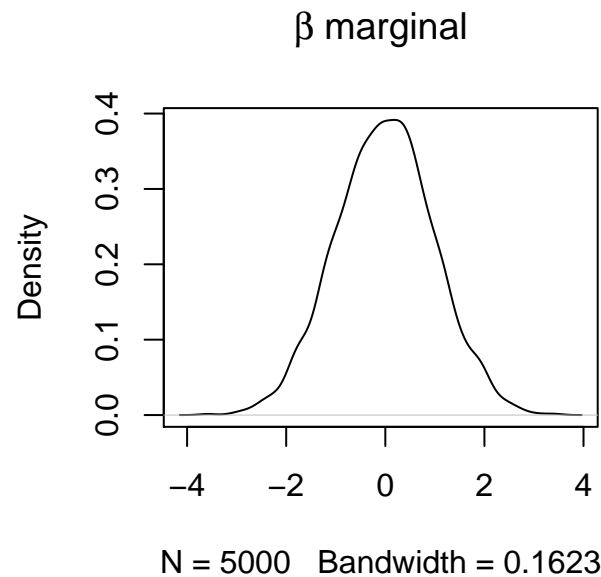# Homework 1

Ian Frankenburg

4/7/2020

Bayesian Adaptive Lasso

Part a.

Consider $p = 1$. Simulate 5,000 Monte Carlo samples from the conditional prior $\beta|\tau^2 = 1$ and obtain a plot of the density using the R function density.

```
n <- 5000
plot(density(rnorm(n,0,1)), main=TeX(paste("$\\beta$", "marginal")))
```


β marginal

N = 5000   Bandwidth = 0.1623

Part b.

Consider $p = 1$. Simulate 5,000 Monte Carlo samples from the marginal prior $\boldsymbol{\beta}$, considering $\lambda^2 = 2$, so that $\mathbb{E}(\tau^2 | \lambda) = 1$. Obtain a plot of the density as in **a.**

```r
lambda <- sqrt(2)
tau.sq <- rgamma(n,shape=1,rate = lambda^2/2)
beta.marginal <- rnorm(n,0,sqrt(tau.sq))
plot(density(beta.marginal), main=TeX(paste("$\\lambda^2 = 2$")), xlim=c(-5,5))
```
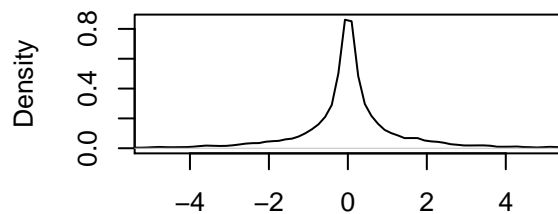
$$\lambda^2 = 2$$
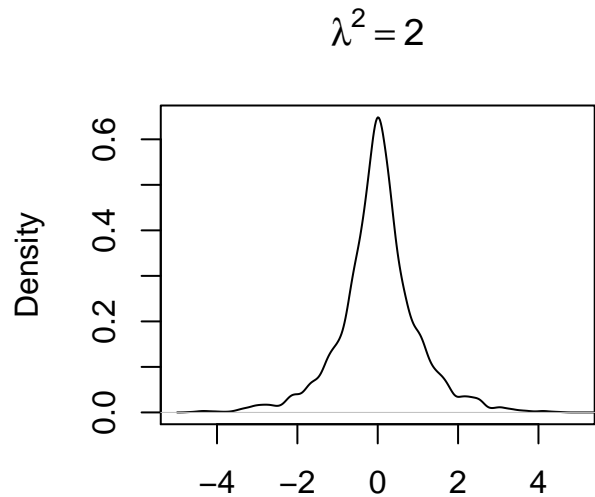
N = 5000   Bandwidth = 0.1147

