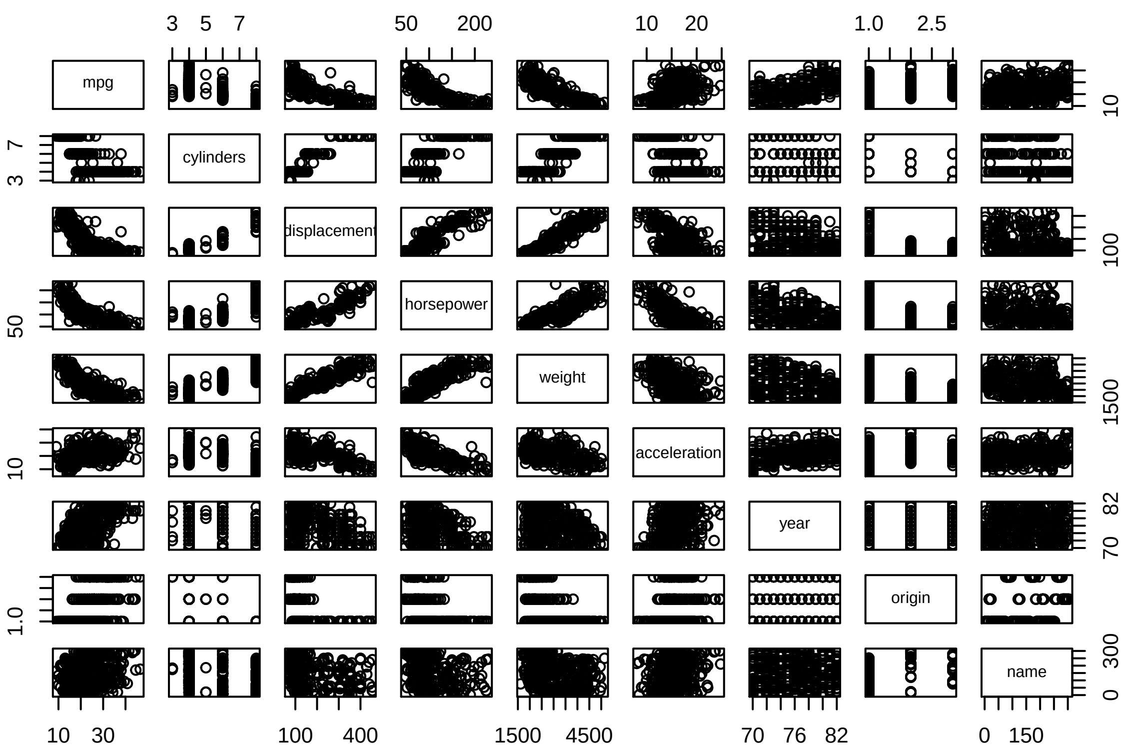
Assignment 6 Chap 7

Problem 1–Exercise 2

(a) g(x)=0. RSS will be ignored because a large penalty forces g→0
(b) g(x)=c. RSS will be ignored because a large penalty forces first derivative g→0
(c) g(x)=ax+c. RSS will be ignored because a large penalty forces second derivative g→0
(d) g(x)=axˆ2+c. RSS will be ignored because a large penalty forces third derivative g→0
(e) The penalty is 0. It is a linear regression to select g by minimizing RSS.

Problem 2–Exercise 8

```r
set.seed(1)
library(ISLR)
attach(Auto)
pairs(Auto)
```



mpg is inversely proprotional to cylinders, displacement, horsepower, and weight. I will try horsepower to check its non-linear relationships.

Polynomial

```r
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-18
```

```r
library(boot)
cv.error = rep(NA, 10)
for (d in 1:10) {
  fit = glm(mpg~poly(horsepower,d), data = Auto)
  cv.error[d] = cv.glm(Auto, fit, K = 10)$delta[1]
}
which.min(cv.error)
```

```
## [1] 7
```

Step functions

```r
cv.error = rep(NA, 10)
for (c in 2:10) {
  Auto$horseCut=cut(Auto$horsepower, c)
  fit = glm(mpg~horseCut, data = Auto)
  cv.error[d] = cv.glm(Auto, fit, K = 10)$delta[1]
}
which.min(cv.error)
```

```
## [1] 10
```

Splines

```r
library(splines)
cv.error = rep(NA, 10)
for (df in 3:10) {
  fit = glm(mpg~ns(horsepower, df=df), data = Auto)
  cv.error[d] = cv.glm(Auto, fit, K = 10)$delta[1]
}
which.min(cv.error)
```

```
## [1] 10
```

