Package ‘SiER’

October 12, 2022

Type Package

Title Signal Extraction Approach for Sparse Multivariate Response Regression

Version 0.1.0

Author Ruiyan, Xin Qi

Maintainer Ruiyan Luo <rluo@gsu.edu>

Description
Methods for regression with high-dimensional predictors and univariate or maltivariate response variables. It considers the decomposition of the coefficient matrix that leads to the best approximation to the signal part in the response given any rank, and estimates the decomposition by solving a penalized generalized eigenvalue problem followed by a least squares procedure. Ruiyan Luo and Xin Qi (2017) <doi:10.1016/j.jmva.2016.09.005>.

License GPL-2

LazyData TRUE

Suggests MASS

RoxygenNote 6.0.1.9000

NeedsCompilation no

Repository CRAN

Date/Publication 2017-09-19 08:50:24 UTC

R topics documented:

cv.SiER *Cross-validation for high-dimensional multivariate regression*

Description

Conduct the cross-validation and build the final model for the following high dimensional multivariate regression model:

$$Y = \mu + X\beta + \epsilon,$$

where Y is the $n \times q$ response matrix with $q \geq 1$, X is the $n \times p$ predictor matrix, and ϵ is the noise matrix. The coefficient matrix β is $p \times q$ and μ is the intercept. The number of predictors p can be much larger than the sample size n. The response is univariate if $q = 1$ and multivariate if $q > 1$.

Usage

```
cv.SiER(X, Y, K.cv = 5, upper.comp = 10, thresh = 0.01)
```

Arguments

`X`	the $n \times p$ predictor matrix.
`Y`	the $n \times q$ response matrix, where $q \geq 1$ is the number of response variables.
`K.cv`	the number of CV sets. Default is 5.
`upper.comp`	the upper bound for the maximum number of components to be calculated. Default is 10.
`thresh`	a number between 0 and 1 specifying the minimum proportion of variation to be explained by each selected component relative to all the selected components. It is used to determine the maximum number of components to be calculated in the CV procedure. The optimal number of components will be selected from the integers from 1 to the minimum of upper.comp and this maximum number. A smaller thresh leads to a larger maximum number of components and a longer running time. A larger thresh value leads to a smaller running time, but may miss some important components and lead to a larger prediction error. Default is 0.01.

Details

Based on the best rank K approximation to $X\beta$, the coefficient matrix has decomposition $\beta = \sum \alpha_k w_k^T$, where α_k is the vector so that $X\alpha_k$ has the maximum correlation with Y under the restriction that $X\alpha_k$ has unit variance and is uncorrelated with $X\alpha_1$,..., $X\alpha_{k-1}$. We estimate α_k by solving a penalized generalized eigenvalue problem with penalty $\tau||\alpha_k||_\lambda^2$ where $||\alpha_k||_\lambda^2 = (1-\lambda)||\alpha_k||_2^2 + \lambda||\alpha_k||_1^2$ is a mixture of the squared l_2 and squared l_1 norms. The w_k is estimated by regressing Y on $X\alpha_k$.

Value

A fitted CV-object, which is used in the function pred.SiER() for prediction and getcoef.SiER() for extracting the estimated coefficient matrix.

mu	the estimated intercept vector.
beta	the estimated slope coefficient matrix.
min.error	minimum CV error.
scale.x	the maximum absolute value of X used to scale X.
X	the input X.
Y	the input Y.
params.set	a 9*2 matrix specifying the set of values of tau and $lambda$ used in CV.
error	a list for CV errors.
opt.K	optimal number of components to be selected.
opt.tau	optimal value for tau.
opt.lambda	optimal value for $lambda$.

Author(s)

Ruiyan Luo and Xin Qi

References

Ruiyan Luo and Xin Qi (2017) Signal extraction approach for sparse multivariate response regression, Journal of Multivariate Statistics. 153: 83-97.