Consider $p = 1$. Add a hyper prior on $\gamma = 1/\gamma \sim Gamma(a, rate = b)$. Assess how the marginal prior of $\boldsymbol{\beta}$ changes for $a = 1$ and values of $b \geq 1$.
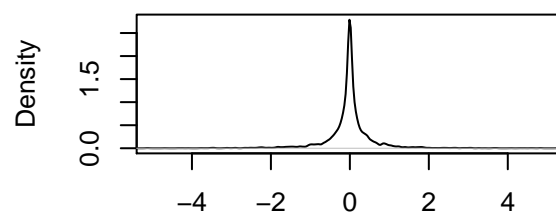
```r
set.seed(1)
par(mfrow=c(2,2))
rates <- c(1,3,5,10)
for(b in rates){
  lambda <- 1/rgamma(n,1,b)
  tau.sq <- rgamma(n,shape=1,rate = lambda^2/2)
  beta.marginal <- rnorm(n,0,sqrt(tau.sq))
  plot(density(beta.marginal), main=paste("rate b = ",b),xlim=c(-5,5))
}
```
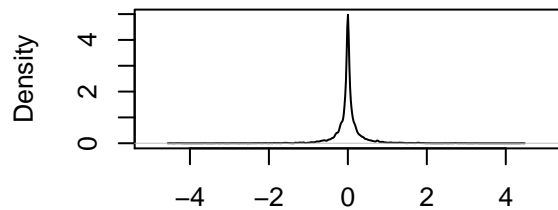
rate b = 1
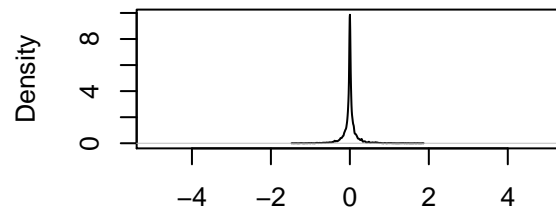

N = 5000   Bandwidth = 0.09665

rate b = 3

N = 5000   Bandwidth = 0.03314

rate b = 5

N = 5000   Bandwidth = 0.01929

rate b = 10

N = 5000   Bandwidth = 0.009661

Considering the hyper prior in **c.**, describe a Markov Chain Monte Carlo algorithm to sample from the posterior distribution of $\boldsymbol{\beta}$ and σ^2.

I will implement a joint Gibbs and Metropolis sampler. The model is

$$\boldsymbol{Y}|\boldsymbol{\beta},\sigma^2 \sim N(\boldsymbol{X}\boldsymbol{\beta},\sigma^2\boldsymbol{I})$$
$$\beta_j|\tau_j^2 \sim N(0,\tau_j^2)$$
$$\tau_j^2 \sim \text{Gamma}(1,\frac{\lambda^2}{2})$$
$$\lambda^2 \sim \text{Inverse-Gamma}(a,1/b)$$
$$\sigma^2 \sim \text{Inverse-Gamma}(0.1,0.1).$$

I need the full conditionals

$$\{\beta_1,\ldots,\beta_p|\boldsymbol{Y},\sigma^2,\tau_1^2,\ldots,\tau_p^2,\lambda\},$$
$$\{\sigma^2|\boldsymbol{Y},\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\lambda\},$$
$$\{\tau_1^2,\ldots,\tau_p^2|\boldsymbol{Y},\beta_1,\ldots,\beta_p,\sigma^2,\lambda\},$$
$$\{\lambda|\boldsymbol{Y},\beta_1,\ldots,\beta_p,\sigma^2,\tau_1^2,\ldots,\tau_p^2\}$$

which are all proportional to

$$p(\boldsymbol{Y}|\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\sigma^2,\lambda) \times p(\beta_1,\ldots,\beta_p|\tau_1^2,\ldots,\tau_p^2) \times p(\tau_1^2,\ldots,\tau_p^2|\lambda)p(\lambda)p(\sigma^2)$$

so I'll start with the posterior

$$p(\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\sigma^2,\lambda|\boldsymbol{Y}) \propto p(\boldsymbol{Y}|\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\sigma^2,\lambda)$$
$$\times p(\beta_1,\ldots,\beta_p|\tau_1^2,\ldots,\tau_p^2)$$
$$\times p(\tau_1^2,\ldots,\tau_p^2|\lambda)p(\lambda)p(\sigma^2).$$

As a function of just σ^2, this is proportional to

$$p(\boldsymbol{Y}|\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\sigma^2,\lambda)p(\sigma^2)$$
$$= N(\boldsymbol{X}\boldsymbol{\beta},\sigma^2\boldsymbol{I})IG(a,b).$$

Time to show this is inverse-gamma distributed.

