Package 'icmm'

October 12, 2017

Type Package

Title Empirical Bayes Variable Selection via ICM/M Algorithm

Version 1.1

Author Vitara Pungpapong [aut, cre],
Min Zhang [aut],
Dabao Zhang [aut]

Maintainer Vitara Pungpapong <vitara@cbs.chula.ac.th>

Description Carries out empirical Bayes variable selection via ICM/M algorithm. The basic problem is to fit high-dimensional regression which most coefficients are assumed to be zero. This package allows incorporating the Ising prior to capture structure of predictors in the modeling process. The current version of this package can handle the normal, binary logistic, and Cox's regression (Pungpapong et. al. (2015) <doi:10.1214/15-EJS1034>, Pungpapong et. al. (2017) <arXiv:1707.08298>).

License GPL (>= 2)

Imports EbayesThresh

Suggests MASS, stats

LazyData TRUE

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-12 03:17:25 UTC

R topics documented:

`icmm-package` *Empirical Bayes Variable Selection via ICM/M*

Description

Carries out empirical Bayes variable selection via ICM/M algorithm. The basic problem is to fit a high-dimensional regression which most of the coefficients are assumed to be zero. This package allows incorporating the Ising prior to capture structure of predictors in the modeling process. The current version of this package can handle the normal, binary logistic, and Cox's regression.

Details

Package:	icmm
Type:	Package
Version:	1.1
Date:	2017-10-11
License:	GPL-2
LazyLoad:	yes

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Maintainer: Vitara Pungpapong <vitara@cbs.chula.ac.th>

References

Pungpapong, V., Zhang, M. and Zhang, D. (2015). Selecting massive variables using an iterated conditional modes/medians algorithm. Electronic Journal of Statistics. 9:1243-1266. <doi:10.1214/15-

EJS1034>.
Pungpapong, V., Zhang, M. and Zhang, D. (2017). Variable selection for high-dimensional generalized linear models using an iterated conditional modes/medians algorithm. Preprint <arXiv:1707.08298>.

`get.ab` *Hyperparameter estimation for* `a` *and* `b`.

Description

This function estimates the hyperparameters `a` and `b` for the Ising prior. This function is for internal use called by the `icmm` function.

Usage

```
get.ab(beta, structure, edgeind)
```

Arguments

`beta`	a (p*1) matrix of regression coefficients.
`structure`	a data frame stores the information of structure among predictors.
`edgeind`	a vector stores primary keys of `structure`.

Details

Estimate hyperparameters, `a` and `b`, using maximum pseudolikelihood estimators.

Value

Return a two-dimensional vector where the fist element is `a` and the second element is `b`.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
data(linearrelation)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
# Suppose obtain beta from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
edgeind<-sort(unique(linearrelation[,1]))
hyperparameter<-get.ab(beta=beta, structure=linearrelation, edgeind=edgeind)
```

get.alpha *Hyperparameter estimation for* alpha.

Description

This function estimates a hyperparameter alpha, a scale parameter in Laplace denstiy. This function is for internal use called by the icmm function.

Usage

```
get.alpha(beta, scaledfactor)
```

Arguments

beta	a (p*1) matrix of regression coefficients.
scaledfactor	a scalar value of multiplicative factor.

Details

This function estimates a hyperparameter alpha, a scale parameter in Laplace density as the mode of its full conditional distribution function.

Value

Return a scalar value of alpha.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
# Obtain initial values of beta from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
# Initiate alpha
alpha<-0.5
# Estimate sigma
e<-Y-X%*%beta
nz<-sum(beta[,1]!=0)
sigma<-get.sigma(Y=Y, X=X, beta=beta, alpha=alpha)
# Update alpha as the mode of its full conditional distribution function
alpha<-get.alpha(beta=beta, scaledfactor=1/(sqrt(n-1)*sum(abs(beta))/sigma))
```

get.beta *Obtain model coefficient without assuming prior on structure of predictors.*

Description

Given a sufficient statistic for a regression coefficient, this funciton estimates a regression coefficient without assuming prior on structure of predictors.

