Package ‘cqrReg’

October 12, 2022

Type Package

Title Quantile, Composite Quantile Regression and Regularized Versions

Version 1.2.1

Date 2022-06-07

Author Jueyu Gao & Linglong Kong

Maintainer Jueyu Gao <jueyu@ualberta.ca>

Description Estimate quantile regression(QR) and composite quantile regression (cqr) and with adaptive lasso penalty using interior point (IP), majorize and minimize(MM), coordinate descent (CD), and alternating direction method of multipliers algorithms(ADMM).

License GPL (>= 2)

Depends Rcpp (>= 0.10.0),quantreg,R (>= 2.6)

LinkingTo Rcpp,RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-07 05:00:02 UTC

R topics documented:

`cqr.admm` *Composite Quantile regression (cqr) use Alternating Direction Method of Multipliers (ADMM) algorithm.*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

Usage

```
cqr.admm(X,y,tau,rho,beta, maxit, toler)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	vector of quantile level
`rho`	augmented Lagrangian parameter
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

cqr.admm(x,y,tau) work properly only if the least square estimation is good.

References

S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein.(2010) Distributed Optimization and Statistical Learning via the Alternating Direction. Method of Multipliers *Foundations and Trends in Machine Learning*, **3**, No. 1, 1–122

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
cqr.admm(x,y,tau)
```

`cqr.cd` *Composite Quantile Regression (cqr) use Coordinate Descent (cd) Algorithms*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

Usage

```
cqr.cd(X,y,tau,beta,maxit,toler)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	vector of quantile level
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

cqr.cd(x,y,tau) work properly only if the least square estimation is good.

References

Wu, T.T. and Lange, K. (2008). Coordinate Descent Algorithms for Lasso Penalized Regression. *Annals of Applied Statistics*, **2**, No 1, 224–244.

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
cqr.cd(x,y,tau)
```

cqr.fit *Composite Quantile Regression (cqr) model fitting*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. High level function for estimating parameter by composite quantile regression.

Usage

```
cqr.fit(X,y,tau,beta,method,maxit,toler,rho)
```

Arguments

X	the design matrix
y	response variable
tau	vector of quantile level
method	"mm" for majorize and minimize method,"cd" for coordinate descent method, "admm" for Alternating method of mulipliers method,"ip" for interior point mehod
rho	augmented Lagrangian parameter
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)
toler	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

cqr.fit(x,y,tau) work properly only if the least square estimation is good. Interior point method is done by quantreg.

`cqr.fit.lasso` *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso)*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. High level function for estimating and selecting parameter by composite quantile regression with adaptive lasso penalty.

Usage

```
cqr.fit.lasso(X,y,tau,lambda,beta,method,maxit,toler,rho)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	vector of quantile level
`method`	"mm" for majorize and minimize method,"cd" for coordinate descent method, "admm" for Alternating method of mulipliers method
`lambda`	The constant coefficient of penalty function. (default lambda=1)
`rho`	augmented Lagrangian parameter
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

cqr.fit.lasso(x,y,tau) work properly only if the least square estimation is good.

cqr.ip *Composite Quantile Regression (cqr) use Interior Point (ip) Method*

Description

The function use the interior point method from quantreg to solve the quantile regression problem.

Usage

```
cqr.ip(X,y,tau)
```

Arguments

X	the design matrix
y	response variable
tau	vector of quantile level

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

Need to install quantreg package from CRAN.

References

Koenker, R. and S. Portnoy (1997). The Gaussian Hare and the Laplacian Tortoise: Computability of squared-error vs. absolute-error estimators, with discussion, *Statistical Science*, **12**, 279-300.

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
#you should install quantreg first to run following command
#cqr.ip(x,y,tau)
```

`cqr.lasso.admm` *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Alternating Direction Method of Multipliers (ADMM) algorithm*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

Usage

```
cqr.lasso.admm(X,y,tau,lambda,rho,beta,maxit)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	vector of quantile level
`lambda`	The constant coefficient of penalty function. (default lambda=1)
`rho`	augmented Lagrangian parameter
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

cqr.lasso.admm(x,y,tau) work properly only if the least square estimation is good.

References

S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein.(2010) Distributed Optimization and Statistical Learning via the Alternating Direction. Method of Multipliers *Foundations and Trends in Machine Learning*, **3**, No. 1, 1–122

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
cqr.lasso.admm(x,y,tau)
```

