STA721 HW6

Zheng Yuan Evan Wyse

2019/10/30

Problem 1

(a)

First,the mle for $\hat{\beta}$ conditional on X is

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

The expected MSE for OLS under the full model can be written as

$$E[(\beta - \hat{\beta})^T(\beta - \hat{\beta})] = tr(IVar(\hat{\beta})) + (E(\beta - \hat{\beta}))^T E(\beta - \hat{\beta}) = tr(Var(\hat{\beta})) + (E(\beta - \hat{\beta}))^T E(\beta - \hat{\beta}),$$

and

$$E(\beta - \hat{\beta}) = \beta - E(\hat{\beta}) = \beta - (X^T X)^{-1} X^T X\beta = \beta - \beta = 0$$

$$Var(\hat{\beta}) = Var((X^T X)^{-1} X^T Y) = \sigma^2 (X^T X)^{-1} X^T X (X^T X)^{-1} = \sigma^2 (X^T X)^{-1}$$

so

$$E[(\beta - \hat{\beta})^T(\beta - \hat{\beta})] = tr(\sigma^2 (X^T X)^{-1}) = \sigma^2 \sum_{i=1}^{p} \frac{1}{\lambda_i},$$

here $\lambda_i' s$ are the eigenvalues of $X^T X$.

(b)

First, we calculate the average of observed MSEs,

```
## [1] 49.56834
```

In order to see whether the average of observed MSEs provides a good estimate of the expected MSE, we can also calculate the expected MSE analytically, that is

$$E[(\beta - \hat{\beta})^T(\beta - \hat{\beta})] = tr(\sigma^2 (X^T X)^{-1})$$

and compare with it.

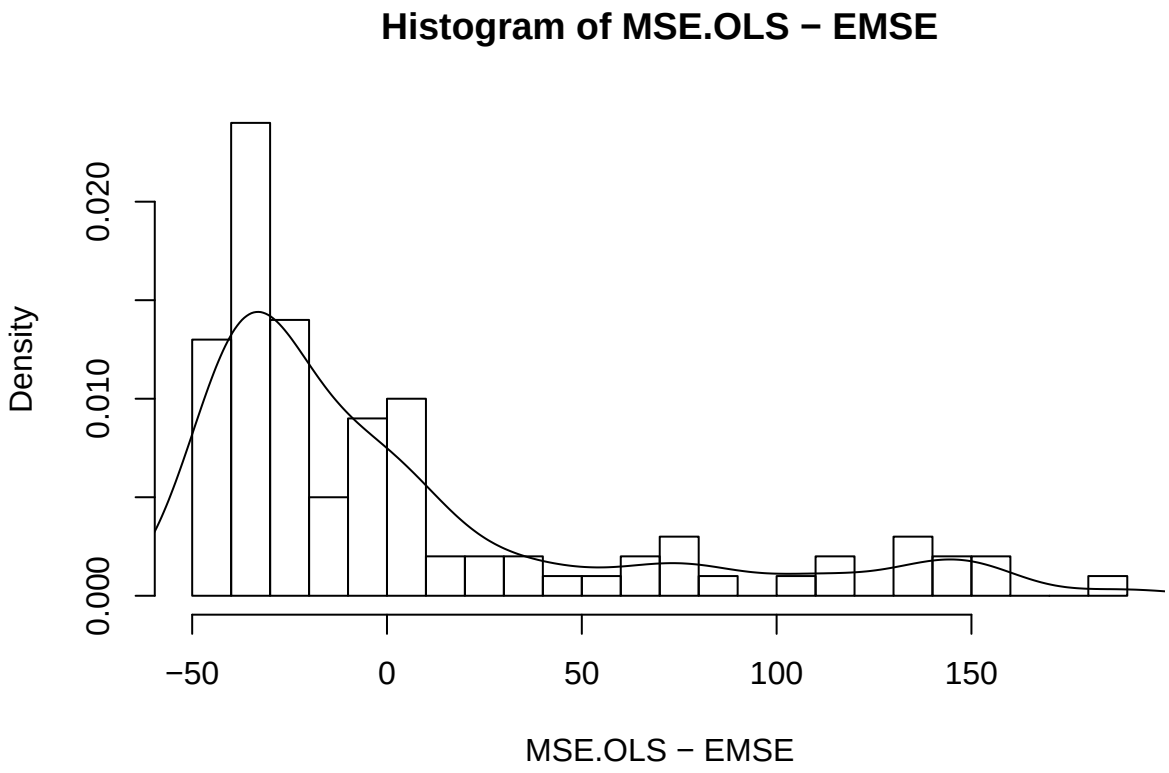
```
EMSE=(sigma^2)*sum(diag(solve(t(X)%*%X)))
EMSE
```

```
## [1] 45.42631
```

After comparison, there is still a unignorable difference between the average of observed MSEs and the expected MSE, so it does not provide a good estimate actually.

The distribution of observed MSE minus expected MSEs:

```
hist(MSE.OLS-EMSE, freq = F, breaks=20);lines(density(MSE.OLS-EMSE))
```



As we can see, the distribution of observed MSE minus expected MSEs looks like a normal distribution centering around its mean, which is around -35. In this case, I don't think increasing the number of simulated date sets would help, because the distribution of observed MSE minus expected MSEs does not center around 0, by the Central Limit Theorem, the average of observed MSEs must not converge to expected MSE in the end. Therefore, in fact, the more simulated date sets we have, the more difference between the average and the expected value there will be, until the difference is stable around a value.

(c)

```
library(lars)
```

```
## Loaded lars 1.2
```

```
library(MASS)
library(matrixStats)
library(monomvn)
```

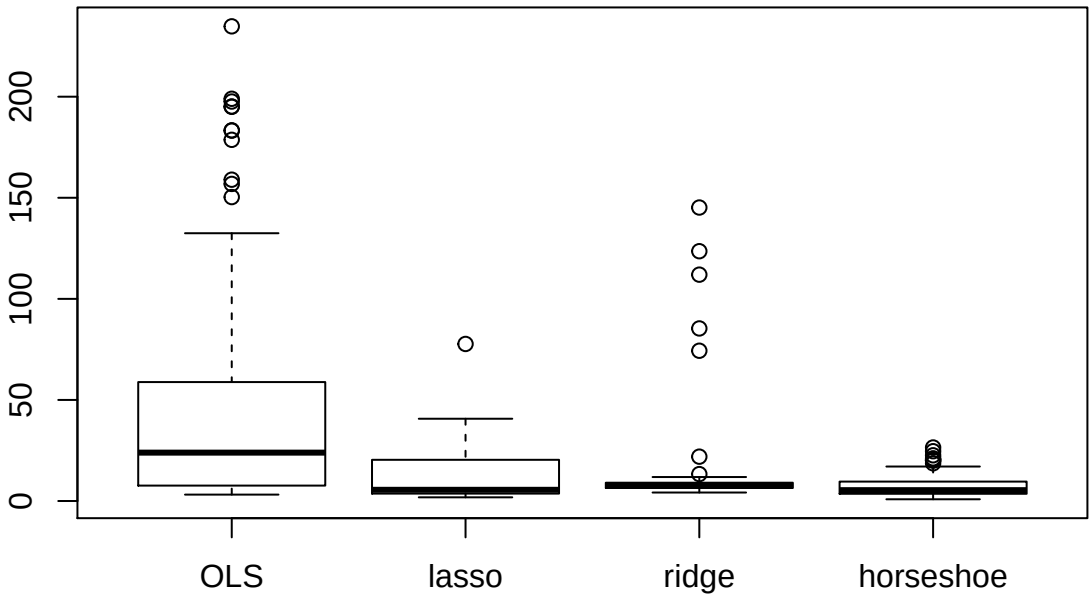
```
## Loading required package: pls
```

```
##
## Attaching package: 'pls'
```