$$N(\boldsymbol{y};\boldsymbol{X}\boldsymbol{\beta},\sigma^2\boldsymbol{I})IG(\sigma^2;q,r)$$
$$\propto (\sigma^2)^{-n/2}\exp\left\{-\frac{1}{2\sigma^2}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})\right\}(\sigma^2)^{q-1}\exp\left\{-\frac{r}{\sigma^2}\right\}$$
$$= (\sigma^2)^{-(n/2+q)-1}\exp\left\{-\frac{1}{\sigma^2}(r+\frac{1}{2}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})\right\}$$
$$= IG(n/2+q,r+(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})/2)|_{q=0.1,r=0.1}$$

As a function of $\boldsymbol{\beta}$, the conditional is proportional to

$$p(\boldsymbol{Y}|\beta_1,\ldots,\beta_p,\tau_1^2,\ldots,\tau_p^2,\sigma^2,\lambda)p(\beta_1,\ldots,\beta_p|\tau_1^2,\ldots,\tau_p^2)$$
$$= N(\boldsymbol{X}\boldsymbol{\beta},\sigma^2\boldsymbol{I})\cdot\prod_{i=1}^{p}N(0,\tau_i^2)$$
$$= N(\boldsymbol{X}\beta,\Sigma)\cdot N(0,\Omega), \text{ where } \Omega=\text{diag}(\tau_1^2,\ldots,\tau_p^2)$$
$$= N(m,\boldsymbol{M})$$

with $m = \boldsymbol{M}\boldsymbol{X}^\top\Sigma^{-1}y$ and $\boldsymbol{M} = (\boldsymbol{X}^\top\Sigma^{-1}\boldsymbol{X} + \Omega^{-1})^{-1}$. As a function of $\tau_1^2, \ldots, \tau_p^2$, the conditional is non-standard, but it's proportional to

$$p(\beta_1, \ldots, \beta_p | \tau_1^2, \ldots, \tau_p^2)p(\tau_1^2, \ldots, \tau_p^2 | \lambda)$$

$$= \prod_{i=1}^{p} N(0, \tau_i^2) \cdot \prod_{i=1}^{p} IG(1, \frac{\lambda^2}{2})$$

Finally, as a function of λ, it's proportional to

$$p(\tau_1^2, \ldots, \tau_p^2 | \lambda)p(\lambda)$$

$$= \prod_{i=1}^{p} IG(1, \frac{\lambda^2}{2}) \cdot IG(0.1, 0.1)$$

Now I can build an algorithm to iteratively update through these conditional distributions. I take the starting value of $\boldsymbol{\beta}^{(0)}$ to be the least-squares solution $\hat{\boldsymbol{\beta}}$.

Result: Samples from joint posterior $p(\boldsymbol{\beta}, \sigma^2 | \boldsymbol{y})$

for s *in # samples* **do**

$\quad$ $\sigma^{2(s+1)} \sim IG(n/2 + a, 2b + (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{(s)})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{(s)})/2)$

$\quad$ $\boldsymbol{\beta}^s \sim N_p(m, \boldsymbol{M})$

$\quad$ $m^\top = \qquad \boldsymbol{M} = \tau_1^2, \ldots, \tau_p^2 \sim N(\boldsymbol{\tau}^{(s)}, \delta\boldsymbol{I})^2$

$\quad$ $\lambda^* \sim N(\lambda.s, \delta)^2$

$\quad$ $\log(r) = \log p(\boldsymbol{y} | \boldsymbol{\beta}^*, \tau_1^2, \ldots, \tau_p^2, \sigma^{2(s)}) + \log p(\boldsymbol{\beta}^* | \tau_1^2, \ldots, \tau_p^2)$

$\quad\qquad\qquad - \log p(\boldsymbol{y} | \boldsymbol{\beta}^{(s)}, \tau_1^2, \ldots, \tau_p^2, \sigma^{2(s)}) - \log p(\boldsymbol{\beta}^{(s)} | \tau_1^2, \ldots, \tau_p^2)$

$\quad$ $u \sim Unif(0, 1)$

$\quad$ **if** $\log(u) < \log(r)$ **then**

$\quad\quad |\quad \boldsymbol{\tau}^{(s+1)} = \boldsymbol{\beta}^*$

$\quad$ **else**

$\quad\quad |\quad \boldsymbol{\tau}^{(s+1)} = \boldsymbol{\beta}^{(s)}$

$\quad$ **end**

end

Algorithm 1: Gibbs and Metropolis

5

Part f.

