rm(list=ls())

library(MASS)

library(lme4)

library(nlme)

library(flexclust)

Sn=function(x,t){  
xx=1-t/sqrt(sum(x^2))

if(xx>0){

s=xx\*x

}else{

s=0\*x

}

list(s=s)

}

Theta\_SCAD=function(u,lambda){

L=length(u)

Ta=rep(0,L)

for(k in 1:L){

uabs=abs(u[k])

if(uabs<=(lambda+lambda/eta)){

Ta[k]=Sn(x=u[k],t=lambda/eta)$s

}else if(uabs>(lambda+lambda/eta)&uabs<=(v\*lambda)){

Ta[k]=Sn(x=u[k],t=v\*lambda/((v-1)\*eta))$s/(1-1/((v-1)\*eta))

}else{

Ta[k]=u[k]

}

}

list(Ta=Ta)

}

Fn\_SCAD=function(lambda){

ahat=a0

betahat=b0

theta=theta0

upsilon=upsilon0

counter=1

eps\_rel=0.0001

eps\_abs=0.0001

repeat{

a1=as.vector(Z%\*%ahat+X%\*%betahat)

phi=sum((Y-exp(a1))^2/exp(a1))/(N-p-m)

W=diag(exp(a1)/phi)

Yt=Z%\*%ahat+X%\*%betahat+(1/exp(a1))\*(Y-exp(a1))

xx=solve(t(X)%\*%W%\*%X)

Qx=diag(N)-X%\*%xx%\*%t(X)%\*%W

ahat=solve(t(Z)%\*%W%\*%Qx%\*%Z+eta\*t(Delta)%\*%Delta)%\*%(t(Z)%\*%W%\*%Qx%\*%Yt+eta\*t(Delta)%\*%(theta-1/eta\*upsilon))

betahat=solve(t(X)%\*%W%\*%X)%\*%t(X)%\*%W%\*%(Yt-Z%\*%ahat)

theta1=theta

for(i in 1:(m-1)){

Pi=ahat[i]-ahat[(1:m)>i]+1/eta\*upsilon[((i-1)\*(2\*m-i)/2+1):((i-1)\*(2\*m-i)/2+m-i)]

theta[((i-1)\*(2\*m-i)/2+1):((i-1)\*(2\*m-i)/2+m-i)]=Theta\_SCAD(Pi,lambda)$Ta

upsilon[((i-1)\*(2\*m-i)/2+1):((i-1)\*(2\*m-i)/2+m-i)]=upsilon[((i-1)\*(2\*m-i)/2+1):((i-1)\*(2\*m-i)/2+m-i)]+eta\*(ahat[i]-ahat[(1:m)>i]-theta[((i-1)\*(2\*m-i)/2+1):((i-1)\*(2\*m-i)/2+m-i)])

}

r=Delta%\*%ahat-theta

s=eta\*t(Delta)%\*%(theta-theta1)

primal=sqrt(sum(r^2))

dual=sqrt(sum(s^2))

if(primal>10\*dual){

eta=2\*eta

}else{

eta=eta

}

eps\_p=eps\_abs\*sqrt(m\*(m-1)/2)+eps\_rel\* max(sqrt(sum((Delta%\*%ahat)^2)),sqrt(sum((theta)^2)))

eps\_d=eps\_abs\*sqrt(m)+eps\_rel\*sqrt(sum((t(Delta)%\*%upsilon)^2))

if(primal<eps\_p&dual<eps\_d){

break

}

if(counter>500){

break

}

counter=counter+1

}

list(ahat=ahat,betahat=betahat,phi=phi)

}

nrep=100

m=50

p=2

c1=m\*0.1

c2=m\*0.2

a1=-1

a2=1

eta=1

v=3

ngroup=rep(0,nrep)

aest=matrix(0,nrep,m)

betaest=matrix(0,nrep,p)

aMSE=rep(0,nrep)

se.beta=matrix(0,nrep,p)

cp.beta=matrix(0,nrep,p)

RI=matrix(0,nrep,4)

se.a=NULL

cp.a=NULL

Aest=NULL

Mu=NULL

betaran=matrix(0,nrep,p)

aMSE\_ran=rep(0,nrep)

se.beta\_ran=matrix(0,nrep,p)

cp.beta\_ran=matrix(0,nrep,p)

betafix=matrix(0,nrep,p)

se.beta\_fix=matrix(0,nrep,p)

cp.beta\_fix=matrix(0,nrep,p)

aMSE\_fix=rep(0,nrep)

betaor=matrix(0,nrep,p)

se.beta\_or=matrix(0,nrep,p)

cp.beta\_or=matrix(0,nrep,p)

aMSE\_or=rep(0,nrep)

aor=matrix(0,nrep,m)

