Package ‘ADMMsigma’

March 1, 2018

Type Package

Title Penalized Precision Matrix Estimation via ADMM

Version 1.0

Date 2018-02-23

Description This R package produces penalized precision matrix estimates via the alternating direction method of multipliers (ADMM) algorithm

License MIT + file LICENSE

ByteCompile TRUE

NeedsCompilation yes

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Depends Rcpp (>= 0.12.10),
RcppArmadillo,
doParallel,
foreach,
dplyr,
ggplot2

LinkingTo Rcpp,
RcppArmadillo

Suggests testthat

SystemRequirements GNU make

R topics documented:

ADMMsigma *ADMM penalized precision matrix estimation (using ADMMsigmac)*

Description

Penalized Gaussian likelihood precision matrix estimation using the ADMM algorithm.

Usage

```
ADMMsigma(X = NULL, S = NULL, lam = 10^seq(-5, 5, 0.5), alpha = seq(0,
  1, 0.1), rho = 2, mu = 10, tau1 = 2, tau2 = 2, crit = "ADMM",
  tol1 = 1e-04, tol2 = 1e-04, maxit = 1000, K = 5, parallel = FALSE,
  quiet = TRUE)
```

Arguments

`X` data matrix

`S` option to specify sample covariance matrix (denominator n)

`lam` tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)

`alpha` elasticnet mixing parameter [0, 1]: 0 = ridge, 1 = lasso/bridge. Defaults to seq(-1, 1, 0.1)

`rho` initial step size for ADMM

`mu` factor for primal and residual norms

`tau1` adjustment for rho

`tau2` adjustment for rho

`crit` criterion for convergence c('ADMM', 'grad', 'loglik'). Option crit != 'ADMM' will use tol1 as tolerance. Default is 'ADMM'

`tol1` absolute tolerance. Defaults to 1e-4

`tol2` relative tolerance. Defaults to 1e-4

`maxit` maximum number of iterations

`K` specify the number of folds for cross validation

`parallel` option to run CV in parallel. Defaults to FALSE

`quiet` specify whether the function returns progress of CV or not

Value

iterations, lam, omega, and gradient

Examples

```
ADMM_sigma(X, lam = 0.1, rho = 10)
```

`ADMMsigmac` *ADMM penalized precision matrix estimation (c++)*

Description

Penalized Gaussian likelihood precision matrix estimation using the ADMM algorithm.

Usage

```
ADMMsigmac(S, initZ2, initY, lam, alpha = 1, rho = 2, mu = 10, tau1 = 2,
  tau2 = 2, crit = "ADMM", tol1 = 1e-04, tol2 = 1e-04, maxit = 1000L)
```

Arguments

`S` option to specify sample covariance matrix (denominator n)

`initY` initialization matrix for Y

`lam` tuning parameter for penalty

`alpha` elasticnet mixing parameter [0, 1]: 0 = ridge, 1 = lasso/bridge

`rho` initial step size for ADMM

`mu` factor for primal and residual norms

`tau1` adjustment for rho

`tau2` adjustment for rho

`crit` criterion for convergence c("ADMM", "grad", "lik"). Option crit != "ADMM" will use tol1 as tolerance. Defaults to "ADMM"

`tol1` absolute tolerance. Defaults to 1e-4

`tol2` relative tolerance. Defaults to 1e-4

`maxit` maximum number of iterations

`initZ` initialization matrix for Z2

Value

iterations, lam, omega

Examples

```
ADMMsigmac(X, lam = 0.1)
```

CVP_ADMMsigmac *CV (no folds) ADMM penalized precision matrix estimation (c++)*

Description

Cross validation (no folds) function for ADMM_sigma. This function is to be used with ParallelCV.

Usage

```
CVP_ADMMsigmac(S_train, S_valid, lam, alpha, rho = 2, mu = 10, tau1 = 2,
  tau2 = 2, crit = "ADMM", tol1 = 1e-04, tol2 = 1e-04, maxit = 1000L,
  K = 5L, quiet = TRUE)
```

Arguments

S_train	matrix or data frame. This is pxp sample covariance for training data
S_valid	matrix or data frame. This is pxp sample covariance for validation data
lam	tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)
alpha	elasticnet mixing parameter [0, 1]: 0 = ridge, 1 = lasso/bridge
rho	initial step size for ADMM
mu	factor for primal and residual norms
tau1	adjustment for rho
tau2	adjustment for rho
crit	criterion for convergence c('ADMM', 'grad', 'lik'). Option crit != 'ADMM' will use tol1 as tolerance. Defaults to 'ADMM'
tol1	absolute tolerance. Defaults to 1e-4
tol2	relative tolerance. Defaults to 1e-4
maxit	maximum number of iterations
quiet	specify whether the function returns progress of CV or not

Value

iterations, lam, S, Omega, and cv.errors

CV_ADMMsigmac *CV ADMM penalized precision matrix estimation (c++)*

Description

Cross validation function for ADMM_sigma.