GAM

```r
library(gam)
```

```
## Loaded gam 1.16.1
```

```r
fit = gam(mpg~ s(horsepower, 10) + s(horsepower, 7) + s(horsepower, 3) + s(horsepower, 4), data = Auto)
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```r
summary(fit)
```

```
##
## Call: gam(formula = mpg ~ s(horsepower, 10) + s(horsepower, 7) + s(horsepower,
##     3) + s(horsepower, 4), data = Auto)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -15.4032  -2.4340  -0.1092   2.1326  14.9656
##
## (Dispersion Parameter for gaussian family taken to be 18.7168)
##
##     Null Deviance: 23818.99 on 391 degrees of freedom
## Residual Deviance: 6925.222 on 370 degrees of freedom
## AIC: 2284.14
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                    Df  Sum Sq Mean Sq F value    Pr(>F)
## s(horsepower, 10)   1 14433.1 14433.1  771.13 < 2.2e-16 ***
## Residuals         370  6925.2    18.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                   Npar Df  Npar F  Pr(F)
## (Intercept)
## s(horsepower, 10)       9 12.4581 <2e-16 ***
## s(horsepower, 7)        6  0.4215 0.8646
## s(horsepower, 3)        2  0.0306 0.9698
## s(horsepower, 4)        3  0.0646 0.9786
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As all the methods show, the relationship between horsepower and mpg is highly non-linear, which is almost about df 10.

Problem 3–Exercise 9

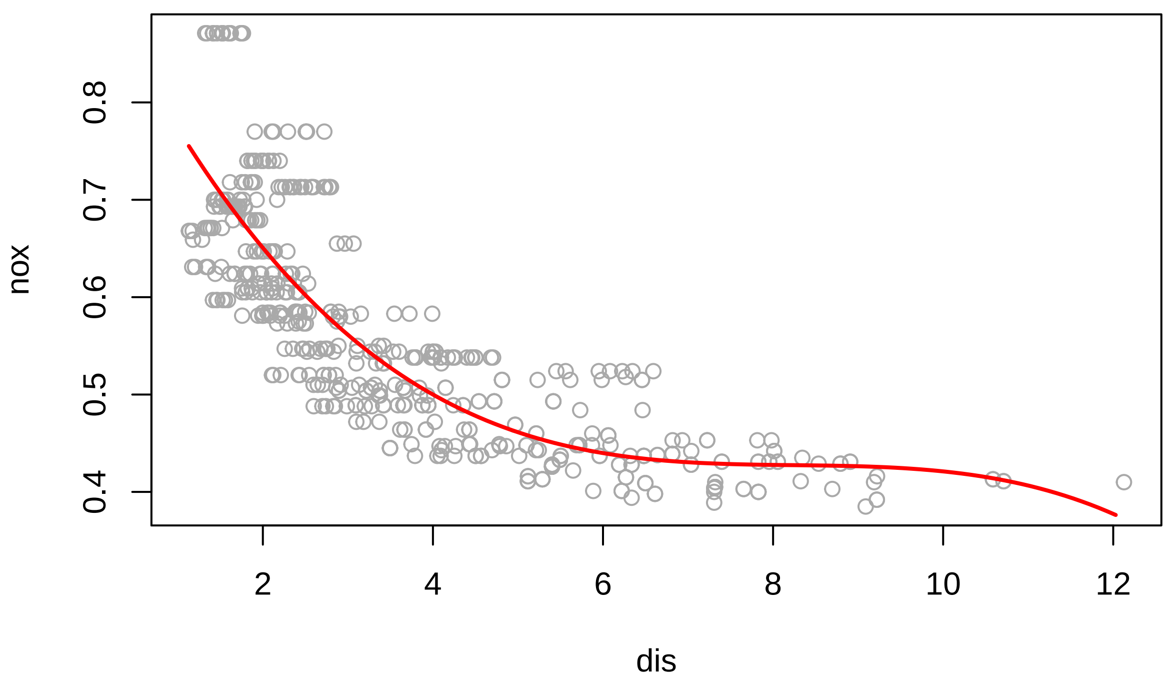
(a)

```r
set.seed(1)
library(MASS)
attach(Boston)
lm.9a=lm(nox ~ poly(dis, 3), data = Boston)
summary(lm.9a)
```

```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
dislim=range(dis)
dis.grid = seq(from=dislim[1], to = dislim[2], by=0.1)
lm.pred= predict(lm.9a, list(dis = dis.grid))
plot(nox~dis, data = Boston, col = "darkgrey")
lines(dis.grid, lm.pred, col="red", lwd = 2)
```

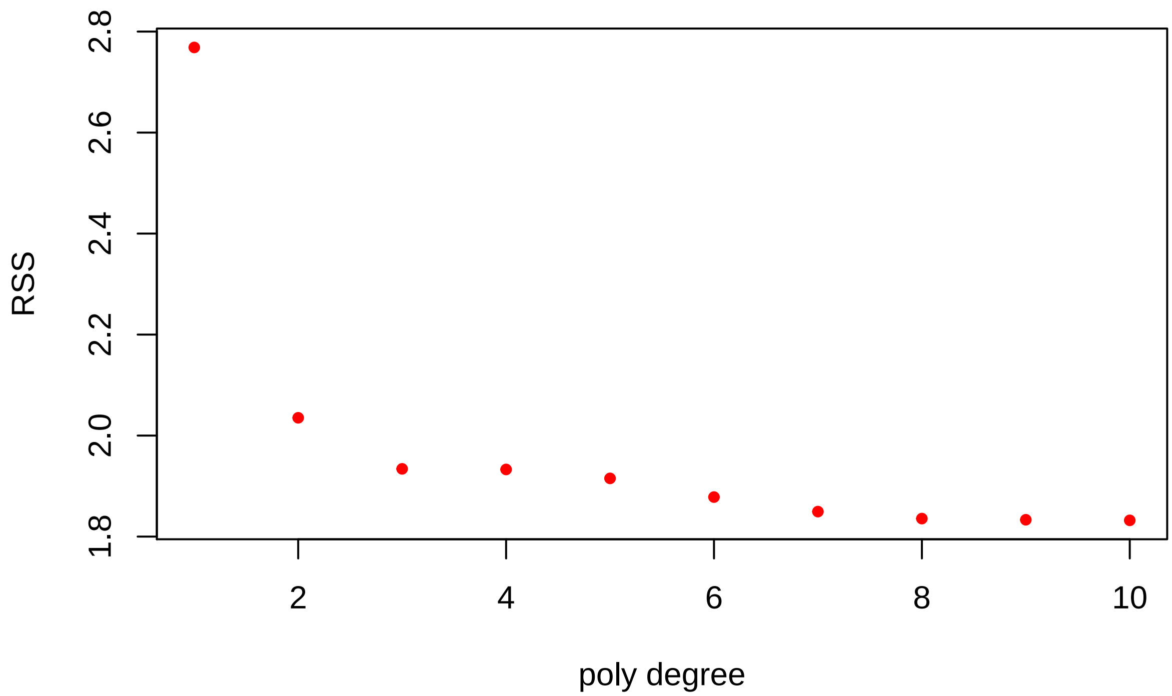


(b)

```
poly.resi = rep(NA, 10)
for (d in 1:10) {
  fit = lm(nox~poly(dis,d), data = Auto)
  poly.resi[d]= sum(fit$residuals^2)
}
poly.resi
```

```
##  [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484
##  [8] 1.835630 1.833331 1.832171
```

