Package 'HSDiC'

October 12, 2022

Type Package

Version 0.1

Title Homogeneity and Sparsity Detection Incorporating Prior Constraint Information

Author Yaguang Li [aut, cre], Baisuo Jin [aut]

Maintainer Yaguang Li <liyg@mail.ustc.edu.cn>

Depends R (>= 3.5.0), ncvreg, glmnet, quadprog, Matrix

Description We explore sparsity and homogeneity of regression coefficients incorporating prior constraint information. A general pairwise fusion approach is proposed to deal with the sparsity and homogeneity detection when combining prior convex constraints. We develop an modified alternating direction method of multipliers algorithm (ADMM) to obtain the estimators.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2018-10-19 14:10:03 UTC

R topics documented:

HSDiC_ADMM *Homogeneity Detection Incorporating Prior Constraint Information by ADMM*

Description

simultaneous homogeneity detection and variable selection incorporating prior constraint by ADMM algorithm. The problem turn to solving quadratic programming problems of the form *min(-d^T b + 1/2 b^T D b)* with the constraints *A^T b >= b_0*. The penalty is the pairwise fusion with p(p-1)/2 number of penalties.

Usage

```
HSDiC_ADMM(X, Y, A.eq, A.ge, A.lbs, A.ubs, b.eq, b.ge, b.lbs, b.ubs,
  penalty = c("MCP", "SCAD", "adlasso", "lasso"), lambda2,
  admmScale1 = 1/nrow(X), admmScale2 = 1, admmAbsTol = 1e-04,
  admmRelTol = 1e-04, nADMM = 2000, admmVaryScale = FALSE)
```

Arguments

X	n-by-p design matrix.
Y	n-by-1 response matrix.
A.eq	equality constraint matrix.
A.ge	inequality constraint matrix.
A.lbs	low-bounds matrix on variables, see examples.
A.ubs	upper-bounds matrix on variables, see examples.
b.eq	equality constraint vector.
b.ge	inequality constraint vector.
b.lbs	low-bounds on variables, see details.
b.ubs	upper-bounds on variables, see details.
penalty	The penalty to be applied to the model. Either "lasso" (the default), "SCAD", or "MCP".
lambda2	penalty tuning parameter for thresholding function.
admmScale1	first ADMM scale parameter, 1/nrow(X) is default.
admmScale2	second ADMM scale parameter, 1 is default.
admmAbsTol	absolute tolerance for ADMM, 1e-04 is default.
admmRelTol	relative tolerance for ADMM, 1e-04 is default.
nADMM	maximum number of iterations for ADMM, 2000 is default.
admmVaryScale	dynamically chance the ADMM scale parameter, FALSE is default

Value

`betahat` solution vector.

`stats.ADMM_inters` number of iterations.

References

'Pairwise Fusion Approach Incorporating Prior Constraint Information' by Yaguang Li

See Also

`solve.QP`

Examples

```
## data generation
set.seed(111)
n=100
p=50
r <- 1 #0.5, 0.8, 1

beta <- r*c(sample(rep(1:2, each = 10)), rep(0,10), -sample(rep(1:2, each = 10)) )
X <- matrix(rnorm(n*p),nrow = n)
sigma = 1
Y <- X %*% beta + sigma * rnorm(n, 0, 1)


# equalities
A.eq <- rbind(rep(1,p))
b.eq <- c(0)

# inequalities
A.ge <- diag( c(rep(1,30), rep(-1,20)) )
b.ge <- rep(0,p)

# lower-bounds
A.lbs <- diag(1, p)
b.lbs <- rep(-2, p)

# upper-bounds on variables
A.ubs <- diag(-1, p)
b.ubs <- rep(-2, p)

ptm <- proc.time()
fit <- HSDiC_ADMM(X, Y, A.eq, A.ge, A.lbs, A.ubs, b.eq, b.ge, b.lbs, b.ubs,
                  penalty = "adlasso", lambda2 = 0.8, admmScale2 = 1)
proc.time() - ptm

## table(round(fit$beta,1))

plot(beta, type="p", pch = 20, cex = 1)
```

```
points(fit$beta, col = 3)
```

---

mBIC *Modified Bayesian Information Criterion*

Description

Calculate the modified Bayesian information criterion for estimated model.

Usage

```
mBIC(beta, Y, X)
```

Arguments

`beta`	the estimated coefficients.
`Y`	the response.
`X`	design matrix with the same order of the columns in `beta`.

Value

Returns an object with

`BIC`	a numeric value with the corresponding BIC.
`K`	the corresponding number of groups.

References

'Pairwise Fusion Approach Incorporating Prior Constraint Information' by Yaguang Li

See Also

BIC

`thresh_est` *Threshold estimation*

Description

Function to implement the soft-, MCP, SCAD thresholding rule in the ADMM method.

Usage

```
thresh_est(z, lambda, tau, a = 3, penalty = c("MCP", "SCAD", "lasso"))
```

Arguments

`z`	a vector where the function is to be evaluated.
`lambda`	a number representing a tuning parameter.
`tau`	the penalty parameter in the ADMM method.
`a`	the tuning parameter of the MCP/SCAD penalty (see details). Default is 3 for MCP and 3.7 for SCAD.
`penalty`	The penalty to be applied to the model. Either "lasso" (the default), "SCAD", or "MCP".

Value

A vector containing the threshlding values at z.

References

'Pairwise Fusion Approach Incorporating Prior Constraint Information' by Yaguang Li

Index