{BIOSTAT 285 Spring 2020 Homework 1}

Due 11:00 PM 04/17/2020 (Submit to CCLE)

Nan Liu

Remark. For **Computational Part**, please complete your answer in the **RMarkdown** file and summit the generated PDF and RMD files. Related packages have been loaded in setup.

Computational Part

1. (Model Selection, [ISL] 6.8, *25 pt*) In this exercise, we will generate simulated data, and will then use this data to perform model selection.

(a) Use the `rnorm` function to generate a predictor $\boldsymbol{X}$ of length $n = 100$, as well as a noise vector $\boldsymbol{\epsilon}$ of length $n = 100$.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.6.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
set.seed(1)
x <- rnorm(100)
epsilon <- rnorm(100)
```

(b) Generate a response vector $\boldsymbol{Y}$ of length $n = 100$ according to the model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon,$$

where $\beta_0 = 3$, $\beta_1 = 2$, $\beta_2 = -3$, $\beta_3 = 0.3$.

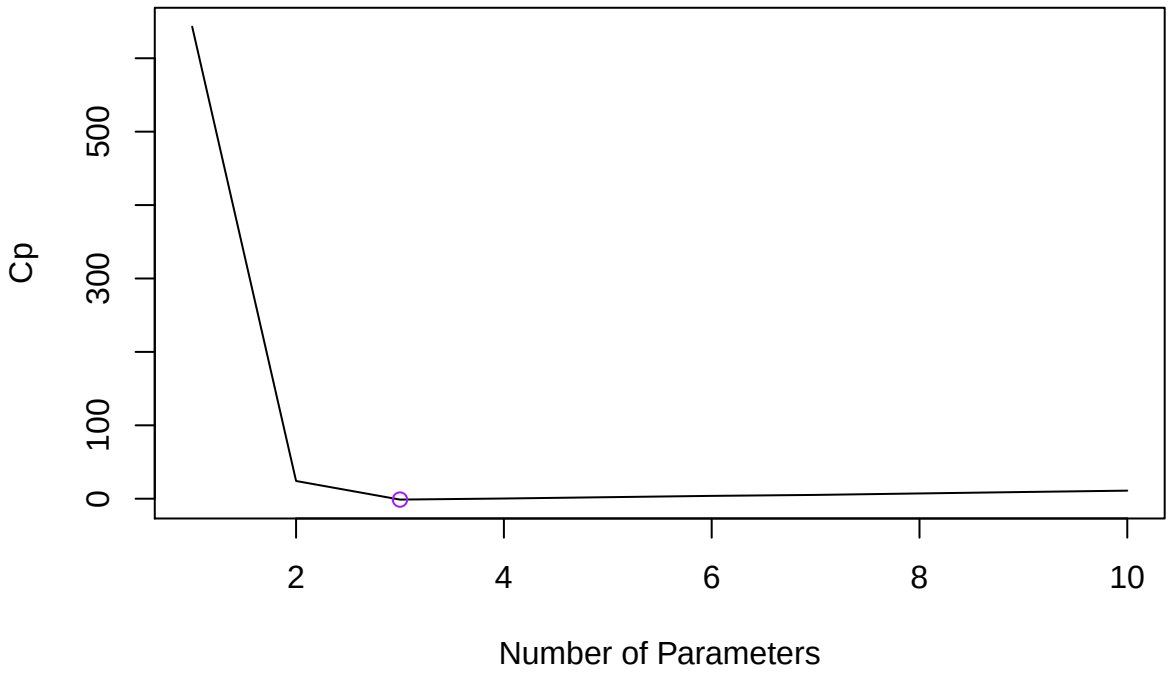
```
y <- 3 + 2 * x -3 * x^2 + 0.3 * x^3 + epsilon
```

(c) Use the `regsubsets` function from `leaps` package to perform best subset selection in order to choose the best model from the set of predictors $(X, X^2, \cdots, X^{10})$. What are the best models obtained according to C_p, BIC, and adjusted R^2, respectively? Show some plots to provide evidence for your answer, and report the coefficients of the best model obtained.

```
library(leaps)
leaps <- regsubsets(y~x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) +
                      I(x^7) + I(x^8) + I(x^9) + I(x^10),
                    data = data.frame(x = x, y = y), nvmax = 10)
(summary <- summary(leaps))
```

```
## Subset selection object
## Call: regsubsets.formula(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data.frame(x = x,
##     y = y), nvmax = 10)
## 10 Variables  (and intercept)
##          Forced in Forced out
## x            FALSE      FALSE
## I(x^2)       FALSE      FALSE
## I(x^3)       FALSE      FALSE
## I(x^4)       FALSE      FALSE
## I(x^5)       FALSE      FALSE
## I(x^6)       FALSE      FALSE
## I(x^7)       FALSE      FALSE
## I(x^8)       FALSE      FALSE
## I(x^9)       FALSE      FALSE
## I(x^10)      FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: exhaustive
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9) I(x^10)
## 1  ( 1 )  " " "*"    " "    " "    " "    " "    " "    " "    " "    " "
## 2  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    " "    " "    " "
## 3  ( 1 )  "*" "*"    " "    " "    " "    " "    "*"    " "    " "    " "
## 4  ( 1 )  "*" "*"    " "    " "    "*"    "*"    " "    " "    " "    " "
## 5  ( 1 )  "*" "*"    " "    " "    " "    " "    "*"    "*"    "*"    " "
## 6  ( 1 )  "*" "*"    " "    " "    " "    "*"    "*"    "*"    " "    "*"
## 7  ( 1 )  "*" "*"    " "    "*"    " "    "*"    "*"    "*"    " "    "*"
## 8  ( 1 )  "*" "*"    " "    "*"    "*"    "*"    " "    "*"    "*"    "*"
## 9  ( 1 )  "*" "*"    " "    "*"    "*"    "*"    "*"    "*"    "*"    "*"
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"
```

```
#Cp
plot(1:10, summary$cp, xlab = "Number of Parameters", ylab = "Cp", type = "l")
min_cp <- min(summary$cp)
points(c(1:10)[summary$cp == min_cp], min_cp, pch = 1, col = "purple")
```



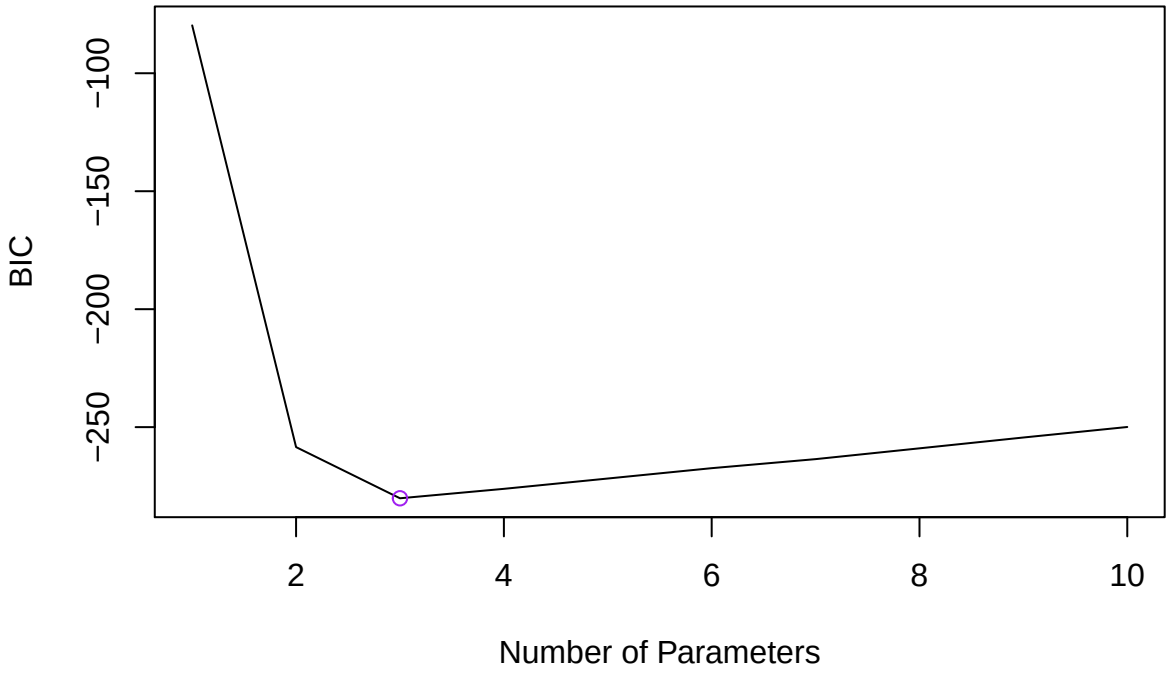