Examples

```
# q=1
library(MASS)
nvar=100
nvarq <- 1
sigmaY <- 0.1
sigmaX=0.1
nvar.eff=15
rho <- 0.3
Sigma=matrix(0,nvar.eff,nvar.eff)
for(i in 1:nvar.eff){
    for(j in 1:nvar.eff){
        Sigma[i,j]=rho^(abs(i-j))
    }
}

betas.true <- matrix(0, nvar, 1)
betas.true[1:15,1]=rep(1,15)/sqrt(15)

ntest <- 100
ntrain <- 90
ntot <- ntest+ntrain
```

```
X <- matrix(0,ntot,nvar)
X[,1:nvar.eff] <- mvrnorm(n=ntot, rep(0, nvar.eff), Sigma)
X[,-(1:nvar.eff)] <- matrix(sigmaX*rnorm((nvar-nvar.eff)*dim(X)[1]),
                            dim(X)[1],(nvar-nvar.eff))
Y <- X%*%betas.true
Y <- Y+rnorm(ntot, 0, sigmaY)

X.train <- X[1:ntrain,]
Y.train <- Y[1:ntrain,]
X.test <- X[-(1:ntrain),]
Y.test <- Y[-(1:ntrain),]

cv.fit <- cv.SiER(X.train,Y.train, K.cv=5)

Y.pred=pred.SiER(cv.fit, X.test)
error=sum((Y.pred-Y.test)^2)/ntest
print(c("predict error=", error))
coefs=getcoef.SiER(cv.fit)


#q>1
library(MASS)
total.noise <- 0.1
rho <- 0.3
rho.e <- 0.2
nvar=500
nvarq <- 3
sigma2 <- total.noise/nvarq
sigmaX=0.1
nvar.eff=150

Sigma=matrix(0,nvar.eff,nvar.eff)
for(i in 1:nvar.eff){
    for(j in 1:nvar.eff){
        Sigma[i,j]=rho^(abs(i-j))
    }
}
Sigma2.y <- matrix(sigma2*rho.e,nvarq, nvarq)
diag(Sigma2.y) <- sigma2

betas.true <- matrix(0, nvar, 3)
betas.true[1:15,1]=rep(1,15)/sqrt(15)
betas.true[16:45,2]=rep(0.5,30)/sqrt(30)
betas.true[46:105,3]=rep(0.25,60)/sqrt(60)

ntest <- 500
ntrain <- 90
ntot <- ntest+ntrain
X <- matrix(0,ntot,nvar)
X[,1:nvar.eff] <- mvrnorm(n=ntot, rep(0, nvar.eff), Sigma)
X[,-(1:nvar.eff)] <- matrix(sigmaX*rnorm((nvar-nvar.eff)*dim(X)[1]),
                           dim(X)[1],(nvar-nvar.eff))
Y <- X%*%betas.true
```

```
Y <- Y+mvrnorm(n=ntot, rep(0,nvarq), Sigma2.y)

X.train <- X[1:ntrain,]
Y.train <- Y[1:ntrain,]
X.test <- X[-(1:ntrain),]
Y.test <- Y[-(1:ntrain),]

cv.fit <- cv.SiER(X.train,Y.train, K.cv=5)

Y.pred=pred.SiER(cv.fit, X.test)
error=sum((Y.pred-Y.test)^2)/ntest
print(c("predict error=", error))
```

`getcoef.SiER` *Get the estimated intercept and coefficient.*

Description

Get the estimates for μ, β based on the object obtained from `cv.SiER`().

Usage

```
getcoef.SiER(cv.fit)
```

Arguments

`cv.fit`	the object obtained from `cv.SiER`().

Value

a list containing

`mu`	the vector of estimated μ.
`beta`	the estimated matrix for β.
`nonzero.ind`	the vector of indices for selected variables.

Author(s)

Ruiyan Luo and Xin Qi

See Also

`cv.SiER`

Examples

```
#See the examples in cv.Sier().
```

pred.SiER — *Prediction for high-dimensional multivariate regression*

Description

Make prediction for the univariate or multivariate response based on new observations of predictors from the CV object obtained by `cv.SiER`.

Usage

```
pred.SiER(cv.fit, X.new)
```

Arguments

`cv.fit`	the CV object obtained by `cv.SiER`().
`X.new`	a new data matrix for predictors. The number of columns equals to the number of variables.

Value

A matrix containing the predicted response for new observations. The number of rows is equal to the number of observations in the new data set, and the number of columns is equal to the number of the responses.

Author(s)

Ruiyan Luo and Xin Qi

See Also

`cv.SiER`

Examples

```
#See the examples in cv.SiER().
```

Index