# STK2100: Solutions Week 7

Lars H. B. Olsen

24.02.2021

## Textbook

### Exercise 4.7

Consider a non-parametric model $Y_i = f(x_{i1}, x_{i2}, \ldots, x_{ip}) + \epsilon_i$, where $\mathrm{E}[\epsilon_i] = 0$ and $\mathrm{var}(\epsilon_i) = \sigma^2$, for $i = 1, 2, \ldots, n$. Assume that all error terms $\epsilon_i$ are independent of each other. This means that $\mathrm{var}(Y_i) = \sigma^2$, which is used in the computations below, indicated by *ind* above the equality sign. Consider the linear smoother $\hat{Y} = SY$, where $S$ is a $n \times n$ matrix, $s_i$ is the $i$th row of $S$, and $Y = (Y_1, Y_2, \ldots, Y_n)^T$. We can then express $\mathrm{cov}(\hat{Y}_i, Y_i)$ as

$$\begin{aligned}
\mathrm{cov}(\hat{Y}_i, Y_i) &= \mathrm{cov}(s_i Y, Y_i) \\
&= \sum_{j=1}^{n} \mathrm{cov}(s_{ij} Y_j, Y_i) \\
&= \sum_{j=1}^{n} s_{ij} \, \mathrm{cov}(Y_j, Y_i) \\
&\overset{ind}{=} s_{ii} \, \mathrm{cov}(Y_i, Y_i) \\
&= s_{ii} \, \mathrm{var}(Y_i) \\
&= s_{ii} \sigma^2.
\end{aligned}$$

Futhermore, this means that

$$\sum_{i=1}^{n} \mathrm{cov}(\hat{Y}_i, Y_i) = \sum_{i=1}^{n} s_{ii} \sigma^2 = \mathrm{tr}(S) \sigma^2,$$

which was what we were asked to prove.

### Exercise 4.8

We are asked to show that for the tree growth algorithm, see section 4.8.2 on page 99, we have that $D_j - D_j^* > 0$, aparat from a degenerate case. Let $R_j$ be the region of interest in the current iteration of the tree growth algorithm. That is, the region $R_j$ will be divided into two subregions denoted by $R_j'$ and $R_j''$.

From (4.16) on page 102, we have that $D_j = \sum_{i \in R_j} (y_i - \hat{c}_j)^2$, where $\hat{c}_j$ is the average value of the response values in ragion $R_j$. That is, $\hat{c}_j = \mathrm{argmin}_c \sum_{i \in R_j} (y_i - c)^2$. Thus any other value than $\hat{c}_j$ will increase $D_j$. The same is true for the two subregions $R_j'$ and $R_j''$ and their corresponing averages $\hat{c}_j'$ and $\hat{c}_j''$, respectively.

We then look at $D_j^*$, see page 103, and we get that

$$\begin{aligned} D_j^* &= \sum_{i \in R_j'} (y_i - \hat{c}_j')^2 + \sum_{i \in R_j''} (y_i - \hat{c}_j'')^2 \\ &< \sum_{i \in R_j'} (y_i - \hat{c}_j)^2 + \sum_{i \in R_j''} (y_i - \hat{c}_j)^2 \\ &= \sum_{i \in R_j} (y_i - \hat{c}_j)^2 \\ &= D_j. \end{aligned}$$

If we subtract $D_j^*$ from both sides, we get that $D_j - D_j^* > 0$, which was what we were asked to show. This will not be the case in degenerate settings, such as, if $R_j$ only contains one response, or if $R_j$ contains several responses but with identical response values. It the latter case, we would then have that $\hat{c}_j = \hat{c}_j' = \hat{c}_j''$ and thereby no gain.

# ISL book

## Exercise 7.1

**This exercise is related to *Exercise 4.3* from last week.**

It was mentioned in the chapter that a cubic regression spline with one knot at $\xi$ can be obtained using a basis of the form $x$; $x^2$, $x^3$, $(x-\xi)_+^3$, where $(x-\xi)_+^3 = (x-\xi)^3$ if $x > \xi$ and equals 0 otherwise. We will now show that a function of the form

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-\xi)_+^3$$

is indeed a cubic regression spline, regardless of the values of $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$.

### a)

Find a cubic polynomial

$$f_1(x) = a_1 + b_1 x + c_1 x^2 + d_1 x^3$$

such that $f(x) = f_1(x)$ for all $x \leq \xi$. Express $a_1, b_1, c_1, d_1$ in terms of $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$.

*For $x \leq \xi$, we have*

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3,$$

*so we take $a_1 = \beta_0$, $b_1 = \beta_1$, $c_1 = \beta_2$ and $d_1 = \beta_3$.*

### b)

Find a cubic polynomial

$$f_2(x) = a_2 + b_2 x + c_2 x^2 + d_2 x^3$$

such that $f(x) = f_2(x)$ for all $x > \xi$. Express $a_2, b_2, c_2, d_2$ in terms of $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$. We have now established that $f(x)$ is a piecewie polynomial.

*For $x > \xi$, we have*

$$\begin{aligned} f(x) &= \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-\xi)^3 \\ &= (\beta_0 - \beta_4 \xi^3) + (\beta_1 + 3\xi^2 \beta_4) x + (\beta_2 - 3\beta_4 \xi) x^2 + (\beta_3 + \beta_4) x^3, \end{aligned}$$

*so we take $a_2 = \beta_0 - \beta_4 \xi^3$, $b_2 = \beta_1 + 3\xi^2 \beta_4$, $c_2 = \beta_2 - 3\beta_4 \xi$ and $d_2 = \beta_3 + \beta_4$.*

**c)**

Show that $f_1(\xi) = f_2(\xi)$. That is $f(x)$ is continuous at $\xi$.

