# HW2-R-code

Chapter 6, Problem 8

(a)

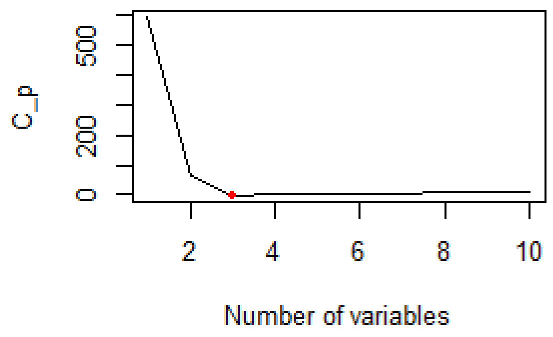
```
#Problem8(a)
set.seed(1)
x <- rnorm(100)
eps <- rnorm(100)
```

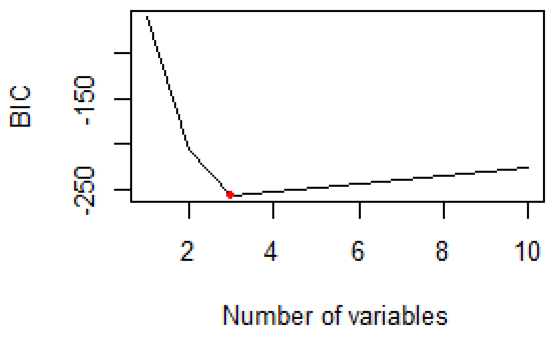
(b)

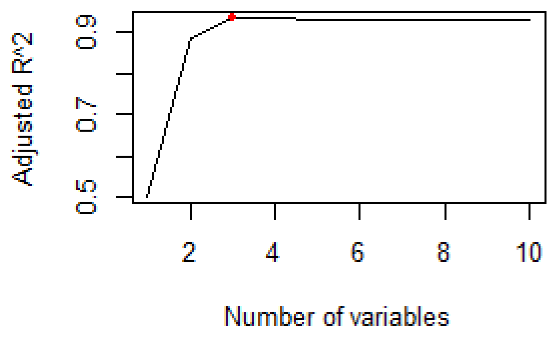
```
#Problem8(b)
set.seed(2)
#select coeffiencients:b0=1,b1=2,b2=-2,b3=0.5
b0 <- 1
b1 <- 2
b2 <- -2
b3 <- 0.5
y <- b0 + b1 * x + b2 * x^2 + b3 * x^3 +eps
```

(c)

```
#Problem8(c)
library(leaps)
#use the data.frame() function to create a single data set containing both X and Y
data.full <- data.frame(y = y, x = x)
regfit.full <- regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)
reg.summary <- summary(regfit.full)
#Use regsubsets function to choose best model
par(mfrow = c(2, 2))
plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
points(which.min(reg.summary$cp), reg.summary$cp[which.min(reg.summary$cp)], col = "red", cex =
1, pch = 20)
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
points(which.min(reg.summary$bic), reg.summary$bic[which.min(reg.summary$bic)], col = "red", cex
= 1, pch = 20)
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
points(which.max(reg.summary$adjr2), reg.summary$adjr2[which.max(reg.summary$adjr2)], col =
"red", cex = 1, pch = 20)
```







We find that with Cp, BIC and Adjusted R2 criteria, we all pick 3 variables in our models.

```
coef(regfit.full, which.max(reg.summary$adjr2))
```

```
## Output:
(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2
1.07219472            2.44514720           -2.15676236
poly(x, 10, raw = T)5
0.09022577
```