Usage

```
get.beta(SS, w, alpha, scaledfactor)
```

Arguments

SS	a scalar value of sufficient statistic for a regression coefficient.
w	a scalar value of mixing weight.
alpha	a scalar value of hyperparameter alpha.
scaledfactor	a scalar value of multiplicative factor.

Details

Empirical Bayes thresholding is employed to obtain a posterior median of a regression coefficient.

Value

a scalar value of regression coefficient.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
# Obtain initial values from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
# Initiate all other parameters
w<-0.5
alpha<-0.5
sigma<-get.sigma(Y=Y, X=X, beta=beta, alpha=alpha)
# Obtain a sufficient statistic
j<-1
Yres<-Y-X%*%beta+X[,j]*beta[j,1]
```

```
sxy<-t(Yres)%*%X[,j]
ssx<-sum(X[,j]^2)
SS<-sqrt(n-1)*sxy/(sigma*ssx)
beta[j,1]<-get.beta(SS=SS, w=w, alpha=alpha, scaledfactor=sigma/sqrt(n-1))
```

`get.beta.ising` *Obtain a regression coefficient when assuming Ising prior (with structured predictors).*

Description

Given a sufficient statistic for a regression coefficient, this function estimates a coefficient when assuming the Ising model to incorporate the prior of structured predictors.

Usage

```
get.beta.ising(SS, wpost, alpha, scaledfactor)
```

Arguments

`SS`	a sufficient statistic for a regression coefficient.
`wpost`	a posterior probability of mixing weight.
`alpha`	a scalar value for hyperparameter `alpha`.
`scaledfactor`	a scalar value for multiplicative factor.

Details

Given a posterior probability of mixing weight, empirical Bayes thresholding is employed to obtain a posterior median of a regression coefficient.

Value

a scalar value of regression coefficient.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
data(linearrelation)
edgeind<-sort(unique(linearrelation[,1]))
# Obtain initial values from lasso
data(initbetaGaussian)
```

```
beta<-as.matrix(initbetaGaussian)
# Initiate all other parameters
alpha<-0.5
sigma<-get.sigma(Y=Y, X=X, beta=beta, alpha=alpha)
hyperparam<-get.ab(beta, linearrelation, edgeind)
# Obtain regression coefficient
j<-1
Yres<-Y-X%*%beta+X[,j]*beta[j,1]
sxy<-t(Yres)%*%X[,j]
ssx<-sum(X[,j]^2)
SS<-sqrt(n-1)*sxy/(sigma*ssx)
wpost<-get.wpost(SS, beta, alpha, hyperparam, linearrelation, edgeind, j)
beta[j,1]<-get.beta.ising(SS=SS, wpost=wpost, alpha=alpha,
           scaledfactor=sigma/sqrt(n-1))
```

`get.pseudodata.binomial` *Obtain pseudodata based on the binary logistic regression model.*

Description

For logistic regression, given the current estimates of regression coefficients, working responses and their corresponding weights are obtained.

Usage

```
get.pseudodata.binomial(Y, X, beta0, beta, niter)
```

Arguments

`Y`	an (n*1) numeric matrix of responses.
`X`	an (n*p) numeric design matrix.
`beta0`	a scalar value of intercept term.
`beta`	a (p*1) matrix of regression coefficients.
`niter`	number of iterations in ICM/M algorithm.

Value

Return a list including elements

`z`	an (n*1) matrix of working responses
`sigma2`	an (n*1) matrix of inverse of weights.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simBinomial)
Y<-as.matrix(simBinomial[,1])
X<-as.matrix(simBinomial[,-1])
p<-dim(X)[2]
# Obtain initial values from lasso
data(initbetaBinomial)
initbeta<-as.matrix(initbetaBinomial)
# Get Pseudodata
pseudodata<-get.pseudodata.binomial(Y=Y, X=X, beta0=0, beta=initbeta, niter=1)
z<-pseudodata$z
sigma<-sqrt(pseudodata$sigma2)
```

`get.pseudodata.cox` *Obtain pseudodata based on the Cox's regression model.*

Description

For Cox's regression model, given the current estimates of regression coefficients, working responses and their corresponding weights are obtained.

Usage