```r
plot(1:10,poly.resi, xlab = "poly degree", ylab = "RSS", col="red", pch=20)
```

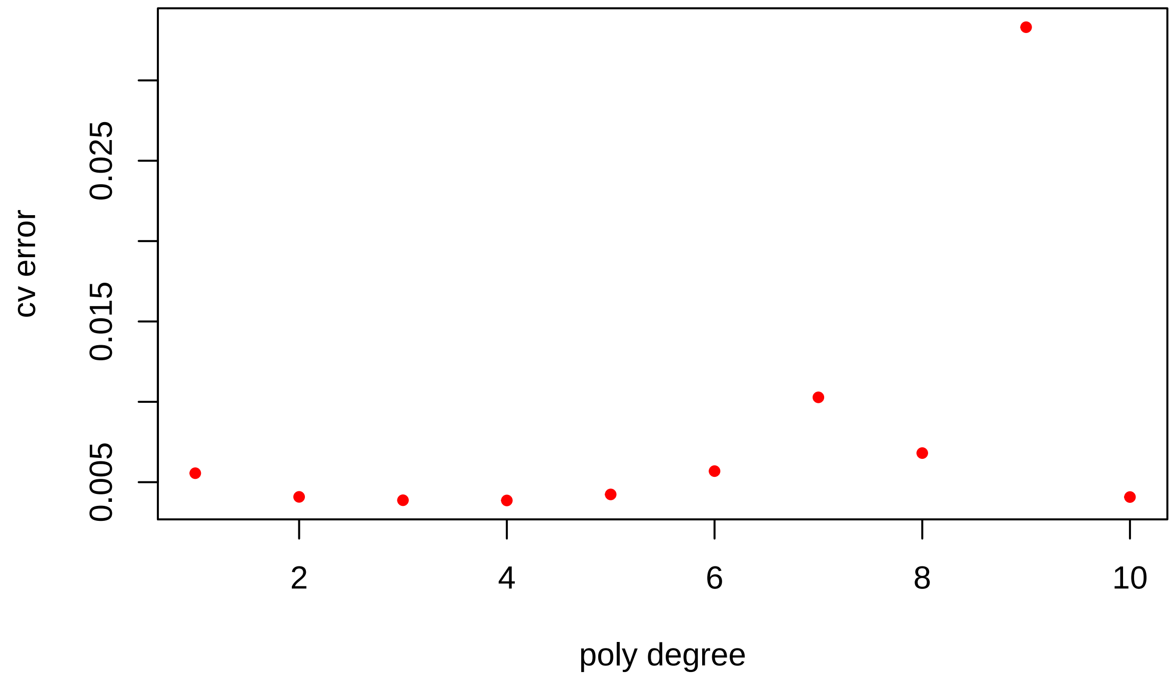


(c)

```r
cv.error = rep(NA, 10)
for (d in 1:10) {
  fit = glm(nox~poly(dis,d), data = Auto)
  cv.error[d]= cv.glm(Boston, fit, K = 10)$delta[1]
}
cv.error
```

```
##  [1] 0.005558263 0.004085706 0.003876521 0.003863342 0.004237452
##  [6] 0.005686862 0.010278897 0.006810868 0.033308607 0.004075599
```

```r
plot(1:10,cv.error, xlab = "poly degree", ylab = "cv error", col="red", pch=20)
```



I will choose 3 as my optimal degree, since it got the second best result, and is simpler than the best model, poly=4.

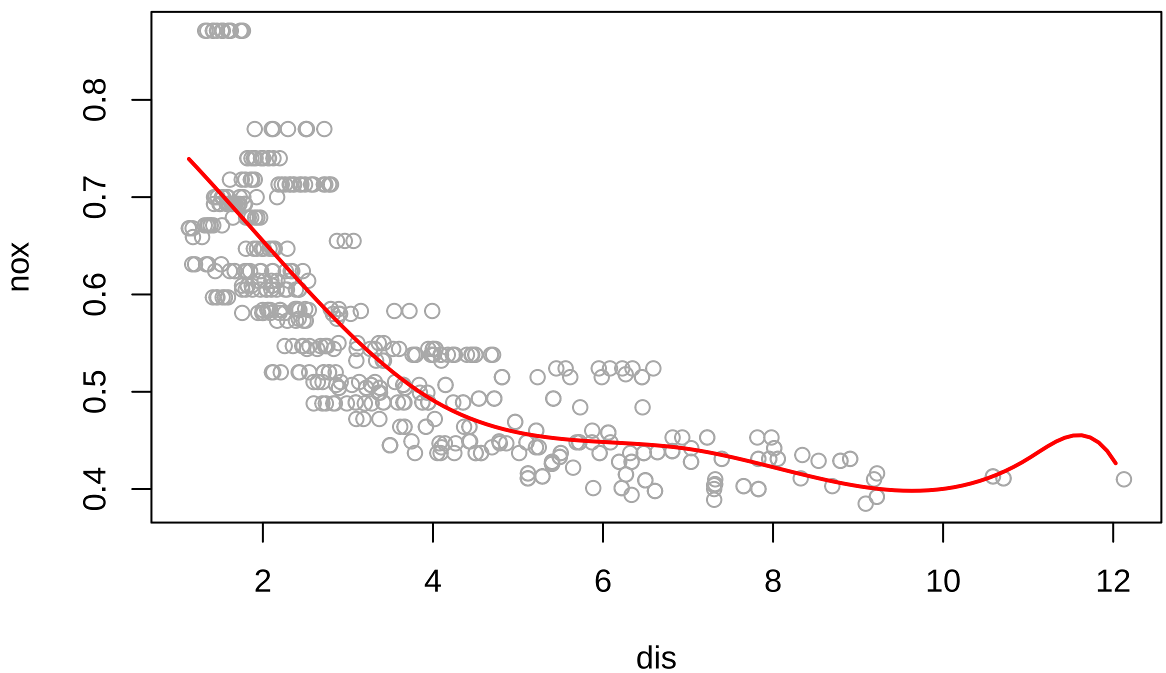
(d) The range of dis is from 1 to 13, then we split in into 4 parts, 3 knots.