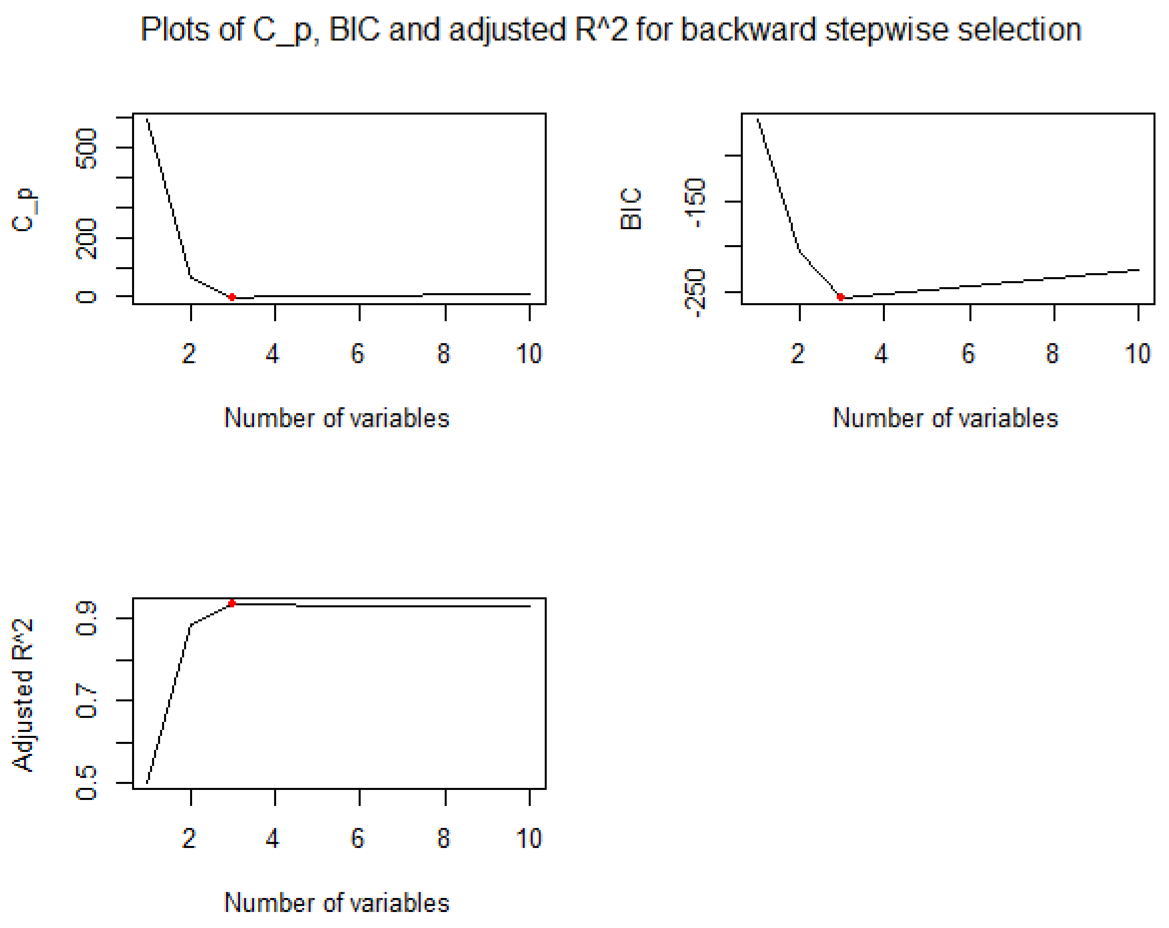
(d)

```
#Problem8(d)
#forward stepwise selection
regfit.fwd <- regsubsets(y ~ poly(x, 10, raw = T),, data = data.full, nvmax = 10, method =
"forward")
reg.summary.fwd <- summary(regfit.fwd)
par(mfrow = c(2, 2))
plot(reg.summary.fwd$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
points(which.min(reg.summary.fwd$cp), reg.summary.fwd$cp[which.min(reg.summary.fwd$cp)], col =
"red", cex = 1, pch = 20)
plot(reg.summary.fwd$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
points(which.min(reg.summary.fwd$bic), reg.summary.fwd$bic[which.min(reg.summary.fwd$bic)], col
= "red", cex = 1, pch = 20)
plot(reg.summary.fwd$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
points(which.max(reg.summary.fwd$adjr2),
reg.summary.fwd$adjr2[which.max(reg.summary.fwd$adjr2)], col = "red", cex = 1, pch = 20)
mtext("Plots of C_p, BIC and adjusted R^2 for forward stepwise selection", side = 3, line = -2,
outer = TRUE)
```

```
coef(regfit.fwd, which.max(reg.summary.fwd$adjr2))
```

```
Output:
(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2
1.07219472            2.44514720           -2.15676236
poly(x, 10, raw = T)5
0.09022577
```

```
#backward stepwise selection
regfit.bwd <- regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method =
"backward")
reg.summary.bwd <- summary(regfit.bwd)
par(mfrow = c(2, 2))
plot(reg.summary.bwd$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
points(which.min(reg.summary.bwd$cp), reg.summary.bwd$cp[which.min(reg.summary.bwd$cp)], col =
"red", cex = 1, pch = 20)
plot(reg.summary.bwd$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
points(which.min(reg.summary.bwd$bic), reg.summary.bwd$bic[which.min(reg.summary.bwd$bic)], col
= "red", cex = 1, pch = 20)
plot(reg.summary.bwd$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
points(which.max(reg.summary.bwd$adjr2),
reg.summary.bwd$adjr2[which.max(reg.summary.bwd$adjr2)], col = "red", cex = 1, pch = 20)
mtext("Plots of C_p, BIC and adjusted R^2 for backward stepwise selection", side = 3, line = -2,
outer = TRUE)
```