```
get.pseudodata.cox(Y, X, event, beta, time, ntime, sumevent)
```

Arguments

`Y`	an (n*1) numeric matrix of time response.
`X`	an (n*p) numeric design matrix.
`event`	an (n*1) numeric matrix of status: of status indicator: `0`=right censored, `1`=event at time.
`beta`	a (p*1) matrix of regression coefficients.
`time`	a vector or sorted value of `Y`.
`ntime`	length of the vector `time`.
`sumevent`	a vector of size `ntime` where each element is the sum of event where `Y` is equal to each value in `time`.

Value

Return a list including elements

`z`	an (n*1) matrix of working responses
`sigma2`	an (n*1) matrix of inverse of weights.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simCox)
Y<-as.matrix(simCox[,1])
event<-as.matrix(simCox[,2])
X<-as.matrix(simCox[,-(1:2)])
time<-sort(unique(Y))
ntime<-length(time)
# sum of event_i where y_i =time_k
sumevent<-rep(0, ntime)
for(j in 1:ntime)
{
  sumevent[j]<-sum(event[Y[,1]==time[j]])
}
# Obtain initial values from lasso
data(initbetaCox)
initbeta<-as.matrix(initbetaCox)
# Get Pseudodata
pseudodata<-get.pseudodata.cox(Y, X, event, initbeta, time, ntime, sumevent)
z<-pseudodata$z
sigma<-sqrt(pseudodata$sigma2)
```

`get.sigma` *Standard deviation estimation.*

Description

This function estimates the standard deviation when `family="gaussian"`. This function is for internal use called by the `icmm` function.

Usage

```
get.sigma(Y, X, beta, alpha)
```

Arguments

`Y`	an (n*1) numeric matrix of responses.
`X`	an (n*p) numeric design matrix.
`beta`	a (p*1) matrix of regression coefficients.
`alpha`	a scalar value of hyperparmeter `alpha`.

Details

Estimate standard deviation as the mode of its full conditional distribution function when specify `family="gaussian"`. This function is for internal use called by the `icmm` function.

Value

Return a scalar value of standard deviation.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
alpha<-0.5
# Obtain initial values from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
# Obtain sigma
sigma<-get.sigma(Y=Y, X=X, beta=beta, alpha=alpha)
```

---

get.wpost *Estimate posterior probability of mixing weight.*

Description

With the Ising prior on structured predictors, this function gets the posterior probability of mixing weight.

Usage

```
get.wpost(SS, beta, alpha, hyperparam, structure, edgeind, j)
```

Arguments

SS	a scalar value of sufficient statistic for regression coefficient.
beta	a (p*1) matrix of regression coefficients.
alpha	a scalar value of hyperparameter alpha.
hyperparam	a two-dimensional vector of hyperparameters a and b.
structure	a data frame stores the information of structure among predictors.
edgeind	a vector stores primary keys of structure.
j	an index ranges from 1 to p. This function estimates a posterior probability of a mixing weight corresponding to predictor j.

Details

With the Ising prior on structured predictors, the problem is transformed into the realm of empirical Bayes thresholding with Laplace prior by estimating the posterior probability of mixing weight. The posterior probability is used to find the posterior median of a regression coefficient.

Value

Return a scalar value of a posterior probability of mixing weight for predictor.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
data(linearrelation)
edgeind<-sort(unique(linearrelation[,1]))
# Obtain initial values from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
# Initiate all other parameters
alpha<-0.5
sigma<-get.sigma(Y=Y, X=X, beta=beta, alpha=alpha)
hyperparam<-get.ab(beta, linearrelation, edgeind)
# Estimate the posterior probability of first predictor
j<-1
Yres<-Y-X%*%beta+X[,j]*beta[j,1]
sxy<-t(Yres)%*%X[,j]
ssx<-sum(X[,j]^2)
SS<-sqrt(n-1)*sxy/(sigma*ssx)
wpost<-get.wpost(SS=SS, beta=beta, alpha=alpha, hyperparam=hyperparam,
       structure=linearrelation, edgeind=edgeind, j=j)
```

`get.wprior` *Mixing weight estimation.*

Description

Given other parameters, this function estimates a mixing weight from the mode of its full conditional distribution function.