*We have immediately that*

$$f_1(\xi) = \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3$$

*and*

$$\begin{aligned} f_2(\xi) &= (\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\xi^2\beta_4)\xi + (\beta_2 - 3\beta_4\xi)\xi^2 + (\beta_3 + \beta_4)\xi^3 \\ &= \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3. \end{aligned}$$

**d)**

Show that $f_1'(\xi) = f_2'(\xi)$. That is $f'(x)$ is continuous at $\xi$.

*We also have immediately that*

$$f_1'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2$$

*and*

$$\begin{aligned} f_2'(\xi) &= \beta_1 + 3\xi^2\beta_4 + 2(\beta_2 - 3\beta_4\xi)\xi + 3(\beta_3 + \beta_4)\xi^2 \\ &= \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2. \end{aligned}$$

**e)**

Show that $f_1''(\xi) = f_2''(\xi)$. That is $f''(x)$ is continuous at $\xi$. Therefore, $f(x)$ is indeed a cubic spline.

*We finally have that*

$$f_1''(\xi) = 2\beta_2 + 6\beta_3\xi$$

*and*

$$\begin{aligned} f_2''(\xi) &= 2(\beta_2 - 3\beta_4\xi) + 6(\beta_3 + \beta_4)\xi \\ &= 2\beta_2 + 6\beta_3\xi. \end{aligned}$$

## Exercise 7.9

This question uses the variables "dis" (the weighted mean of distances to five Boston employment centers) and "nox" (nitrogen oxides concentration in parts per 10 million) from the "Boston" data. We will treat "dis" as the predictor and "nox" as the response.

**a)**

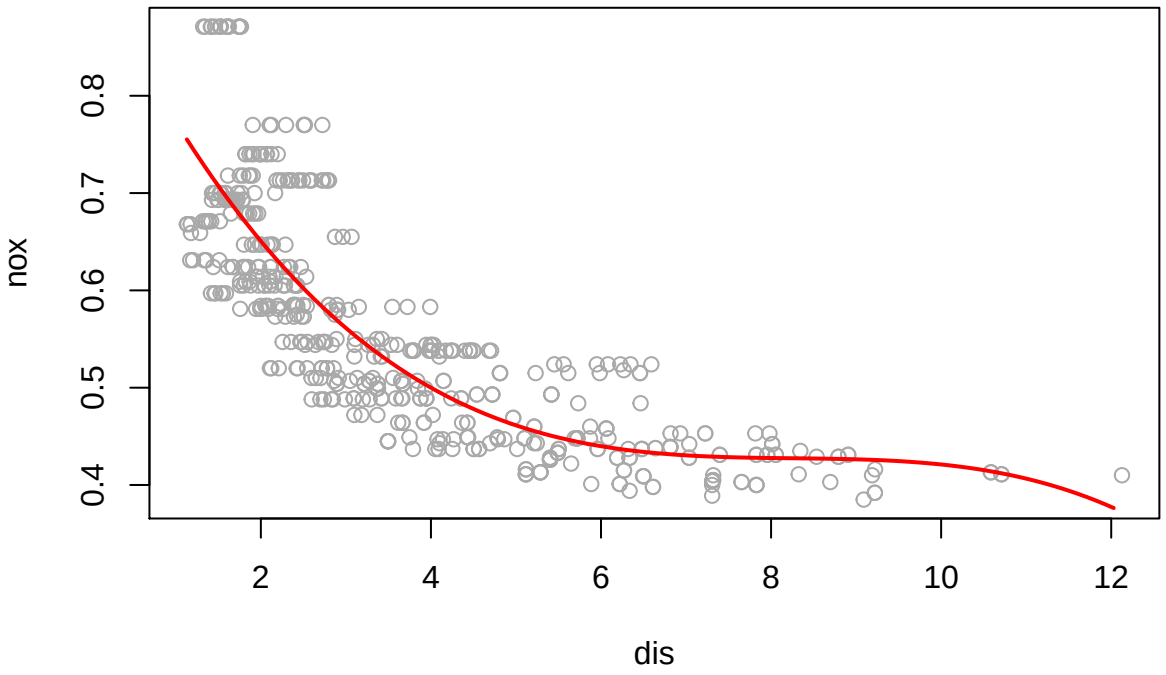
Use the "poly()" function to fit a cubic polynomial regression to predict "nox" using "dis". Report the regression output, and plot the resulting data and polynomial fits.

```
# Include 'MASS' to get the data
library(MASS)

# Set seed for reproducibility
set.seed(1)
```

```r
# Fit the model using cubic polynomial regression
fit = lm(nox ~ poly(dis, 3), data = Boston)

# Look at the model
summary(fit)
```

```
##
## 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
```

```r
# Get the range of the predictor dis
dislims = range(Boston$dis)

# Create a grid for the dis predictor
dis.grid = seq(from = dislims[1], to = dislims[2], by = 0.1)

# Compute the predicted values at the grid points
preds = predict(fit, list(dis = dis.grid))

# Plot the data
plot(nox ~ dis, data = Boston, col = "darkgrey")

# Add the cubic polynomial. Follows the data quite nice
lines(dis.grid, preds, col = "red", lwd = 2)
```