```
sp.9d=lm(nox~bs(dis, df=4, knots = c(4,7,11)), data=Boston)
summary(sp.9d)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, df = 4, knots = c(4, 7, 11)), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.124567 -0.040355 -0.008702  0.024740  0.192920
##
## Coefficients:
##                                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)                              0.73926    0.01331  55.537  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))1 -0.08861    0.02504  -3.539  0.00044
## bs(dis, df = 4, knots = c(4, 7, 11))2 -0.31341    0.01680 -18.658  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))3 -0.26618    0.03147  -8.459 3.00e-16
## bs(dis, df = 4, knots = c(4, 7, 11))4 -0.39802    0.04647  -8.565  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))5 -0.25681    0.09001  -2.853  0.00451
## bs(dis, df = 4, knots = c(4, 7, 11))6 -0.32926    0.06327  -5.204 2.85e-07
##
## (Intercept)                           ***
## bs(dis, df = 4, knots = c(4, 7, 11))1 ***
## bs(dis, df = 4, knots = c(4, 7, 11))2 ***
## bs(dis, df = 4, knots = c(4, 7, 11))3 ***
## bs(dis, df = 4, knots = c(4, 7, 11))4 ***
## bs(dis, df = 4, knots = c(4, 7, 11))5 **
```

```
## bs(dis, df = 4, knots = c(4, 7, 11))6 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06185 on 499 degrees of freedom
## Multiple R-squared:  0.7185, Adjusted R-squared:  0.7151
## F-statistic: 212.3 on 6 and 499 DF,  p-value: < 2.2e-16
```

```
sp.pred=predict(sp.9d, list(dis=dis.grid))
plot(nox~dis, data = Boston, col="darkgrey")
lines(dis.grid, sp.pred, col="red", lwd=2)
```



As the plot shows, this 3 knots split result is quiet close to the dataset, excpet dis>10, where data become scarce.

(e) The range of dis is from 1 to 13, then we split in into 4 parts, 3 knots.

```
cv.rss = rep(NA, 20)
for (df in 3:20) {
  fit = lm(nox~bs(dis, df=df), data=Boston)
  cv.rss[df] = sum(fit$residuals^2)
}
cv.rss[3:20]
```

```
##  [1] 1.934107 1.922775 1.840173 1.833966 1.829884 1.816995 1.825653
##  [8] 1.792535 1.796992 1.788999 1.782350 1.781838 1.782798 1.783546
## [15] 1.779789 1.775838 1.774487 1.776727
```

As you can see, train rss decreases continuly till df=14, meaning df=14 produce the best fit.

(f) The range of dis is from 1 to 13, then we split in into 4 parts, 3 knots.

```r
cv.error = rep(NA, 20)
for (i in 3:20) {
  fit = glm(nox~bs(dis, df=i), data=Boston)
  cv.error[i] = cv.glm(Boston, fit, K =10)$delta[1]
}
```

```
## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.0992), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.0992), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.2157), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.2157), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.354, `66.66667%`
## = 4.2474: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.354, `66.66667%`
## = 4.2474: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.4212, `66.66667%`
## = 4.38856666666667: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.4212, `66.66667%`
## = 4.38856666666667: some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1105, `50%` = 3.2721, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1105, `50%` = 3.2721, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1, `50%` = 3.1323,
## `75%` = 5.118: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1, `50%` = 3.1323,
## `75%` = 5.118: some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.92938, `40%` =
## 2.55946, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.92938, `40%` =
## 2.55946, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.93736, `40%` =
## 2.59666, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.93736, `40%` =
## 2.59666, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.86156666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.86156666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.79777142857143, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.79777142857143, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.7936, `28.57143%`
## = 2.16771428571429, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.7936, `28.57143%`
## = 2.16771428571429, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.734325, `25%` =
## 2.0941, : some 'x' values beyond boundary knots may cause ill-conditioned
```

```
## bases

## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.734325, `25%` =
## 2.0941, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.751575, `25%` =
## 2.1084, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.751575, `25%` =
## 2.1084, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.71552222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.71552222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.66286666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.66286666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.62008, `20%` =
## 1.92938, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.62008, `20%` =
## 1.92938, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.6283, `20%` = 1.9512, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.6283, `20%` = 1.9512, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61225454545455, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61225454545455, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61066363636364, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61066363636364, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