Usage

```
get.wprior(beta)
```

Arguments

`beta`	a (p*1) matrix of regression coefficients.

Details

Given other parameters, this function estimates a mixing weight from the mode of its full conditional distribution function. This function is called when use the independent prior of predictors (no prior on structured predictors).

Value

Return a scalar value of a mixing weight.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
# Obtain initial values from lasso
data(initbetaGaussian)
beta<-as.matrix(initbetaGaussian)
# Estimate the mixing weight
w<-get.wprior(beta)
```

---

get.zeta *Local posterior probability estimation*

Description

This function estimates the local posterior probability when assuming no prior on structured predictors.

Usage

```
get.zeta(SS, w, alpha)
```

Arguments

SS	a scalar value of sufficient statistic for regression coefficient.
w	a scalar value of mixing weight.
alpha	a scalar value of hyperparameter alpha.

Details

Given all other parameters, this function estimates the local posterior probability or the probability that a regression coefficient is not zero conditional on other parameters. This function is called when assuming no prior on structured predictors.

Value

Return a scalar value of local posterior probability.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
# Obtain initial values from lasso
data(initbetaGaussian)
initbeta<-as.matrix(initbetaGaussian)
# Obtain the final output from ebvs
output<-icmm(Y, X, b0.start=0, b.start=initbeta, family = "gaussian",
        ising.prior = FALSE, estalpha = FALSE, alpha = 0.5, maxiter = 100)
b0<-output$coef[1]
beta<-matrix(output$coef[-1], ncol=1)
# Get all parameters for function arguments
w<-get.wprior(beta)
alpha<-0.5
sigma<-get.sigma(Y,X,beta,alpha)
# Estimate local posterior probability
j<-1
Yres<-Y-b0-X%*%beta+X[,j]*beta[j,1]
sxy<-t(Yres)%*%X[,j]
ssx<-sum(X[,j]^2)
SS<-sqrt(n-1)*sxy/(sigma*ssx)
zeta<-get.zeta(SS=SS, w=w, alpha=alpha)
```

---

get.zeta.ising *Local posterior probability estimation.*

Description

This function estimates the local posterior probability when assuming Ising prior on structured predictors.

Usage

```
get.zeta.ising(SS, beta, alpha, hyperparam, structure, edgeind, j)
```

Arguments

SS	a scalar value of sufficient statistic for regression coefficient.
beta	a (p*1) matrix of regression coefficients.
alpha	a scalar value of hyperparameter alpha.
hyperparam	a two-dimensional vector of hyperparameters a and b.
structure	a data frame stores the information of structure among predictors.

edgeind	a vector stores primary keys of structure.
j	an index ranges from 1 to p. This function estimate a local posterior probability of predictor j.

Details

Given all other parameters, this function estimates the local posterior probability or the probability that a regression coefficient is not zero conditional on other par ameters. This function is called when assuming Ising prior on structured predictors.

Value

Return a scalar value of local posterior probability.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

Examples

```
data(simGaussian)
data(linearrelation)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
n<-dim(X)[1]
# Obtain initial values from lasso
data(initbetaGaussian)
initbeta<-as.matrix(initbetaGaussian)
# Get final output from ebvs
output<-icmm(Y, X, b0.start=0, b.start=initbeta, family = "gaussian",
        ising.prior = TRUE, structure=linearrelation, estalpha = FALSE,
        alpha = 0.5, maxiter = 100)
b0<-output$coef[1]
beta<-matrix(output$coef[-1], ncol=1)
# Get all parameters for function arguments
w<-get.wprior(beta)
alpha<-0.5
sigma<-get.sigma(Y,X,beta,alpha)
edgeind<-sort(unique(linearrelation[,1]))
hyperparam<-get.ab(beta, linearrelation, edgeind)
# Estimate local posterior probability
j<-1
Yres<-Y-b0-X%*%beta+X[,j]*beta[j,1]
sxy<-t(Yres)%*%X[,j]
ssx<-sum(X[,j]^2)
SS<-sqrt(n-1)*sxy/(sigma*ssx)
zeta<-get.zeta.ising(SS=SS, beta=beta, alpha=alpha, hyperparam=hyperparam,
      structure=linearrelation, edgeind=edgeind, j=j)
```

icmm *Empirical Bayes Variable Selection*

Description

Empirical Bayes variable selection via the ICM/M algorithm.