```
#report the coefficients
lm1 <- lm(y ~ I(x) + I(x^2) + I(x^7), data = data.frame(x = x,y = y))
summary(lm1)
```

```
##
## Call:
## lm(formula = y ~ I(x) + I(x^2) + I(x^7), data = data.frame(x = x,
##     y = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9289 -0.5827 -0.1852  0.5620  2.1319
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.076274   0.117608  26.157  < 2e-16 ***
## I(x)         2.356236   0.127197  18.524  < 2e-16 ***
## I(x^2)      -3.165149   0.087021 -36.372  < 2e-16 ***
## I(x^7)       0.010468   0.001949   5.372 5.42e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9453 on 96 degrees of freedom
## Multiple R-squared:  0.9495, Adjusted R-squared:  0.9479
## F-statistic: 601.6 on 3 and 96 DF,  p-value: < 2.2e-16
```

According to Cp, the model contains 3 predictors which are X, X^2, X^7 The best models obtained according to C_p is: $Y = 3.08 + 2.36 * X\text{-}3.17* X^2 + 0.01 * X^7$

```
#BIC
plot(1:10, summary$bic, xlab = "Number of Parameters", ylab = "BIC", type = "l")
```

```r
min_bic <- min(summary$bic)
points(c(1:10)[summary$bic == min_bic], min_bic, pch=1, col = "purple")
```

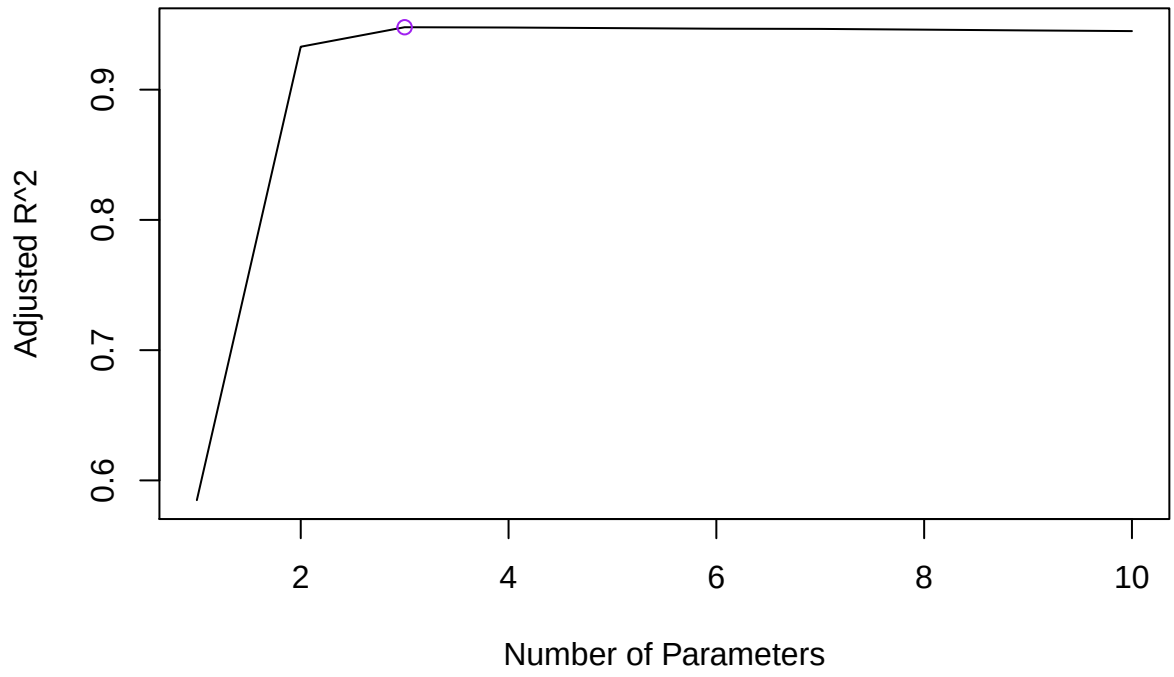



```r
#report the coefficients
lm2 <- lm(y ~ I(x) + I(x^2) + I(x^7), data = data.frame(x = x,y = y))
summary(lm2)
```

```
##
## Call:
## lm(formula = y ~ I(x) + I(x^2) + I(x^7), data = data.frame(x = x,
##     y = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9289 -0.5827 -0.1852  0.5620  2.1319
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.076274   0.117608  26.157  < 2e-16 ***
## I(x)         2.356236   0.127197  18.524  < 2e-16 ***
## I(x^2)      -3.165149   0.087021 -36.372  < 2e-16 ***
## I(x^7)       0.010468   0.001949   5.372 5.42e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9453 on 96 degrees of freedom
## Multiple R-squared:  0.9495, Adjusted R-squared:  0.9479
## F-statistic: 601.6 on 3 and 96 DF,  p-value: < 2.2e-16
```

According to BIC, the model contains 3 predictors which are X, X^2, X^7 The best models obtained according to BIC is: $Y = 3.08 + 2.36 * X$-$3.17* X^2 + 0.01 * X^7$

```r
#Adjusted R^2
plot(1:10, summary$adjr2, xlab = "Number of Parameters", ylab = "Adjusted R^2", type = "l")
max_adjr2 <- max(summary$adjr2)
points(c(1:10)[summary$adjr2 == max_adjr2], max_adjr2, pch = 1, col = "purple")
```




```r
#report the coefficients
lm3 <- lm(y ~ I(x) + I(x^2) + I(x^7), data=data.frame(x = x, y = y))
summary(lm3)
```