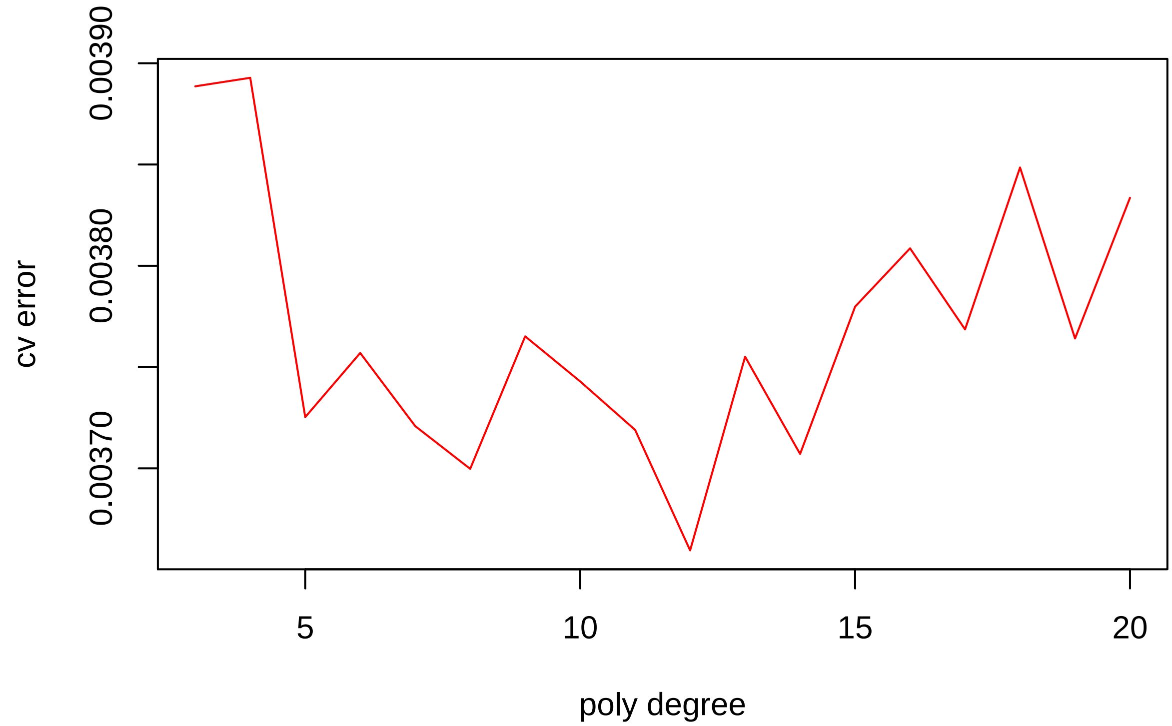
```
## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.60476666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.60476666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.5881, `16.66667%`
## = 1.82231666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.5881, `16.66667%`
## = 1.82231666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.58949230769231, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.58949230769231, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5741, `15.38462%`
## = 1.8209, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5741, `15.38462%`
## = 1.8209, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.54201428571429, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.54201428571429, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.5768, `14.28571%`
## = 1.81652857142857, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.5768, `14.28571%`
## = 1.81652857142857, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.5282, `13.33333%`
## = 1.77806666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.5282, `13.33333%`
## = 1.77806666666667, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.52093333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.52093333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.517275, `12.5%` =
## 1.75675, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.517275, `12.5%` =
## 1.75675, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.526375, `12.5%` =
## 1.754625, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.526375, `12.5%` =
## 1.754625, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.51483529411765, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.51483529411765, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.46587777777778, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.46587777777778, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.46597222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.46597222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
cv.error
```

```
##  [1]          NA          NA 0.003888599 0.003892806 0.003725278
##  [6] 0.003756957 0.003720852 0.003699744 0.003765164 0.003742916
## [11] 0.003718950 0.003659489 0.003755108 0.003707088 0.003779820
## [16] 0.003808616 0.003768584 0.003848543 0.003764094 0.003833573
```

```
?ylim
plot(3:20,cv.error[3:20], xlab = "poly degree", ylab = "cv error", type = "l", col="red", pch=20)
```



Test

MSE is minimum for 10 degrees of freedom.

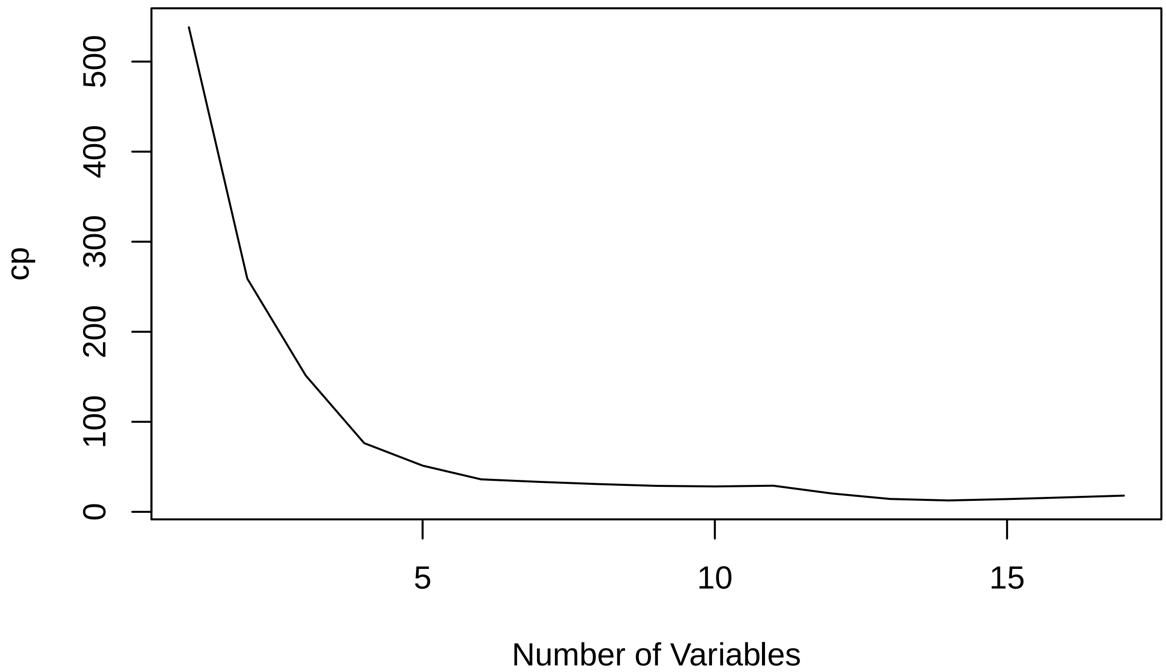
Problem 4–Exercise 10

(a)

```
set.seed(1)
library(leaps)
library(ISLR)
attach(College)
train= sample(length(Outstate), length(Outstate)/2)
test=-train
College.train=College[train, ]
College.test=College[test, ]
reg.fit = regsubsets(Outstate ~ ., data = College.train, nvmax = 17, method = "forward")
reg.summary = summary(reg.fit)
which.min(reg.summary$cp)
```

```
## [1] 14
```