`cqr.lasso.cd` *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Coordinate Descent (cd) Algorithms*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

Usage

```
cqr.lasso.cd(X,y,tau,lambda,beta,maxit,toler)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	vector of quantile level
`lambda`	The constant coefficient of penalty function. (default lambda=1)
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

cqr.lasso.cd(x,y,tau) work properly only if the least square estimation is good.

References

Wu, T.T. and Lange, K. (2008). Coordinate Descent Algorithms for Lasso Penalized Regression. *Annals of Applied Statistics*, **2**, No 1, 224–244.

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
cqr.lasso.cd(x,y,tau)
```

`cqr.lasso.mm` *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Majorize and Minimize (mm) Algorithm*

Description

The adaptive lasso penalty parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

Usage

```
cqr.lasso.mm(X,y,tau,lambda,beta,maxit,toler)
```

Arguments

X	the design matrix
y	response variable
tau	vector of quantile level
lambda	The constant coefficient of penalty function. (default lambda=1)
beta	initial value of estimate coefficient (default naive guess by least square estimation)

`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept for various quantile level

Note

cqr.lasso.mm(x,y,tau) work properly only if the least square estimation is good.

References

David R.Hunter and Runze Li.(2005) Variable Selection Using MM Algorithms,*The Annals of Statistics* **33**, Number 4, Page 1617–1642.

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
cqr.lasso.mm(x,y,tau)
```

---

cqr.mm *Composite Quantile Regression (cqr) use Majorize and Minimize (mm) Algorithm*

Description

Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

Usage

```
cqr.mm(X,y,tau,beta,maxit,toler)
```

Arguments

X	the design matrix
y	response variable
tau	vector of quantile level
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)
toler	the tolerance critical for stop the algorithm (default 1e-3)

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept for various quantile level

Note

cqr.mm(x,y,tau) work properly only if the least square estimation is good.

References

David R.Hunter and Kenneth Lange. Quantile Regression via an MM Algorithm,*Journal of Computational and Graphical Statistics*, **9**, Number 1, Page 60–77.

Hui Zou and Ming Yuan(2008). Composite Quantile Regression and the Oracle Model Selection Theory, *The Annals of Statistics*, **36**, Number 3, Page 1108–1126.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
tau=1:5/6
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
cqr.mm(x,y,tau)
```

`CQRADMMCPP` *Composite Quantile regression (cqr) use Alternating Direction Method of Multipliers (ADMM) algorithm core computational part*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

`CQRCDCPP` *Composite Quantile Regression (cqr) use Coordinate Descent (cd) Algorithms core computational part*

Description

Composite quantile regression (cqr) find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

`CQRMMCPP` *Composite Quantile Regression (cqr) use Majorize and Minimize (mm) Algorithm core computational part*

Description

Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

`CQRPADMMCPP` *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Alternating Direction Method of Multipliers (ADMM) algorithm core computational part*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

CQRPCDCPP *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Coordinate Descent (cd) Algorithms core computational part*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

CQRPMMCPP *Composite Quantile Regression (cqr) with Adaptive Lasso Penalty (lasso) use Majorize and Minimize (mm) Algorithm core computational part*

Description

The adaptive lasso penalty parameter base on the estimated coefficient without penalty function. Composite quantile regression find the estimated coefficient which minimize the absolute error for various quantile level. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

QR.admm *Quantile Regression (QR) use Alternating Direction Method of Multipliers (ADMM) algorithm*

Description

The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

Usage

```
QR.admm(X,y,tau,rho,beta, maxit, toler)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level
rho	augmented Lagrangian parameter
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)
toler	the tolerance critical for stop the algorithm (default 1e-3)

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

QR.admm(x,y,tau) work properly only if the least square estimation is good.

References

S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein.(2010) Distributed Optimization and Statistical Learning via the Alternating Direction.Method of Multipliers *Foundations and Trends in Machine Learning*, **3**, No.1, 1–122

Koenker, Roger. *Quantile Regression*, New York, 2005. Print.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
QR.admm(x,y,0.1)
```

QR.cd *Quantile Regression (QR) use Coordinate Descent (cd) Algorithms*

Description

The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

Usage

```
QR.cd(X,y,tau,beta,maxit,toler)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	quantile level
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

QR.cd(x,y,tau) work properly only if the least square estimation is good.

References

Wu, T.T. and Lange, K. (2008). Coordinate Descent Algorithms for Lasso Penalized Regression. *Annals of Applied Statistics*, **2**, No 1, 224–244.

Koenker, Roger. *Quantile Regression*, New York, 2005. Print.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
```

```
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
QR.cd(x,y,0.1)
```

QR.ip *Quantile Regression (QR) use Interior Point (ip) Method*

Description

The function use the interior point method from quantreg to solve the quantile regression problem.