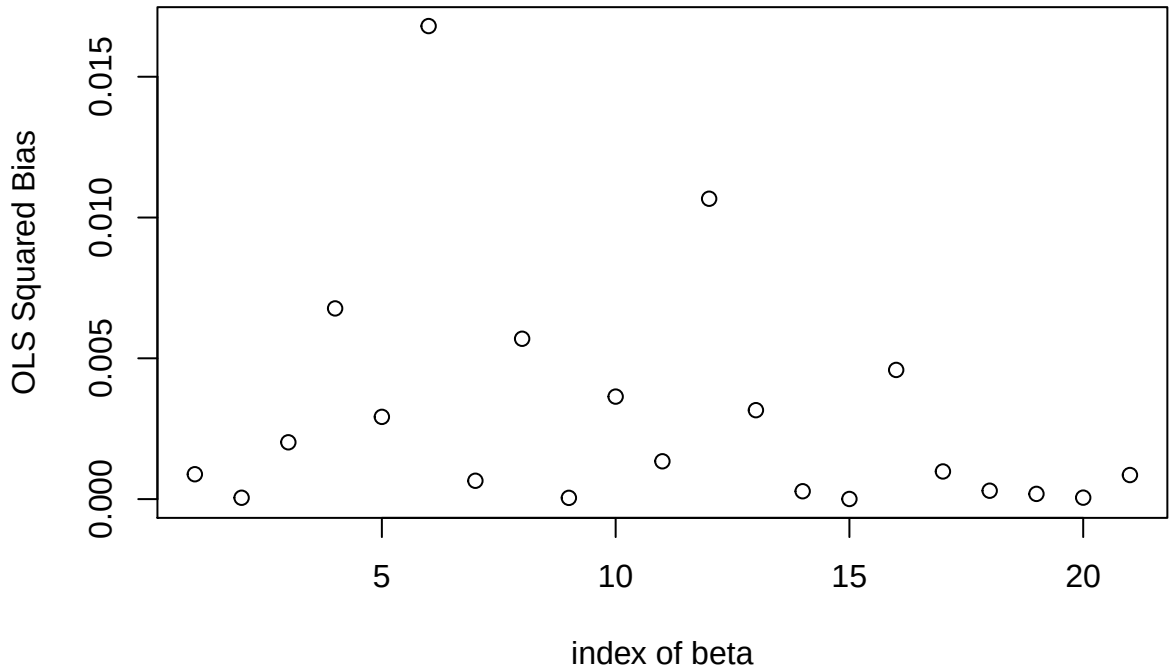
```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
OLS.MSE = rep(0,100)
lasso.MSE = rep(0,100)
```

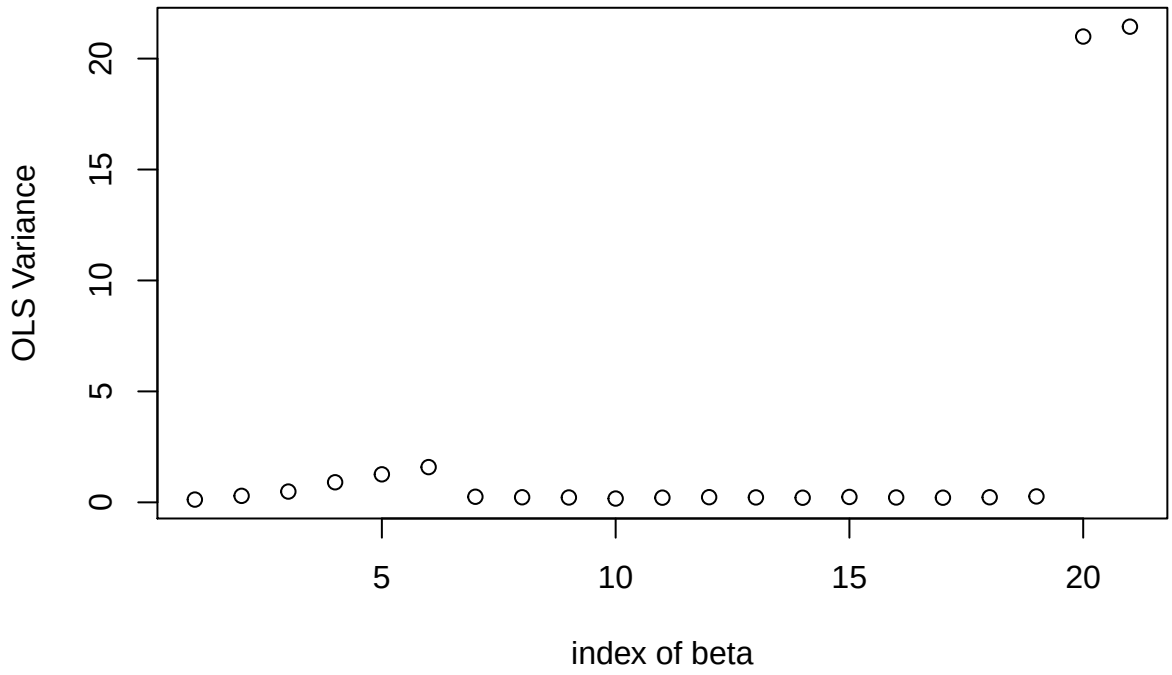
```r
ridge.MSE = rep(0,100)
bhs.MSE = rep(0,100)

Coef.OLS = matrix(rep(0,2100),nrow=100) ## record OLS estimates of beta for each simulation
Coef.lasso = matrix(rep(0,2000),nrow=100)## record lasso estimates of beta for each simulation
Coef.ridge = matrix(rep(0,2100),nrow=100)## record ridge estimates of beta for each simulation
Coef.bhs = matrix(rep(0,2100),nrow=100)## record bhs estimates of beta for each simulation


for( i in 1:100) {
  rm(df)
  load(fname[i])
  X = as.matrix(df[,-1]); Y = df[,1]
  X.scale = scale(X); n = length(Y); X.mean = colMeans(X); X.sd = colSds(X)
  coeftrue = betatrue
  coeftrue[1] = betatrue[1]+sum(betatrue[-1]*X.mean)
  coeftrue[-1] = betatrue[-1]*X.sd


  ## ols
  nk.ols = lm(Y ~ X)
  coef.ols = coef(nk.ols)
  OLS.MSE[i] = sum((betatrue - coef.ols)^2)
  Coef.OLS[i,1:21]=coef.ols

  ## lasso
  nk.lasso = lars(X, Y, type="lasso")
  coef.lasso = coef(nk.lasso, s=which.min(nk.lasso$Cp))
  lasso.MSE[i] = sum((betatrue[-1] - coef.lasso)^2)
  Coef.lasso[i,1:20]=coef.lasso

  ## ridge
  seq.lambda = seq(0,10,0.01)
  nk.ridge = lm.ridge(Y ~ X, lambda = seq.lambda)
  lambda = seq.lambda[which.min(nk.ridge$GCV)]
  nk.ridge = lm.ridge(Y ~ X, lambda = lambda)
  coef.ridge = coef(nk.ridge)
  ridge.MSE[i] = sum((betatrue - coef.ridge)^2)
  Coef.ridge[i,1:21]=coef.ridge

  ## horseshoe
  nk.bhs = blasso(X, Y, case="hs", RJ=FALSE, normalize=F, verb=0)
  coef.bhs = c(mean(nk.bhs$mu), apply(nk.bhs$beta, 2, mean))
  bhs.MSE[i] = sum((betatrue - coef.bhs)^2)
  Coef.bhs[i,1:21]=coef.bhs

}
## Observed MSE for each simulation
boxplot(OLS.MSE, lasso.MSE, ridge.MSE, bhs.MSE, names=c("OLS", "lasso", "ridge", "horseshoe"), main="Ob
```