```
##
## Call:
## lm(formula = y ~ I(x) + I(x^2) + I(x^7), data = data.frame(x = x,
##     y = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9289 -0.5827 -0.1852  0.5620  2.1319
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.076274   0.117608  26.157  < 2e-16 ***
## I(x)         2.356236   0.127197  18.524  < 2e-16 ***
## I(x^2)      -3.165149   0.087021 -36.372  < 2e-16 ***
## I(x^7)       0.010468   0.001949   5.372 5.42e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9453 on 96 degrees of freedom
## Multiple R-squared:  0.9495, Adjusted R-squared:  0.9479
## F-statistic: 601.6 on 3 and 96 DF,  p-value: < 2.2e-16
```

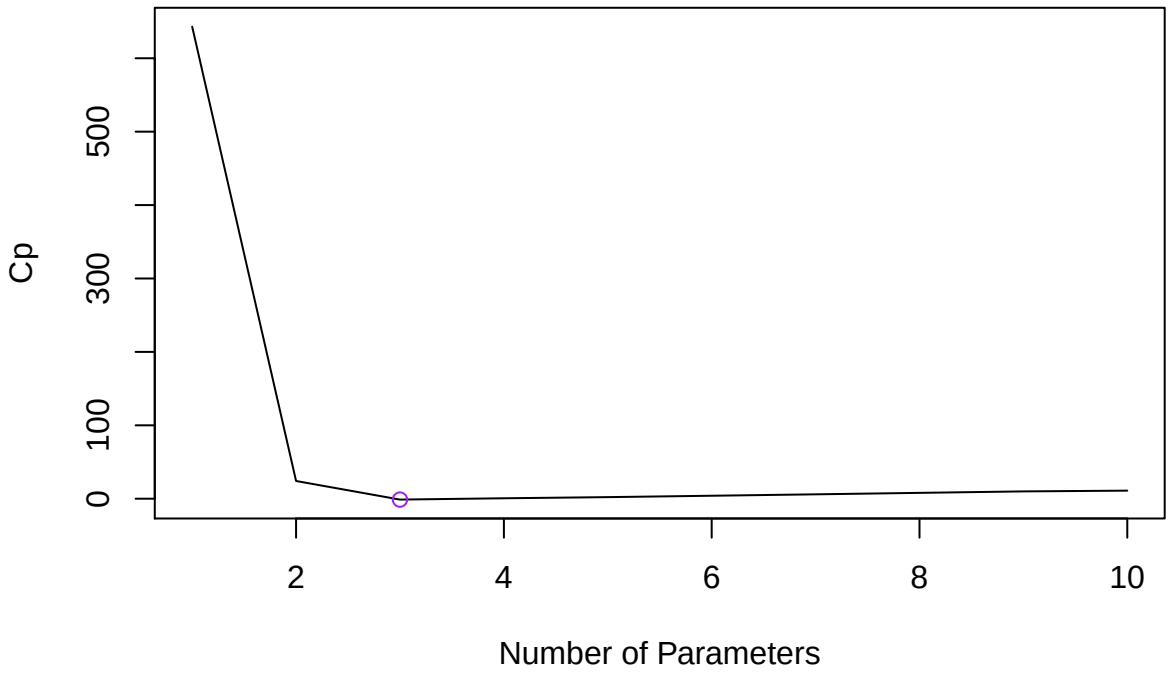
According to adjusted R^2, the model contains 3 predictors which are X, X^2, X^7 The best models obtained according to adjusted R^2 is: $Y = 3.08 + 2.36 * X\text{-}3.17* X^2 + 0.01 * X^7$

(d) Repeat (c), using forward stepwise selection and also using backward stepwise selection. How does your answer compare to the results in (c)?

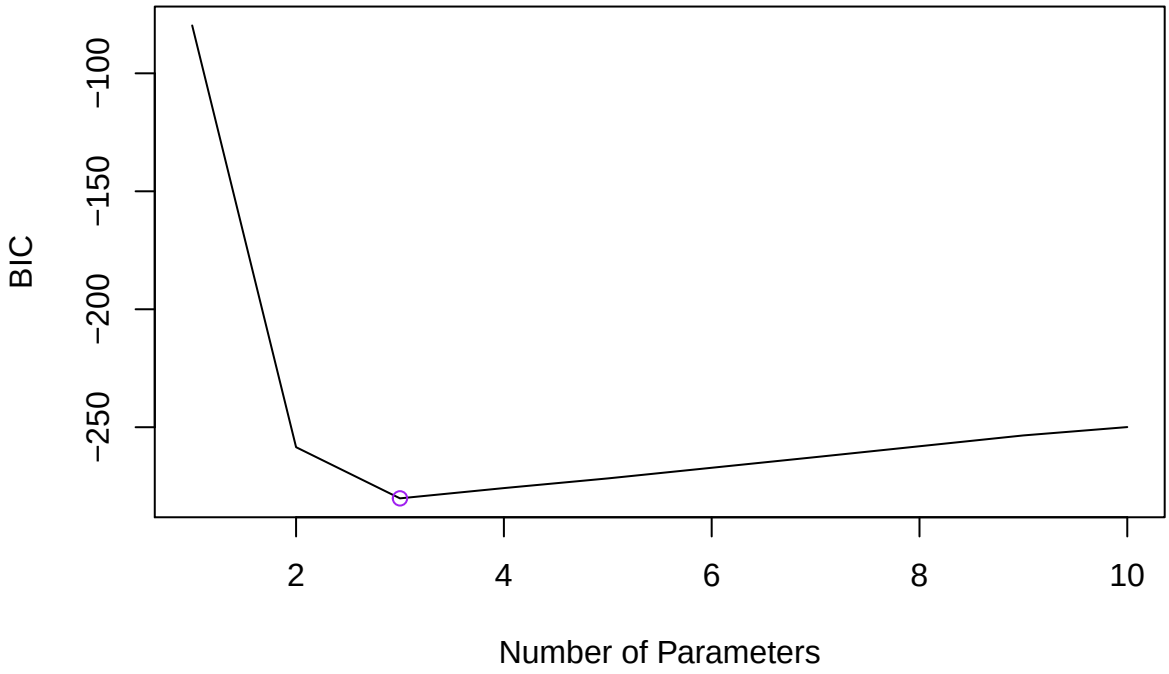
```
#forward stepwise selection
leaps1 <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
                       I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10),
                     data = data.frame(x = x, y = y), nvmax = 10, method = "forward")
(summary1 <- summary(leaps1))
```

```
## Subset selection object
## Call: regsubsets.formula(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data.frame(x = x,
##     y = y), nvmax = 10, method = "forward")
## 10 Variables  (and intercept)
##          Forced in Forced out
## x            FALSE      FALSE
## I(x^2)       FALSE      FALSE
## I(x^3)       FALSE      FALSE
## I(x^4)       FALSE      FALSE
## I(x^5)       FALSE      FALSE
## I(x^6)       FALSE      FALSE
## I(x^7)       FALSE      FALSE
## I(x^8)       FALSE      FALSE
## I(x^9)       FALSE      FALSE
## I(x^10)      FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: forward
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9) I(x^10)
## 1  ( 1 )  " " "*"    " "    " "    " "    " "    " "    " "    " "    " "
## 2  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    " "    " "    " "
## 3  ( 1 )  "*" "*"    " "    " "    " "    " "    "*"    " "    " "    " "
## 4  ( 1 )  "*" "*"    " "    "*"    " "    " "    "*"    " "    " "    " "
## 5  ( 1 )  "*" "*"    " "    "*"    " "    " "    "*"    " "    "*"    " "
## 6  ( 1 )  "*" "*"    " "    "*"    " "    "*"    "*"    " "    "*"    " "
## 7  ( 1 )  "*" "*"    " "    "*"    "*"    "*"    "*"    " "    "*"    " "
## 8  ( 1 )  "*" "*"    "*"    "*"    "*"    "*"    "*"    " "    "*"    " "
## 9  ( 1 )  "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"    " "
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"
```

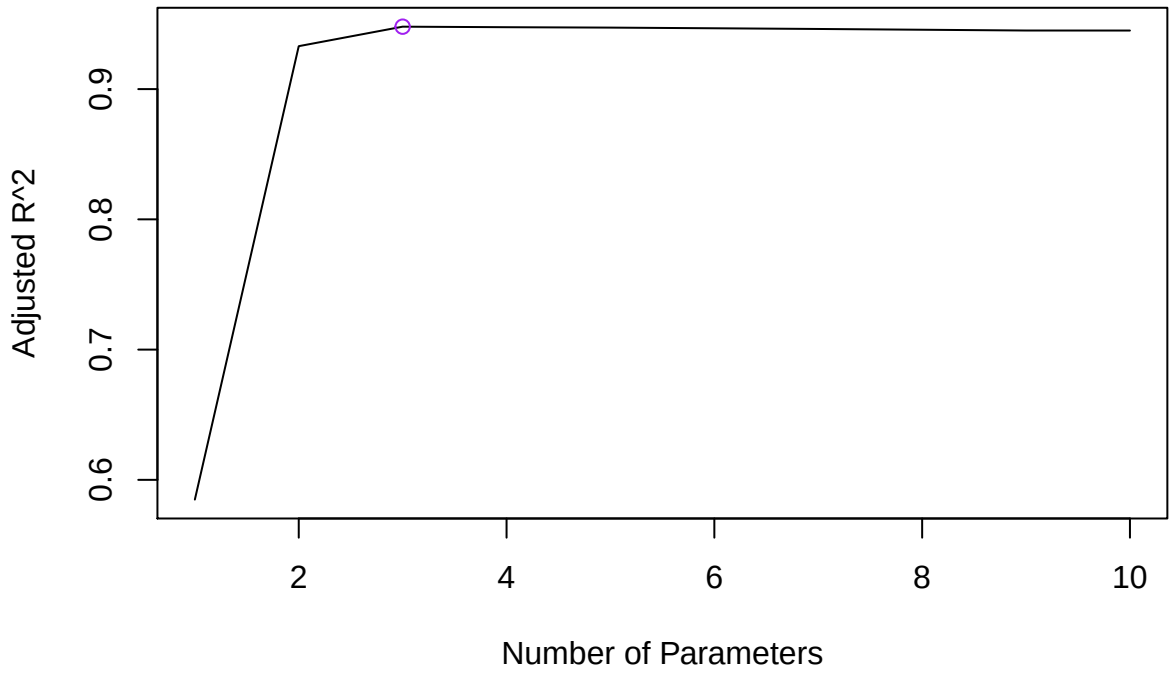
```
#Cp
plot(1:10, summary1$cp, xlab = "Number of Parameters", ylab = "Cp", type = "l")
min_cp1 <- min(summary1$cp)
points(c(1:10)[summary1$cp == min_cp1], min_cp1, pch = 1, col = "purple")
```




