayesian variable selection methods for continuous data and zero-inflated count data.

Details

The package includes the following function:

`mvnBvs`	Bayesian variable selection for data with multivariate continuous responses
`mzipBvs`	Bayesian variable selection for data with multivariate zero-inflated count responses

Package:	mBvs
Type:	Package
Version:	1.3
Date:	2018-8-27
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

Kyu Ha Lee, Mahlet G. Tadesse, Brent A. Coull, Jacqueline R. Starr
Maintainer: Kyu Ha Lee <klee@forsyth.org>

References

Lee, K. H., Tadesse, M. G., Baccarelli, A. A., Schwartz J., and Coull, B. A. (2016), Multivariate Bayesian variable selection exploiting dependence structure among outcomes: application to air pollution effects on DNA methylation, *Biometrics*, Volume 73, Issue 1, pages 232-241.

Lee, K. H., Coull, B. A., Moscicki, A.-B., Paster, B. J., Starr, J. R. (2018+), Bayesian variable selection for multivariate zero-inflated models: application to microbiome count data, *under review*.

`methods` *Methods for objects of class,* `mvnBvs` *and* `mzipBvs`.

Description

The `mvnBvs` class represents results from Bayesian variable selection using multivariate normal regression models. The `mzipBvs` class represents results from Bayesian variable selection using multivariate zero-inflated regression models.

Usage

```
## S3 method for class 'mvnBvs'
print(x, digits=3, ...)
## S3 method for class 'mzipBvs'
print(x, digits=3, ...)
## S3 method for class 'summ.mvnBvs'
print(x, digits=3, ...)
## S3 method for class 'summ.mzipBvs'
print(x, digits=3, ...)
## S3 method for class 'mvnBvs'
summary(object, digits=3, ...)
## S3 method for class 'mzipBvs'
summary(object, digits=3, ...)
```

Arguments

`x`	an object of class `mvnBvs`, `summ.mvnBvs`, `mzipBvs`, `summ.mzipBvs`.
`digits`	a numeric value indicating the number of digits to display.
`object`	an object of class `mvnBvs` or `mzipBvs`.
`...`	additional arguments.

See Also

`mvnBvs`, `mzipBvs`

`mvnBvs` *The function to perform variable selection for multivariate normal responses*

Description

The function can be used to perform variable selection for multivariate normal responses incorporating not only information on the mean model, but also information on the variance-covariance structure of the outcomes. A multivariate prior is specified on the latent binary selection indicators to incorporate the dependence between outcomes into the variable selection procedure.

Usage

```
mvnBvs(Y, lin.pred, data, model = "unstructured", hyperParams, startValues, mcmcParams)
```

Arguments

`Y`	a data.frame containing q continuous multivariate outcomes from `n` subjects. It is of dimension $n \times q$.
`lin.pred`	a list containing two formula objects: the first formula specifies the p covariates for which variable selection is to be performed; the second formula specifies the confounders to be adjusted for (but on which variable selection is not to be performed) in the regression analysis.
`data`	a data.frame containing the variables named in the formulas in `lin.pred`.
`model`	a character that specifies the covariance structure of the model: either "unstructured" or "factor-analytic".
`hyperParams`	a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, `eta` (a numeric value for the hyperparameter η that regulates the extent to which the correlation between response variables influences the prior of the variable selection indicator), `v` (a numeric vector of length q for the standard deviation hyperparameter v of the regression parameter β prior), `omega` (a numeric vector of length p for the hyperparameter ω in the prior of the variable selection indicator), `beta0` (a numeric vector of length $q+1$ for hyperparameter μ_0 and h_0 in the prior of the intercept β_0), `US` (a list containing numeric vectors for hyperparameters in the unstructured model: `US.Sigma`), `FA` (a list containing numeric vectors for hyperparameters in the factor-analytic model: `lambda` and `sigmaSq`). See Examples below.
`startValues`	a numeric vector containing starting values for model parameters: c(`beta0`, `B`, `gamma`, `Sigma`) for the unstructured model; c(`beta0`, `B`, `gamma`, `sigmaSq`, `lambda`) for the factor-analytic model. See Examples below.
`mcmcParams`	a list containing variables required for MCMC sampling. Components include, `run` (a list containing numeric values for setting the overall run: `numReps`, total number of scans; `thin`, extent of thinning; `burninPerc`, the proportion of burn-in). `tuning` (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings algorithm: `mhProp_beta_var`, variance of the proposal density for B; `mhrho_prop`, degrees of freedom of the inverse-Wishart proposal density for Σ in the unstructured model; `mhPsi_prop`, scale matrix of inverse-Wishart proposal density for Σ in the unstructured model; `mhProp_lambda_var`, variance of the proposal density for λ in the factor-analytic model.) See Examples below.

Value

`mvnBvs` returns an object of class `mvnBvs`.