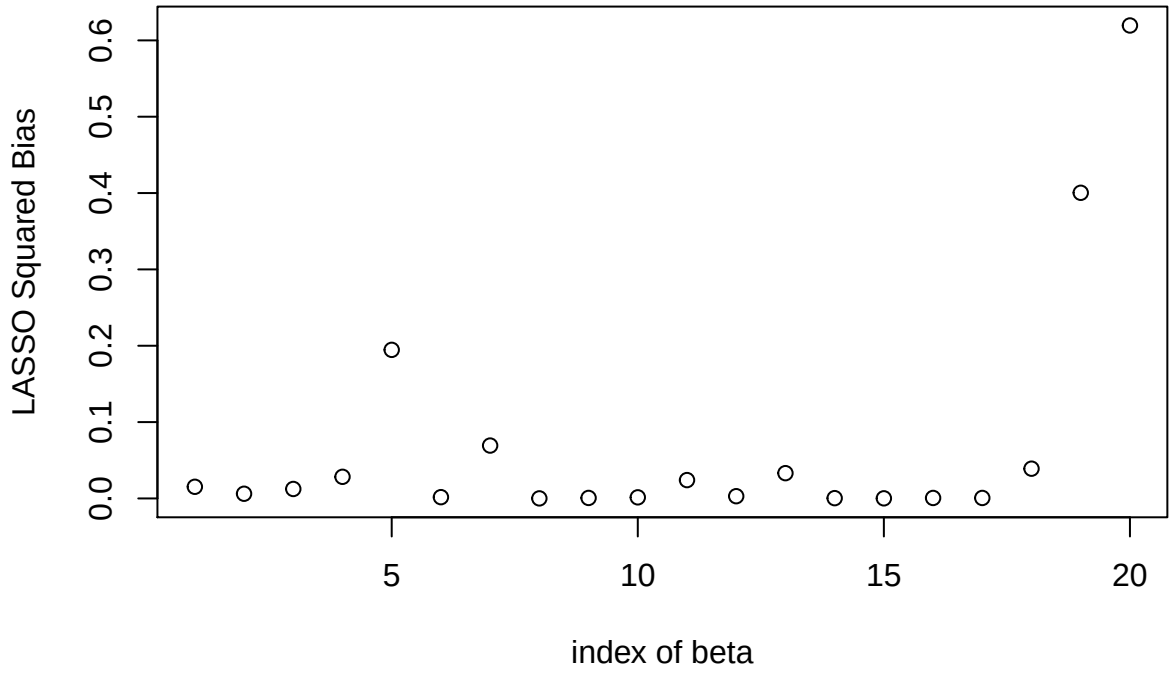

```
## eatimation of squared bias and variance for each estimation

mean.OLS=apply(Coef.OLS,2, mean)
bias2.OLS=(mean.OLS-betatrue)^2
var.OLS = apply(Coef.OLS, 2, var)
plot(bias2.OLS, xlab="index of beta",ylab="OLS Squared Bias" )
```



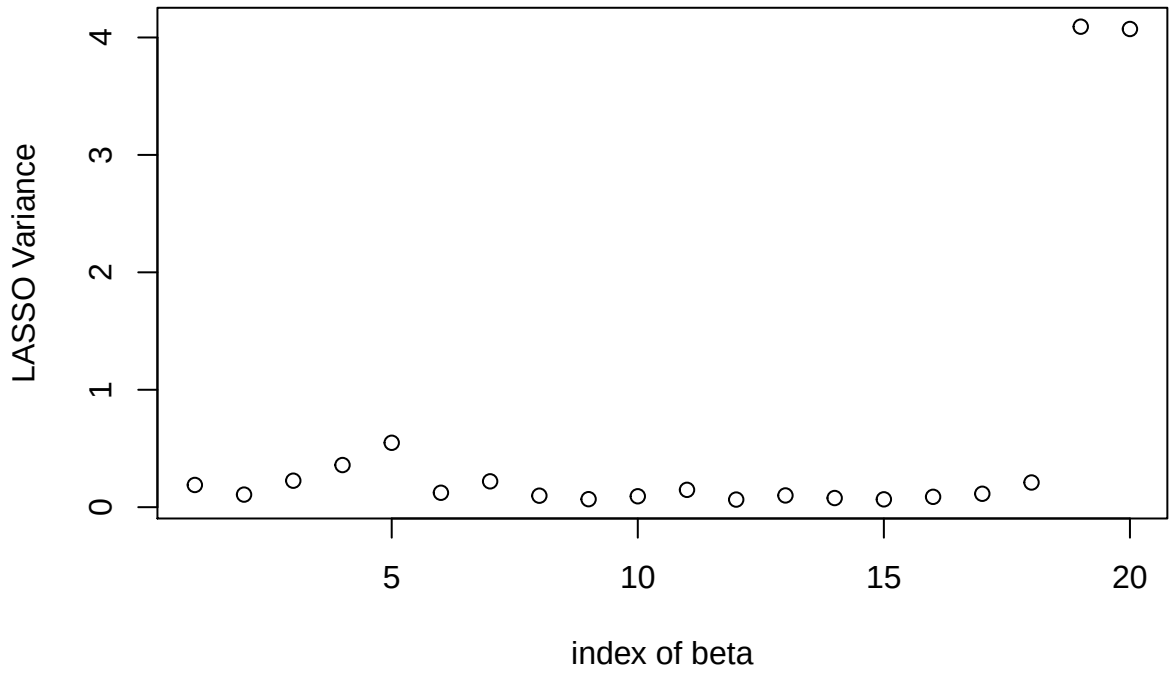

```
plot(var.OLS, xlab ="index of beta", ylab="OLS Variance" )
```