```
#BIC
plot(1:10, summary1$bic, xlab="Number of Parameters", ylab="BIC", type = "l")
min_bic1 <- min(summary1$bic)
points(c(1:10)[summary1$bic == min_bic1], min_bic1, pch = 1, col = "purple")
```




```
#Adjusted R^2
plot(1:10, summary1$adjr2, xlab = "Number of Parameters", ylab = "Adjusted R^2", type = "l")
max_adjr2_1 <- max(summary1$adjr2)
points(c(1:10)[summary1$adjr2 == max_adjr2_1], max_adjr2_1, pch = 1, col = "purple")
```



Using forward stepwise selection, the best model selected from C_p, BIC, and adjusted R^2 all contain 3 predictors which are X, X^2, X^7. The best model is the same as we got in (c) which is: $Y = 3.08 + 2.36 * X$-3.17* $X^2 + 0.01 * X^7$

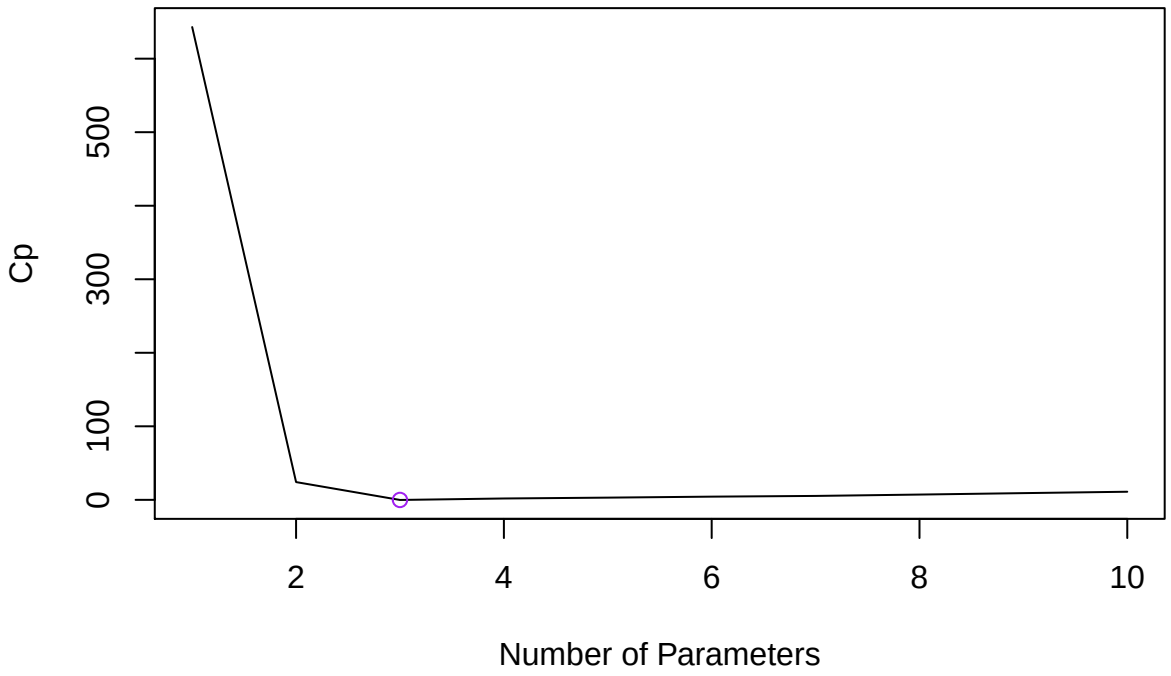
Next let's do backwards stepwise selection:

```
#backward
leapsback <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
                        I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10),
                        data = data.frame(x = x, y = y), nvmax = 10, method = "backward")
summaryback <- summary(leapsback)
summaryback
```

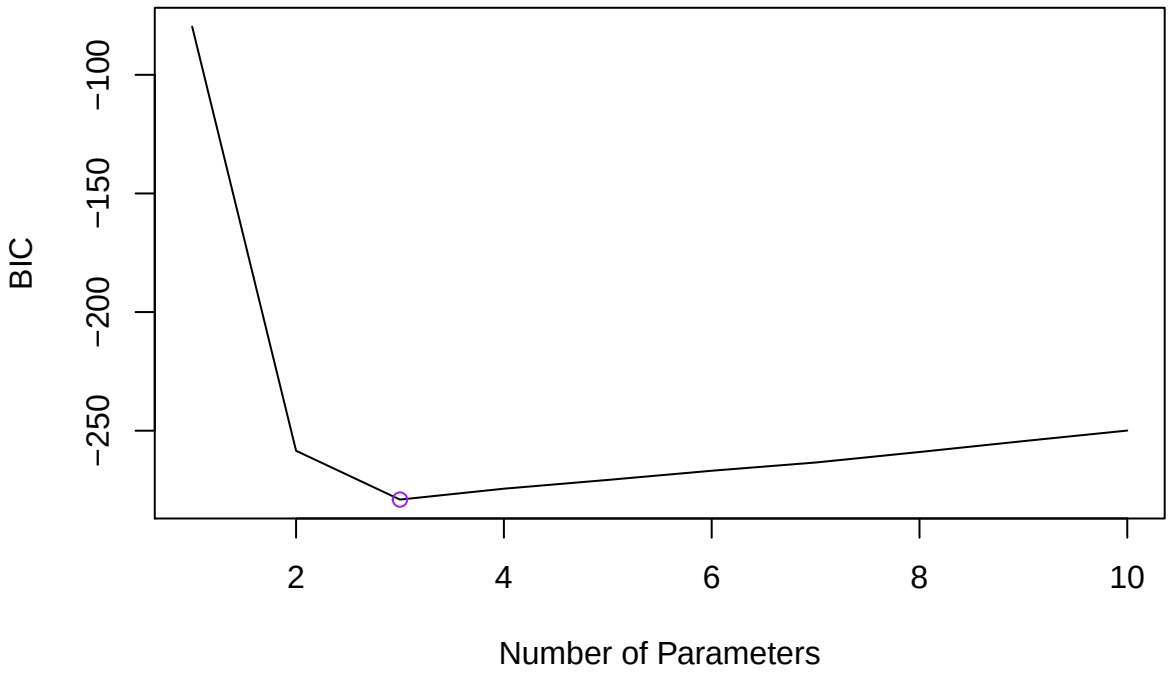
```
## Subset selection object
## Call: regsubsets.formula(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) +
##     I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10), data = data.frame(x = x,
##     y = y), nvmax = 10, method = "backward")
## 10 Variables  (and intercept)
##          Forced in Forced out
## x            FALSE      FALSE
## I(x^2)       FALSE      FALSE
## I(x^3)       FALSE      FALSE
## I(x^4)       FALSE      FALSE
## I(x^5)       FALSE      FALSE
## I(x^6)       FALSE      FALSE
## I(x^7)       FALSE      FALSE
## I(x^8)       FALSE      FALSE
## I(x^9)       FALSE      FALSE
## I(x^10)      FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: backward
##           x   I(x^2) I(x^3) I(x^4) I(x^5) I(x^6) I(x^7) I(x^8) I(x^9) I(x^10)
## 1  ( 1 )  " " "*"    " "    " "    " "    " "    " "    " "    " "    " "
```