Implement such algorithm in R and compare your results with estimates obtained using **glmnet()**. In particular, you should test your results on the diabetes data available from lars, (use the matrix of predictors x).

```r
loglambda.target <- function(lambda, a, b, tau2){
  (-a - 1)*log(lambda) - (lambda^2/2 * sum(tau2)) - 1/(b*lambda)
}
logtau2j.target <- function(tau2j, lambda, betaj){
  -log(sqrt(tau2j)) - 1/2*(1/tau2j*betaj^2 + lambda^2*tau2j)
}

set.seed(1)
data("diabetes")
X <- cbind(diabetes$x); y <- diabetes$y; n <- nrow(X)
X <- cbind(rep(1,n),X)
p <- ncol(X)
samples <- 50000;
#lambda <- 1e-10
lambda <- 1
tau2 <- rep(1000,p)
beta <- solve(t(X)%*%X)%*%t(X)%*%y
sigma2.keep <- lambda.keep <-rep(0,samples)
beta.keep <-tau2.keep<- matrix(NA,nrow=p,ncol=samples)
sigma2 <-var(y)
sigma2.keep[1] <- sigma2
beta.keep[,1] <- solve(t(X)%*%X)%*%t(X)%*%y
lambda.keep[1] <- lambda
tau2.keep[,1] <- tau2
s <- 2
post.means <- matrix(NA,nrow=p,ncol=length(lambdas))
index <- 1
b<-1
for(s in 2:samples){
  lambda.p <- exp(log(lambda)+rnorm(1,0,lambda))
  logr <- loglambda.target(lambda.p,1,b,tau2)-loglambda.target(lambda,1,b,tau2)+log(lambda.p)-log(lambda
  if(log(runif(1))<logr){lambda <- lambda.p}
  lambda.keep[s] <- lambda
  for(j in 1:length(tau2)){
    tau2j.p <- exp(log(tau2[j])+rnorm(1,0,5))
    logr <-
      logtau2j.target(tau2j.p, lambda, beta[j]) -
      logtau2j.target(tau2[j], lambda, beta[j]) +
      log(tau2j.p)-log(tau2[j])
    if(log(runif(1))<logr){tau2[j] <- tau2j.p}
  }
  # tau2 <- 1/rinvgauss(n = length(beta), mean = lambda/abs(beta),
  #                     shape = lambda^2)
  sigma2 <- rinvgamma(1,shape=n/2+0.1,rate=(0.1+t(y-X%*%beta)%*%(y-X%*%beta)/2))
  M <- solve(t(X)%*%X*1/sigma2+diag(1/tau2))
  m <- M%*%t(X)%*%y/sigma2
  beta <- t(rmvnorm(n=1,mean=m,sigma=M))
```

6

```r
  sigma2.keep[s] <- sigma2
  beta.keep[,s] <- beta
  tau2.keep[,s] <- tau2
}
post.means[,index] <- matrix(rowMeans(beta.keep[,floor(samples/4):samples]),nrow = p)
index <- index +1
}
fit <- glmnet(X, y)
glmnetCoef <- coef(fit,s=1)
plot(fit, xvar="lambda")

fit <- glmnet(X, y)
glmnetCoef <- coef(fit,s=1)
# df <- cbind(beta.ls,colMeans(beta.keep[floor(samples/4):samples,]),glmnetCoef[-1,])
# colnames(df) <- c("Least Square", "Bayes", "Glmnet")
# df
plot(log(lambda.keep[1:samples]), type="l")
beta.ls <- solve(t(X)%*%X)%*%t(X)%*%y
cbind(beta.ls,betaChain)
beta.keep <- t(beta.keep)
tau2.keep <- t(tau2.keep)
plot(sigma2.keep[1:samples],type="l")
plot(log(tau2.keep[,1]),type="l")
plot((tau2.keep[,2]),type="l")
plot((tau2.keep[,3]),type="l")
plot((tau2.keep[,4]),type="l")
plot((tau2.keep[,5]),type="l")
plot((tau2.keep[,6]),type="l")
plot((tau2.keep[,7]),type="l")
```