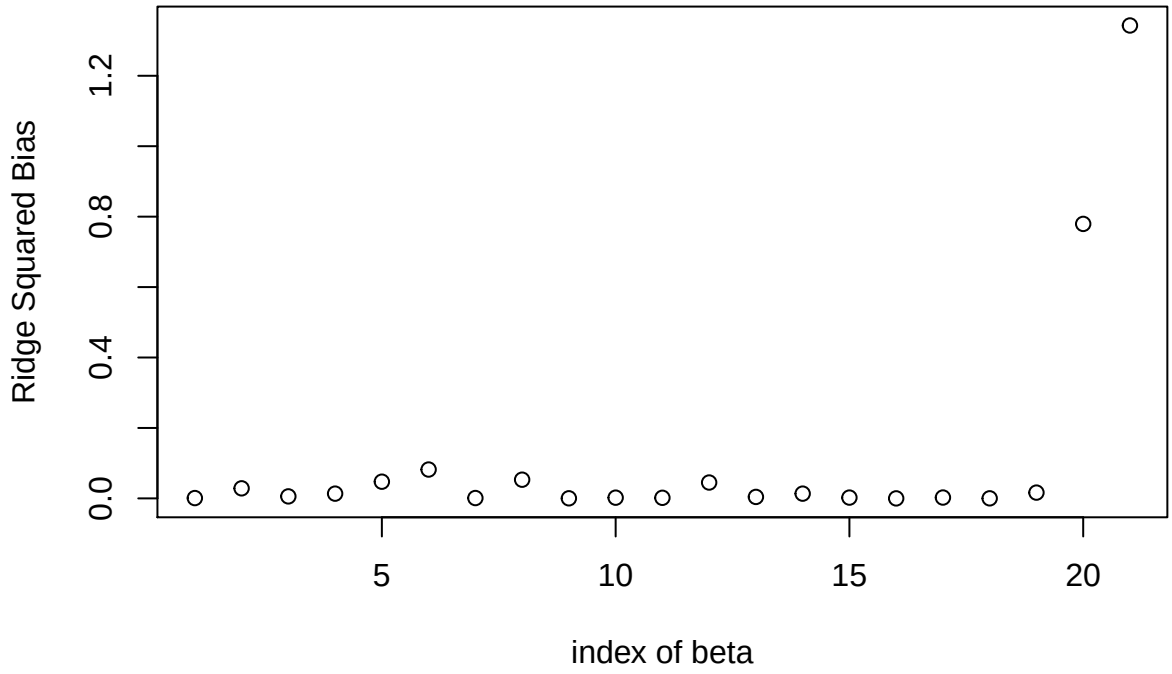

```
mean.lasso=apply(Coef.lasso,2, mean)
bias2.lasso=(mean.lasso-betatrue[-1])^2
var.lasso = apply(Coef.lasso, 2, var)
plot(bias2.lasso, xlab="index of beta",ylab="LASSO Squared Bias" )
```



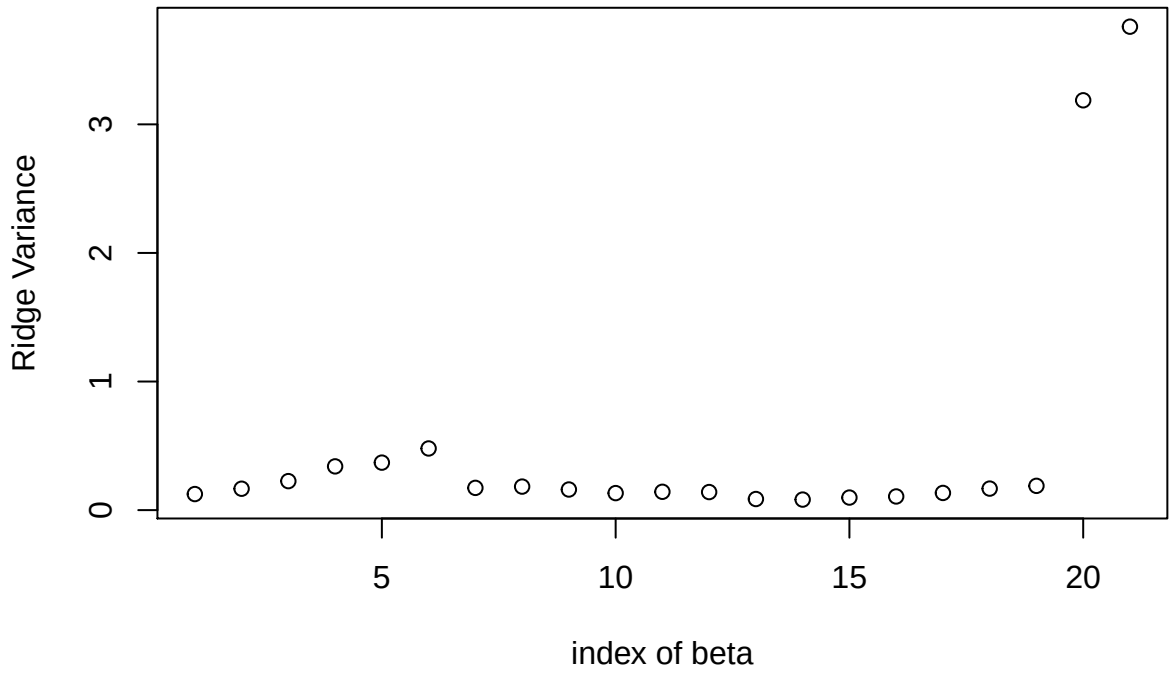

```
plot(var.lasso, xlab ="index of beta", ylab="LASSO Variance" )
```