```
## 2  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    " "    " "    " "
## 3  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    " "    "*"    " "
## 4  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    "*"    "*"    " "
## 5  ( 1 )  "*" "*"    " "    " "    " "    " "    " "    "*"    "*"    "*"
## 6  ( 1 )  "*" "*"    " "    " "    " "    "*"    " "    "*"    "*"    "*"
## 7  ( 1 )  "*" "*"    " "    "*"    " "    "*"    " "    "*"    "*"    "*"
## 8  ( 1 )  "*" "*"    " "    "*"    "*"    "*"    " "    "*"    "*"    "*"
## 9  ( 1 )  "*" "*"    " "    "*"    "*"    "*"    "*"    "*"    "*"    "*"
## 10  ( 1 ) "*" "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"    "*"
```

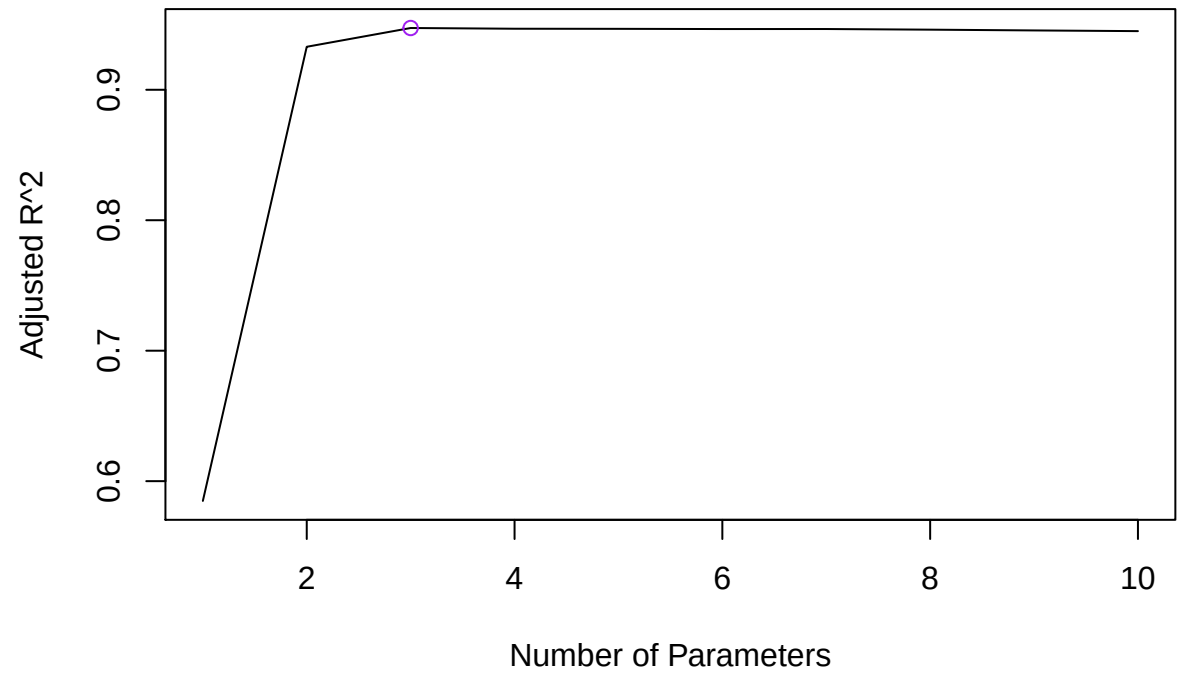
```r
#Cp
plot(1:10, summaryback$cp, xlab = "Number of Parameters", ylab = "Cp", type = "l")
min_cpback <- min(summaryback$cp)
points(c(1:10)[summaryback$cp == min_cpback], min_cpback, pch = 1, col = "purple")
```




```r
#BIC
plot(1:10, summaryback$bic, xlab = "Number of Parameters", ylab = "BIC", type = "l")
min_bicback <- min(summaryback$bic)
points(c(1:10)[summaryback$bic == min_bicback], min_bicback, pch = 1, col = "purple")
```




```
#Adjusted R^2
plot(1:10, summaryback$adjr2, xlab = "Number of Parameters", ylab = "Adjusted R^2",
     type = "l")
max_adjr2_back <- max(summaryback$adjr2)
points(c(1:10)[summaryback$adjr2 == max_adjr2_back], max_adjr2_back, pch = 1, col = "purple")
```



Using backwards stepwise selection, the best model selected from C_p, BIC, and adjusted R^2 all contain 3 predictors which are X, X^2, X^9. Next let's report the coefficients:

```r
#report the coefficients
lmback <- lm(y ~ I(x) + I(x^2) + I(x^9), data = data.frame(x = x, y = y) )
summary(lmback)
```

```
##
## Call:
## lm(formula = y ~ I(x) + I(x^2) + I(x^9), data = data.frame(x = x,
##     y = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9207 -0.5848 -0.1950  0.5785  2.1018
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.0788814  0.1184085  26.002  < 2e-16 ***
## I(x)         2.4198180  0.1223545  19.777  < 2e-16 ***
## I(x^2)      -3.1772356  0.0883149 -35.976  < 2e-16 ***
## I(x^9)       0.0018705  0.0003569   5.241 9.44e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9506 on 96 degrees of freedom
## Multiple R-squared:  0.9489, Adjusted R-squared:  0.9473
## F-statistic: 594.5 on 3 and 96 DF,  p-value: < 2.2e-16
```

The best model according to backwards stepwise selection is: $Y = 3.079 + 2.420 * X$-$3.177* X^2 + 0.002 * X^9$ This model is different from that in (c).

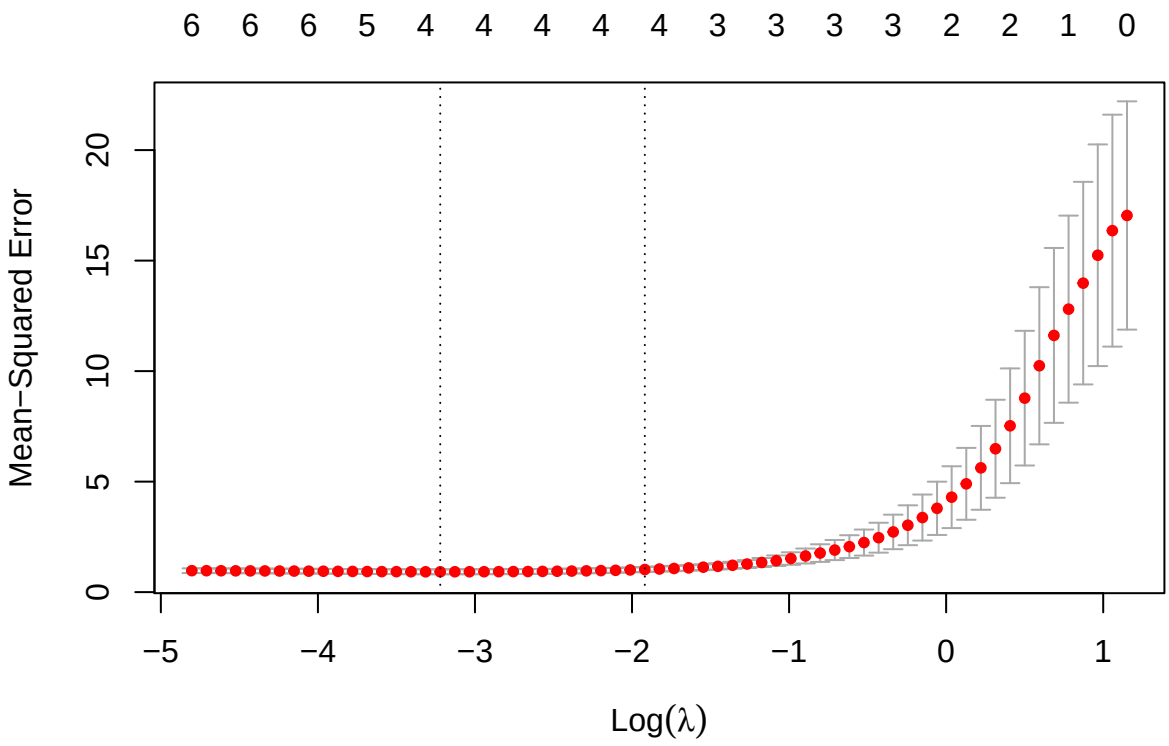
(e) Now fit a LASSO model with `glmnet` function from `glmnet` package to the simulated data, again using $(X, X^2, \cdots, X^{10})$ as predictors. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates, and discuss the results obtained.

