Package: rpc (via r-universe)

March 23, 2025

Type Package

Title Ridge Partial Correlation

Version 2.0.3

Date 2025-03-20

Encoding UTF-8

Maintainer Somak Dutta <somakd@iastate.edu>

Description Computes the ridge partial correlation coefficients in a high or ultra-high dimensional linear regression problem. An extended Bayesian information criterion is also implemented for variable selection. Users provide the matrix of covariates as a usual dense matrix or a sparse matrix stored in a compressed sparse column format. Detail of the method is given in the manual.

License GPL (>= 2)

Imports Rcpp (>= 1.0.11), Matrix

Suggests MatrixExtra

LinkingTo Rcpp

RoxygenNote 7.3.2

Repository https://somakd.r-universe.dev

RemoteUrl https://github.com/somakd/rpc

RemoteRef HEAD

RemoteSha 2fe95246a6d1aae2e84bf7d473c7a9d15898b65d

Contents

eBIC *Model selection using extended BIC*

Description

This function performs model selection using extended BIC and the ridge partial correlation coefficients

Usage

```
eBIC(rpc.obj)
```

Arguments

rpc.obj	an object of class rpc, containing the following items: - X: matrix of covariates (excluding intercept) - y: response vector - rpc: vector of ridge partial correlation coefficients, one for each column in X.

Value

BIC.PATH vector of eBIC of each model, starting with the intercept only model.

model.best sorted indices (in increased order) of the model with the smallest EBIC

Author(s)

An Nguyen

Somak Dutta

Maintainer: Somak Dutta <somakd@iastate.edu>

Examples

```
n <- 50; p <- 400;
trueidx <- 1:3
truebeta <- c(4,5,6)
X <- matrix(rnorm(n*p), n, p) # n x p covariate matrix
y <- 0.5 + X[,trueidx] %*% truebeta + rnorm(n)
res <- rpc(X,y, lambda = 0.1, ncores = 1)
eBIC(res) # model.best: model selected by eBIC
```

rpc *Ridge Partial Correlations*

Description

The sample ridge partial correlations (RPC) can be used for a simple, efficient, and scalable variables screening method in ultra-high dimensional linear regression. These coefficients incorporate both the ridge regularized estimates of the regression coefficients and the ridge regularized partial variances of the predictor variables, providing the screening method with sure screening property without strong assumptions on the marginal correlations. For more details, please see Wang et al. (2025).

This function computes the ridge partial correlations for each variable in the covariate matrix.

Usage

```
rpc(X, y, lambda = 1, XXt = NULL, ncores = 1, RAM = 6)
```

Arguments

`X`	The input matrix of type 'matrix' or 'dgCMatrix'. No need to center or scale X as that would be done implicitly.
`y`	vector containing the response variable.
`lambda`	the regularization parameter.
`XXt`	the matrix UU' where U = scale(X), will be computed if not provided. When the matrix X is large, recomputing XXt can be costly, for example if the rpc's for a new lambda value is required. It is thus wise to pass XXt (see function `XXt.compute`).
`ncores`	the number of cores to be used. Default is 1.
`RAM`	The algorithm will use a maximum of this much additional RAM. Default is 6GB. Increasing this number, within the limit of the machine, will allow the algorithm to run faster.

Details

Consider the linear regression model:

$$y = \beta_0 + X\beta + \epsilon,$$

where X is the $n \times p$ design matrix, β is a p-dimensional vector, and ϵ is the n-dimensional vector of iid residual errors with mean 0 and variance σ^2, ϵ is independent of X.

Assuming the design matrix X is centered and scaled, and letting $\tilde{Y} = Y - \bar{Y}_n$, the (sample) partial correlation between the i^{th} predictor and Y given the remaining predictors is $-P_{i+1,1}/\sqrt{P_{1,1}P_{i+1,i+1}}$, where P is the $(p+1) \times (p+1)$ joint precision matrix of the response and the covariates. In cases

when $p > n$, P is not invertible. Hence, we consider the ridge regularized version of P, given by R, where:

$$R := (n-1)\begin{bmatrix} \tilde{Y}^T\tilde{Y} & \tilde{Y}^T X \\ X^T\tilde{Y} & X^T X + \lambda I_p \end{bmatrix}^{-1}.$$

The ridge partial correlations between y and X_i is then defined in terms of elements of R as follows:

$$\rho_{i,\lambda} = -\frac{r_{i+1,1}}{\sqrt{r_{1,1} r_{i+1,i+1}}}$$

where $r_{i,j}$ is the $(i.j)$th entry in R.

For variable screening, one may choose the K variables with largest absolute RPC values, for some $K > 0$ (e.g., $K = n$ or $n/\log p$). Alternatively, the extended BIC criteria may be used. See `eBIC` function in this package.

N.B. Further hyper-threading speed-up can be obtained by loading the `MatrixExtra` package.

Value

an object containing the following items:
- rpc: vector of rpc coefficients.
- X: the original matrix of covariates.
- XXt: the matrix UU', where U is the centered-scaled X.
- y: vector of centered and scaled response variable.
- lambda: the original lambda used.

Author(s)

Somak Dutta

An Nguyen

Maintainer: Somak Dutta <somakd@iastate.edu>

See Also

[eBIC()] for model selecting, [XXt.compute()] for computing crossproduct.

Examples

```
## Toy example:
n <- 50; p <- 400;
trueidx <- 1:3 ## First three variables are important
truebeta <- c(4,5,6)
X <- matrix(rnorm(n*p), n, p) ## n x p covariate matrix
y <- 0.5 + X[,trueidx] %*% truebeta + rnorm(n) ## Response
res <- rpc(X,y, lambda = 0.1, ncores = 1)
order(abs(res$rpc),decreasing = TRUE)[1:10] # Top 10 variables
## Run another case with the same X and y, but pass XXt to make it faster
res2 <- rpc(X,y, lambda = 0.001,XXt = res$XXt , ncores = 1)
order(abs(res2$rpc),decreasing = TRUE)[1:10] # Top 10 variables
```

XXt.compute *XXt Computation*

Description

For an input matrix X, This function computes UU' = tcrossprod(U) where U = scale(X) in a memory efficient way.

Usage

```
XXt.compute(
  X,
  meanX = NULL,
  varX = NULL,
  ncores = 4,
  check = TRUE,
  chunksize = 1000
)
```

Arguments

X	The input matrix of type 'matrix' or 'dgCMatrix'. No need to center or scale X as that would be done implicitly.
meanX	Vector of column means of X. Will be computed if not provided.
varX	Vector of column variances of X. Will be computed if not provided.
ncores	the number of cores to be used. Default is 1.
check	check for zero variances.
chunksize	When using openmp parallelization, this would be the chunk size under a dynamic scheduling.

Value

a nrow(X) x nrow(X) dense symmetric matrix.

Author(s)

Somak Dutta

Maintainer: Somak Dutta <somakd@iastate.edu>

Examples

```
require("Matrix")
set.seed(123)
x <- rsparsematrix(100,100,density = 0.3)
norm(XXt.compute(x) - tcrossprod(scale(x))) # (4.783951e-13 on my machine)
```

Index