Free λ and carry out a sensitivity analysis assessing the behavior of the posterior distribution of $\boldsymbol{\beta}$ and σ^2, as hyper parameters a and b are changed. Explain clearly the rationale you use to assess sensitivity and provide recommendations for the analysis of the diabetes data.

```
loglambda.target <- function(lambda, a, b, tau2){
  (-a - 1)*log(lambda) - (lambda^2/2 * sum(tau2)) - 1/(b*lambda)
}
logtau2j.target <- function(tau2j, lambda, betaj){
  -log(sqrt(tau2j)) - 1/2*(1/tau2j*betaj^2 + lambda^2*tau2j)
}

set.seed(1)
data("diabetes")
X <- cbind(diabetes$x); y <- diabetes$y; n <- nrow(X)
X <- cbind(rep(1,n),X)
p <- ncol(X)
samples <- 5000;
#lambda <- 1e-10
lambda <- 1
tau2 <- rep(1000,p)
beta <- solve(t(X)%*%X)%*%t(X)%*%y
sigma2.keep <- lambda.keep <-rep(0,samples)
beta.keep <-tau2.keep<- matrix(NA,nrow=p,ncol=samples)
sigma2 <-var(y)
sigma2.keep[1] <- sigma2
beta.keep[,1] <- solve(t(X)%*%X)%*%t(X)%*%y
lambda.keep[1] <- lambda
tau2.keep[,1] <- tau2
s <- 2
lambdas <- seq(from=-12, to = -1, length.out = 12)
post.means <- matrix(NA,nrow=p,ncol=length(lambdas))
index <- 1
for(l in lambdas){
for(s in 2:samples){
  # lambda.p <- exp(log(lambda)+rnorm(1,0,lambda))
  # logr <- loglambda.target(lambda.p,1,b,tau2)-loglambda.target(lambda,1,b,tau2)+log(lambda.p)-log(lam
  # if(log(runif(1))<logr){lambda <- lambda.p}
  # lambda.keep[s] <- lambda
  lambda <- exp(l)
  j<-1
  for(j in 1:length(tau2)){
    tau2j.p <- exp(log(tau2[j])+rnorm(1,0,5))
    logr <-
      logtau2j.target(tau2j.p, lambda, beta[j]) -
      logtau2j.target(tau2[j], lambda, beta[j]) +
      log(tau2j.p)-log(tau2[j])
    if(log(runif(1))<logr){tau2[j] <- tau2j.p}
  }
  # tau2 <- 1/rinvgauss(n = length(beta), mean = lambda/abs(beta), shape = lambda^2)
  sigma2 <- rinvgamma(1,shape=n/2+0.1,rate=(0.1+t(y-X%*%beta)%*%(y-X%*%beta)/2))
  M <- solve(t(X)%*%X*1/sigma2+diag(1/tau2))
```

```r
  m <- M%*%t(X)%*%y/sigma2
  beta <- t(rmvnorm(n=1,mean=m,sigma=M))
  sigma2.keep[s] <- sigma2
  beta.keep[,s] <- beta
  tau2.keep[,s] <- tau2
}
post.means[,index] <- matrix(rowMeans(beta.keep[,floor(samples/4):samples]),nrow = p)
index <- index +1
}
fit <- glmnet(X, y)
glmnetCoef <- coef(fit,s=1)
plot(fit, xvar="lambda")
betaChain <- colMeans(beta.keep[,1:samples])

plot(lambdas,post.means[2,],ylab="Coefficients",xlab="log lambdas",ylim=c(min(post.means),max(post.means
lines(lambdas,post.means[2,])
for(j in 3:p){
  points(lambdas,post.means[j,],ylab="Coefficients",xlab="log lambdas", col=j,type="l")
lines(lambdas,post.means[j,], col=j)
}

fit <- glmnet(X, y)
plot(fit,xvar = "lambda")
glmnetCoef <- coef(fit,s=1)
# df <- cbind(beta.ls,colMeans(beta.keep[floor(samples/4):samples,]),glmnetCoef[-1,])
# colnames(df) <- c("Least Square", "Bayes", "Glmnet")
# df
beta.ls <- solve(t(X)%*%X)%*%t(X)%*%y
sajd <- glmnetCoef
cbind(beta.ls,post.means,coef(fit,s=1)[-2,])
```

Part g.

