

# R-code for Extra exercise 3.3

## Linear regression by OLS, best subset regression, ridge regression, the lasso and least angle regression.

Part a)

```
library(MASS)
library(leaps)
library(lars)
```

```
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/div.sim.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/cv.k.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/lm.best.subset.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/calc.err.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/calc.res.r")
```

```
p<-15
```

```
beta0<-3
beta<-c(rep(2,5),rep(1,5),rep(0,5))
```

```
Sigma.x<-matrix(0.8,nrow=p,ncol=p)
diag(Sigma.x)<-1
```

```
sd.eps<-5
```

```
### set k for k-fold cross validation
k.in.cv<-10
```

```
n.train<-20
```

```
### simulate X in the training data
X.train<-sim.X(n.train,p,Sigma.x)
```

```
### Simulate y in the training set and in the test set
y.train<-sim.y.given.X(X.train,beta0,beta,sd.eps)
y.test<-sim.y.given.X(X.test,beta0,beta,sd.eps)
```

```
### construct the training data
```

```

training.data<-data.frame(y=y.train,X=I(X.train))

ind.cv<-1:nrow(training.data)
ind.cv<-sample(ind.cv)

### best subset regression
k.vec<-seq(0,p,1)
obj.cv.bs<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,lm.method="best.subset",
                tuning.par=k.vec)
obj.bs<-lm.best.subset(x=training.data$X,y=training.data$y,
                      k.vec=obj.cv.bs$opt.tuning.par)
beta0.hat<-obj.bs$coef[1]
beta.hat<-obj.bs$coef[-1]

### ridge regression
lambda.vec<-c(1e20,
              1000,500,100,50,10,5,1,
              0.5,0.1,0.05,0.01,0.005,0.001,
              0)
obj.cv.ridge<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,lm.method="ridge",
                  tuning.par=lambda.vec)
obj.ridge<-lm.ridge(y~X,data=training.data,
                   lambda=obj.cv.ridge$opt.tuning.par)
beta0.hat<-coef(obj.ridge)[1]
beta.hat<-coef(obj.ridge)[-1]

### lasso
s.vec<-seq(0,1,0.05)
obj.cv.lasso<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,lm.method="lasso",
                  tuning.par=s.vec)
obj.lasso<-lars(x=training.data$X,y=training.data$y,type="lasso")
beta.hat<-predict(obj.lasso,s=obj.cv.lasso$opt.tuning.par,type="coef",mode="fraction")
Xbar<-apply(training.data$X,2,mean)
ybar<-mean(training.data$y)
beta0.hat<-ybar-beta.hat%*%Xbar

### lar
s.vec<-seq(0,1,0.05)
obj.cv.lasso<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,lm.method="lar",
                  tuning.par=s.vec)

```

```
obj.lasso<-lars(x=training.data$X,y=training.data$y,type="lar")
beta.hat<-predict(obj.lasso,s=obj.cv.lasso$opt.tuning.par,type="coef",mode="fraction")
Xbar<-apply(training.data$X,2,mean)
ybar<-mean(training.data$y)
beta0.hat<-ybar-beta.hat%*%Xbar
```

```
pdf(file="tuning.pdf")
par(mfrow=c(2,2))
```

```
plot(k.vec,obj.cv.bs$RMSE,main="best subset",ylim=c(0,30))
abline(h=5,lty=2)
```

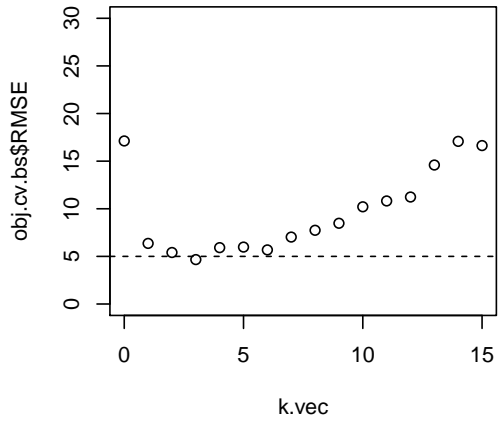
```
plot(log(1/(lambda.vec+0.00000001)),obj.cv.ridge$RMSE,main="ridge",ylim=c(0,30))
abline(h=5,lty=2)
```

```
plot(s.vec,obj.cv.lasso$RMSE,main="lasso",ylim=c(0,30))
abline(h=5,lty=2)
```

