Here is a brief explanation of the PPADS package.

**Two main functions:** PPADS(Parallel PPA algorithm for solving Dantzig selectors), IPPADS(Parallel Improved PPA Algorithm for Solving Dantzig selectors).

1. **PPADS(pen,lambda,X,Xy,G)**

#' @param lambda Parameter tuning or regularization term parameters.

#' @param X Matrix of predictors, of dimension (n\*p); each row is an observation.

#' @param Xy Xy represents X multiplied by y (Responses0).

#' @param G The number of blocks in the observation matrix divided by columns.

#' @returns \item{beta}{Regression coefficient}

#' @returns \item{ite}{number of iterations}

#' @returns \item{time}{calculation time}

**Two examples for PPADS**

#' #######Example 1(Example of sparse coefficients)

n=500

p=1000

beta=rep(0,p)

beta[1:8]=c(3,1.5,10,4,2,5,2.5,4.5)

rho <- 0.5

R <- matrix(0,p,p)#

for(i in 1:p){

for(j in 1:p){

R[i,j] <- rho^abs(i-j)

}

}

error=rnorm(n,0,1)#rt(n,1.5),rnorm(n,0,1),rcauchy(n,0,1)

X <- matrix(rnorm(n\*p),n,p) %\*% t(chol(R))#chol

#X=scale(X,center=FALSE,scale=TRUE)

delte=0.1#0.5,1.5

y <- X %\*% beta + delte\*error

p = ncol(X)

n = nrow(X)

lambda = 15\*10^{-1}\*sqrt(log(p)/n) #15\*10^{-1}\*sqrt(log(p)/n)

Xy = t(X)%\*%y

###Lasso

Model1\_lasso = PPADS(pen="lasso",lambda,X,Xy ,G=1)#It has insensitivity to partitioning, which #means that no matter how G changes, the solution will not change.

Model1\_lasso$beta[1:8]

Model1\_lasso$time

#L1-error

sum(abs(Model1\_lasso$beta - beta))

#L2-error

sum((Model1\_lasso$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_lasso$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_lasso$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_lasso$beta)>10^-5))

###MCP

Model1\_mcp = PPADS(pen="mcp",lambda,X,Xy ,G=1)

Model1\_mcp$beta[1:8]

Model1\_mcp$time

#L1-error

sum(abs(Model1\_mcp$beta - beta))

#L2-error

sum((Model1\_mcp$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_mcp$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_mcp$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_mcp$beta)>10^-5))

###SCAD

Model1\_scad = PPADS(pen="scad",lambda,X,Xy ,G=1)

Model1\_scad$beta[1:8]

Model1\_scad$time

#L1-error

sum(abs(Model1\_scad$beta - beta))

#L2-error

sum((Model1\_scad$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_scad$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_scad$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_scad$beta)>10^-5))

#' #######Example 2(Example of Dense coefficients)

s=1#s=1,2,3....

n=720\*s

p=2560\*s

index=1:80

ind=sample(index,10)

beta=rep(0,p)

for (i in 1:10) {

for (j in 1:(32\*s)) {

betag=sign(runif(1,-1,1))\*(1+ abs(rnorm(32\*s,0,1)))

}

beta[((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))]=betag

}

rho <- 0.5

R <- matrix(0,p,p)#

for(i in 1:p){

for(j in 1:p){

R[i,j] <- rho^abs(i-j)

}

}

error=rnorm(n,0,1)#rt(n,1.5),rnorm(n,0,1),rcauchy(n,0,1)

X <- matrix(rnorm(n\*p),n,p) %\*% t(chol(R))#chol

X=scale(X,center=FALSE,scale=TRUE)

delte=0.5#0.5,1.5

y <- X %\*% beta + delte\*error

Xy = t(X)%\*%y

p = ncol(X)

n = nrow(X)

lambda = 5\*10^{-2}\*sqrt(log(p)/n)# best 5\*10^{-2}\*sqrt(log(p)/n)

###Lasso

Model2\_lasso = PPADS(pen="lasso",lambda,X,Xy ,G=1)#It has insensitivity to partitioning, which #means that no matter how G changes, the solution will not change.

plot(beta)

plot(Model2\_lasso$beta)

#AE

sum(abs(Model2\_lasso$beta - beta))/p

#number of iterations

Model2\_lasso$ite

#computing time

Model2\_lasso$time

#

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_lasso$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_lasso$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num

###MCP

Model2\_mcp = PPADS(pen="mcp",lambda,X,Xy ,G=1)

plot(beta)

plot(Model2$mcp)

sum(abs(Model2\_mcp$beta - beta))/p

#AE

sum(abs(Model2\_mcp$beta - beta))/p

#number of iterations

Model2\_mcp$ite

#computing time

Model2\_mcp$time

#

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_mcp$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_mcp$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num

###SCAD

Model2\_scad = PPADS(pen="scad",lambda,X,Xy ,G=1)

plot(beta)

plot(Model2$scad)

sum(abs(Model2\_scad$beta - beta))/p

#AE

sum(abs(Model2\_scad$beta - beta))/p

#number of iterations

Model2\_scad$ite

#computing time

Model2\_scad$time

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_scad$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_scad$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num

1. **IPPADS(pen,lambda,X,Xy,G)**

#' @param lambda Parameter tuning or regularization term parameters.

#' @param X Matrix of predictors, of dimension (n\*p); each row is an observation.

#' @param Xy Xy represents X multiplied by y (Responses0).