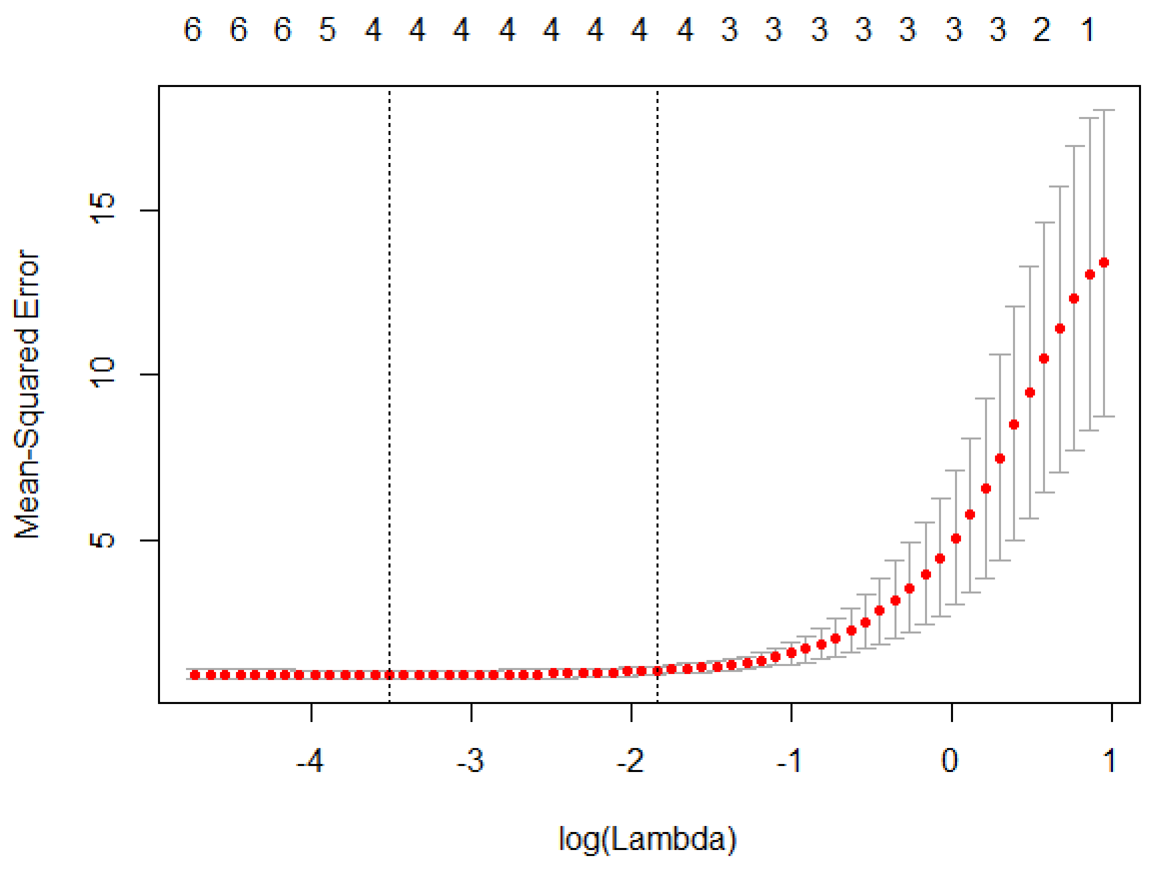


```
coef(regfit.bwd, which.max(reg.summary.bwd$adjr2))
```

```
Output:
(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2
1.07219472            2.44514720           -2.15676236
poly(x, 10, raw = T)5
0.09022577
```