Usage

```
QR.ip(X,y,tau)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

Need to install quantreg package from CRAN.

References

Koenker, Roger. *Quantile Regression*, New York, 2005. Print.

Koenker, R. and S. Portnoy (1997). The Gaussian Hare and the Laplacian Tortoise: Computability of squared-error vs. absolute-error estimators, with discussion, *Statistical Science*, **12**, 279-300.

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
```

```
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
#you should install Rmosek first to run following command
#QR.ip(x,y,0.1)
```

QR.lasso.admm *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Alternating Direction Method of Multipliers (ADMM) algorithm*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

Usage

```
QR.lasso.admm(X,y,tau,lambda,rho,beta,maxit)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level
lambda	The constant coefficient of penalty function. (default lambda=1)
rho	augmented Lagrangian parameter
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

QR.lasso.admm(x,y,tau) work properly only if the least square estimation is good.

References

S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein.(2010) Distributed Optimization and Statistical Learning via the Alternating Direction. Method of Multipliers *Foundations and Trends in Machine Learning*, **3**, No.1, 1–122

Wu, Yichao and Liu, Yufeng (2009). Variable selection in quantile regression. *Statistica Sinica*, **19**, 801–817.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
QR.lasso.admm(x,y,0.1)
```

`QR.lasso.cd` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Coordinate Descent (cd) Algorithms*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression. As explored by Tong Tong Wu and Kenneth Lange.

Usage

```
QR.lasso.cd(X,y,tau,lambda,beta,maxit,toler)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	quantile level
`lambda`	The constant coefficient of penalty function. (default lambda=1)
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

QR.lasso.cd(x,y,tau) work properly only if the least square estimation is good.

References

Wu, T.T. and Lange, K. (2008). Coordinate Descent Algorithms for Lasso Penalized Regression. *Annals of Applied Statistics*, **2**, No 1, 224–244.

Wu, Yichao and Liu, Yufeng (2009). Variable selection in quantile regression. *Statistica Sinica*, **19**, 801–817.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
QR.lasso.cd(x,y,0.1)
```

`QR.lasso.ip` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Interior Point (ip) Method*

Description

The function use the interior point method from quantreg to solve the quantile regression problem.

Usage

```
QR.lasso.ip(X,y,tau,lambda)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level
lambda	The constant coefficient of penalty function. (default lambda=1)

Value

a `list` structure is with components

beta	the vector of estimated coefficient
b	intercept
lambda	The constant coefficient of penalty function. (default lambda=1)

Note

Need to install quantreg package from CRAN.

References

Koenker, R. and S. Portnoy (1997). The Gaussian Hare and the Laplacian Tortoise: Computability of squared-error vs. absolute-error estimators, with discussion, *Statistical Science*, **12**, 279-300.

Wu, Yichao and Liu, Yufeng (2009). Variable selection in quantile regression. *Statistica Sinica*, **19**, 801–817.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
#you should install Rmosek first to run following command
#QR.lasso.ip(x,y,0.1)
```

`QR.lasso.mm` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Majorize and Minimize (mm) algorithm*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

Usage

```
QR.lasso.mm(X,y,tau,lambda,beta,maxit,toler)
```

Arguments

X	the design matrix.
y	response variable.
tau	quantile level.
lambda	The constant coefficient of penalty function. (default lambda=1)
beta	initial value of estimate coefficient.(default naive guess by least square estimation)
maxit	maxim iteration. (default 200)
toler	the tolerance critical for stop the algorithm. (default 1e-3)

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

QR.lasso.mm(x,y,tau) work properly only if the least square estimation is good.

References

David R.Hunter and Runze Li.(2005) Variable Selection Using MM Algorithms,*The Annals of Statistics* **33**, Number 4, Page 1617–1642.