Aor=matrix(0,nrep,3)

se.a\_or=matrix(0,nrep,3)

cp.a\_or=matrix(0,nrep,3)

Im=diag(m)

Delta=NULL

for(i in 1:(m-1)){

Delta=cbind(Delta,(Im[,i]-Im[,(1:m)>i]))

}

Delta=t(Delta)

for(irep in 1:nrep){

set.seed(irep)

ni=round(runif(m,50,100))

N=sum(ni)

beta=c(0.2,0.2)

xi=seq(0.2,0.3,by=0.02)

nxi=length(xi)

atrue=c(rep(a1,c1),rep(0,m-c1-c2),rep(a2,c2))

J1=matrix(rep(1:p,times=p),byrow=FALSE, nrow=p)

K1=matrix(rep(1:p,times=p),byrow=TRUE, nrow=p)

sigma=0.2^abs(J1-K1)

X=mvrnorm(N,rep(0,p),sigma)

Z=matrix(0,N,m)

id=NULL

a=NULL

for(k in 1:m){

Z[(sum(ni[(1:m)<k])+1):(sum(ni[(1:m)<k])+ni[k]),k]=rep(1,ni[k])

id=c(id,rep(k,ni[k]))

a=c(a,rep(atrue[k],ni[k]))

}

ax=as.vector(Z%\*%atrue+X%\*%beta)

Y=rnbinom(N,exp(ax),0.5)

Data=as.data.frame(cbind(id,X,Y,a))

####fixed effect model#####

fix=glm(Y~-1+X+factor(id),family=quasipoisson,data=Data)

summary(fix)

afix=as.vector(coef(fix)[-(1:p)])

betafix[irep,]=coef(fix)[1:p]

aMSE\_fix[irep]=sum((atrue-afix)^2)/m

se.beta\_fix[irep,]=summary(fix)$coef[1:p,2]

cp.beta\_fix[irep,]=ifelse(abs(betafix[irep,]-beta)<1.96\*se.beta\_fix[irep,],1,0)

Varf=as.vector(summary(fix)$coef[-(1:p),2])

####random effect model####

model=glmer.nb(Y~-1+X+(1|factor(id)),family=quasipoisson,data=Data)

summary(model)

betaran[irep,]=as.numeric(coef(model)$"factor(id)"[1,-1])

aran=coef(model)$"factor(id)"[,1]

aMSE\_ran[irep]=sum((atrue-aran)^2)/m

se.beta\_ran[irep,]=summary(model)$coef[,2]

cp.beta\_ran[irep,]=ifelse(abs(betaran[irep,]-beta)<1.96\*se.beta\_ran[irep,],1,0)

#####fused effects model####

a0=as.vector(afix)

b0=as.matrix(as.numeric(betafix[irep,]))

up0=rep(0,m)

theta0=NULL

upsilon0=NULL

for(i in 1:(m-1)){

theta0=c(theta0,(a0[i]-a0[(1:m)>i]))

upsilon0=c(upsilon0,(up0[i]-up0[(1:m)>i]))

}

Group=rep(0,nxi)

bic=rep(0,nxi)

b1=rep(0,nxi)

b2=rep(0,nxi)

ascad=list()

betascad=list()

Phi=rep(0,nxi)

for(i in 1:nxi){

value=Fn\_SCAD(lambda=xi[i])

ascad[[i]]=value$ahat

betascad[[i]]=value$betahat

Phi[i]=value$phi

ai=ascad[[i]]

j=1

group=list()

group[[1]]=which(abs(ai-ai[1])<0.2)

while(length(unlist(group))<m){

ai[-unlist(group)][which(abs(ai[-unlist(group)]-ai[-unlist(group)][1])<0.2)]=j

group[[j+1]]=which(ai==j)

j=j+1

}

mu=as.numeric(lapply(1:length(group),function(x)1/length(group[[x]])\*sum(ascad[[i]][group[[x]]])))

df=cbind(lengths(group),mu)

df=df[order(df[,2]),]

print(df)