(e)

```
#Problem8(e)
library(glmnet)
xmat <- model.matrix(y ~ poly(x, 10, raw = T),data = data.full)[, -1]
cv.lasso <- cv.glmnet(xmat, y, alpha = 1)
plot(cv.lasso)
```



```
bestlam <- cv.lasso$lambda.min
bestlam
```

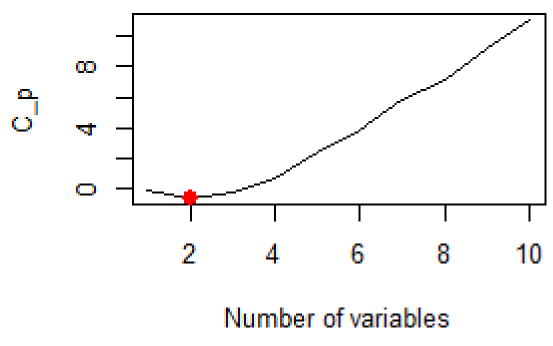
```
Output:
[1] 0.02980709
```

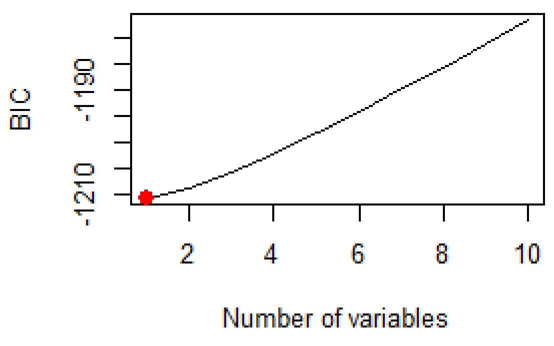
```
fit.lasso <- glmnet(xmat, y, alpha = 1)
predict(fit.lasso, s = bestlam, type = "coefficients")
```

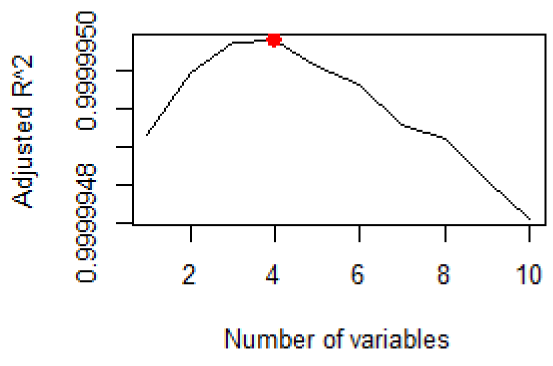
```
11 x 1 sparse Matrix of class "dgCMatrix"
                                 1
(Intercept)              1.04760945
poly(x, 10, raw = T)1    2.30385224
poly(x, 10, raw = T)2   -2.11665883
poly(x, 10, raw = T)3    0.12474433
poly(x, 10, raw = T)4    .
poly(x, 10, raw = T)5    0.06726153
poly(x, 10, raw = T)6    .
poly(x, 10, raw = T)7    .
poly(x, 10, raw = T)8    .
poly(x, 10, raw = T)9    .
poly(x, 10, raw = T)10   .
```