Usage

```
CV_ADMMsigmac(X, lam, alpha, rho = 2, mu = 10, tau1 = 2, tau2 = 2,
  crit = "ADMM", tol1 = 1e-04, tol2 = 1e-04, maxit = 1000L, K = 5L,
  quiet = TRUE)
```

Arguments

`X` matrix or data frame. This is the n x p column matrix where the rows are a realization of n independent copies of a p-variate random vector

`lam` tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)

`alpha` elasticnet mixing parameter [0, 1]: 0 = ridge, 1 = lasso/bridge

`rho` initial step size for ADMM

`mu` factor for primal and residual norms

`tau1` adjustment for rho

`tau2` adjustment for rho

`crit` criterion for convergence c('ADMM', 'grad', 'lik'). Option crit != 'ADMM' will use tol1 as tolerance. Defaults to 'ADMM'

`tol1` absolute tolerance. Defaults to 1e-4

`tol2` relative tolerance. Defaults to 1e-4

`maxit` maximum number of iterations

`K` specify the number of folds for cross validation

`quiet` specify whether the function returns progress of CV or not

Value

iterations, lam, S, Omega, and cv.errors

Examples

```
CV_ADMMsigmac(X, lam = seq(0.1, 3, 0.1))
```

`CV_RIDGEsigmac` *CV ridge penalized precision matrix estimation (c++)*

Description

Cross validation function for RIDGEsigma.

Usage

```
CV_RIDGEsigmac(X, lam, K = 3L, quiet = TRUE)
```

Arguments

`X` matrix or data frame. This is the n x p column matrix where the rows are a realization of n independent copies of a p-variate random vector

`lam` tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)

`K` specify the number of folds for cross validation

`quiet` specify whether the function returns progress of CV or not

Value

iterations, lam, S, Omega, and cv.errors

Examples

```
CV_RIDGEsigmac(X, lam = seq(0.1, 3, 0.1))
```

ParallelCV *Parallel CV (uses CV_ADMMsigmac)*

Description

Parallel implementation of cross validation.

Usage

```
ParallelCV(X = NULL, S = NULL, lam = 10^seq(-5, 5, 0.5), alpha = seq(0,
  1, 0.1), rho = 2, mu = 10, tau1 = 2, tau2 = 2, crit = "ADMM",
  tol1 = 1e-04, tol2 = 1e-04, maxit = 1000, K = 5, quiet = TRUE)
```

Arguments

lam	tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)
alpha	elasticnet mixing parameter [0, 1]: 0 = ridge, 1 = lasso/bridge
rho	initial step size for ADMM
mu	factor for primal and residual norms
tau1	adjustment for rho
tau2	adjustment for rho
crit	criterion for convergence c('ADMM', 'grad', 'lik'). Option crit != 'ADMM' will use tol1 as tolerance. Default is 'ADMM'
tol1	absolute tolerance. Defaults to 1e-4
tol2	relative tolerance. Defaults to 1e-4
maxit	maximum number of iterations
K	specify the number of folds for cross validation
quiet	specify whether the function returns progress of CV or not

Value

iterations, lam, omega, and gradient

`plot.ADMMsigma` *Plot ADMMsigma object*

Description

produces a heat plot for the cross validation errors

Usage

```
## S3 method for class 'ADMMsigma'
plot(x, ...)
```

Arguments

`x`	ADMMsigma class object

`plot.RIDGEsigma` *Plot RIDGEsigma object*

Description

produces a heat plot for the cross validation errors

Usage

```
## S3 method for class 'RIDGEsigma'
plot(x, ...)
```

Arguments

`x`	RIDGEsigma class object

`print.ADMMsigma` *Print ADMMsigma object*

Usage

```
## S3 method for class 'ADMMsigma'
print(x, ...)
```

Arguments

`x`	ADMMsigma class object

print.RIDGEsigma *Print RIDGEsigma object*

Usage

```
## S3 method for class 'RIDGEsigma'
print(x, ...)
```

Arguments

x	RIDGEsigma class object

RIDGEsigma *Ridge penalized precision matrix estimation (using RIDGEsigmac)*

Description

Penalized Gaussian likelihood precision matrix estimation using the ADMM algorithm.

Usage

```
RIDGEsigma(X = NULL, S = NULL, lam = 10^seq(-5, 5, 0.5), K = 3,
  quiet = TRUE)
```

Arguments

X	data matrix
S	option to specify sample covariance matrix (denominator n)
lam	tuning parameter for penalty. Defaults to 10^seq(-5, 5, 0.5)
K	specify the number of folds for cross validation
quiet	specify whether the function returns progress of CV or not

Value

lam, omega, and gradient

Examples

```
RIDGEsigma(X, lam = 0.1)
```

RIDGEsigmac *Ridge-penalized precision matrix estimation (c++)*

Description

Ridge-penalized Gaussian likelihood precision matrix estimation. Augmented from Adam Rothman's STAT 8931 code.

Usage

```
RIDGEsigmac(S, lam)
```

Arguments

S	sample covariance matrix (denominator n)
lam	tuning parameter for penalty

Value

matrix of omega hat

Examples

```
n = nrow(X)
RIDGEsigmac(S = (n-1)/n*cov(X), lam = 0.1)
```

Index