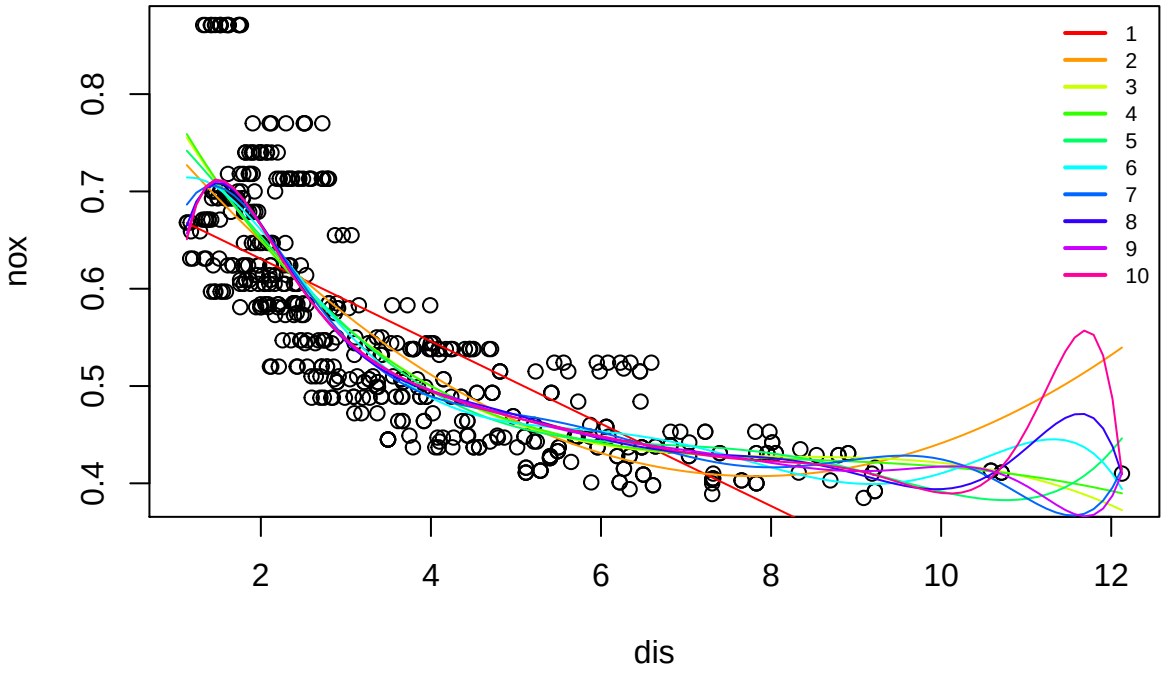

*We may conclude that all polynomial terms are significant.*

**b)**

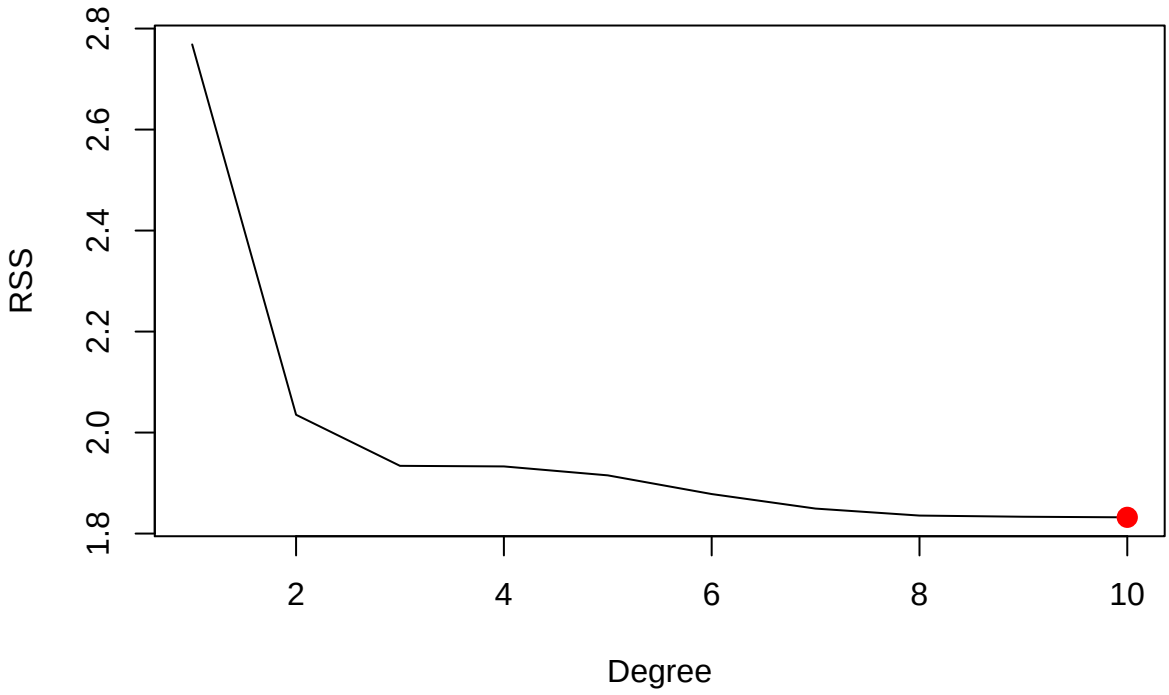
Plot the polynomial fits for a range of different polynomial degrees (say, from 1 to 10), and report the associated residual sum of squares.

```
# Create some new values for which we will evaluate the models
x=seq(dislims[1], dislims[2],length.out = 100)

# Get 10 different colors
clrs=rainbow(10)

{
  # Plot the data
  plot(Boston[,c('dis','nox')])

  # Array to store the RSS
  rss = rep(NA, 10)

  # Iterate over the different degrees
  for (i in 1:10) {
      # Fit the models
      fit = lm(nox ~ poly(dis, i), data = Boston)

      # Get the predicted curves
      pred = predict(fit,data.frame(dis=x))

      # Plot the curves
      lines(x,pred,col=clrs[i])

      # Store the RSS
      rss[i] = sum(fit$residuals^2)
```

```
  }
  # Add a legend
  legend(x='topright',legend = 1:10, col=clrs, lty = c(1,1),lwd = c(2,2),
         bty = "n", cex = 0.7)
}
```




```
# Plot the RSS
plot(1:10, rss, xlab = "Degree", ylab = "RSS", type = "l")
points(which.min(rss), rss[which.min(rss)],col='red', pch=20, cex=2)
```