(f)

```
#Problem8(f)
b7 <- 7
y <- b0 + b7 * x^7 + eps
data.full <- data.frame(y = y, x = x)
regfit.full <- regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)
reg.summary <- summary(regfit.full)
par(mfrow = c(2, 2))
plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
points(which.min(reg.summary$cp), reg.summary$cp[which.min(reg.summary$cp)], col = "red", cex =
2, pch = 20)
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
points(which.min(reg.summary$bic), reg.summary$bic[which.min(reg.summary$bic)], col = "red", cex
= 2, pch = 20)
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
points(which.max(reg.summary$adjr2), reg.summary$adjr2[which.max(reg.summary$adjr2)], col =
"red", cex = 2, pch = 20)
```







```
coef(regfit.full, 1)
```

```
Output(BIC):
(Intercept) poly(x, 10, raw = T)7

0.9589402             7.0007705
```

```
coef(regfit.full, 2)
```

```
Output(Cp):
(Intercept) poly(x, 10, raw = T)2 poly(x, 10, raw = T)7
1.0704904            -0.1417084             7.0015552
```

```
coef(regfit.full, 4)
```

```
Output(AIC):
(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2
1.0762524             0.2914016            -0.1617671
poly(x, 10, raw = T)3 poly(x, 10, raw = T)7
-0.2526527             7.0091338
```

```
#lasso
xmat <- model.matrix(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)[, -1]
cv.lasso <- cv.glmnet(xmat, y, alpha = 1)
bestlam <- cv.lasso$lambda.min
bestlam
```

```
Output:
[1] 13.57478
```

```
fit.lasso <- glmnet(xmat, y, alpha = 1)
predict(fit.lasso, s = bestlam, type = "coefficients")
```

```
Output:
11 x 1 sparse Matrix of class "dgCMatrix"
                              1
(Intercept)           1.904188
poly(x, 10, raw = T)1  .
poly(x, 10, raw = T)2  .
poly(x, 10, raw = T)3  .
poly(x, 10, raw = T)4  .
poly(x, 10, raw = T)5  .
poly(x, 10, raw = T)6  .
poly(x, 10, raw = T)7  6.776797
poly(x, 10, raw = T)8  .
poly(x, 10, raw = T)9  .
poly(x, 10, raw = T)10 .
```

Chapter 6, Problem 9

(a)

```
#Problem9(a)
library(ISLR)
data("College")
set.seed(11)
#split the data into training and testing
train = sample(1:dim(College)[1], dim(College)[1] / 2)
test <- -train
College.train <- College[train, ]
College.test <- College[test, ]
```

(b)

```
#Problem9(b)
#Fit a linear model using least squares on the training set, and report the test error obtained
fit.lm <- lm(Apps ~ ., data = College.train)
pred.lm <- predict(fit.lm, College.test)
mean((pred.lm - College.test$Apps)^2)
```

```
Output:
[1] 1538442
```

(c)

```
#Problem9(c)
#Fit a ridge regression model on the training set, with lambda chosen by cross-validation
library(glmnet)
train.mat <- model.matrix(Apps ~ ., data = College.train)
test.mat <- model.matrix(Apps ~ ., data = College.test)
grid <- 10 ^ seq(4, -2, length = 100)
fit.ridge <- glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
cv.ridge <- cv.glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
bestlam.ridge <- cv.ridge$lambda.min
bestlam.ridge
```

```
Output:
[1] 18.73817
```

```
pred.ridge <- predict(fit.ridge, s = bestlam.ridge, newx = test.mat)
mean((pred.ridge - College.test$Apps)^2)
```

```
Output:
[1] 1608859
```

(d)

```
#Problem9(d)
#Fit a lasso model on the training set, with lambda chosen by cross-validation
fit.lasso <- glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
cv.lasso <- cv.glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
bestlam.lasso <- cv.lasso$lambda.min
bestlam.lasso
```

```
Output:
[1] 21.54435
```

```
pred.lasso <- predict(fit.lasso, s = bestlam.lasso, newx = test.mat)
mean((pred.lasso - College.test$Apps)^2)
```