#' @param G The number of blocks in the observation matrix divided by columns.

#' @returns \item{beta}{Regression coefficient}

#' @returns \item{ite}{number of iterations}

#' @returns \item{time}{calculation time}

#' #######Example 1(Example of sparse coefficients)

n=500

p=1000

beta=rep(0,p)

beta[1:8]=c(3,1.5,10,4,2,5,2.5,4.5)

rho <- 0.5

R <- matrix(0,p,p)#

for(i in 1:p){

for(j in 1:p){

R[i,j] <- rho^abs(i-j)

}

}

error=rnorm(n,0,1)#rt(n,1.5),rnorm(n,0,1),rcauchy(n,0,1)

X <- matrix(rnorm(n\*p),n,p) %\*% t(chol(R))#chol

#X=scale(X,center=FALSE,scale=TRUE)

delte=0.1#0.5,1.5

y <- X %\*% beta + delte\*error

p = ncol(X)

n = nrow(X)

lambda = 50\*10^{-1}\*sqrt(log(p)/n) #15\*10^{-1}\*sqrt(log(p)/n)

Xy = t(X)%\*%y

###Lasso

Model1\_lasso = IPPADS(pen="lasso",lambda,X,Xy ,G=1)#It has insensitivity to partitioning, which #means that no matter how G changes, the solution will not change.

Model1\_lasso$beta[1:8]

Model1\_lasso$time

#L1-error

sum(abs(Model1\_lasso$beta - beta))

#L2-error

sum((Model1\_lasso$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_lasso$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_lasso$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_lasso$beta)>10^-5))

###MCP

Model1\_mcp = IPPADS(pen="mcp",lambda,X,Xy ,G=1)

Model1\_mcp$beta[1:8]

Model1\_mcp$time

#L1-error

sum(abs(Model1\_mcp$beta - beta))

#L2-error

sum((Model1\_mcp$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_mcp$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_mcp$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_mcp$beta)>10^-5))

###SCAD

Model1\_scad = IPPADS(pen="scad",lambda,X,Xy ,G=1)

Model1\_scad$beta[1:8]

Model1\_scad$time

#L1-error

sum(abs(Model1\_scad$beta - beta))

#L2-error

sum((Model1\_scad$beta - beta)^2)

#Model error

sum(as.matrix(Model1\_scad$beta - beta,p,1)\*(R%\*%as.matrix(Model1\_scad$beta - beta,p,1)))

#Number of non-zero coefficients

length(which(abs(Model1\_scad$beta)>10^-5))

#' #######Example 2(Example of Dense coefficients)

s=1#s=1,2,3....

n=720\*s

p=2560\*s

index=1:80

ind=sample(index,10)

beta=rep(0,p)

for (i in 1:10) {

for (j in 1:(32\*s)) {

betag=sign(runif(1,-1,1))\*(1+ abs(rnorm(32\*s,0,1)))

}

beta[((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))]=betag

}

rho <- 0.5

R <- matrix(0,p,p)#

for(i in 1:p){

for(j in 1:p){

R[i,j] <- rho^abs(i-j)

}

}

error=rnorm(n,0,1)#rt(n,1.5),rnorm(n,0,1),rcauchy(n,0,1)

X <- matrix(rnorm(n\*p),n,p) %\*% t(chol(R))#chol

X=scale(X,center=FALSE,scale=TRUE)

delte=0.5#0.5,1.5

y <- X %\*% beta + delte\*error

Xy = t(X)%\*%y

p = ncol(X)

n = nrow(X)

lambda = 5\*10^{-2}\*sqrt(log(p)/n)# best 5\*10^{-2}\*sqrt(log(p)/n)

###Lasso

Model2\_lasso = PPADS(pen="lasso",lambda,X,Xy ,G=1)#It has insensitivity to partitioning, which #means that no matter how G changes, the solution will not change.

plot(beta)

plot(Model2\_lasso$beta)

#AE

sum(abs(Model2\_lasso$beta - beta))/p

#number of iterations

Model2\_lasso$ite

#computing time

Model2\_lasso$time

#

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_lasso$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_lasso$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num

###MCP

Model2\_mcp = PPADS(pen="mcp",lambda,X,Xy ,G=1)

plot(beta)

plot(Model2$mcp)

sum(abs(Model2\_mcp$beta - beta))/p

#AE

sum(abs(Model2\_mcp$beta - beta))/p

#number of iterations

Model2\_mcp$ite

#computing time

Model2\_mcp$time

#

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_mcp$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_mcp$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num

###SCAD

Model2\_scad = PPADS(pen="scad",lambda,X,Xy ,G=1)

plot(beta)

plot(Model2$scad)

sum(abs(Model2\_scad$beta - beta))/p

#AE

sum(abs(Model2\_scad$beta - beta))/p

#number of iterations

Model2\_scad$ite

#computing time

Model2\_scad$time

#FN

Nonindex = NULL

for (i in 1:10) {

Nonindex =c(Nonindex, ((ind[i]-1)\*(32\*s)+1):((ind[i]-1)\*(32\*s)+(32\*s))) }

FN\_num = 0

for (i in Nonindex) {

if(abs(Model2\_scad$beta[i]) >10^-5) {FN\_num = FN\_num +0 }

}

FN\_num

#FP

FP\_num = 0

Oindex = (1:p)[-Nonindex]

for (i in Oindex) {

if(abs(Model2\_scad$beta[i]) >10^-4) {FP\_num = FP\_num +1 }

}

FP\_num