Author(s)

Kyu Ha Lee, Mahlet G. Tadesse, Brent A. Coull
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Tadesse, M. G., Baccarelli, A. A., Schwartz J., and Coull, B. A. (2016), Multivariate Bayesian variable selection exploiting dependence structure among outcomes: application to air pollution effects on DNA methylation, *Biometrics*, Volume 73, Issue 1, pages 232-241.

Examples

```
# loading a data set
data(simData_cont)
Y <- simData_cont$Y
data <- simData_cont$X
form1 <- as.formula( ~ cov.1+cov.2)
form2 <- as.formula( ~ 1)
lin.pred <- list(form1, form2)

p  <- dim(data)[2]
p_adj  <- 0
q  <- dim(Y)[2]

#####################
## Hyperparameters ##

## Common hyperparameters
##
eta = 0.1
v = rep(10, q)
omega = rep(log(0.5/(1-0.5)), p-p_adj)
common.beta0 <- c(rep(0, q), 10^6)

## Unstructured model
##
rho0  <- q + 4
Psi0 <- diag(3, q)
US.Sigma <- c(rho0, Psi0)

## Factor-analytic model
##
FA.lam <- c(rep(0, q), 10^6)
FA.sigSq <- c(2, 1)

##
hyperParams <- list(eta=eta, v=v, omega=omega, beta0=common.beta0,
US=list(US.Sigma=US.Sigma),
FA=list(lambda=FA.lam, sigmaSq=FA.sigSq))

###################
## MCMC SETTINGS ##

## Setting for the overall run
##
```

```
numReps    <- 50
thin       <- 1
burninPerc <- 0.5

## Tuning parameters for specific updates
##
##  - those common to all models
mhProp_beta_var  <- matrix(0.5, p+p_adj, q)
##
## - those specific to the unstructured model
mhrho_prop <- 1000
mhPsi_prop <- diag(1, q)
##
## - those specific to the factor-analytic model
mhProp_lambda_var  <- 0.5

##
mcmc.US <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                tuning=list(mhProp_beta_var=mhProp_beta_var,
                 mhrho_prop=mhrho_prop, mhPsi_prop=mhPsi_prop))

##
mcmc.FA  <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                tuning=list(mhProp_beta_var=mhProp_beta_var,
                 mhProp_lambda_var=mhProp_lambda_var))

#####################
## Starting Values ##

##  - those common to all models
beta0 <- rep(0, q)
B <- matrix(sample(x=c(0.3, 0), size=q, replace = TRUE), p+p_adj, q)
gamma <- B
gamma[gamma != 0] <- 1
##
## - those specific to the unstructured model
Sigma <- diag(1, q)
##
## - those specific to the factor-analytic model
lambda <- rep(0.5, q)
sigmaSq <- 1

####################################
## Fitting the unstructured model ##
####################################

startValues <- vector("list", 2)

startValues[[1]] <- as.vector(c(beta0, B, gamma, Sigma))

beta0 <- rep(0.2, q)
Sigma <- diag(0.5, q)
```

```
startValues[[2]] <- as.vector(c(beta0, B, gamma, Sigma))

fit.us <- mvnBvs(Y, lin.pred, data, model="unstructured", hyperParams,
startValues, mcmcParams=mcmc.US)

fit.us
summ.fit.us <- summary(fit.us); names(summ.fit.us)
summ.fit.us

#######################################
## Fitting the factor-analytic model ##
#######################################

startValues <- vector("list", 2)

startValues[[1]] <- as.vector(c(beta0, B, gamma, sigmaSq, lambda))

beta0 <- rep(0.2, q)
sigmaSq <- 0.5
startValues[[2]] <- as.vector(c(beta0, B, gamma, sigmaSq, lambda))

fit.fa <- mvnBvs(Y, lin.pred, data, model="factor-analytic", hyperParams,
startValues, mcmcParams=mcmc.FA)

fit.fa
summ.fit.fa <- summary(fit.fa); names(summ.fit.fa)
summ.fit.fa
```

`mzipBvs` *The function to perform variable selection for multivariate zero-inflated count responses*

Description

The function can be used to perform variable selection for multivariate zero-inflated count responses.

Usage

```
mzipBvs(Y, lin.pred, data, model = "generalized", offset = NULL, hyperParams, startValues,
 mcmcParams)
```

Arguments

`Y`	a data.frame containing q count outcomes from `n` subjects. It is of dimension $n \times q$.