```r
library(glmnet)
xmatrix = model.matrix(y ~ x + I(x^2) + I(x^3) + I(x^4) +
                         I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10),
                       data = data.frame(x = x, y = y))[, -1]

# Cross-validation to select lambda
lasso.cv = cv.glmnet(xmatrix, y, alpha = 1)
lasso.cv$lambda.min
```

```
## [1] 0.03991416
```

```r
plot(lasso.cv)
```




```
# coefficient estimate when refitting model with lamda.min
coef(lasso.cv, lasso.cv$lambda.min)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept)  3.039815105
## x            2.230337134
## I(x^2)      -3.103319268
## I(x^3)       .
## I(x^4)       .
## I(x^5)       0.049841076
## I(x^6)       .
## I(x^7)       0.000806843
## I(x^8)       .
## I(x^9)       .
## I(x^10)      .
```

The optimal value of λ is about 0.04. When we chose λ which gives minimum cv error, the Lasso method shrinks the coefficient of $X^3, X^4, X^6, X^8, X^9, X^{10}$to zero, and selects X, X^2, X^5, X^7 as predictors.

(f) Now generate a response vector Y according to the model

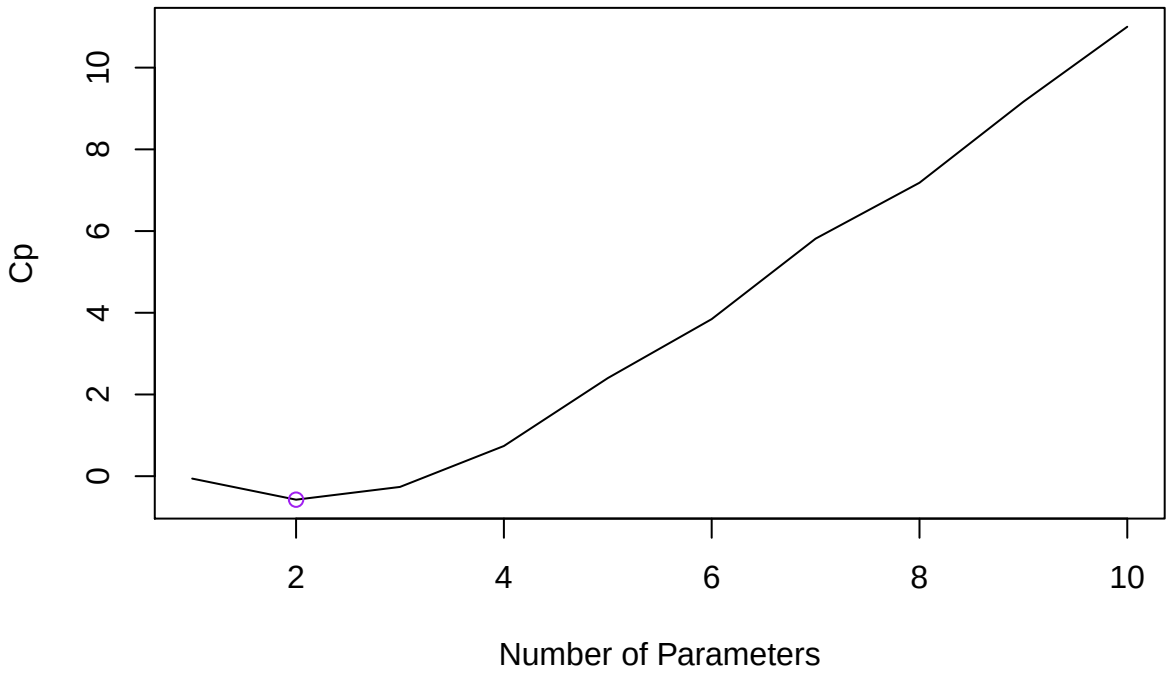
$$Y = \beta_0 + \beta_7 X^7 + \epsilon,$$

where $\beta_7 = 7$, and perform best subset selection and the LASSO. Discuss the results obtained.

First let's perform best subset selection.

```r
y1 <- 3 + 7 * x^7 + epsilon
#best subsets
library(leaps)
sub_full <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8)
                     + I(x^9) + I(x^10), data = data.frame(x = x, y = y1), nvmax = 10)
sub_summary <- summary(sub_full)
```

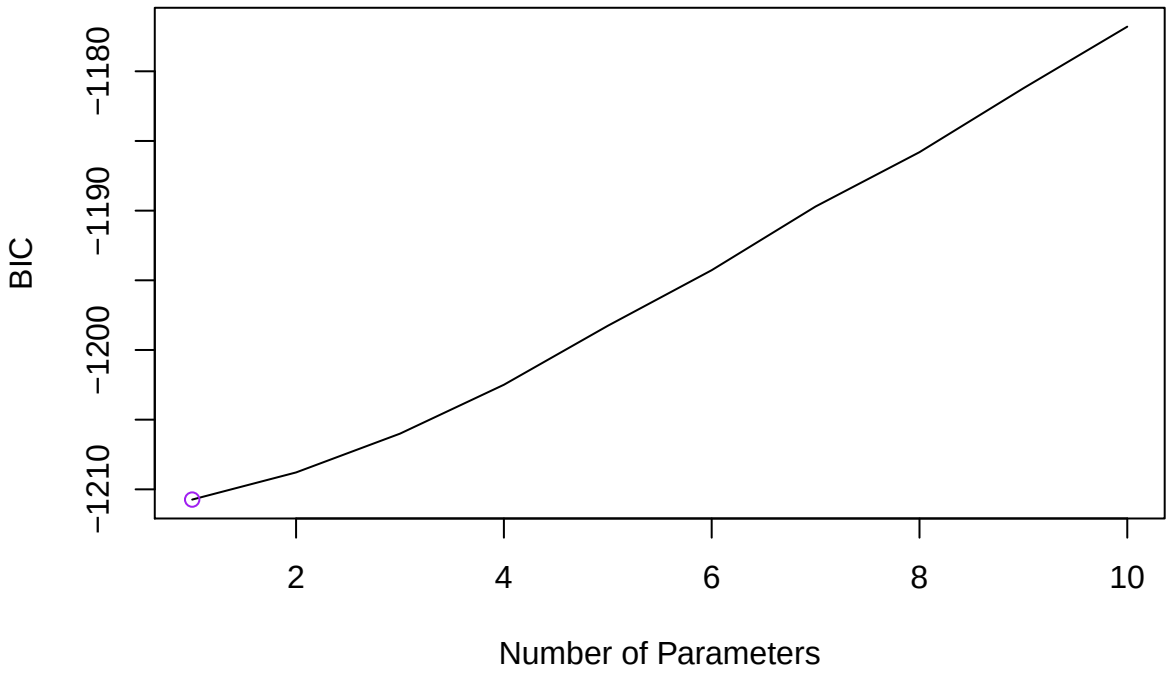
```r
#Cp
plot(1:10, sub_summary$cp, xlab = "Number of Parameters", ylab = "Cp", type = "l")
subset_min_cp <- min(sub_summary$cp)
points(c(1:10)[sub_summary$cp == subset_min_cp], subset_min_cp, pch = 1, col = "purple")
```