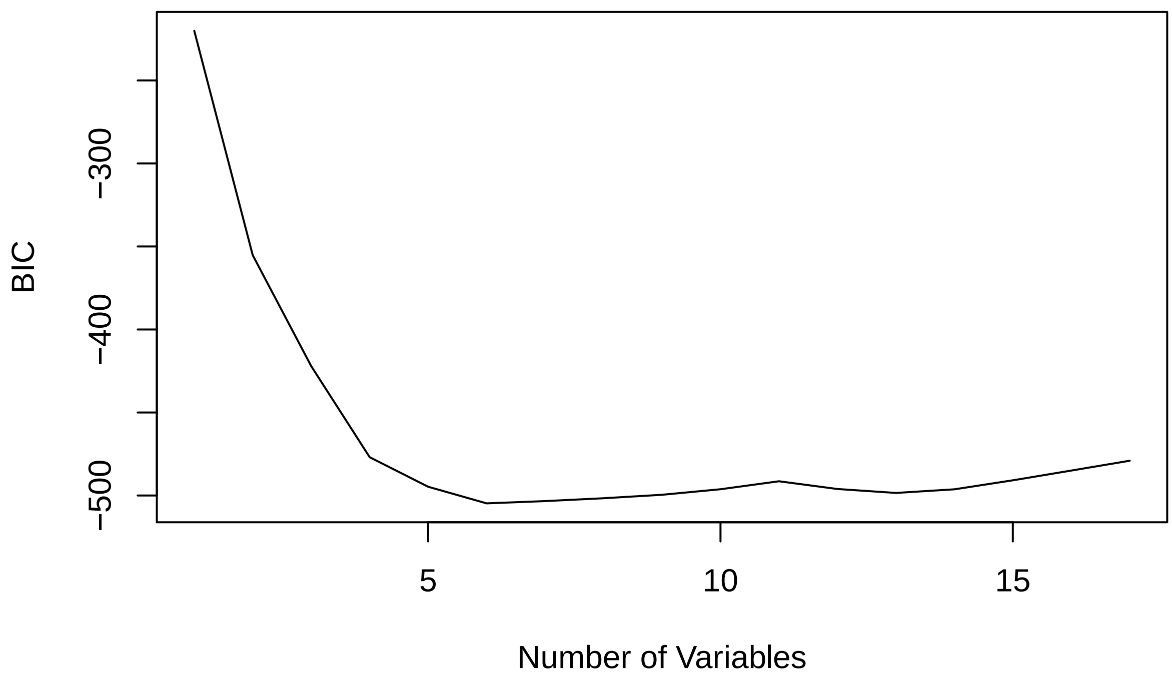
```
plot(reg.summary$cp, xlab = "Number of Variables", ylab = "cp", type = "l")
```




```
which.min(reg.summary$bic)
```

```
## [1] 6
```

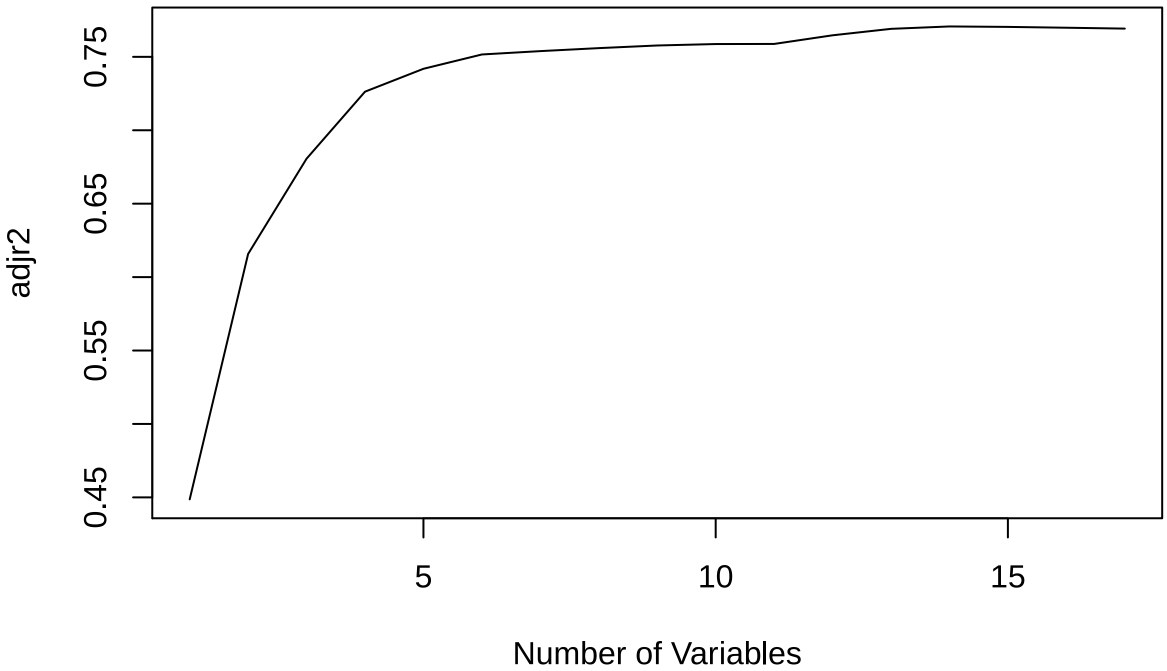
```
plot(reg.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
```




```
which.max(reg.summary$adjr2)
```

```
## [1] 14
```

```r
plot(reg.summary$adjr2, xlab = "Number of Variables", ylab = "adjr2", type = "l")
```



We pick predictors=6 as our minimum size for the subset.

```r
reg.6=regsubsets(Outstate~. , data = College, method = "forward")
reg.6coef= coef(reg.6, id=6)
names(reg.6coef)
```