`lin.pred`	a list containing three formula objects: the first formula specifies the p_z covariates for which variable selection is to be performed in the binary component of the model; the second formula specifies the p_x covariates for which variable selection is to be performed in the count part of the model; the third formula specifies the p_0 confounders to be adjusted for (but on which variable selection is not to be performed) in the regression analysis.
`data`	a data.frame containing the variables named in the formulas in `lin.pred`.
`model`	a character that specifies the type of model: A generalized multivariate Bayesian variable selection method of Lee et al.(2017+) can be implemented by setting `model="generalized"`. A simpler model that assumes one common variable selection indicator ($\gamma_{j,k} = \delta_{j,k}$) and the same covariance pattern ($R = R_V$) for two model parts can be used by setting `model="restricted1"`. iii) Another simpler model that assumes the same covariance pattern ($R = R_V$) but separate variable selection indicators for the binary and count parts of the model can be implemented by setting `model="restricted2"`.
`offset`	an optional numeric vector with an a priori known component to be included as the linear predictor in the count part of model.
`hyperParams`	a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, `rho0` (degrees of freedom for inverse-Wishart prior for Σ_V), `Psi0` (a scale matrix for inverse-Wishart prior for Σ_V), `mu_alpha0` (hyperparameter μ_{α_0} in the prior of α_0), `mu_alpha` (a numeric vector of length q for hyperparameter μ_α in the prior of α), `mu_beta0` (hyperparameter μ_{β_0} in the prior of β_0), `mu_beta` (a numeric vector of length q for hyperparameter μ_β in the prior of β), `a_alpha0` (hyperparameter a_{α_0} in the prior of $\sigma^2_{\alpha_0}$), `b_alpha0` (hyperparameter b_{α_0} in the prior of $\sigma^2_{\alpha_0}$), `a_alpha` (hyperparameter a_α in the prior of σ^2_α), `b_alpha` (hyperparameter b_α in the prior of σ^2_α), `a_beta0` (hyperparameter a_{β_0} in the prior of $\sigma^2_{\beta_0}$), `b_beta0` (hyperparameter b_{β_0} in the prior of $\sigma^2_{\beta_0}$), `a_beta` (hyperparameter a_β in the prior of σ^2_β), `b_beta` (hyperparameter b_β in the prior of σ^2_β), `v_beta` (a numeric vector of length q for the standard deviation hyperparameter v_β of the regression parameter β prior), `omega_beta` (a numeric vector of length $p_x - p_0$ for the hyperparameter ω_β in the prior of the variable selection indicator), `v_alpha` (a numeric vector of length q for the standard deviation hyperparameter v_α of the regression parameter α prior), `omega_alpha` (a numeric vector of length $p_z - p_0$ for the hyperparameter ω_α in the prior of the variable selection indicator), See Examples below.
`startValues`	a numeric vector containing starting values for model parameters. See Examples below.
`mcmcParams`	a list containing variables required for MCMC sampling. Components include, `run` (a list containing numeric values for setting the overall run: `numReps`, total number of scans; `thin`, extent of thinning; `burninPerc`, the proportion of burn-in). `tuning` (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings algorithm: `beta0.prop.var`, variance of the proposal density for β_0;`beta.prop.var`, variance of the proposal density for B;`alpha.prop.var`, variance of the proposal density for A;`V.prop.var`, variance of the proposal density for V.) See Examples below.

Value

`mzipBvs` returns an object of class `mzipBvs`.

Author(s)

Kyu Ha Lee, Brent A. Coull, Jacqueline R. Starr
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Coull, B. A., Moscicki, A.-B., Paster, B. J., Starr, J. R. (2018+), Bayesian variable selection for multivariate zero-inflated models: application to microbiome count data, *under review*.