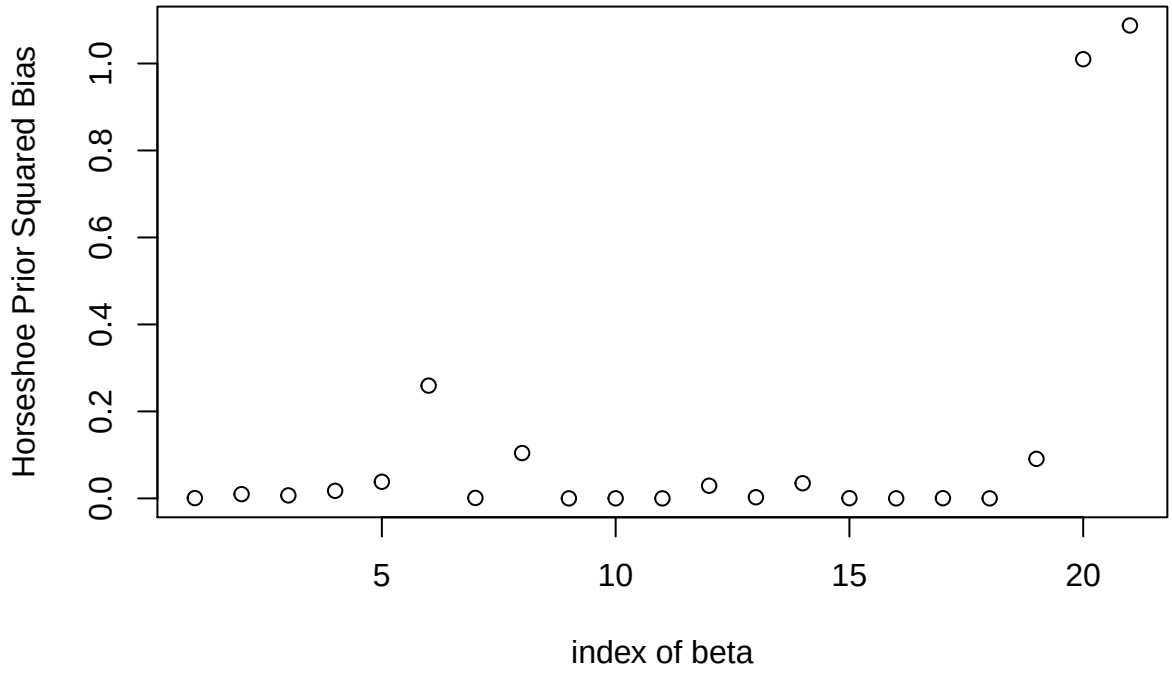

```
mean.ridge=apply(Coef.ridge,2, mean)
bias2.ridge=(mean.ridge-betatrue)^2
var.ridge = apply(Coef.ridge, 2, var)
plot(bias2.ridge, xlab="index of beta",ylab="Ridge Squared Bias" )
```



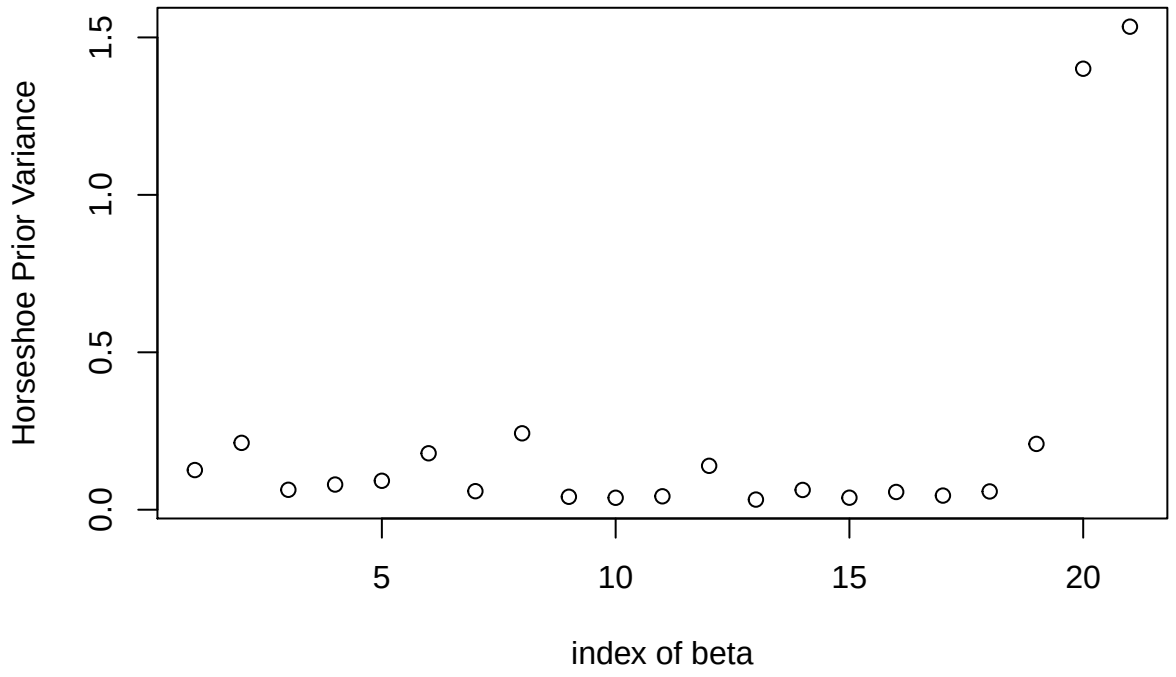

```
plot(var.ridge, xlab ="index of beta", ylab="Ridge Variance" )
```




```
mean.bhs=apply(Coef.bhs,2, mean)
bias2.bhs=(mean.bhs-betatrue)^2
var.bhs = apply(Coef.bhs, 2, var)
plot(bias2.bhs, xlab="index of beta",ylab="Horseshoe Prior Squared Bias" )
```




```
plot(var.bhs, xlab ="index of beta", ylab="Horseshoe Prior Variance" )
```



(d)

We assume here a centered, rotated $\mathbf{X}$. Also, τ, λ represent diagonal matrices in our notation, with τ_j denoting a particular diagonal element.

We can write the joint distribution $p(\mathbf{Y}, \boldsymbol{\beta}, \tau, \lambda, \phi) \propto \phi^{n/2} exp(-\frac{\phi}{2}(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}))exp(-\frac{\phi}{2}SSE)exp(-\frac{\phi}{2}(\boldsymbol{\beta}_0-\bar{y})^2 n)p(\boldsymbol{\beta}|\tau,\phi)p(\boldsymbol{\beta}_0,\phi)p(\tau|\lambda)p(\lambda)$

So

$$\boldsymbol{\beta} \propto exp(-\frac{\phi}{2}\boldsymbol{\beta}^T\tau^{-1}\boldsymbol{\beta})exp(-frac\phi 2(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}))$$

$$\propto exp(-\frac{\phi}{2}(\boldsymbol{\beta}^T(\mathbf{X}^T\mathbf{X}+\tau^{-1})\boldsymbol{\beta} - 2\boldsymbol{\beta}^T(\mathbf{X}^T\mathbf{X}+\tau^{-1})(\mathbf{X}^T\mathbf{X}+\tau^{-1})^{-1}\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}}))$$