Usage

```
icmm(Y, X, event, b0.start, b.start, family = "gaussian",
     ising.prior = FALSE, structure, estalpha = FALSE,
     alpha = 0.5, maxiter = 100)
```

Arguments

Y	an (n*1) numeric matrix of responses.
X	an (n*p) numeric design matrix.
event	an (n*1) numeric matrix of status for censored data: 0=censored data, 1=event at time. event is required when family="cox".
b0.start	a starting value of intercept term (optional).
b.start	a (p*1) matrix of starting values for regression coefficients.
family	specification of the model. It can be one of these three models: "gaussian", "binomial", "cox". The default is family="gaussian".
ising.prior	a logical flag for Ising prior utilization. ising.prior=FALSE assumes no prior on structure among predictors. ising.prior=TRUE indicates incorporation of Ising prior to capture structure among predictors in modeling process.
structure	a data frame stores the information of structured predictors (need to specify when ising.prior=TRUE).
estalpha	a logical flag specifying whether to obtain alpha via ICM/M algorithm.
alpha	a scalar value of scale parameter in Laplace density (non-zero part of prior). The default value is 0.5.
maxiter	a maximum values of iterations for ICM/M algorithm.

Details

The main function for empirical Bayes variable selection. Iterative conditional modes/medians (ICM/M) is implemented in this function. The basic problem is to estimate regression coefficients in high-dimensional data (i.e., large p small n) and we assume that most coefficients are zero. This function also allows the prior of structure of covariates to be incorporated in the model.

Value

Return a list including elements

coef	a vector of model coefficients. The first element is an intercept term when specifying family="gaussian" or family="binomial".
iterations	number of iterations of ICM/M.
alpha	a scalar value of alpha.
postprob	a p-vector of local posterior probabilities or zeta.

Author(s)

Vitara Pungpapong, Min Zhang, Dabao Zhang

References

Pungpapong, V., Zhang, M. and Zhang, D. (2015). Selecting massive variables using an iterated conditional modes/medians algorithm. Electronic Journal of Statistics. 9:1243-1266. <doi:10.1214/15-EJS1034>.
Pungpapong, V., Zhang, M. and Zhang, D. (2017). Variable selection for high-dimensional generatlized linear models using an iterated conditional modes/medians algorithm. Preprint <arXiv:1707.08298>.

See Also

get.ab, get.alpha, get.beta, get.beta.ising, get.pseudodata.binomial, get.pseudodata.cox, get.sigma, get.wprior, get.zeta, get.zeta.ising

Examples

```
# Normal linear regression model
# With no prior on structure among predictors
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
# Obtain initial values from lasso
data(initbetaGaussian)
initbeta<-as.matrix(initbetaGaussian)
result<-icmm(Y=Y, X=X, b.start=initbeta, family="gaussian",
        ising.prior=FALSE, estalpha=FALSE, alpha=0.5, maxiter=100)
result$coef
result$iterations
result$alpha
result$wpost

# With prior on structure among predictors
data(linearrelation)
result<-icmm(Y=Y, X=X, b.start=initbeta, family="gaussian",
        ising.prior=TRUE, structure=linearrelation,
        estalpha=FALSE, alpha=0.5, maxiter=100)
result$coef
result$iterations
```

```
result$alpha
result$wpost

# Binary logistic regression model
data(simBinomial)
Y<-as.matrix(simBinomial[,1])
X<-as.matrix(simBinomial[,-1])
p<-dim(X)[2]
# Obtain initial values from lasso
data(initbetaBinomial)
initbeta<-as.matrix(initbetaBinomial)
result<-icmm(Y=Y, X=X, b0.start=0, b.start=initbeta, family="binomial",
        ising.prior=TRUE, structure=linearrelation, estalpha=FALSE,
        alpha=0.5, maxiter=100)
result$coef
result$iterations
result$alpha
result$wpost

# Cox's model
data(simCox)
Y<-as.matrix(simCox[,1])
event<-as.matrix(simCox[,2])
X<-as.matrix(simCox[,-(1:2)])
# Obtain initial values from lasso
data(initbetaCox)
initbeta<-as.matrix(initbetaCox)
result <- icmm(Y=Y, X=X, event=event, b.start=initbeta, family="cox",
          ising.prior=TRUE, structure=linearrelation, estalpha=FALSE,
          alpha=0.5, maxiter=100)
result$coef
result$iterations
result$alpha
result$wpost
```