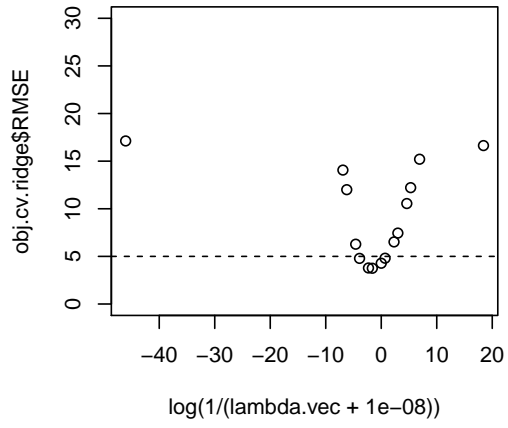
```
plot(s.vec,obj.cv.lasso$RMSE,main="lar",ylim=c(0,30))
abline(h=5,lty=2)
```

```
dev.off()
```

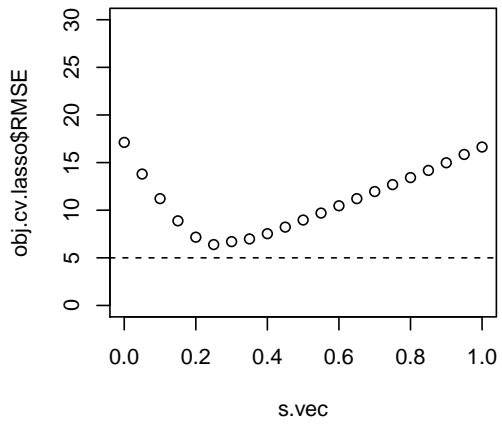
**best subset**



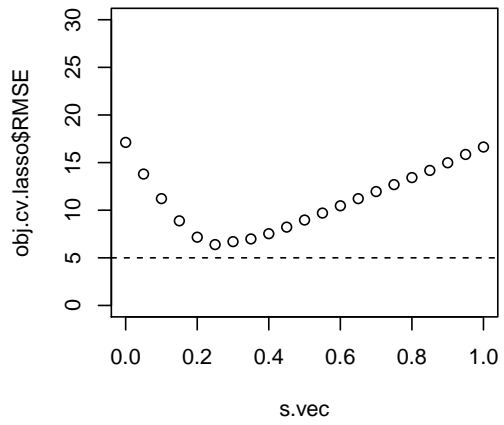
**ridge**



**lasso**



**lar**



```

cv.k<-function(data,k,ind=NULL,lm.method="ridge",tuning.par) {

  n<-nrow(data)
  if (is.null(ind)) {
    ind<-1:n
    ind<-sample(ind)
  }
  data<-data[ind,]

  n.most.parts<-floor(n/k)
  n.last.part<-n-(k-1)*n.most.parts

  ind<-list(k)
  for (i in 1:(k-1)) {
    ind[[i]]<-((i-1)*n.most.parts+1):(i*n.most.parts)
  }
  ind[[k]]<-((k-1)*n.most.parts+1):n

  pred<-matrix(NA,nrow=n,ncol=length(tuning.par))

  for (i in 1:k) {
    training.dat<-data[-ind[[i]],]
    test.dat<-data[ind[[i]],]

    if (lm.method=="ridge") {
      obj<-lm.ridge(y~X,data=training.dat,lambda=tuning.par)
      coef<-coef(obj)
### must use the coef function here, because lm.ridge else returns
### the scaled coefficients
    }

    if (lm.method=="lasso") {
      obj<-lars(x=training.dat$X,y=training.dat$y,type="lasso")
      beta.hat<-predict(obj,s=tuning.par,type="coef",mode="fraction")$coef
      Xbar<-apply(training.dat$X,2,mean)
      ybar<-mean(training.dat$y)
      beta0.hat<-ybar-beta.hat%*%Xbar
      coef<-cbind(beta0.hat,beta.hat)
    }

    if (lm.method=="lar") {
      obj<-lars(x=training.dat$X,y=training.dat$y,type="lar")