```r
#report coefficient
coef(sub_full, 2)
```

```
## (Intercept)      I(x^2)      I(x^7)
##   3.0704904  -0.1417084   7.0015552
```

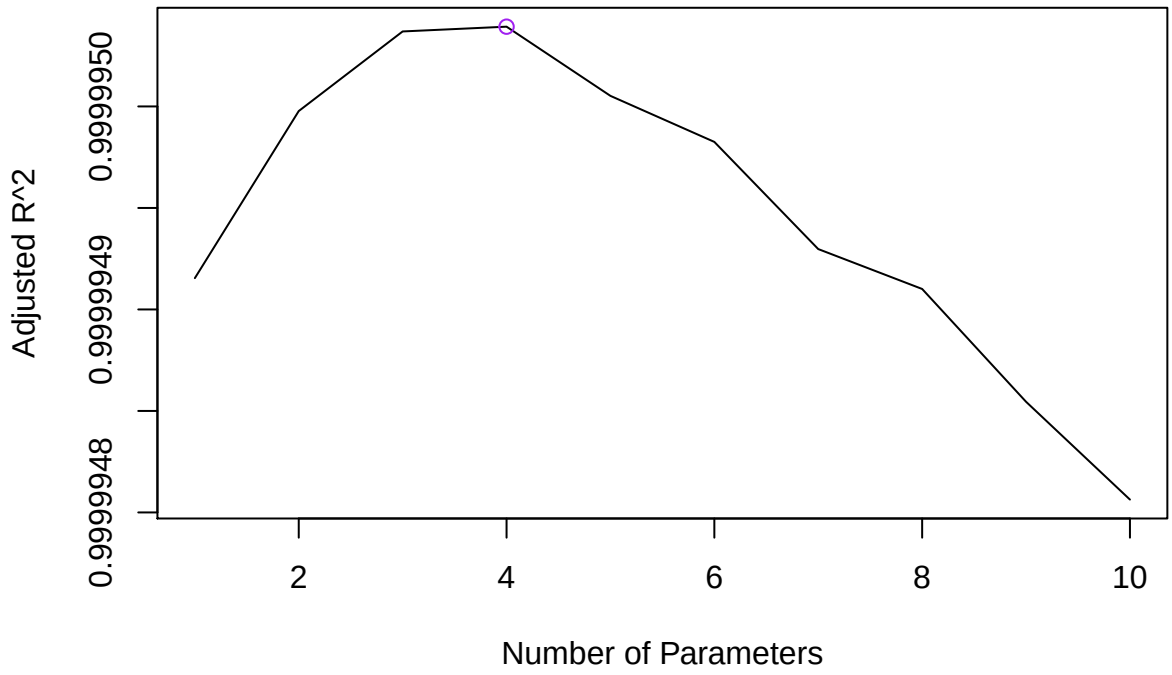
```r
#BIC
plot(1:10, sub_summary$bic, xlab = "Number of Parameters", ylab = "BIC", type = "l")
subset_min_bic <- min(sub_summary$bic)
points(c(1:10)[sub_summary$bic == subset_min_bic], subset_min_bic, pch = 1, col = "purple")
```




```
#report coefficient
coef(sub_full, 1)
```

```
## (Intercept)      I(x^7)
##     2.95894     7.00077
```

```
#adjusted r^2
plot(1:10, sub_summary$adjr2, xlab = "Number of Parameters",
     ylab = "Adjusted R^2", type = "l")
subset_max_adjr2 <- max(sub_summary$adjr2)
points(c(1:10)[sub_summary$adjr2 == subset_max_adjr2],
       subset_max_adjr2, pch = 1, col="purple")
```




```
#report coefficient
coef(sub_full, 4)
```

```
## (Intercept)           x      I(x^2)      I(x^3)      I(x^7)
##   3.0762524   0.2914016  -0.1617671  -0.2526527   7.0091338
```

If we perform best subset selection, the best model obtained according to C_p has 2 parameters, which are X^2 and X^7; the best model obtained according to BIC has only 1 parameter, which is X^7; the best model obtained according to adjusted R^2 has 4 parameters, which are X, X^2, X^3 and X^7.

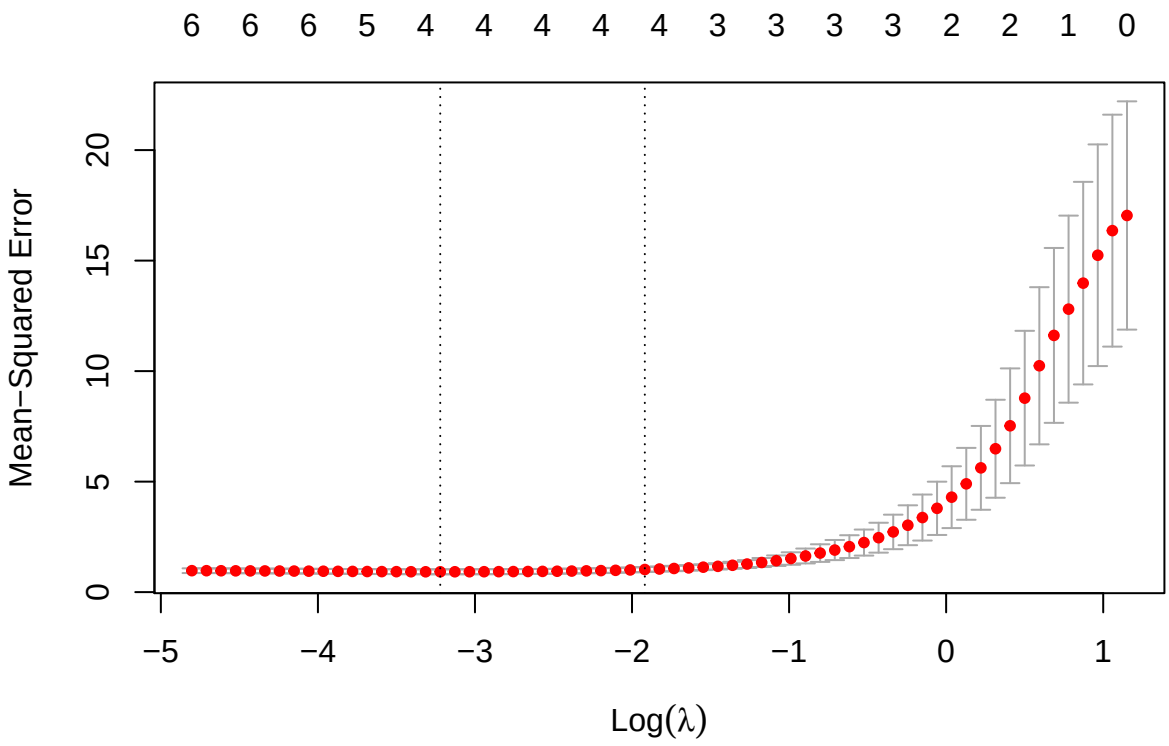
Let's do Lasso:

```
library(glmnet)
xmatrix = model.matrix(y ~ x + I(x^2) + I(x^3) + I(x^4) +
                         I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9) + I(x^10),
                       data = data.frame(x = x, y = y))[, -1]

# Cross-validation to select lambda
lasso.cv1 = cv.glmnet(xmatrix, y1, alpha = 1)
lasso.cv1$lambda.min
```

```
## [1] 12.36884
```

```
plot(lasso.cv)
```




```
# coefficient estimate when refitting model with lamda.min
coef(lasso.cv1, lasso.cv1$lambda.min)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) 3.820215
## x           .
## I(x^2)      .
## I(x^3)      .
## I(x^4)      .
## I(x^5)      .
## I(x^6)      .
## I(x^7)      6.796694
## I(x^8)      .
## I(x^9)      .
## I(x^10)     .
```

If we perform Lasso, the best model has 1 parameter1, which is X^7. The model is$Y = 3.82 + 6.80 * X^7$.

2. (Prediction, [ISL] 6.9, *20 pt*) In this exercise, we will predict the number of applications received (`Apps`) using the other variables in the `College` data set from `ISLR` package.

(a) Randomly split the data set into equal sized training set and test set (1:1).