We recognize a normal kernel, thus $\boldsymbol{\beta} \sim N((\mathbf{X}^T\mathbf{X}+\tau^{-1})^{-1}\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}}, (\mathbf{X}^T\mathbf{X}+\tau^{-1})^{-1})$. We can simplify with the identity $\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}^T\mathbf{Y}$ to obtain the formula in Armagan, Dunson and Lee of $\boldsymbol{\beta} \sim N((\mathbf{X}^T\mathbf{X}+\tau^{-1})^{-1}\mathbf{X}^T\mathbf{Y}, (\mathbf{X}^T\mathbf{X}+\tau^{-1})^{-1}/\phi)$

It is simpler to find τ_j components individually

$$\tau_j \propto p(\boldsymbol{\beta}|\tau_j,\phi)p(\tau_j|\lambda_j) \propto (\phi/\tau_j)^{1/2}exp(-\frac{\phi}{2}\boldsymbol{\beta}_j^T\boldsymbol{\beta}_j/\tau_j)exp(-\lambda^2\tau_j)$$

$$\propto \tau_j^{-1/2}exp(-\frac{1}{2}(\phi\boldsymbol{\beta}_j^2/\tau_j + \lambda_j^2\tau_j))$$

We can recognize a Generalized Inverse Gaussian kernel, with parameters $\mu = 1/2, \nu = \lambda_j^2, \xi = \boldsymbol{\beta}_j^2/\tau\backslash$

$$\phi \propto \phi^{n/2}exp(-\frac{\phi}{2}(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}))exp(-\frac{\phi}{2}SSE)exp(-\frac{\phi}{2}(\boldsymbol{\beta}_0-\bar{y})^2 n)p(\boldsymbol{\beta}|\tau,\phi)p(\boldsymbol{\beta}_0,\phi)$$

$$\propto \phi^{n/2}exp(-\frac{\phi}{2}((\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}) + SSE + n(b_0-\bar{y})^2 + \boldsymbol{\beta}^T\tau^{-1}\boldsymbol{\beta}))|\tau/\phi|^{-1/2}1/\phi$$

$$\propto \phi^{(n+p)/2-1}exp(-\frac{\phi}{2}((\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}) + SSE + n(b_0-\bar{y})^2 + \boldsymbol{\beta}^T\tau^{-1}\boldsymbol{\beta}))$$

We can recognize a (rather ugly) Gamma kernel, with parameters $\alpha = (n+p)/2, \beta = ((\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T(\mathbf{X}^T\mathbf{X})(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + SSE + n(b_0 - \bar{y})^2 + \boldsymbol{\beta}^T\tau^{-1}\boldsymbol{\beta})/2$\

Since $\boldsymbol{\beta}_0 \propto exp(-\frac{\phi}{2}(n(b_0 - \bar{y})^2))$, we can recognize a normal kernel, with $\mu = \bar{y}, \sigma^2 = 1/(n\phi)$\

Finally, $\lambda_j \propto p(\lambda_j|\tau_j)p(\lambda_j) \propto \lambda_j^2 exp(-\frac{1}{2}\lambda_j^2/2)\lambda_j^{\alpha-1}exp(-\lambda_j\eta) = \lambda^{(\alpha+2)-1}exp(-\frac{1}{2}(\lambda_j^2\tau_j + 2\lambda_j\eta)) \propto \lambda^{(\alpha+2)-1}exp(-\frac{\tau_j}{2}(\lambda_j + \frac{\eta}{\tau_j})^2)$. We can recognize a Generalized gamma distribution https://en.wikipedia.org/wiki/Generalized_gamma_distribution with parameters $a = \sqrt{\frac{2}{\tau_j}}, p = 2, d = \alpha + 2$, so $\lambda_j + \frac{\eta}{\tau_j} \sim GenGamma(\frac{2}{\tau_j}, 2, \alpha + 2)$\

Also note that if $\alpha + 2 \in \mathbb{N}$, then we can utilize a scaled χ distribution with $\alpha + 2$ degrees of freedom.

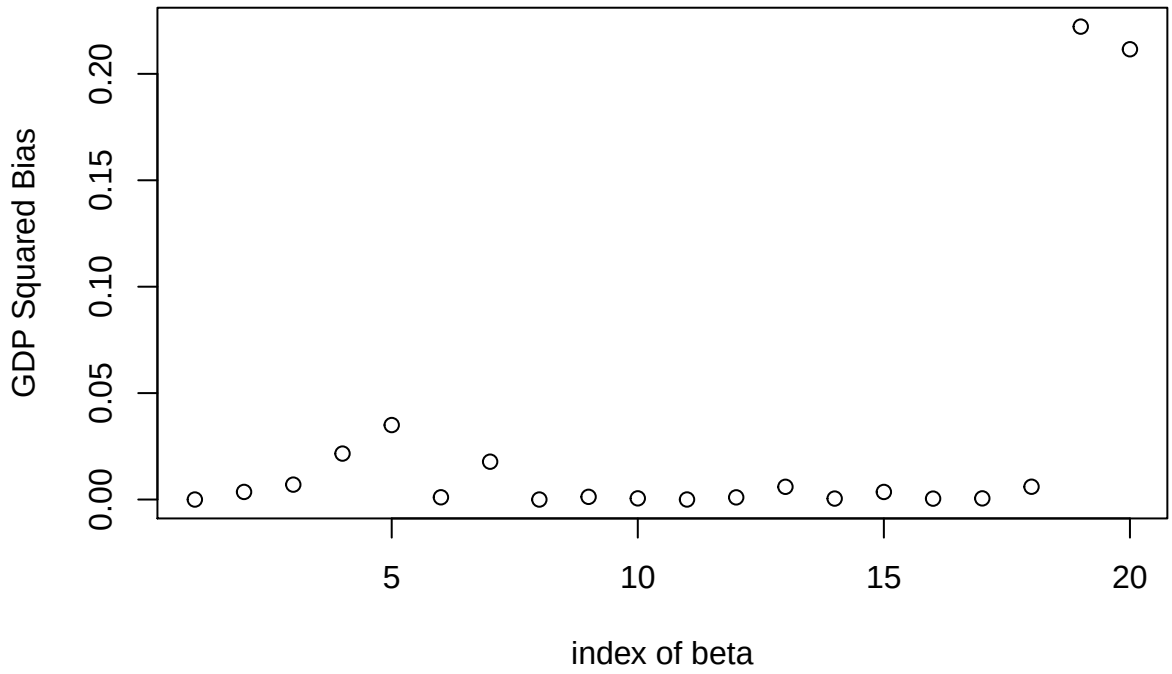
It is alternatively possible to return to our joint distribution, and marginalize τ from the distribution entirely.\ Since $\tau_j \sim GIG(\mu = 1/2, \nu = \lambda_j^2, \xi = \phi\boldsymbol{\beta}_j^2)$, our integral will be proportional to the inverse of the normalizing constant $K_\mu(\sqrt{\nu\xi})(\nu/\xi)^{\mu/2}$, where $K_{1/2}$ is a modified Bessel function of the second kind. Resources online here http://functions.wolfram.com/Bessel-TypeFunctions/BesselK/introductions/Bessels/05/ indicate that $K_{1/2}(z) = \frac{e^{-z}}{\sqrt{z}}$, so we obtain a normalizing constant overall of $e^{-\sqrt{\nu\xi}}(\nu\xi)^{1/4}(\nu)^{1/4}\xi^{-1/4} = e^{-\sqrt{\nu\xi}}(\nu)^{1/2} = e^{-\lambda_j|\boldsymbol{\beta}_j|\sqrt{\phi}}\lambda_j$