`initbetaBinomial` *Initial values for the regression coefficients used in example for running ICM/M algorithm in binary logistic model*

Description

Initial values for the regression coefficients obtained from binary logistic model with lasso regularization for `simBinomial` data set.

Usage

```
data(initbetaBinomial)
```

Format

A data frame with 400 rows.

V1 a numeric vector of the regression coefficients.

Examples

```
data(initbetaBinomial)
```

`initbetaCox` *Initial values for the regression coefficients used in example for running ICM/M algorithm in Cox's model*

Description

Initial values for the regression coefficients obtained from Cox's model with lasso regularization for `simCox` data set.

Usage

```
data(initbetaCox)
```

Format

A data frame with 400 rows.

V1 a numeric vector of the regression coefficients.

Examples

```
data(initbetaCox)
```

`initbetaGaussian` *Initial values for the regression coefficients used in example for running ICM/M algorithm in normal linear regression model*

Description

Initial values for the regression coefficients obtained from normal linear regression model with lasso regularization for `simGaussian` data set.

Usage

```
data(initbetaGaussian)
```

Format

A data frame with 400 rows.

`V1` a numeric vector of the regression coefficients.

Examples

```
data(initbetaGaussian)
```

---

linearrelation *Linear structure of predictors*

Description

This data frame is used as an example to store the structure of predictors or the edge set of an undirected graph. For this data frame, the linear chain is assumed for each predictor.

Usage

```
data(linearrelation)
```

Format

A data frame with 400 observations and 2 variables as follows.

`Index` an index of the predictor/node which has at least one edge.

`EdgeIndices` a string of all indices having an edge connected to `Index` separated by semicolon(;).

Details

This structure of predictors assumes a linear chain for each predictor which its immediate neighbors. For example, j-predictor is connected to (j-1)-predictor and (j+1)-predictor. The example for the entry in the data frame is `Index="5"` and `EdgeIndices="4;6"`.

Examples

```
data(linearrelation)
# To see the format of linearrelation data frame
head(linearrelation)
```

simBinomial *Simulated data from the binary logistic regression model*

Description

Simulated data from the binary logistic regression model. A data frame with 100 observations and 401 variables. The included variables are
`V1` A numeric vector of binary responses where each entry is either `0` or `1`.
`V2-V401` 400 vectors of covariates.

Usage

```
data(simBinomial)
```

Format

A data frame of simulated data from the binary logistic regression with 100 observations and 401 variables.

Examples

```
data(simBinomial)
Y<-as.matrix(simBinomial[,1])
X<-as.matrix(simBinomial[,-1])
```

simCox *Simulated data from Cox's regression model*

Description

Simulated data from Cox's regression model. A data frame with 100 observations and 402 variables. The included variables are
`V1` A numeric vector of responses for right censored data.
`V2` A numeric vector of status indicator: `0`=right censored, `1`=event at time `V1`.
`V3-V402` 400 vectors of covariates.

Usage

```
data(simCox)
```

Format

A data frame of simulated data from Cox's regression model with 100 observations and 402 variables.

Examples

```
data(simCox)
Y<-as.matrix(simCox[,1])
event<-as.matrix(simCox[,2])
X<-as.matrix(simCox[,-(1:2)])
```

---

simGaussian *Simulated data from the normal linear regression model*

Description

Simulated data from the normal linear regression model. A data frame with 100 observations and 401 variables. The included variables are
`V1` A numeric vector of responses.
`V2-V401` 400 vectors of covariates.

Usage

```
data(simGaussian)
```

Format

A data frame of simulated data from the normal linear regression with 100 observations and 401 variables.

Examples

```
data(simGaussian)
Y<-as.matrix(simGaussian[,1])
X<-as.matrix(simGaussian[,-1])
```

Index