Examples

```
## Not run:
# loading a data set
data(simData_mzip)
Y <- simData_mzip$Y
data <- simData_mzip$X

n = dim(Y)[1]
q = dim(Y)[2]

form.bin     <- as.formula(~cov.1)
form.count    <- as.formula(~cov.1)
form.adj    <- as.formula(~1)
lin.pred <- list(form.bin, form.count, form.adj)

Xmat0 <- model.frame(lin.pred[[1]], data=data)
Xmat1 <- model.frame(lin.pred[[2]], data=data)
Xmat_adj <- model.frame(lin.pred[[3]], data=data)

p_adj = ncol(Xmat_adj)
p0 <- ncol(Xmat0) + p_adj
p1 <- ncol(Xmat1) + p_adj

nonz <- rep(NA, q)
for(j in 1:q) nonz[j] <- sum(Y[,j] != 0)

#####################
## Hyperparameters ##

## Generalized model
##
rho0     <- q + 3 + 1
Psi0    <- diag(3, q)
```

```
mu_alpha0     <- 0
mu_alpha    <- rep(0, q)

mu_beta0    <- 0
mu_beta        <- rep(0, q)

a_alpha0    <- 0.7
b_alpha0     <- 0.7

a_alpha        <- rep(0.7, p0)
b_alpha     <- rep(0.7, p0)

a_beta0        <- 0.7
b_beta0     <- 0.7

a_beta        <- rep(0.7, p1)
b_beta         <- rep(0.7, p1)

v_beta = rep(1, q)
omega_beta = rep(0.1, p1-p_adj)
v_alpha = rep(1, q)
omega_alpha = rep(0.1, p0-p_adj)

##
hyperParams.gen <- list(rho0=rho0, Psi0=Psi0, mu_alpha0=mu_alpha0, mu_alpha=mu_alpha,
mu_beta0=mu_beta0, mu_beta=mu_beta, a_alpha0=a_alpha0, b_alpha0=b_alpha0,
a_alpha=a_alpha, b_alpha=b_alpha, a_beta0=a_beta0, b_beta0=b_beta0,
a_beta=a_beta, b_beta=b_beta, v_beta=v_beta, omega_beta=omega_beta,
v_alpha=v_alpha, omega_alpha=omega_alpha)

###################
## MCMC SETTINGS ##

## Setting for the overall run
##
numReps    <- 100
thin       <- 1
burninPerc <- 0.5

## Settings for storage
##
storeV      <-    TRUE
storeW      <-    TRUE

## Tuning parameters for specific updates
##
##  - Generalized model
beta0.prop.var    <- 0.5
alpha.prop.var    <- 0.5
beta.prop.var    <- 0.5
V.prop.var    <- 0.05

##
```

```
mcmc.gen <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
storage=list(storeV=storeV, storeW=storeW),
tuning=list(beta0.prop.var=beta0.prop.var, alpha.prop.var=alpha.prop.var,
beta.prop.var=beta.prop.var, V.prop.var=V.prop.var))

#####################
## Starting Values ##

## Generalized model
##
B <- matrix(0.1, p1, q, byrow = T)
A <- matrix(0.1, p0, q, byrow = T)

V <- matrix(rnorm(n*q, 0, 0.1), n, q)
W <- matrix(rnorm(n*q, 0, 0.1), n, q)

beta0 <- log(as.vector(apply(Y, 2, mean)))
alpha0 <- log(nonz/n / ((n-nonz)/n))

Sigma_V    <- matrix(0, q, q)
diag(Sigma_V) <- 1

R        <- matrix(0, q, q)
diag(R) <- 1

sigSq_alpha0 <- 1
sigSq_alpha <- rep(1, p0)
sigSq_beta0 <- 1
sigSq_beta <- rep(1, p1)

startValues.gen <- vector("list", 2)
startValues.gen[[1]] <- list(B=B, A=A, V=V, W=W, beta0=beta0, alpha0=alpha0, R=R,
sigSq_alpha0=sigSq_alpha0,
sigSq_alpha=sigSq_alpha, sigSq_beta0=sigSq_beta0, sigSq_beta=sigSq_beta, Sigma_V=Sigma_V)

B <- matrix(-0.1, p1, q, byrow = T)
A <- matrix(-0.1, p0, q, byrow = T)

V <- matrix(rnorm(n*q, 0, 0.1), n, q)
W <- matrix(rnorm(n*q, 0, 0.1), n, q)

Sigma_V    <- matrix(0.1, q, q)
diag(Sigma_V) <- 1.1

startValues.gen[[2]] <- list(B=B, A=A, V=V, W=W, beta0=beta0, alpha0=alpha0, R=R,
sigSq_alpha0=sigSq_alpha0,
sigSq_alpha=sigSq_alpha, sigSq_beta0=sigSq_beta0, sigSq_beta=sigSq_beta, Sigma_V=Sigma_V)

###################################
## Fitting the generalized model ##
###################################
fit.gen <- mzipBvs(Y, lin.pred, data, model="generalized", offset=NULL, hyperParams.gen,
startValues.gen, mcmc.gen)
```

```
print(fit.gen)
summ.fit.gen <- summary(fit.gen); names(summ.fit.gen)
summ.fit.gen


## End(Not run)
```

`simData_cont` *A simulated data set containing multivariate normal responses and continuous covariates*

Description

A simulated data set containing multivariate normal responses and continuous covariates

Usage

```
data("simData_cont")
```

Format

a list of two data frame objects. Components include,

`Y` a data frame for 10 multivariate normal responses from 100 observations: `Y.1`-`Y.10`

`X` a data frame for 2 continuous covariates from 100 observations: `cov.1`-`cov.2`

Examples

```
data(simData_cont)
```

`simData_mzip` *A simulated data set containing multivariate zero-inflated count responses and a continuous covariate*

Description

A simulated data set containing multivariate zero-inflated count responses and a continuous covariate

Usage

```
data("simData_mzip")
```

Format

a list of two data frame objects. Components include,

- `Y` a data frame for 10 multivariate count responses from 300 observations: `Y.1`-`Y.10`
- `X` a data frame for a single continuous covariate from 300 observations: `cov.1`

Examples

```
data(simData_mzip)
```

Index