Returning to our original joint distribution and dividing by this normalizing constant, we can obtain $\lambda_j \propto exp(-|\boldsymbol{\beta}_j|\lambda_j^{-1}\sqrt{\phi})\lambda_j^{-1}\lambda_j^2\lambda^{\alpha-1}exp(-\lambda_j\eta) = \lambda_j^{(\alpha+1)-1}exp(-\lambda(|\boldsymbol{\beta}_j|\sqrt{\phi} + \eta))$, and obtain a Gamma kernel with $\alpha = \alpha + 1, \beta = |\boldsymbol{\beta}_j|\sqrt{\phi} + \eta$

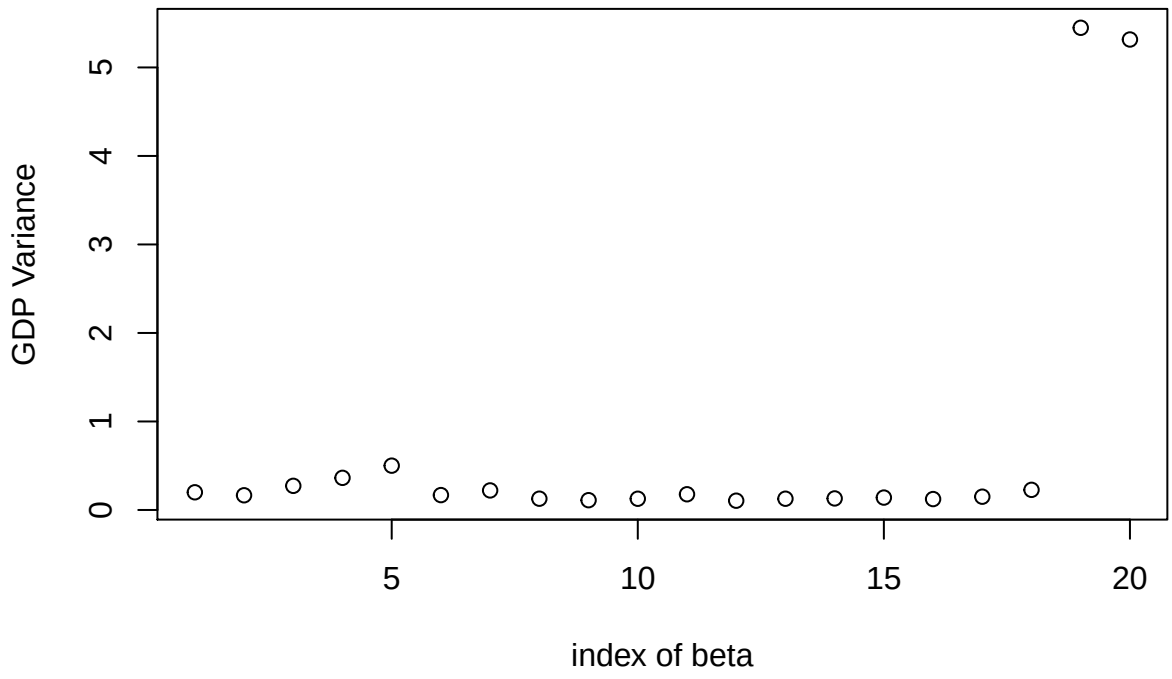
(e)

```
library(lars)
library(MASS)
library(matrixStats)
library(monomvn)
library(Rcpp)
library(RcppArmadillo)
sourceCpp("GDP.cpp")

Coef.gdp = matrix(rep(0,2000),nrow=100) ## record GDP estimates of beta for each simulation
gdp.MSE = rep(0,100) ##MSE

for( i in 1:100){
  rm(df)
  load(fname[i])
  X = as.matrix(df[-1]); Y = df[,1]
  X.scale = scale(X); n = length(Y); X.mean = colMeans(X); X.sd = colSds(X)
  coeftrue = betatrue
  coeftrue[1] = betatrue[1]+sum(betatrue[-1]*X.mean)
  coeftrue[-1] = betatrue[-1]*X.sd
  #Run GDP
  gdp=gdp_em(Y,X,1,1)
  Coef.gdp[i,1:20]=t(gdp$B)
  gdp.MSE[i] = sum((betatrue[-1] - t(gdp$B))^2)
  }

mean.gdp=apply(Coef.gdp,2, mean)
bias2.gdp=(mean.gdp-betatrue[-1])^2
var.gdp = apply(Coef.gdp, 2, var)
```

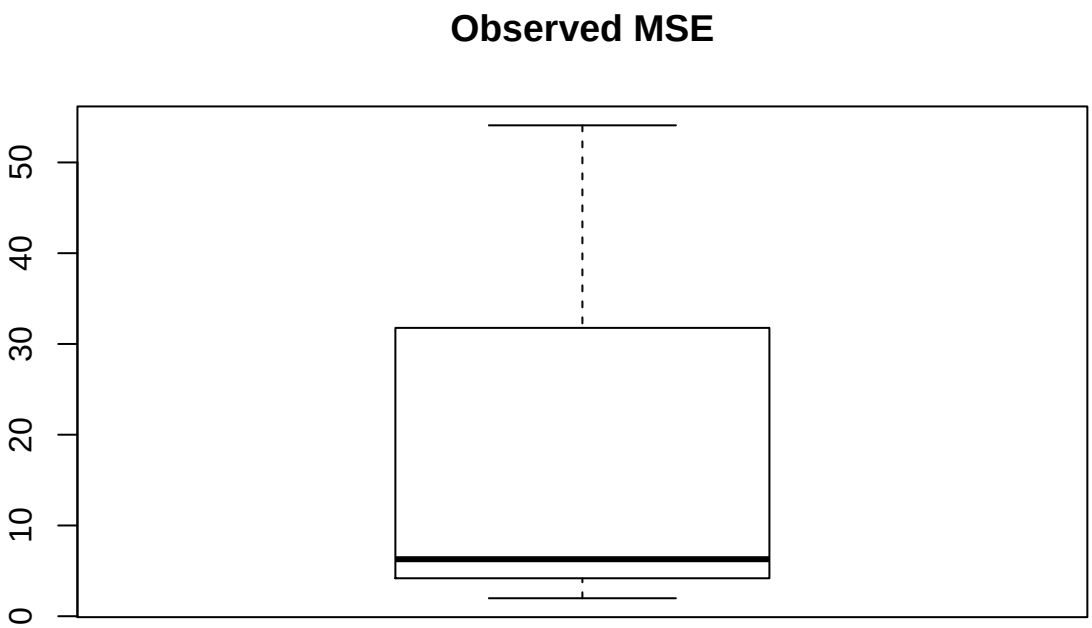
```r
plot(bias2.gdp, xlab="index of beta",ylab="GDP Squared Bias" )
```




```r
plot(var.gdp, xlab ="index of beta", ylab="GDP Variance" )
```