```

```

    beta.hat<-predict(obj,s=tuning.par,type="coef",mode="fraction")$coef
    Xbar<-apply(training.dat$X,2,mean)
    ybar<-mean(training.dat$y)
    beta0.hat<-ybar-beta.hat%*%Xbar
    coef<-cbind(beta0.hat,beta.hat)
  }

  if (lm.method=="best.subset") {
    obj<-lm.best.subset(x=training.dat$X,y=training.dat$y,k.vec=tuning.par)
    coef<-obj$coef
  }

  X.test<-cbind(1,test.dat$X)
  pred[ind[[i]],]<-X.test%*%t(coef)
}

RMSE<-sqrt(apply((data$y-pred)^2,2,mean))

which.opt<-which.min(RMSE)
opt.tuning.par<-tuning.par[which.opt]
opt.RMSE<-RMSE[which.opt]

res<-list(opt.tuning.par=opt.tuning.par,
          opt.RMSE=opt.RMSE,
          which.opt=which.opt,
          tuning.par=tuning.par,
          RMSE=RMSE)

res
}

```

Part b)

```

library(MASS)
library(leaps)
library(lars)

source("/nr/user/aldrin/Ui0/STK4030/Rfunc/div.sim.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/cv.k.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/lm.best.subset.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/calc.err.r")
source("/nr/user/aldrin/Ui0/STK4030/Rfunc/calc.res.r")

```

```

p<-15

beta0<-3
beta<-c(rep(2,5),rep(1,5),rep(0,5))

Sigma.x<-matrix(0.8,nrow=p,ncol=p)
diag(Sigma.x)<-1

sd.eps<-5

### set k for k-fold cross validation
k.in.cv<-10

### number of observations in training sets and test sets
n.train1<-20
n.train2<-100
n.train3<-1000
n.test<-1000

### number of simulations
n.sim1<-1000
n.sim2<-300
n.sim3<-100
###n.sim<-1000

### number of methods investigated
n.met<-5

### The simulation results will be put in this object,
### which is a list of lists,
### and the element of the inner list will be a matrix
res<-list()
for (j in 1:3) {
  res[[j]]<-list()
  for (no.met in 1:n.met) {
    res[[j]][[no.met]]<-NULL
  }
}

```

```

### Varies the number of observations in the training data
for (j in 1:3) {
  print(paste("j=",j))
  if (j==1) {
    n.train<-n.train1
    n.sim<-n.sim1
  }
  if (j==2) {
    n.train<-n.train2
    n.sim<-n.sim2
  }
  if (j==3) {
    n.train<-n.train3
    n.sim<-n.sim3
  }
}

### Construct matrices where we will store errors
beta.err<-matrix(NA,n.sim,p+1)
pred.err<-matrix(NA,n.sim,n.test)
f.err<-matrix(NA,n.sim,n.test)

err<-list()
for (no.met in 1:n.met) {
  err[[no.met]]<-list(beta.err=beta.err,pred.err=pred.err,f.err=f.err)
}

### loop over the simulations
for (i in 1:n.sim) {
  print(paste("j=",j," i=",i))

### construction of the matrix X in the test set
  X.test<-sim.X(n.test,p,Sigma.x)

### Copy the matrix X into a data frame
### The I function is used to get correct variable names in the prediction
### Not used now
### X.test.df<-data.frame(X=I(X.test))

### construction of E(Y|X) in the test set
  Ey.test<-calc.Ey.given.X(X.test,beta0,beta)