Free λ and carry out a sensitivity analysis assessing the behavior of the posterior distribution of β and σ^2, as hyper parameters a and b are changed. Explain clearly the rationale you use to assess sensitivity and provide recommendations for the analysis of the diabetes data.

```r
loglambda.target <- function(lambda, a, b, tau2){
  (-a - 1)*log(lambda) - (lambda^2/2 * sum(tau2)) - 1/(b*lambda)
}
logtau2j.target <- function(tau2j, lambda, betaj){
  -log(sqrt(tau2j)) - 1/2*(1/tau2j*betaj^2 + lambda^2*tau2j)
}
bs <- c(seq(from=0.001,to=1,length.out = 10),seq(from=1.5,to=20,length.out = 10))
set.seed(1)
data("diabetes")
X <- cbind(diabetes$x); y <- diabetes$y; n <- nrow(X)
X <- cbind(rep(1,n),X)
p <- ncol(X)
betasB <- matrix(0,nrow=p,ncol=length(bs))
samples <- 5000;
#lambda <- 1e-10
lambda <- 1
tau2 <- rep(1000,p)
beta <- solve(t(X)%*%X)%*%t(X)%*%y
sigma2.keep <- lambda.keep <-rep(0,samples)
beta.keep <-tau2.keep<- matrix(NA,nrow=p,ncol=samples)
sigma2 <-var(y)
sigma2.keep[1] <- sigma2
beta.keep[,1] <- solve(t(X)%*%X)%*%t(X)%*%y
lambda.keep[1] <- lambda
tau2.keep[,1] <- tau2
s <- 2
lambdas <- seq(from=-12, to = -2, length.out = 12)
post.means <- matrix(NA,nrow=p,ncol=length(lambdas))
index <- 1
bindex <- 1
for(b in bs){
for(s in 2:samples){
  lambda.p <- exp(log(lambda)+rnorm(1,0,lambda))
  logr <- loglambda.target(lambda.p,1,b,tau2)-loglambda.target(lambda,1,b,tau2)+log(lambda.p)-log(lambda
  if(log(runif(1))<logr){lambda <- lambda.p}
  lambda.keep[s] <- lambda
  for(j in 1:length(tau2)){
    tau2j.p <- exp(log(tau2[j])+rnorm(1,0,5))
    logr <-
      logtau2j.target(tau2j.p, lambda, beta[j]) -
      logtau2j.target(tau2[j], lambda, beta[j]) +
      log(tau2j.p)-log(tau2[j])
    if(log(runif(1))<logr){tau2[j] <- tau2j.p}
  }
  # tau2 <- 1/rinvgauss(n = length(beta), mean = lambda/abs(beta),
  #                     shape = lambda^2)
  sigma2 <- rinvgamma(1,shape=n/2+0.1,rate=(0.1+t(y-X%*%beta)%*%(y-X%*%beta)/2))
```

```r
  M <- solve(t(X)%*%X*1/sigma2+diag(1/tau2))
  m <- M%*%t(X)%*%y/sigma2
  beta <- t(rmvnorm(n=1,mean=m,sigma=M))
  sigma2.keep[s] <- sigma2
  beta.keep[,s] <- beta
  tau2.keep[,s] <- tau2
}
post.means[,index] <- matrix(rowMeans(beta.keep[,floor(samples/4):samples]),nrow = p)
post.meansindex <- index +1
betasB[,bindex] <- rowMeans(beta.keep[,1000:samples])
bindex <- bindex+1
}


# df <- cbind(beta.ls,colMeans(beta.keep[floor(samples/4):samples,]),glmnetCoef[-1,])
# colnames(df) <- c("Least Square", "Bayes", "Glmnet")
# df
plot(bs,betasB[2,],ylim = c(min(betasB),max(betasB)),type="l", main="Effect of b", ylab="Coefficients",
for(j in 3:length(bs)){
  points(bs,betasB[j,],ylim = c(min(betasB[j,]),max(betasB[j,])), type="l", col=j)
}
```