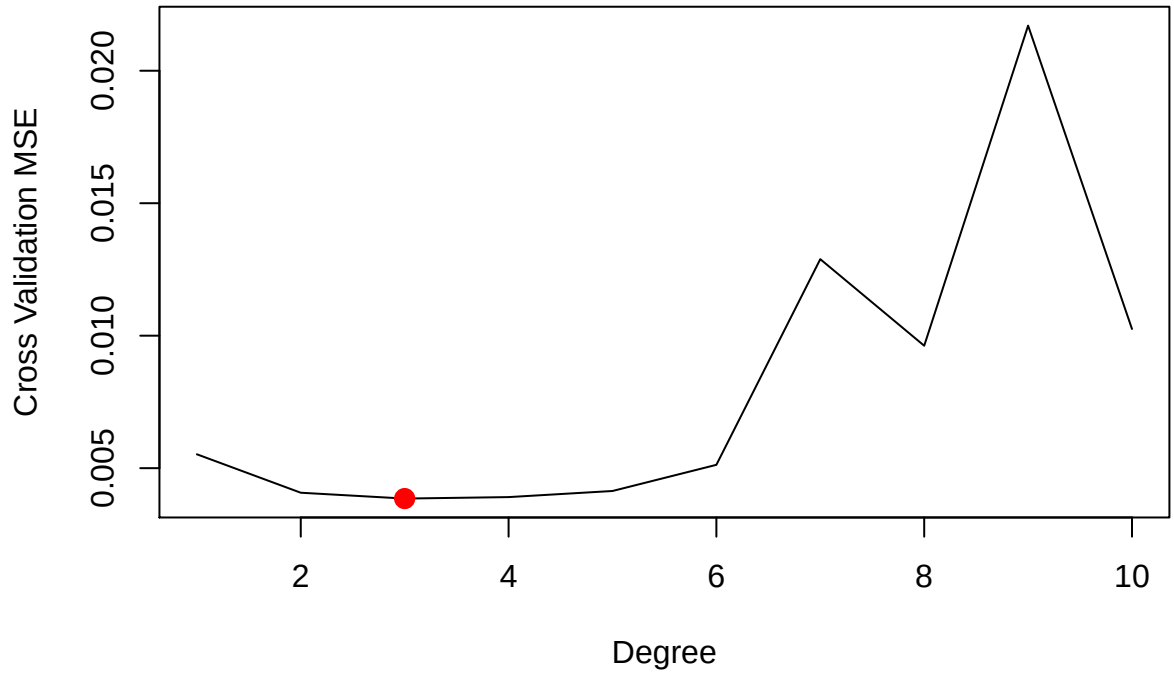

*It seems that the RSS decreases with the degree of the polynomial, and so it is minimum for a polynomial of*

*degree 10.*

**c)**

Perform cross-validation or another approach to select the optimal degree for the polynomial, and explain your results.

```r
# Include the 'boot' library foor the 'cv.glm' function
library(boot)

# Set seed for reproducibility
set.seed(532)

# Array to store the cross validation MSE
deltas = rep(NA, 10)

# Iterate over the different degrees
for (i in 1:10) {
    # Fit the model
    fit = glm(nox ~ poly(dis, i), data = Boston)

    # Conduct 10-fold cross validation and extract the cv MSE.
    deltas[i] = cv.glm(Boston, fit, K = 10)$delta[1]
}

# Plot the cv MSE
plot(1:10, deltas, xlab = "Degree", ylab = "Cross Validation MSE", type = "l")
points(which.min(deltas),deltas[which.min(deltas)],col='red',pch=20,cex=2)
```




