STA601: Homework 8

Akbota Anuarbek

Randomly generate β_i's:

- randomly choose 5 integers from 1 to 100, say *2, 12, 48, 61, 99.*
- draw $\beta_i \sim N(0,2)$ for $i \in \{2,12,48,61,99\}$
- set $\beta_i = 0$ for $i \notin \{2,12,48,61,99\}$, also set $\beta_0 = 0$

Generate n data observations using β_i's:

- $Y_i = \beta_1 x_1 + \beta_2 x_2 ... + \beta_{100} x_{100} + e_i$
- $e_i \sim N(0,1)$
- $X \sim N(\mu, \Sigma)$, where μ is a zero vector and $\Sigma = I_{100}$

(i) Fully Bayes Ridge

- $y \sim N(X\beta, \sigma^2 I_n)$
- $\beta \sim N(0, \tau^{-1}\sigma^2 I_p)$
- $\sigma^{-2} \sim Gamma(a,b)$
- $\tau \sim Gamma(c,d)$

<u>Full conditionals:</u>

$\pi(\beta|\sigma^2,\tau) \propto N(M,V)$, where $M = Vx^T y\sigma^{-2}$, $V = (x^T x\sigma^{-2} + \sigma^{-2}\tau I_p)^{-1}$

$\pi(\sigma^{-2}|\beta,\tau) \propto Gamma(a + n/2, b + \frac{1}{2}(y - X\beta)^T(y - X\beta))$

$\pi(\tau|\sigma^2,\beta) \propto Gamma(c + p/2, d + \frac{1}{2}\sigma^{-2}\beta^T\beta)$

Run Gibbs sampler in R, letting $a = b = 1$, $c = d = 0.5$

(ii) t-prior with v=1

- $y \sim N(X\beta, \sigma^2 I_n)$
- $\beta \sim N(0, \Sigma_0)$, where $\Sigma_0 = \tau^{-1}\sigma^2\Sigma_\lambda$, Σ_λ is a diag. matrix with elements from λ_1^{-1} to λ_p^{-1}
- $\lambda_j \sim Gamma(\frac{v}{2}, \frac{v}{2})$ are i.i.d.
- $\sigma^{-2} \sim Gamma(a,b)$
- $\tau \sim Gamma(c,d)$

Full conditionals:

$\pi(\beta|\sigma^2, \tau, \lambda_{1:p}) \propto N(M, V)$, where $M = Vx^T y\sigma^{-2}$, $V = (x^T x\sigma^{-2} + \Sigma_0^{-1})^{-1}$

$\pi(\sigma^{-2}|\beta, \tau, \lambda_{1:p}) \propto Gamma(a + n/2, b + \frac{1}{2}(y - X\beta)^T(y - X\beta))$

$\pi(\tau|\lambda_{1:p}, \sigma^2, \beta) \propto Gamma(c + \frac{p}{2}, d + \frac{1}{2}\sigma^{-2}\beta^T\Sigma_0^{-1}\beta)$

$\pi(\lambda_j|\tau, \beta, \sigma^2) \propto Gamma(\frac{v+2}{2}, \frac{v}{2} + \frac{1}{2}\tau\sigma^{-2}\beta_j^2)$

Run Gibbs sampler in R for v=1.

(ii) t-prior with v=1

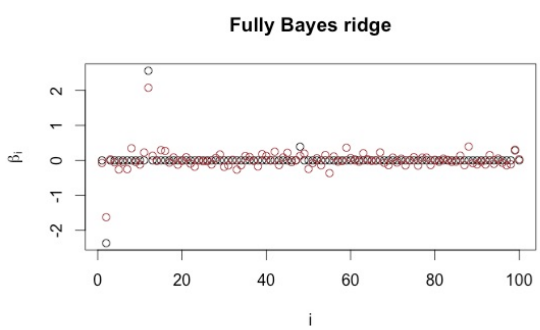
Full conditionals are the same as in part (ii), run Gibbs sampler in R for v=0.01.