```
library(ISLR)
data(College)
#split data in to 50% traing set and 50% test set
s <- sample(nrow(College), floor(nrow(College)*0.5), replace = F)
training <- College[s,]
test <- College[-s,]
```

(b) Fit a linear model using least squares on the training set, and report the test error obtained.

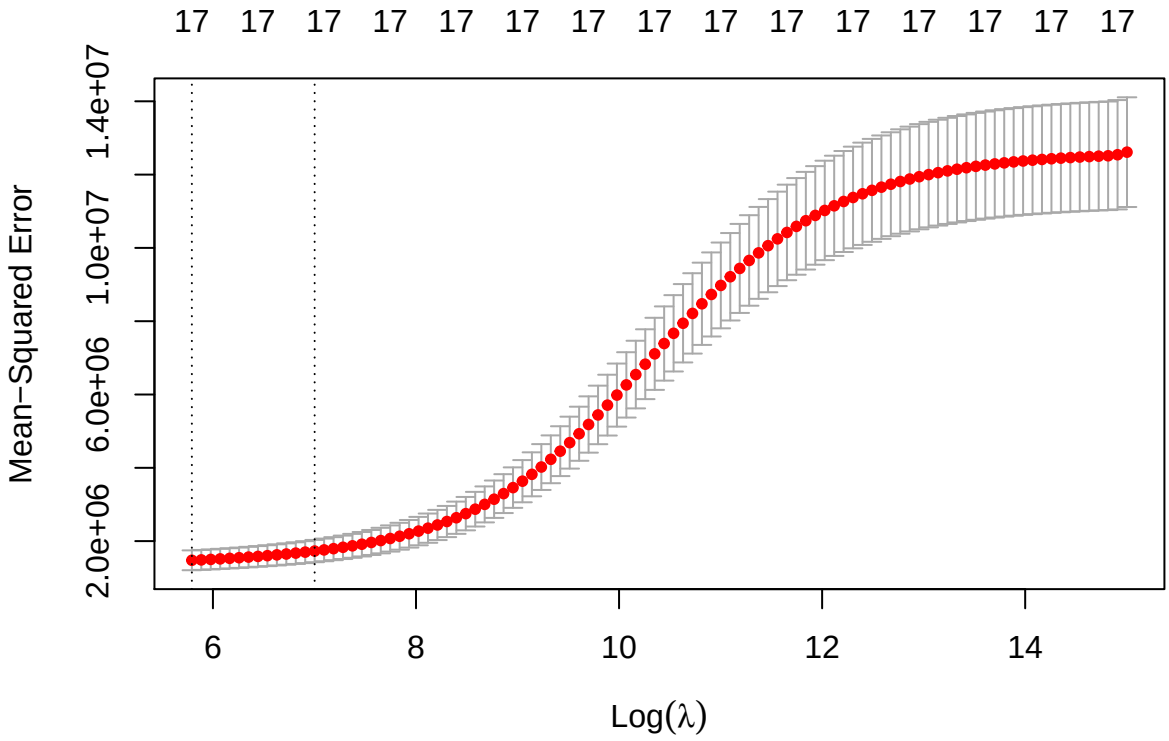
```
fit <- lm(Apps ~.,  data = training)
pred <- predict(fit, test)
mean((pred - test$Apps)^2)
```

```
## [1] 1365049
```

The test error is 1365049.

(c) Fit a ridge regression model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained.

```
xtrain = model.matrix (Apps~., training)
ytrain = training$Apps
xtest = model.matrix (Apps~., test)
ytest = test$Appscv
#use 5-fold cv:
cv.out = cv.glmnet (xtrain, ytrain, alpha = 0, nfolds = 5)
plot(cv.out)
```




```
(bestlamda = cv.out$lambda.min)
```

```
## [1] 327.8098
```

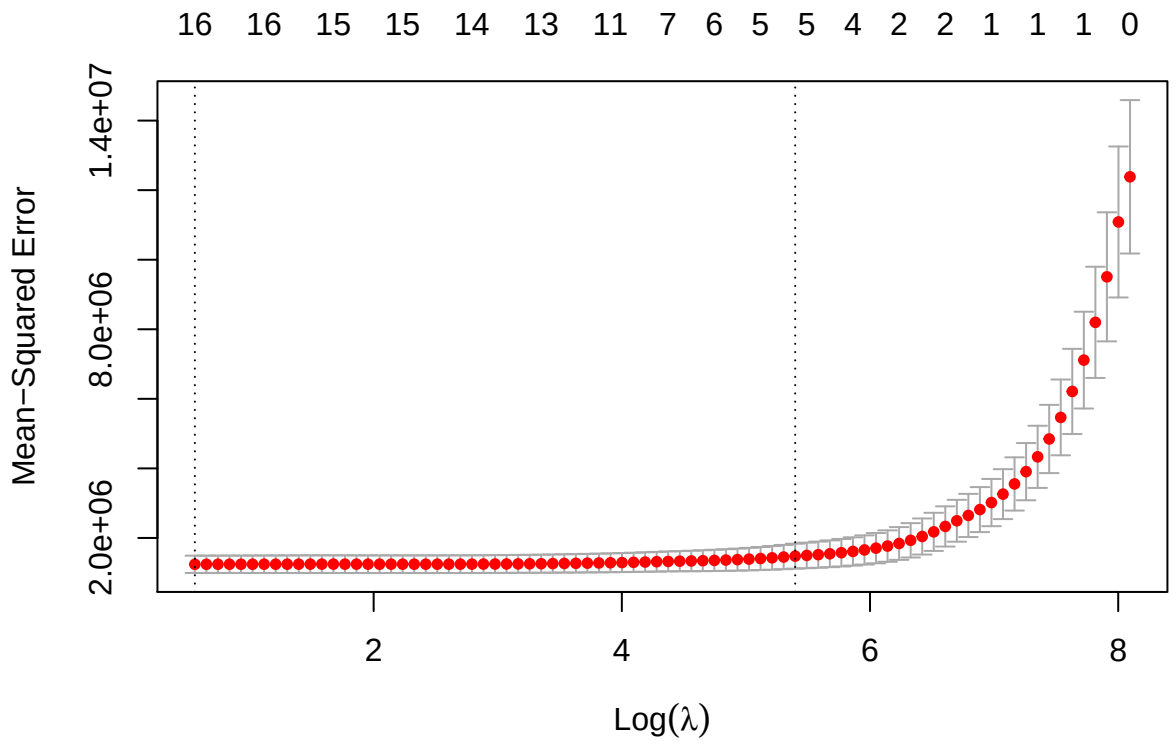
```
#modelfrom ridge regression
ridge_rg = glmnet(xtrain, ytrain, alpha = 0)
#test error
#Fitting trainning model on test set
ridge_pred <- predict(ridge_rg, s = bestlamda, newx = xtest)
#Calculating test error
mean((ridge_pred - ytest)^2)
```

```
## [1] NaN
```

The chosen λ is 327.8098 and the test error result from ridge regression model is 2079222.

(d) Fit a LASSO model on the training set, with λ chosen by 5-fold cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.

```
#use 5-fold cv:
cv.outlasso = cv.glmnet (xtrain, ytrain, alpha = 1, nfolds = 5)
plot(cv.outlasso)
```




```
(bestlamlasso = cv.outlasso$lambda.min)
```

```
## [1] 1.749422
```

```
#modelfrom lasso regression
fit.lasso <- glmnet(xtrain, ytrain, alpha = 1)
lasso_rg = glmnet(xtrain, ytrain, alpha = 1, lambda = bestlamlasso)
#the coefficient
```

```
lasso_coef <- coef(lasso_rg)
#test error
#Fitting trainning model on test set
lasso_pred = predict(fit.lasso, s = bestlamlasso, newx = xtest)
#Calculating test error
mean((lasso_pred - ytest)^2)
```

```
## [1] NaN
```

```
#the number of non-zero coefficient estimates.
predict(fit.lasso, type = "coefficients", s = bestlamlasso)
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                           1
## (Intercept) -415.41151489
## (Intercept)    .
## PrivateYes  -928.35328935
## Accept         1.26564141
## Enroll        -0.64401763
## Top10perc     49.94486035
## Top25perc    -16.33532750
## F.Undergrad    0.14788769
## P.Undergrad   -0.08810625
## Outstate      -0.06601327
## Room.Board     0.19532427
## Books         -0.02018463
## Personal       .
## PhD           -7.14924058
## Terminal      -7.93478509
## S.F.Ratio      6.74996160
## perc.alumni   -5.02324489
## Expend         0.10214012
## Grad.Rate     17.02026757
```

The chosen λ is 1.749422 The test error is 1359773. There are 16 non-zero coefficient estimates (not including intercept).

(e) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these three approaches?

To compare accuracy, we need to calculate R^2 for each approach:

```
avgapps <- mean(ytest)
```

```
## Warning in mean.default(ytest): argument is not numeric or logical: returning NA
```

```
(lm_r2 <- 1 - mean((pred - test$Apps)^2) / mean((avgapps - ytest)^2))
```

```
## [1] NaN
```

```
(ridge_r2 <- 1 - mean((ridge_pred-ytest)^2) /
  mean((avgapps - ytest)^2))
```

```
## [1] NaN
```

```
(lasso_r2 <- 1- mean((lasso_pred - ytest)^2) /
  mean((avgapps - ytest)^2))
```

```
## [1] NaN
```

The R^2 of least square linear model and lasso model is about 92%, which are higher than that of ridge regression model, which is about 88%. So the least square model and lasso model can predict the number of college applications received more accurately.

The model generated from Lasso method has much smaller test error (1359773) than that from Ridge regression method which is 2079222 in this dataset. The test error from linear model using least squares is similiar to the test error resulting from Lasso model which is 1365049.