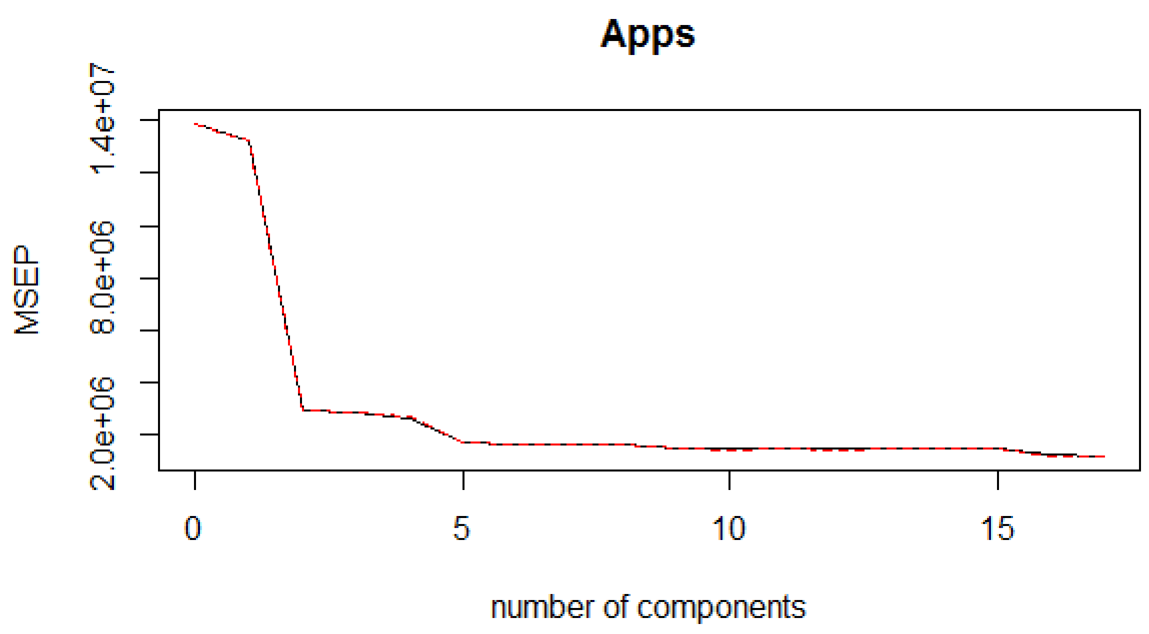
```
Output:
[1] 1635280
```

```
predict(fit.lasso, s = bestlam.lasso, type = "coefficients")
```

```
Output:
19 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept) -836.50402310
(Intercept)    .
PrivateYes  -385.73749394
Accept         1.17935134
Enroll         .
Top10perc     22.70211938
Top25perc      .
F.Undergrad    0.07062149
P.Undergrad    0.01366763
Outstate      -0.03424677
Room.Board     0.01281659
Books         -0.02167770
Personal       .
PhD           -1.46396964
Terminal      -5.17281004
S.F.Ratio      5.70969524
perc.alumni   -9.95007567
Expend         0.14852541
Grad.Rate      5.79789861
```

(e)

```
#Problem9(e)
#Fit a PCR model on the training set, with MM chosen by cross-validation
library(pls)
fit.pcr <- pcr(Apps ~ ., data = College.train, scale = TRUE, validation = "CV")
validationplot(fit.pcr, val.type = "MSEP")
```

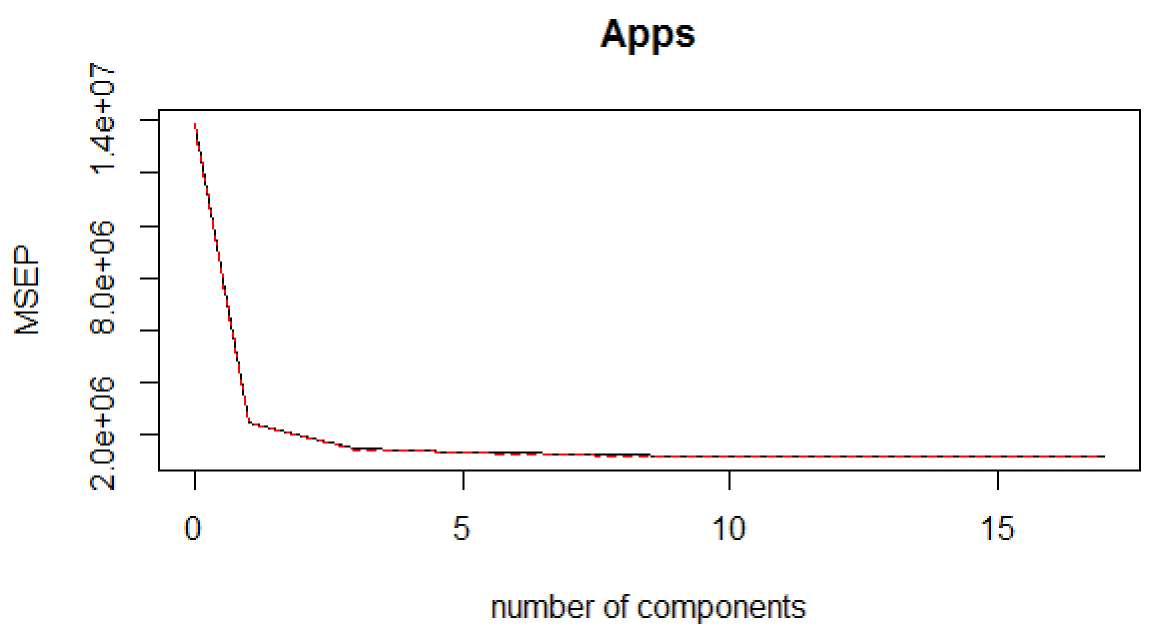


```
pred.pcr <- predict(fit.pcr, College.test, ncomp = 10)
mean((pred.pcr - College.test$Apps)^2)
```

```
Output:
[1] 3014496
```