*We may see that a polynomial of degree 3 minimizes the cv MSE.*

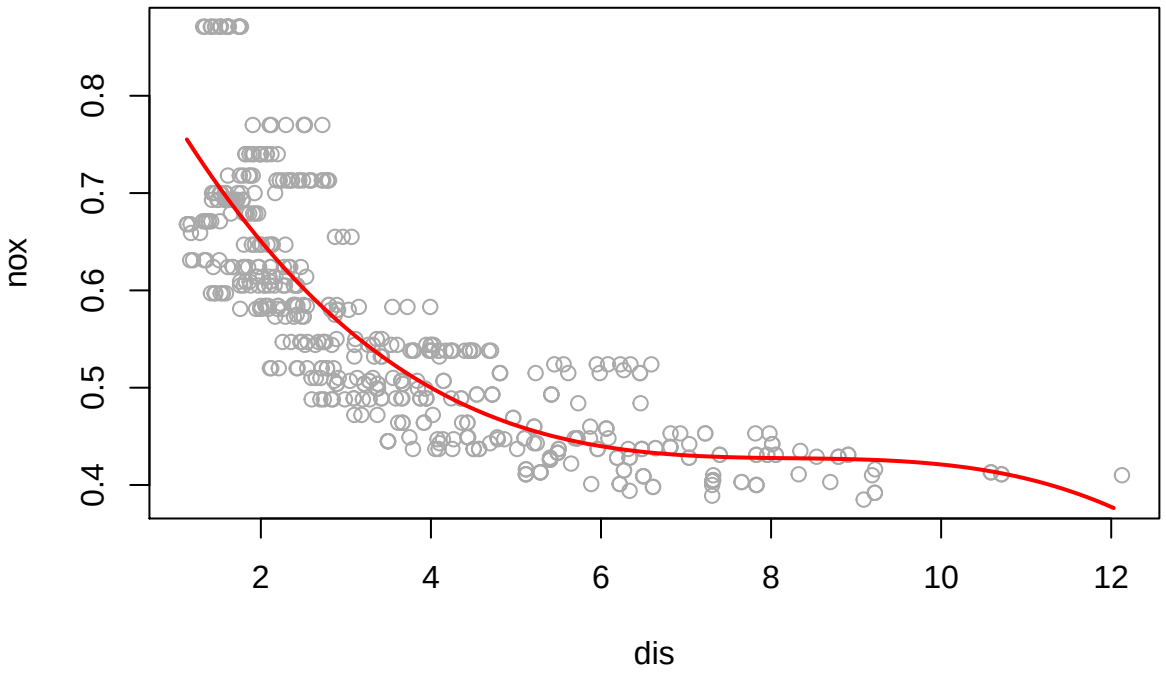
**d)**

Use the `bs()` function to fit a regression spline to predict "nox" using "dis". Report the output for the fit using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

```
# Include the 'splines' library with the 'bs' function
library(splines)

# Fit the spline model. Define knots at 4, 7, and 11.
fit = lm(nox ~ bs(dis, knots = c(4, 7, 11)), data = Boston)

# Look at the model
summary(fit)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, 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, knots = c(4, 7, 11))1 -0.08861    0.02504  -3.539  0.00044 ***
## bs(dis, knots = c(4, 7, 11))2 -0.31341    0.01680 -18.658  < 2e-16 ***
## bs(dis, knots = c(4, 7, 11))3 -0.26618    0.03147  -8.459 3.00e-16 ***
## bs(dis, knots = c(4, 7, 11))4 -0.39802    0.04647  -8.565  < 2e-16 ***
## bs(dis, knots = c(4, 7, 11))5 -0.25681    0.09001  -2.853  0.00451 **
## bs(dis, knots = c(4, 7, 11))6 -0.32926    0.06327  -5.204 2.85e-07 ***
## ---
## 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
```

```
# Compute the predictions
pred = predict(fit, list(dis = dis.grid))

# Plot the data and the predicted spline. Quite good fit
plot(nox ~ dis, data = Boston, col = "darkgrey")
lines(dis.grid, preds, col = "red", lwd = 2)
```