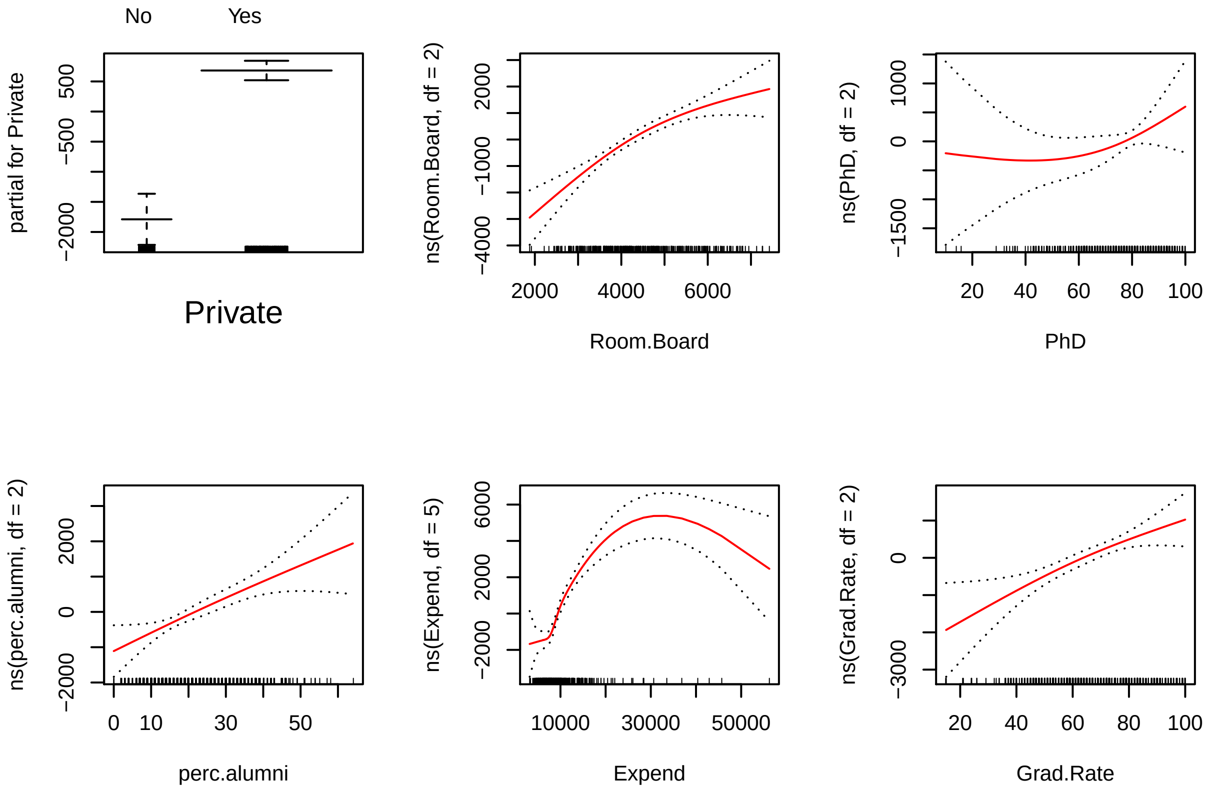
```
## [1] "(Intercept)" "PrivateYes"  "Room.Board"  "PhD"         "perc.alumni"
## [6] "Expend"      "Grad.Rate"
```

(b)

```r
library(gam)
gam.6=gam(Outstate~ Private +ns(Room.Board, df=2) +ns(PhD, df=2) +ns(perc.alumni, df=2) +ns(Expend, df=5
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```r
par(mfrow = c(2,3))
plot(gam.6, se=T, col="red")
```



(c)

```r
gam.pred = predict(gam.6, College.test)
gem.err=mean((College.test$Outstate - gam.pred)^2)
gam.tss=mean((College.test$Outstate - mean(College.test$Outstate))^2)
test.rss= 1-gem.err/gam.tss
test.rss
```

```
## [1] 0.7613443
```

the test R-squared is 0.76, which is quiet good. The model can account for 76% variation in the test data set.

(d)

```r
summary(gam.6)
```

```
##
## Call: gam(formula = Outstate ~ Private + ns(Room.Board, df = 2) + ns(PhD,
##     df = 2) + ns(perc.alumni, df = 2) + ns(Expend, df = 5) +
##     ns(Grad.Rate, df = 2), data = College.train)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -7151.7 -1109.0   -36.1  1305.9  7654.9
##
## (Dispersion Parameter for gaussian family taken to be 3826578)
```

```
##
##     Null Deviance: 6989966760 on 387 degrees of freedom
## Residual Deviance: 1427313633 on 373 degrees of freedom
## AIC: 6998.901
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                            Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                     1 2022785948 2022785948 528.615 < 2.2e-16 ***
## ns(Room.Board, df = 2)      2 2003024353 1001512176 261.725 < 2.2e-16 ***
## ns(PhD, df = 2)             2  705105726  352552863  92.133 < 2.2e-16 ***
## ns(perc.alumni, df = 2)     2  326167935  163083967  42.619 < 2.2e-16 ***
## ns(Expend, df = 5)          5  426700704   85340141  22.302 < 2.2e-16 ***
## ns(Grad.Rate, df = 2)       2   78868463   39434231  10.305 4.403e-05 ***
## Residuals                 373 1427313633    3826578
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA shows a strong evidence of non-linear relationship between "Outstate" and "Expend"", and a less strong non-linear relationship between"Outstate" and "Grad.Rate"" or "PhD".

Problem 5–Exercise 11

(a)

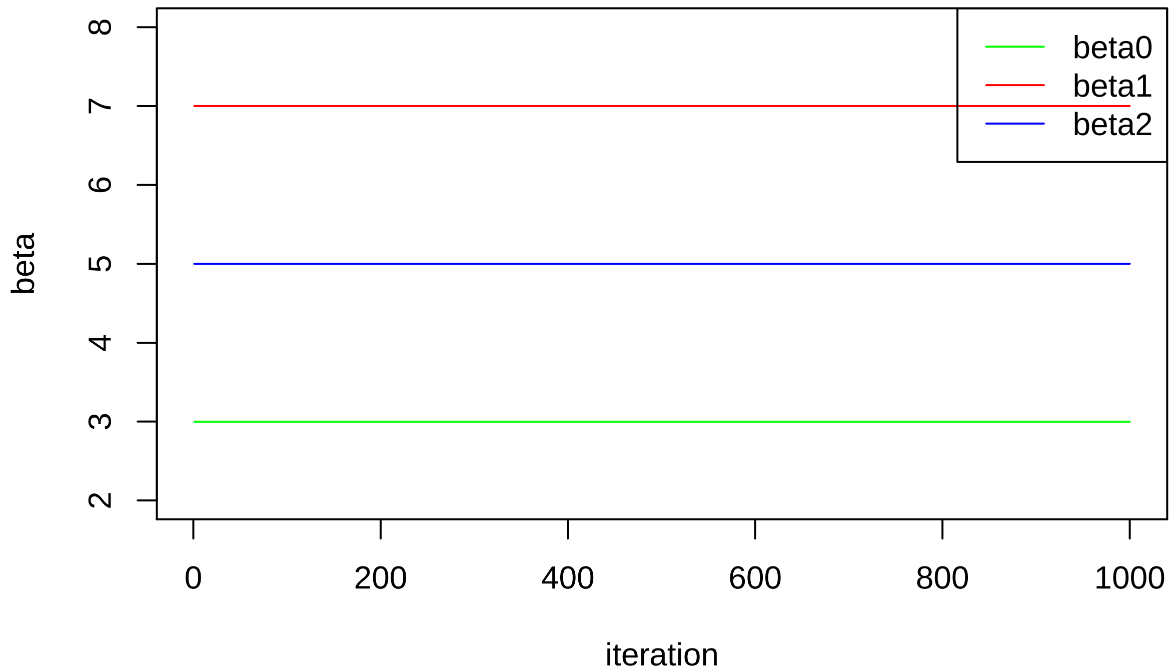
```
set.seed(1)
x1=rnorm(100)
x2=rnorm(100)
noise=rnorm(100, sd=0.03)
y = 7+ 5*x1 +3*x2 + noise
```

(b)