Examples

```
set.seed(1)
n=100
p=2
a=2*rnorm(n*2*p, mean = 1, sd =1)
x=matrix(a,n,2*p)
beta=2*rnorm(p,1,1)
beta=rbind(matrix(beta,p,1),matrix(0,p,1))
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*20 matrix, y is 1000*1 vector, beta is 20*1 vector with last ten zero value elements.
QR.lasso.mm(x,y,0.1)
```

QR.mm — *Quantile Regression (QR) use Majorize and Minimize (mm) algorithm*

Description

The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

Usage

```
QR.mm(X,y,tau,beta,maxit,toler)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)
toler	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

QR.mm(x,y,tau) work properly only if the least square estimation is good.

References

David R.Hunter and Kenneth Lange. Quantile Regression via an MM Algorithm, *Journal of Computational and Graphical Statistics*, **9**, Number 1, Page 60–77

Examples

```
set.seed(1)
n=100
p=2
a=rnorm(n*p, mean = 1, sd =1)
x=matrix(a,n,p)
beta=rnorm(p,1,1)
beta=matrix(beta,p,1)
y=x%*%beta-matrix(rnorm(n,0.1,1),n,1)
# x is 1000*10 matrix, y is 1000*1 vector, beta is 10*1 vector
QR.mm(x,y,0.1)
```

QRADMMCPP — *Quantile Regression (QR) use Alternating Direction Method of Multipliers (ADMM) algorithm core computational part*

Description

The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

QRCDCPP — *Quantile Regression (QR) use Coordinate Descent (cd) Algorithms core computational part*

Description

The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression.

qrfit *Quantile Regression (qr) model fitting*

Description

High level function for estimating parameters by quantile regression

Usage

```
qrfit(X,y,tau,beta,method,maxit,toler,rho)
```

Arguments

X	the design matrix
y	response variable
tau	quantile level
method	"mm" for majorize and minimize method,"cd" for coordinate descent method, "admm" for Alternating method of mulipliers method,"ip" for interior point mehod
rho	augmented Lagrangian parameter
beta	initial value of estimate coefficient (default naive guess by least square estimation)
maxit	maxim iteration (default 200)
toler	the tolerance critical for stop the algorithm (default 1e-3)

Value

a list structure is with components

beta	the vector of estimated coefficient
b	intercept

Note

qrfit(x,y,tau) work properly only if the least square estimation is good. Interior point method is done by quantreg.

`qrfit.lasso` *Quantile Regression (qr) with Adaptive Lasso Penalty (lasso)*

Description

High level function for estimating and selecting parameter by quantile regression with adaptive lasso penalty.

Usage

```
qrfit.lasso(X,y,tau,lambda,beta,method,maxit,toler,rho)
```

Arguments

`X`	the design matrix
`y`	response variable
`tau`	quantile level
`method`	"mm" for majorize and minimize method,"cd" for coordinate descent method, "admm" for Alternating method of mulipliers method,"ip" for interior point mehod
`lambda`	The constant coefficient of penalty function. (default lambda=1)
`rho`	augmented Lagrangian parameter
`beta`	initial value of estimate coefficient (default naive guess by least square estimation)
`maxit`	maxim iteration (default 200)
`toler`	the tolerance critical for stop the algorithm (default 1e-3)

Value

a `list` structure is with components

`beta`	the vector of estimated coefficient
`b`	intercept

Note

qrfit.lasso(x,y,tau) work properly only if the least square estimation is good. Interior point method is done by quantreg.

`QRMMCPP` *Quantile Regression (QR) use Majorize and Minimize (mm) algorithm core computational part*

Description

The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

`QRPADMMCPP` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Alternating Direction Method of Multipliers (ADMM) algorithm core computational part*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The problem is well suited to distributed convex optimization and is based on Alternating Direction Method of Multipliers (ADMM) algorithm .

`QRPCDCPP` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Coordinate Descent (cd) Algorithms core computational part*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The algorithm base on greedy coordinate descent and Edgeworth's for ordinary l_1 regression. As explored by Tong Tong Wu and Kenneth Lange.

`QRPMMCPP` *Quantile Regression (QR) with Adaptive Lasso Penalty (lasso) use Majorize and Minimize (mm) algorithm core computational part*

Description

The adaptive lasso parameter base on the estimated coefficient without penalty function. The algorithm majorizing the objective function by a quadratic function followed by minimizing that quadratic.

Index