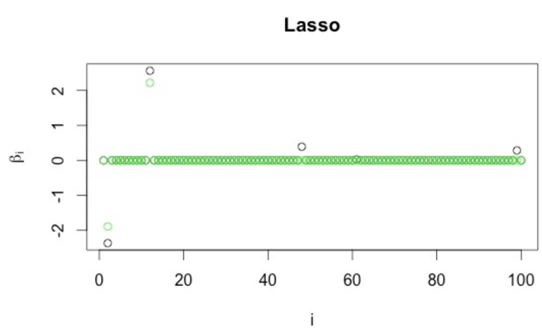
(iv) Lasso

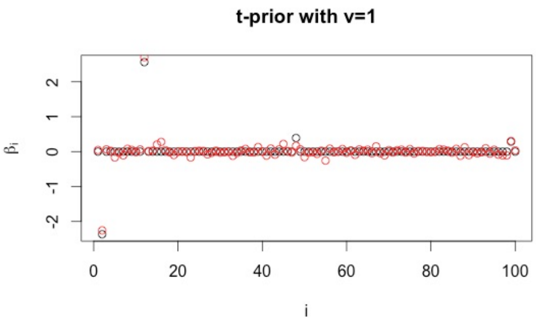
Use R lars package for estimating coefficients.

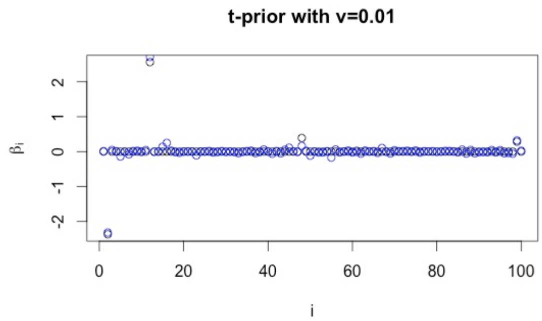
Now let's compare all 4 methods. Plots of coefficients β_i are given below for all methods. Black points are true values of parameters β_i.

Overall, Bayes ridge does not perform very well, this is because we are using only n=100 observations to estimate p=100 parameters in linear regression.
However, when we use t-prior the performance becomes very well, it estimates non-zero parameters quite precisely. T-prior with v=0.001 performs even better in estimating parameters with value 0, compared to t-prior with v=1.
Finally, lasso does a very good job in estimating parameters with value zero, however its estimation of non-zero parameters are less precise than that of t-priors.









We can generate different data an check performance of our parameters for different methods. MSE's for Y are given below. Overall, t-prior with v=0.001 has the lowest MSE, whereas Bayes Ridge has the highest MSE.

	Bayes Ridge	**t-prior v=1**	**t-prior v=0.001**	**Lasso**
MSE	3.46	1.49	1.16	1.34