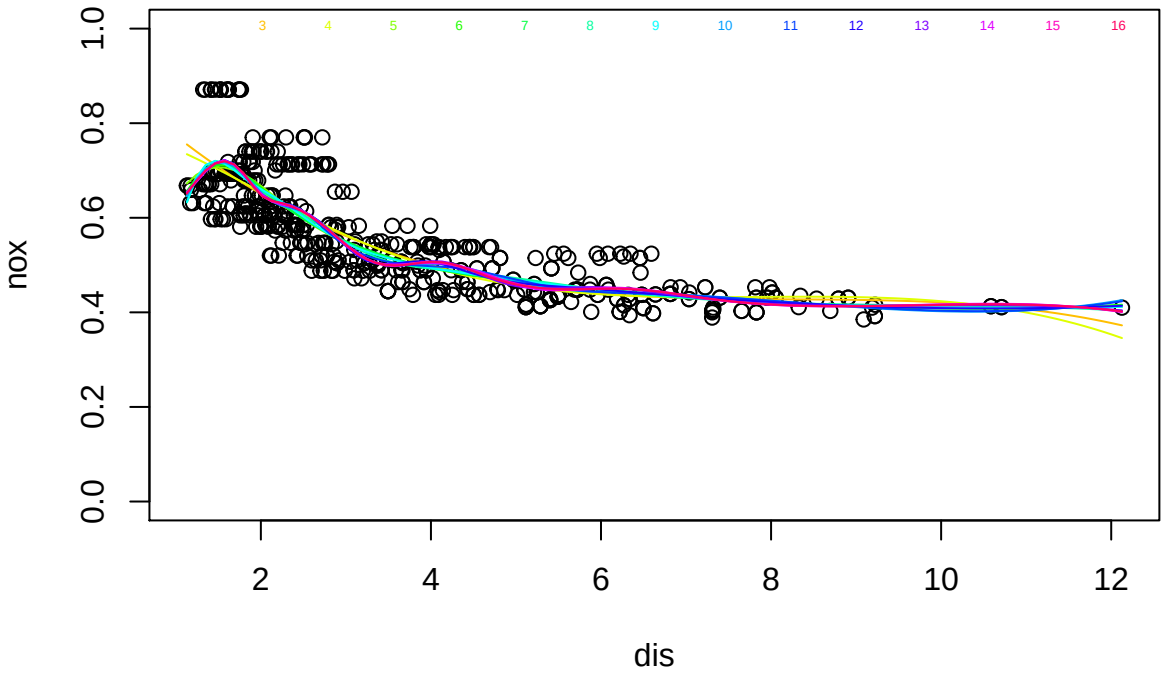

*We may conclude that all terms in spline fit are significant.*

**e)**

Now fit a regression spline for a range of degrees of freedom, and plot the resulting fits and report the resulting RSS. Describe the results obtained.

```
# Array to store the RSS
rss = rep(NA, 16)

# Get colors
clrs=rainbow(16)

# Create values at which we plot the fitted model
x=seq(min(Boston[,'dis']), max(Boston[,'dis']), length.out = 100)
{
  # Plot the data
  plot(Boston[,c('dis','nox')], ylim=c(0,1))

  # Iterate over the degrees of freedom
  for (i in 3:16) {
      # Fit the splines
      fit = lm(nox ~ bs(dis, df = i), data = Boston)

      # Get the predictions
      y=predict(fit,data.frame(dis=x))

      # Plot the predictions
      lines(x,y,col=clrs[i])

      # Store the rss
      rss[i] = sum(fit$residuals^2)
  }
```

```
  # Add a legend
  legend(x='topright',legend=3:16,text.col=clrs[3:16],
         text.width=0.5,bty = 'n',horiz = T, cex = 0.4)
}
```



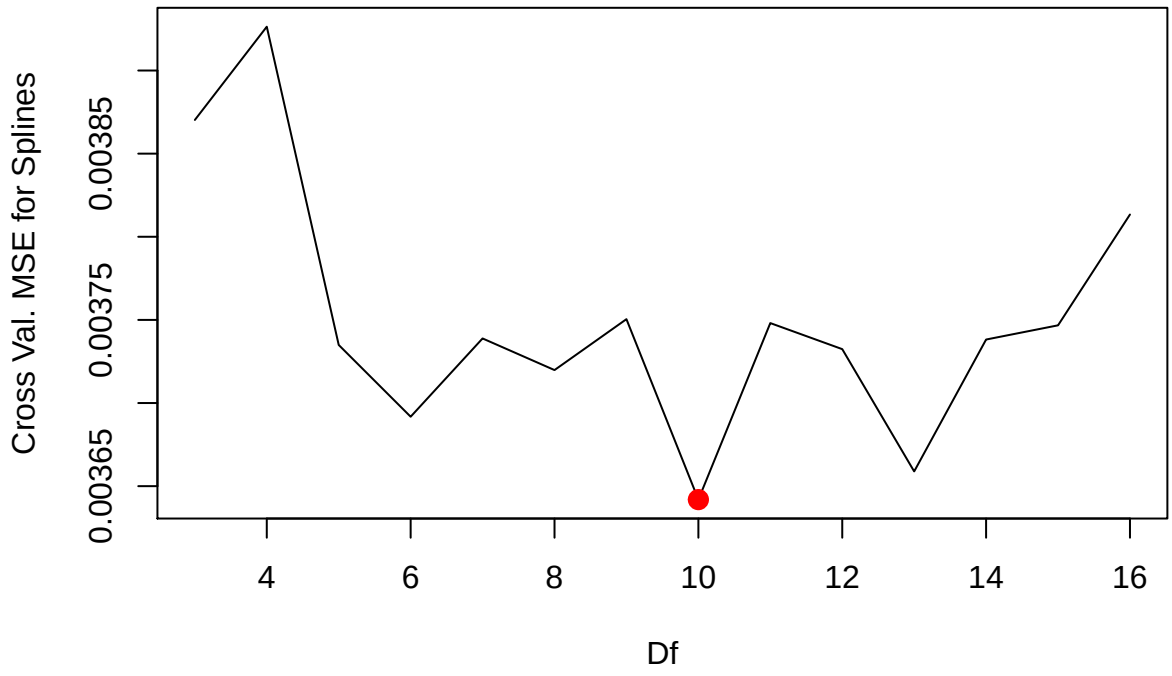

```
# Plot the RSS
plot(3:16, rss[-c(1, 2)], xlab = "Degrees of freedom", ylab = "RSS", type = "l")
points(which.min(rss),rss[which.min(rss)],col='red',pch=20,cex=2)
```


*We may see that RSS decreases until 14 and then slightly increases after that.*

f)

Perform cross-validation or another approach in order to select the best degrees of freedom for a regression spline on this data. Describe your results.

```
# Set seed for reproducibility
set.seed(42)

# Array to store the spline mse
spline.mse=rep(NA, 16)

# Iterate over the different degrees of freedom
for(df in 3:16){
  # Create the model frame
  Boston.model=model.frame(nox~bs(dis,df=df), data=Boston)

  # Need set the names for the methods to work
  names(Boston.model) = c('nox','bs.dis')

  # Fit the splines
  spline.fit=glm(nox~bs.dis,data=Boston.model)

  # Conduct 10-fold cross validation and extract the cv MSE
  spline.mse[df]=cv.glm(spline.fit,data=Boston.model,K=10)$delta[1]
}

# Plot the cv MSE for splines
plot(3:16,spline.mse[-c(1,2)],type='l',xlab='Df',ylab='Cross Val. MSE for Splines')
points(which.min(spline.mse),spline.mse[which.min(spline.mse)],col='red',pch=20,cex=2)
```