Group[i]=length(group)

b1[i]=-2\*sum(Y\*(Z%\*%ascad[[i]]+X%\*%betascad[[i]])-exp(Z%\*%ascad[[i]]+X%\*%betascad[[i]]))

Cn=log(N+p)

b2[i]=log(N)\*(Group[i]+p)

bic[i]=b1[i]+Cn\*b2[i]

}

I=which.min(bic)

aI=ascad[[I]]

phiest=Phi[I]

j=1

group0=list()

group0[[1]]=which(abs(aI-aI[1])<0.2)

while(length(unlist(group0))<m){

aI[-unlist(group0)][which(abs(aI[-unlist(group0)]-aI[-unlist(group0)][1])<0.2)]=j

group0[[j+1]]=which(aI==j)

j=j+1

}

mu0=as.numeric(lapply(1:length(group0),function(x)1/length(group0[[x]])\*sum(ascad[[I]][group0[[x]]])))

group=lapply(order(mu0),function(x)group0[[x]])

mu=mu0[order(mu0)]

df=cbind(lengths(group),mu)

print(df)

ngroup[irep]=length(group)

id1=rep(0,N)

Z1=matrix(0,m,ngroup[irep])

for(k in 1:ngroup[irep]){

Z1[group[[k]],k]=1

id1[unlist(lapply(group[[k]],function(x)(sum(ni[1:x])-ni[x]+1):sum(ni[1:x])))]=k

}

data1=data.frame(X,Y,id1)

para=c(betascad[[I]],mu)

aest[irep,]=Z1%\*%para[-(1:p)]

aMSE[irep]=sum((atrue-aest[irep,])^2)/m

betaest[irep,]=para[1:p]

#####calculate the se#########

s1=as.vector(Z%\*%aest[irep,]+X%\*%betaest[irep,])

W=diag(exp(s1)/phiest)

U=cbind(X,Z%\*%Z1)

A=solve(t(U)%\*%W%\*%U)

se=sqrt(diag(A))

###################################

se.beta[irep,]=se[1:p]

cp.beta[irep,]=ifelse(abs(betaest[irep,]-beta)<1.96\*se.beta[irep,],1,0)

RI[irep,]=comPart(round(aest[irep,]),round(atrue))

if(ngroup[irep]==3){

Aest=rbind(Aest,para[-(1:p)])

se.a=rbind(se.a,se[-(1:p)])

cp.a=rbind(cp.a,ifelse(abs(para[-(1:p)]-c(a1,0,a2))<1.96\*se[-(1:p)],1,0))

Mu=rbind(Mu,mu)

}else{

Aest=rbind(Aest,c(0,0,0))

se.a=rbind(se.a,c(0,0,0))

cp.a=rbind(cp.a,c(0,0,0))

Mu=rbind(Mu,c(0,0,0))

}

###Oracle####

gk=list()

id\_or=rep(0,N)

gk[[1]]=1:c1

gk[[2]]=(c1+1):(m-c2)

gk[[3]]=(m-c2+1):m

id\_or[unlist(lapply(gk[[1]],function(t)which(id==t)))]=1

id\_or[unlist(lapply(gk[[2]],function(t)which(id==t)))]=2

id\_or[unlist(lapply(gk[[3]],function(t)which(id==t)))]=3

data2=data.frame(X,Y,id\_or)

model\_or=glm(Y~-1+X+factor(id\_or),family=poisson,data=data2)

summary(model\_or)

para\_or=as.numeric(coef(model\_or))

se\_or=as.numeric(summary(model\_or)$coef[,2])

betaor[irep,]=para\_or[1:p]

aor[irep,gk[[1]]]=para\_or[p+1]

aor[irep,gk[[2]]]=para\_or[p+2]

aor[irep,gk[[3]]]=para\_or[p+3]

se.beta\_or[irep,]=se\_or[1:p]

cp.beta\_or[irep,]=ifelse(abs(betaor[irep,]-beta)<1.96\*se.beta\_or[irep,],1,0)

aMSE\_or[irep]=sum((atrue-aor[irep,])^2)/m

Aor[irep,]=para\_or[-(1:p)]

se.a\_or[irep,]=se\_or[-(1:p)]

cp.a\_or[irep,]=ifelse(abs(para\_or[-(1:p)]-c(a1,0,a2))<1.96\*se\_or[-(1:p)],1,0)

}