```



```

### simulate X in the training data
X.train<-sim.X(n.train,p,Sigma.x)

### Simulate y in the training set and in the test set
y.train<-sim.y.given.X(X.train,beta0,beta,sd.eps)
y.test<-sim.y.given.X(X.test,beta0,beta,sd.eps)

### construct the training data
training.data<-data.frame(y=y.train,X=I(X.train))

### For the same random order of observations for all methods
### when performing cross validation
ind.cv<-1:nrow(training.data)
ind.cv<-sample(ind.cv)

### Estimate the model by OLS
met.no<-1
obj.lm<-lm(y~X,data=training.data)
beta0.hat<-obj.lm$coef[1]
beta.hat<-obj.lm$coef[-1]
err[[met.no]]<-calc.err(err[[met.no]],
                        beta0,beta,y.test,Ey.test,
                        beta0.hat,beta.hat,X.test)

### Estimate the model by best subset regression and k-fold cross validation
met.no<-2
obj.cv.bs<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,
                lm.method="best.subset",
                tuning.par=seq(0,p,1))
obj.bs<-lm.best.subset(x=training.data$X,y=training.data$y,
                       k.vec=obj.cv.bs$opt.tuning.par)
beta0.hat<-obj.bs$coef[1]
beta.hat<-obj.bs$coef[-1]
err[[met.no]]<-calc.err(err[[met.no]],
                        beta0,beta,y.test,Ey.test,
                        beta0.hat,beta.hat,X.test)

### Estimate the model by ridge regression and k-fold cross validation
met.no<-3
lambda.vec<-c(1e20,
              1000,500,100,50,10,5,1,
              0.5,0.1,0.05,0.01,0.005,0.001,
              0)

```

```

obj.cv.ridge<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,
                 lm.method="ridge",
                 tuning.par=lambda.vec)
obj.ridge<-lm.ridge(y~X,data=training.data,
                  lambda=obj.cv.ridge$opt.tuning.par)
beta0.hat<-coef(obj.ridge)[1]
beta.hat<-coef(obj.ridge)[-1]
err[[met.no]]<-calc.err(err[[met.no]],
                       beta0,beta,y.test,Ey.test,
                       beta0.hat,beta.hat,X.test)

```

### Estimate the model by the lasso and k-fold cross validation

```

met.no<-4
s.vec<-seq(0,1,0.05)
obj.cv.lasso<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,
                  lm.method="lasso",
                  tuning.par=s.vec)
obj.lasso<-lars(x=training.data$X,y=training.data$y,type="lasso")
beta.hat<-predict(obj.lasso,s=obj.cv.lasso$opt.tuning.par,type="coef",mode="fraction")
Xbar<-apply(training.data$X,2,mean)
ybar<-mean(training.data$y)
beta0.hat<-ybar-as.vector(beta.hat*%Xbar)
err[[met.no]]<-calc.err(err[[met.no]],
                       beta0,beta,y.test,Ey.test,
                       beta0.hat,beta.hat,X.test)

```

### Estimate the model by the lar and k-fold cross validation

```

met.no<-5
s.vec<-seq(0,1,0.05)
obj.cv.lar<-cv.k(data=training.data,k=k.in.cv,ind=ind.cv,
                 lm.method="lar",
                 tuning.par=s.vec)
obj.lar<-lars(x=training.data$X,y=training.data$y,type="lar")
beta.hat<-predict(obj.lar,s=obj.cv.lar$opt.tuning.par,type="coef",mode="fraction")
Xbar<-apply(training.data$X,2,mean)
ybar<-mean(training.data$y)
beta0.hat<-ybar-as.vector(beta.hat*%Xbar)
err[[met.no]]<-calc.err(err[[met.no]],
                       beta0,beta,y.test,Ey.test,
                       beta0.hat,beta.hat,X.test)