```
beta0= rep(NA,1000)
beta1= rep(NA,1000)
beta2= rep(NA,1000)
beta1[1]=5
```

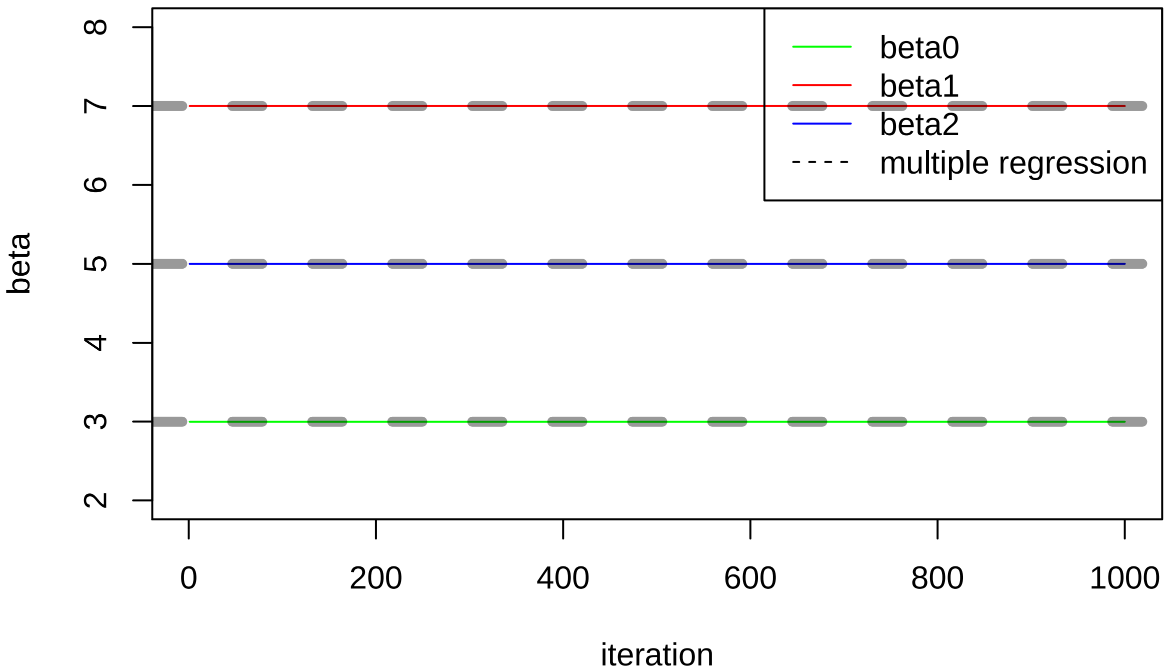
(c,d,e)

```
for(i in 1:1000){
  a=y-beta1[i]*x1
  beta2[i]=lm(a ~ x2)$coef[2]
  a=y-beta2[i]*x2
  beta1[i+1]=lm(a ~ x1)$coef[2]
  beta0[i]= lm(a~ x1)$coef[1]
}
plot(1:1000, beta0, type="l", xlab = "iteration", ylab = "beta", ylim = c(2,8), col="red")
lines(1:1000, beta1[1:1000], col="blue")
lines(1:1000, beta2, col="green")
legend("topright", c("beta0", "beta1","beta2"),lty=1, col = c("green", "red","blue"))
```



(f)

```
multi.lm=lm(y~ x1+x2)
plot(1:1000, beta0, type="l", xlab = "iteration", ylab = "beta", ylim = c(2,8), col="red")
lines(1:1000, beta1[1:1000], col="blue")
lines(1:1000, beta2, col="green")
?abline
abline(h=multi.lm$coef[1], lty="dashed", lwd=5, col = rgb(0, 0, 0, alpha = 0.4))
abline(h=multi.lm$coef[2], lty="dashed", lwd=5, col = rgb(0, 0, 0, alpha = 0.4))
abline(h=multi.lm$coef[3], lty="dashed", lwd=5, col = rgb(0, 0, 0, alpha = 0.4))
legend("topright", c("beta0", "beta1","beta2","multiple regression"), lty=c(1,1,1,2), col = c("green",
```



hmm, I think they are so close to each other, multiregression and backfitting.

(g) it seems like because the relationship between y and x is quiet simple, therefore, it can obtain a good approximation at first few tries.