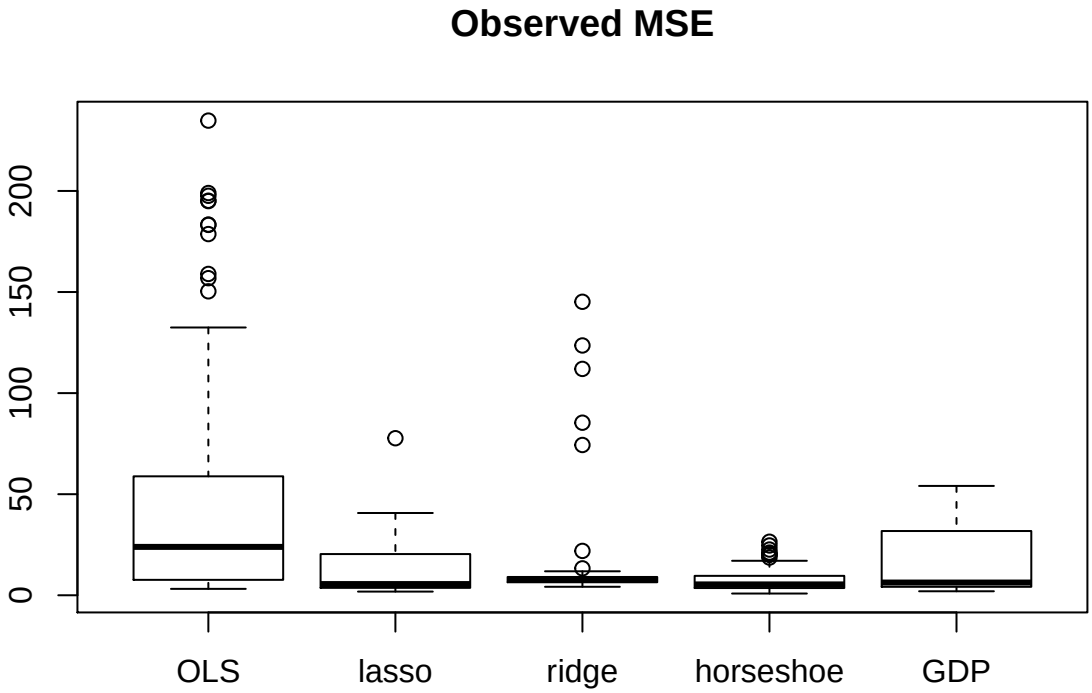



```r
## Observed MSE for each simulation
boxplot(gdp.MSE, names=c("GDP"), main="Observed MSE")
```



(f)

```
boxplot(OLS.MSE, lasso.MSE, ridge.MSE, bhs.MSE, gdp.MSE, names=c("OLS", "lasso", "ridge", "horseshoe","
```



From the result, we can see that estimation from Horseshoe Prior enjoys the lowest MSE. Blasso appears to be the best method. In terms of bias and variability, OLS estimator has both the least squared bias and the highest variance, besides, the variance seems far higher than those of other methods. Therefore, the other methods all performs better than OLS estimator in overall sense. It means that for these datas sets, shrinkage methods works much better than the normal OLS, if we can sacrifies a little bit bias, we can obatin much lower variance. When the covariates X has large correlation structure between each other, it's better to apple shrinkage methods than the traditional OLS.

Problem 2

Ridge prior

$p_\lambda(|\theta|) = \lambda|\theta|^2$

$p'_\lambda(|\theta|) = 2\lambda|\theta|$

1. Unbiasedness: No. $p'_\lambda(|\theta|) = 2\lambda|\theta|$ is not 0 when $|\theta|$ is large. So ridge is not unbiased.
2. Sparsity: No. $\min\ |\theta| + p'_\lambda(|\theta|) = 0$. So it does not have sparsity.
3. Continuity: Yes. $\min\ |\theta| + p'_\lambda(|\theta|)$ is attained at 0. So it have continuity.

Lasso prior

$p_\lambda(|\theta|) = \lambda|\theta|$

$p'_\lambda(|\theta|) = 2\lambda$

1. Unbiasedness: No. $p'_\lambda(|\theta|) = 2\lambda$ is not 0 when $|\theta|$ is large. So ridge is not unbiased.
2. Sparsity: Yes. $\min\ |\theta| + p'_\lambda(|\theta|) = |\theta| + 2\lambda > 0$. So it has sparsity.
3. Continuity: Yes. the minimum is attained at 0. So it have continuity.

Cauchy prior

1. Unbiasedness: Yes.
2. Sparsity: Yes, handling Sparsity.
3. Continuity: No, does not hold.

The horseshoe penalty function $p_\lambda(\theta_i) = -\log\log(1 + \frac{2\tau^2}{\theta_i^2})$.

$p'_\lambda(\theta_i) = \frac{4\tau^2/|\theta_i|^3}{(1+2\tau^2/|\theta_i|^2)\log(1+2\tau^2/|\theta_i|^2)}$. Then unbiasedness follows.

For $\theta \neq 0$, $|\theta_i| + p'_\lambda(\theta_i) = |\theta_i| + \frac{4\tau^2/|\theta_i|^3}{(1+2\tau^2/|\theta_i|^2)\log(1+2\tau^2/|\theta_i|^2)}$ is strictly larger than 0. So it hanles Sparsity.

Since $|\theta_i| + p'_\lambda(\theta_i) \to \infty$, as $|\theta| \to 0$. So continuity does not hold.

Generalized Double Pareto prior

The penalty function is $p_\lambda(|\theta|) = (\alpha + 1)\log(|\theta| + \sigma\eta)$.

$p'_\lambda(|\theta|) = (\alpha + 1)\frac{1}{|\theta|}$ goes to 0 as $|\theta| \to \infty$. Then unbiasedness holds.

1. Unbiasedness: Yes.
2. Sparsity: Holds when $\eta < 2\sqrt{1+\alpha}$
3. Continuity: Yes when $\eta <= \sqrt{1+\alpha}$