*Cross validation MSE is minimum for 10 degrees of freedom.*

## Exercise 7.10

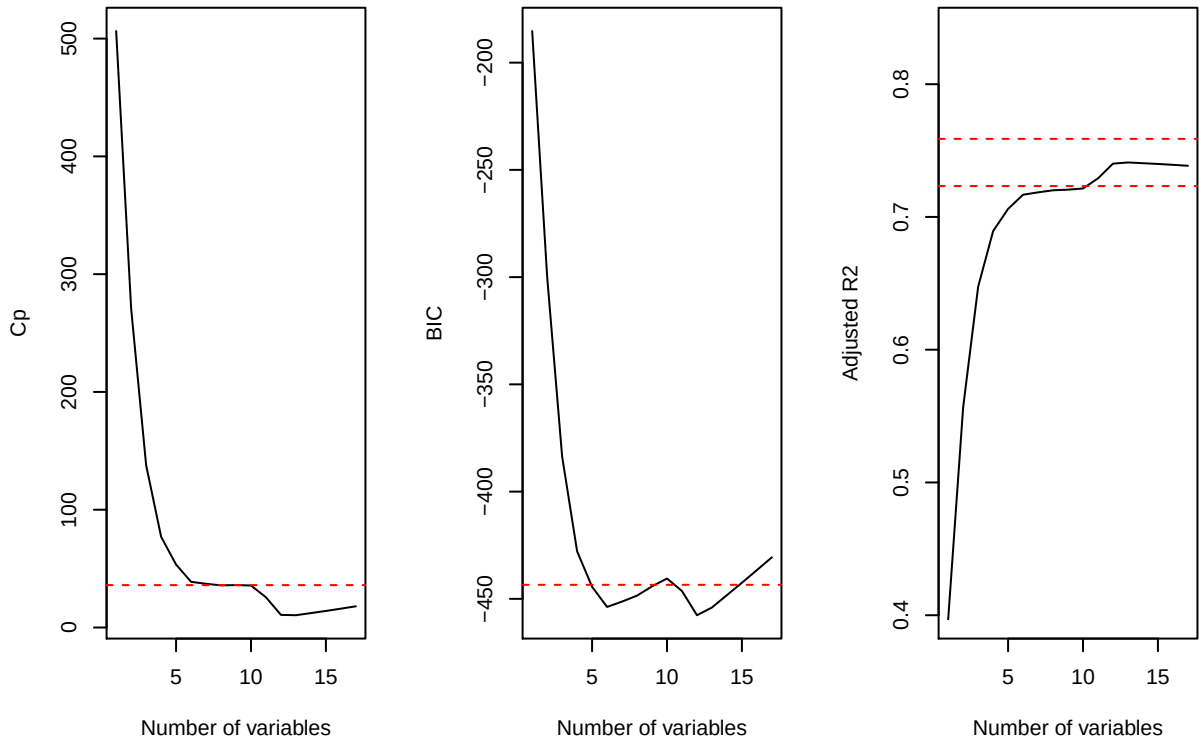
This question relates to the "College" data set.

### a)

Split the data into a training set and a test set. Using out-of-state tuition as the response and the other variables as the predictors, perform forward stepwise selection on the training set in order to identify a satisfactory model that uses just a subset of the predictors.

```
# Import necessary libraries
library(leaps)
library(ISLR)

# attach the college data
attach(College)

# Set seed for reproducibility.
# The results later on depends on the seed
set.seed(2020)

# Create train and test data
train = sample(length(Outstate), length(Outstate) / 2)
test = -train
College.train = College[train, ]
College.test = College[test, ]

# Do forward selection
fit = regsubsets(Outstate ~ ., data = College.train, nvmax = 17, method = "forward")

# Compute summary of the fit
fit.summary = summary(fit)

# Create plot of different selction criteria
par(mfrow = c(1, 3))

# Cp
plot(fit.summary$cp, xlab = "Number of variables",
     ylab = "Cp", type = "l")
min.cp = min(fit.summary$cp)
std.cp = sd(fit.summary$cp)
abline(h = min.cp + 0.2 * std.cp, col = "red", lty = 2)
abline(h = min.cp - 0.2 * std.cp, col = "red", lty = 2)

# BIC
plot(fit.summary$bic, xlab = "Number of variables",
     ylab = "BIC", type='l')
min.bic = min(fit.summary$bic)
std.bic = sd(fit.summary$bic)
abline(h = min.bic + 0.2 * std.bic, col = "red", lty = 2)
abline(h = min.bic - 0.2 * std.bic, col = "red", lty = 2)

# Adjusted R^2
plot(fit.summary$adjr2, xlab = "Number of variables",
     ylab = "Adjusted R2", type = "l", ylim = c(0.4, 0.84))
```

```
max.adjr2 = max(fit.summary$adjr2)
std.adjr2 = sd(fit.summary$adjr2)
abline(h = max.adjr2 + 0.2 * std.adjr2, col = "red", lty = 2)
abline(h = max.adjr2 - 0.2 * std.adjr2, col = "red", lty = 2)
```




*Cp, BIC and adjr2 gives different results. The red lines are 0.2 standard deviations from the optimum. Seems that a model of size 6 is fair compromise between the three evaluation methods. This can be argued. Could also look at training MSE. Then size 7 would win.*

```
# Do forward selection
fit = regsubsets(Outstate ~ ., data = College, method = "forward")

# Select the best model with 6 predictores
coeffs = coef(fit, id = 6)

# Look at which predictors we use, in addition to intercept.
names(coeffs)
```