```

```

}

for (met.no in 1:n.met) {
  res[[j]][[met.no]]<-calc.res(err[[met.no]])
}

}

### divides results into
### bias, standard deviation and root mean squared errors

rowlen<-p+1+2

tmp<-rep(NA,p+1)
for (i in 1:(p+1)) {
  tmp[i]<-paste("beta",i-1,sep="")
}
row.names<-c(tmp,"pred","f")
met.names<-c("OLS","Best-ss","Ridge","Lasso","LAR")

BIAS<-list()
SD<-list()
RMSE<-list()
for (j in 1:3) {
  BIAS[[j]]<-matrix(NA,rowlen,n.met)
  SD[[j]]<-matrix(NA,rowlen,n.met)
  RMSE[[j]]<-matrix(NA,rowlen,n.met)

  colnames(BIAS[[j]])<-met.names
  colnames(SD[[j]])<-met.names
  colnames(RMSE[[j]])<-met.names

  rownames(BIAS[[j]])<-row.names
  rownames(SD[[j]])<-row.names
  rownames(RMSE[[j]])<-row.names

  for (met.no in 1:n.met) {
    BIAS[[j]][,met.no]<-res[[j]][[met.no]][,"bias"]
    SD[[j]][,met.no]<-sqrt(res[[j]][[met.no]][,"var"])
    RMSE[[j]][,met.no]<-sqrt(res[[j]][[met.no]][,"MSE"])
  }
}

```

```
}
```

```
### Print results on screen
```

```
round(BIAS[[1]],1)  
round(SD[[1]],1)  
round(RMSE[[1]],1)
```

```
round(BIAS[[2]],1)  
round(SD[[2]],1)  
round(RMSE[[2]],1)
```

```
round(BIAS[[3]],1)  
round(SD[[3]],1)  
round(RMSE[[3]],1)
```

```

n.train=20
> round(BIAS[[1]],1)
      OLS Best-ss Ridge Lasso LAR
beta0  0.1    0.1  0.1  0.1  0.1
beta1 -0.3    0.3  0.7  0.8  0.8
beta2  0.1    0.5  0.8  0.8  0.8
beta3 -0.2    0.4  0.6  0.7  0.7
beta4 -0.1    0.2  0.7  0.7  0.7
beta5 -0.2    0.5  0.7  0.7  0.7
beta6  0.2    0.1  0.1  0.3  0.3
beta7  0.0    0.2  0.0  0.3  0.3
beta8 -0.2   -0.2 -0.1  0.1  0.1
beta9  0.3    0.3  0.1  0.4  0.4
beta10 0.1    0.0  0.1  0.2  0.2
beta11 0.1   -0.4 -0.6 -0.4 -0.4
beta12 0.1   -0.5 -0.6 -0.4 -0.4
beta13 -0.1  -0.5 -0.6 -0.4 -0.4
beta14 0.1   -0.4 -0.6 -0.4 -0.4
beta15 0.0   -0.3 -0.5 -0.3 -0.3
pred   0.1    0.2  0.1  0.1  0.1
f      0.1    0.1  0.1  0.1  0.1
> round(SD[[1]],1)
      OLS Best-ss Ridge Lasso LAR
beta0  2.8    1.7  1.4  1.7  1.7
beta1  6.1    3.5  1.6  2.0  2.0
beta2  6.4    3.6  1.4  1.9  2.0
beta3  6.5    3.5  1.7  2.1  2.1
beta4  6.2    3.6  1.6  2.1  2.1
beta5  6.0    3.3  1.3  2.1  2.1
beta6  6.4    3.2  1.8  2.0  2.0
beta7  6.3    2.9  1.4  1.7  1.7
beta8  6.3    3.2  1.4  1.7  1.7
beta9  6.4    3.0  1.4  1.7  1.8
beta10 6.3    3.0  1.4  1.7  1.8
beta11 6.6    2.8  1.8  1.8  1.8
beta12 6.3    2.6  1.4  1.5  1.4
beta13 6.3    2.7  1.4  1.6  1.7
beta14 5.8    2.5  1.4  1.4  1.5
beta15 6.3    2.9  1.5  1.5  1.6
pred  12.7    7.8  6.1  7.4  7.5
f     11.6    6.0  3.5  5.5  5.5