(f)

```
#Problem9(f)
#Fit a PLS model on the training set, with MM chosen by cross-validation
fit.pls <- plsr(Apps ~ ., data = College.train, scale = TRUE, validation = "CV")
validationplot(fit.pls, val.type = "MSEP")
```

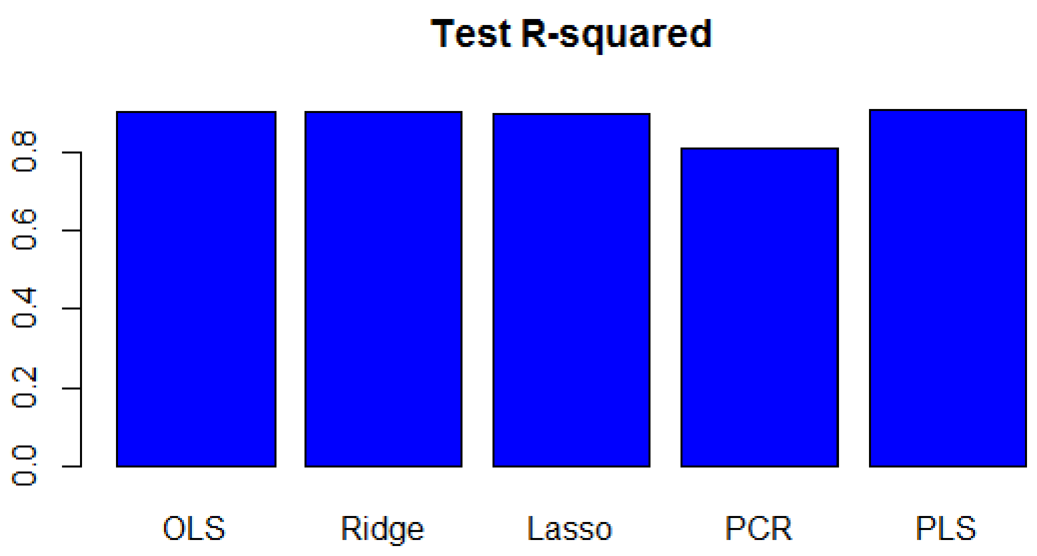


```
pred.pls <- predict(fit.pls, College.test, ncomp = 10)
mean((pred.pls - College.test$Apps)^2)
```

```
Output:
[1] 1508987
```

(g)

```
#Problem9(g)
#compute the test R square for all models
test.avg <- mean(College.test$Apps)
lm.r2 <- 1 - mean((pred.lm - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
ridge.r2 <- 1 - mean((pred.ridge - College.test$Apps)^2) / mean((test.avg -
College.test$Apps)^2)
lasso.r2 <- 1 - mean((pred.lasso - College.test$Apps)^2) / mean((test.avg -
College.test$Apps)^2)
pcr.r2 <- 1 - mean((pred.pcr - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
pls.r2 <- 1 - mean((pred.pls - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
barplot(c(lm.r2, ridge.r2, lasso.r2, pcr.r2, pls.r2), col="blue", names.arg=c("OLS", "Ridge",
"Lasso", "PCR", "PLS"), main="Test R-squared")
```


Test R-squared
0.0
0.2
0.4
0.6
0.8
OLS
Ridge
Lasso
PCR
PLS