R code for the assignment is given below:

```
#randomly generate betas
p<-100
z<-sample(1:p, 5, replace=FALSE)
beta<-rep(0, p)
for(i in 1:5){
  beta[z[i]]<-rnorm(1, mean=0, sd=sqrt(2))
}

#generate data using betas
library(MASS)
n<-100
err<-rnorm(n, mean=0, sd=1)
mu<-rep(0,p)
Sigma=diag(p)
x<-mvrnorm(n=n,mu=mu,Sigma=Sigma)
y<-x%*%beta+err

#fully Bayes ridge
N<-2000
a=1
b=1
c=0.5
d=0.5
```

```
tau<-c(1)
sig_sq<-c(1)
B<-matrix(rep(1, p), N, p)

for(i in 2:N){
  V=solve((t(x)%*%x/sig_sq[i-1]+tau[i-1]*diag(p)/sig_sq[i-1]))
  M=V%*%t(x)%*%y/sig_sq[i-1]
  B[i,]<-mvrnorm(n=1, mu=M , Sigma=V)

  SSR=t(y-x%*%B[i,])%*%(y-x%*%B[i,])
  f=rgamma(1, shape=a+n/2, rate=b+SSR/2)
  sig_sq[i]=1/f

  tau[i]=rgamma(1, shape=c+p/2, rate=d+0.5*t(B[i,])%*%B[i,]/sig_sq[i])

  if(i%%500==0){
    print(i)
  }
}

beta_bayes<-c()
for(j in 1:p){
  beta_bayes[j]<-mean(B[200:N,j])
}

#t-prior with v=1
N<-2000
a=1
b=1
c=0.5
d=0.5
v=1

tau<-c(1)
sig_sq<-c(1)
B<-matrix(rep(1, p), N, p)
lambda<-matrix(rep(1,p), N, p)

for(i in 2:N){
  Sigma_l=diag(1/lambda[i-1,1:p])
  Sigma0=sig_sq[i-1]*(tau[i-1]^(-1))*Sigma_l
  V=solve(t(x)%*%x/sig_sq[i-1]+solve(Sigma0))
  M=V%*%t(x)%*%y/sig_sq[i-1]
  B[i,]=mvrnorm(n=1, mu=M, Sigma=V)
```

```
  SSR=t(y-x%*%B[i,])%*%(y-x%*%B[i,])
  f=rgamma(1, shape=a+n/2, rate=b+SSR/2)
  sig_sq[i]=1/f

 tau[i]=rgamma(1, shape=c+p/2, rate=d+0.5/sig_sq[i]*t(B[i,])%*%solve(Sigma_l)%*%B[i,])

  for(j in 1:p){
    lambda[i,j]=rgamma(1,shape=(v+1)/2, rate=(v+tau[i]/sig_sq[i]*B[i,j]^2)/2)
  }

  if(i%%500==0){
    print(i)
  }
}

beta_t1<-c()
for(j in 1:p){
  beta_t1[j]<-mean(B[200:N,j])
}

#t-prior with v=0.001
N<-2000
a=1
b=1
c=0.5
d=0.5
v=0.001

tau<-c(1)
sig_sq<-c(1)
B<-matrix(rep(1, p), N, p)
lambda<-matrix(rep(1,p), N, p)

for(i in 2:N){
  Sigma_l=diag(1/lambda[i-1,1:p])
  Sigma0=sig_sq[i-1]*(tau[i-1]^(-1))*Sigma_l
  V=solve(t(x)%*%x/sig_sq[i-1]+solve(Sigma0))
  M=V%*%t(x)%*%y/sig_sq[i-1]
  B[i,]=mvrnorm(n=1, mu=M, Sigma=V)

  SSR=t(y-x%*%B[i,])%*%(y-x%*%B[i,])
  f=rgamma(1, shape=a+n/2, rate=b+SSR/2)
  sig_sq[i]=1/f
```

```
 tau[i]=rgamma(1, shape=c+p/2, rate=d+0.5/sig_sq[i]*t(B[i,])%*%solve(Sigma_l)%*%B[i,])

  for(j in 1:p){
    lambda[i,j]=rgamma(1,shape=(v+1)/2, rate=(v+tau[i]/sig_sq[i]*B[i,j]^2)/2)
  }

  if(i%%500==0){
    print(i)
  }
}

beta_t2<-c()
for(j in 1:p){
  beta_t2[j]<-mean(B[200:N,j])
}

#Lasso
las <- lars(x, y, type="lasso", intercept=FALSE)
beta_lasso = coef(las, s=5, mode="lambda")

#plots
plot(beta, ylab=expression(beta[i]), xlab=expression(i), main="Fully Bayes ridge")
points(beta_bayes, col="brown")

plot(beta, ylab=expression(beta[i]), xlab=expression(i), main="t-prior with v=1")
points(beta_t1, col="red")

plot(beta, ylab=expression(beta[i]), xlab=expression(i), main="t-prior with v=0.01")
points(beta_t2, col="blue")

plot(beta, ylab=expression(beta[i]), xlab=expression(i), main="Lasso")
points(beta_lasso, col="green")

#MSE for Y
mse_bayes=t(y-x%*%beta_bayes)%*%(y-x%*%beta_bayes)/n
mse_t1=t(y-x%*%beta_t1)%*%(y-x%*%beta_t1)/n
mse_t2=t(y-x%*%beta_t2)%*%(y-x%*%beta_t2)/n
mse_lasso=t(y-x%*%beta_lasso)%*%(y-x%*%beta_lasso)/n
```