```

```

> round(RMSE[[1]],1)
      OLS Best-ss Ridge Lasso LAR
beta0  2.8    1.8   1.4   1.7 1.7
beta1  6.1    3.6   1.7   2.1 2.1
beta2  6.4    3.6   1.6   2.1 2.1
beta3  6.5    3.5   1.8   2.2 2.2
beta4  6.2    3.6   1.7   2.2 2.2
beta5  6.0    3.4   1.5   2.2 2.3
beta6  6.4    3.2   1.8   2.0 2.0
beta7  6.2    2.9   1.4   1.7 1.7
beta8  6.3    3.2   1.4   1.7 1.7
beta9  6.4    3.0   1.4   1.8 1.9
beta10 6.3    3.0   1.4   1.7 1.8
beta11 6.6    2.8   1.9   1.9 1.9
beta12 6.3    2.6   1.5   1.5 1.5
beta13 6.3    2.8   1.6   1.7 1.7
beta14 5.8    2.6   1.6   1.5 1.6
beta15 6.3    2.9   1.6   1.6 1.7
pred  12.7    7.8   6.1   7.4 7.5
f     11.6    6.0   3.5   5.5 5.5

```

```

n.train=100
> round(BIAS[[2]],1)
      OLS Best-ss Ridge Lasso LAR
beta0  0.0    0.0   0.0  0.0  0.0
beta1  0.0    0.0   0.5  0.2  0.2
beta2 -0.1   -0.1   0.5  0.2  0.2
beta3  0.1    0.1   0.6  0.3  0.3
beta4  0.0   -0.1   0.5  0.2  0.2
beta5  0.0    0.0   0.6  0.2  0.2
beta6  0.1    0.2   0.1  0.2  0.2
beta7  0.0    0.1   0.1  0.1  0.1
beta8  0.0    0.1   0.0  0.1  0.1
beta9  0.0    0.1   0.0  0.1  0.1
beta10 0.0    0.1   0.0  0.1  0.1
beta11 0.0   -0.2  -0.5 -0.3 -0.3
beta12 0.1    0.0  -0.5 -0.2 -0.2
beta13 0.0   -0.1  -0.5 -0.3 -0.3
beta14 0.0   -0.1  -0.5 -0.3 -0.3
beta15 -0.1  -0.2  -0.6 -0.3 -0.3
pred   0.0    0.0   0.0  0.0  0.0
f      0.0    0.0   0.0  0.0  0.0
> round(SD[[2]],1)
      OLS Best-ss Ridge Lasso LAR
beta0  0.6    0.6   0.6  0.6  0.6
beta1  1.2    1.4   0.6  1.1  1.1
beta2  1.2    1.6   0.6  1.1  1.1
beta3  1.1    1.4   0.6  1.0  1.0
beta4  1.3    1.5   0.7  1.1  1.1
beta5  1.2    1.6   0.6  1.1  1.1
beta6  1.1    1.2   0.5  0.8  0.8
beta7  1.2    1.3   0.5  0.9  0.9
beta8  1.2    1.3   0.6  0.9  0.9
beta9  1.2    1.3   0.6  0.9  0.9
beta10 1.2    1.3   0.6  0.9  0.9
beta11 1.2    1.0   0.6  0.7  0.7
beta12 1.2    1.0   0.6  0.7  0.7
beta13 1.1    0.9   0.6  0.6  0.6
beta14 1.2    1.1   0.6  0.7  0.7
beta15 1.1    1.0   0.5  0.7  0.7
pred   5.5    5.5   5.2  5.3  5.3
f      2.2    2.3   1.6  1.9  1.9

```

```

> round(RMSE[[2]],1)
      OLS Best-ss Ridge Lasso LAR
beta0 0.6      0.6  0.6  0.6 0.6
beta1 1.2      1.4  0.8  1.1 1.1
beta2 1.2      1.6  0.8  1.1 1.1
beta3 1.1      1.4  0.8  1.0 1.0
beta4 1.3      1.5  0.9  1.1 1.1
beta5 1.2      1.6  0.9  1.1 1.1
beta6 1.1      1.2  0.5  0.9 0.9
beta7 1.2      1.3  0.5  0.9 0.9
beta8 1.2      1.3  0.6  0.9 0.9
beta9 1.2      1.3  0.6  0.9 1.0
beta10 1.2     1.3  0.6  0.9 0.9
beta11 1.2     1.0  0.8  0.7 0.7
beta12 1.2     1.0  0.8  0.7 0.7
beta13 1.1     0.9  0.8  0.7 0.7
beta14 1.2     1.1  0.8  0.8 0.8
beta15 1.1     1.0  0.8  0.7 0.7
pred  5.5     5.5  5.2  5.3 5.3
f      2.2     2.3  1.6  1.9 1.9

```



```

n.train=1000
> round(BIAS[[3]],1)
      OLS Best-ss Ridge Lasso LAR
beta0 0.0    0.0  0.0  0.0  0.0
beta1 0.0    0.0  0.2  0.1  0.1
beta2 0.0   -0.1  0.1  0.0  0.0
beta3 0.0    0.0  0.2  0.1  0.1
beta4 0.1    0.0  0.2  0.1  0.1
beta5 0.0   -0.1  0.1  0.0  0.0
beta6 0.0    0.0  0.0  0.0  0.0
beta7 0.0    0.1  0.0  0.1  0.1
beta8 0.0    0.1  0.0  0.1  0.1
beta9 0.0    0.1  0.0  0.1  0.1
beta10 0.0   0.0  0.0  0.0  0.0
beta11 0.0   0.0 -0.2 -0.1 -0.1
beta12 0.0   0.0 -0.2 -0.1 -0.1
beta13 0.0   0.0 -0.2 -0.1 -0.1
beta14 -0.1  0.0 -0.2 -0.1 -0.1
beta15 0.0  -0.1 -0.2 -0.1 -0.1
pred  0.0   0.0  0.0  0.0  0.0
f      0.0   0.0  0.0  0.0  0.0
> round(SD[[3]],1)
      OLS Best-ss Ridge Lasso LAR
beta0 0.2    0.2  0.2  0.2  0.2
beta1 0.4    0.4  0.3  0.4  0.4
beta2 0.3    0.3  0.3  0.3  0.3
beta3 0.4    0.4  0.3  0.4  0.4
beta4 0.4    0.4  0.3  0.4  0.4
beta5 0.3    0.3  0.3  0.3  0.3
beta6 0.4    0.5  0.3  0.4  0.4
beta7 0.3    0.4  0.3  0.3  0.3
beta8 0.3    0.4  0.3  0.3  0.3
beta9 0.4    0.5  0.3  0.3  0.3
beta10 0.3   0.4  0.3  0.3  0.3
beta11 0.4   0.3  0.3  0.3  0.3
beta12 0.3   0.3  0.3  0.3  0.3
beta13 0.3   0.3  0.3  0.3  0.3
beta14 0.3   0.2  0.3  0.3  0.3
beta15 0.4   0.3  0.3  0.3  0.3
pred  5.0   5.0  5.0  5.0  5.0
f      0.6   0.7  0.6  0.6  0.6

```

```
> round(RMSE[[3]],1)
      OLS Best-ss Ridge Lasso LAR
beta0 0.2      0.2  0.2  0.2 0.2
beta1 0.4      0.4  0.4  0.4 0.4
beta2 0.3      0.3  0.3  0.3 0.3
beta3 0.4      0.4  0.4  0.4 0.4
beta4 0.4      0.4  0.4  0.4 0.4
beta5 0.3      0.3  0.3  0.3 0.3
beta6 0.4      0.5  0.3  0.4 0.4
beta7 0.3      0.4  0.3  0.3 0.3
beta8 0.3      0.4  0.3  0.3 0.3
beta9 0.4      0.5  0.3  0.4 0.4
beta10 0.3     0.4  0.3  0.3 0.3
beta11 0.4     0.3  0.4  0.3 0.3
beta12 0.3     0.3  0.4  0.3 0.3
beta13 0.3     0.3  0.4  0.3 0.3
beta14 0.3     0.2  0.4  0.3 0.3
beta15 0.4     0.3  0.4  0.3 0.3
pred   5.0     5.0  5.0  5.0 5.0
f      0.6     0.7  0.6  0.6 0.6
```