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

### b)

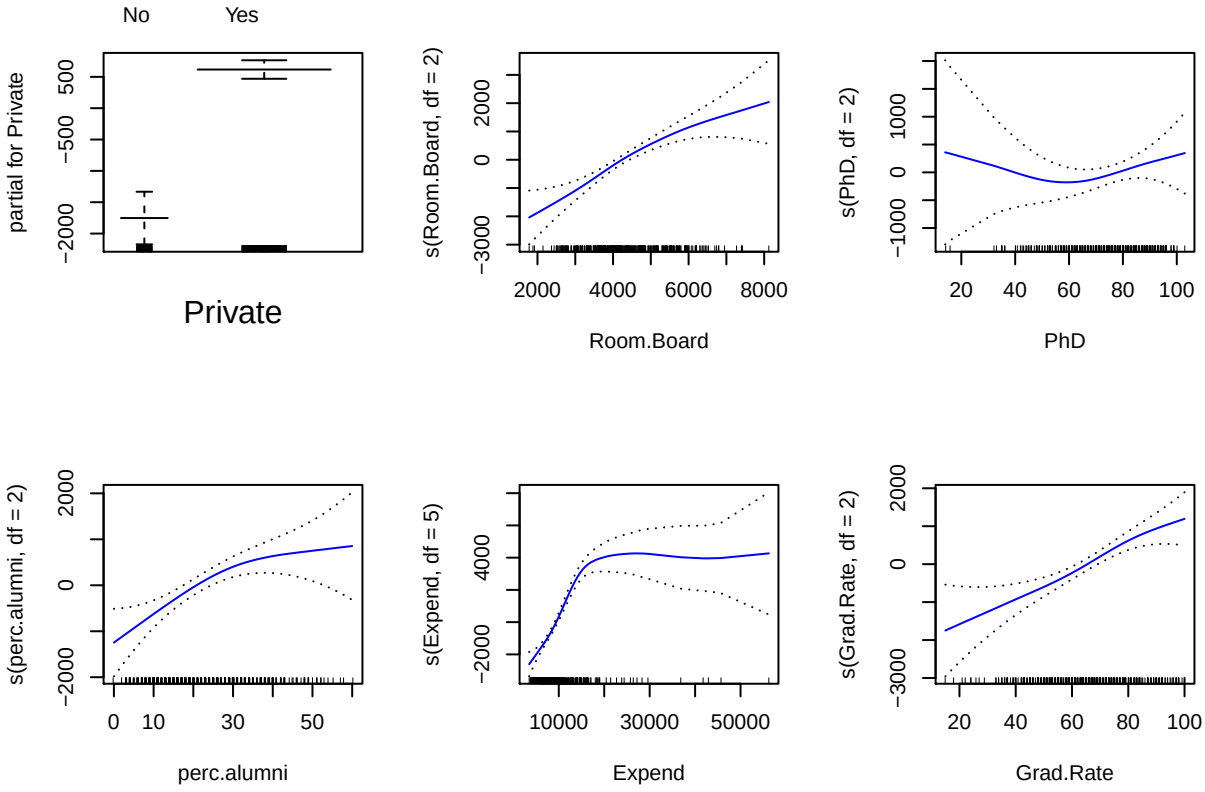
Fit a GAM on the training data, using out-of-state tuition as the response and the features selected in the previous step as the predictors. Plot the results, and explain your findings.

```
# Include the 'gam' library for the 'gam()' function
library(gam)
```

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

```
## Loaded gam 1.20
```

```
# Fit a gam using the predictors above on the training data
fit = gam(Outstate ~ Private + s(Room.Board, df = 2) + s(PhD, df = 2) +
            s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate, df = 2),
          data=College.train)

# Plot the six fitted additve models
par(mfrow = c(2, 3))
plot(fit, se = T, col = "blue")
```




*Grad.Rate, Room.Board, and somewhat PhD seems quite linear, while the other continuous variable are higly non-linear.*

**c)**

Evaluate the model obtained on the test set, and explain the results obtained.

```
# Use the model to predict the test responses
preds = predict(fit, College.test)

# Compute the mean squared test error
err = mean((College.test$Outstate - preds)^2)
err
```

```
## [1] 3171279
```

```
# Compute the test TSS and then RSS
tss = mean((College.test$Outstate - mean(College.test$Outstate))^2)
rss = 1 - err / tss
rss
```

```
## [1] 0.7951092
```

*We obtain a test R^2 of 0.795 using GAM with 6 predictors. That is, about 79.5% of the variance encountered in the data is explained by this model.*

### d)

For which variables, if any, is there evidence of a non-linear relationship with the response?

```
# Use summary to see if there is evidence of a non-linear relationship
summary(fit)
```

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 2) + s(PhD,
##     df = 2) + s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate,
##     df = 2), data = College.train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -7440.52 -1225.07    22.48  1270.47  6919.54
##
## (Dispersion Parameter for gaussian family taken to be 3886232)
##
##     Null Deviance: 6526177771 on 387 degrees of freedom
## Residual Deviance: 1449565520 on 373.0002 degrees of freedom
## AIC: 7004.903
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                          Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                   1 1597314625 1597314625 411.019 < 2.2e-16 ***
## s(Room.Board, df = 2)     1 1141857525 1141857525 293.821 < 2.2e-16 ***
## s(PhD, df = 2)            1  306261193  306261193  78.807 < 2.2e-16 ***
## s(perc.alumni, df = 2)    1  267921177  267921177  68.941  1.89e-15 ***
## s(Expend, df = 5)         1  566110411  566110411 145.671 < 2.2e-16 ***
## s(Grad.Rate, df = 2)      1  103679645  103679645  26.679  3.92e-07 ***
## Residuals               373 1449565520    3886232
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                          Npar Df  Npar F     Pr(F)
## (Intercept)
## Private
## s(Room.Board, df = 2)          1  3.5363   0.06082 .
## s(PhD, df = 2)                 1  2.7823   0.09615 .
## s(perc.alumni, df = 2)         1  3.9421   0.04783 *
## s(Expend, df = 5)              4 20.8798 1.554e-15 ***
## s(Grad.Rate, df = 2)           1  2.0846   0.14963
## ---
## 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 moderately strong non-linear relationship (using p-value of 0.05) between"Outstate" and "perc.alumni". Note that this highly depends on the seed. Furthermore, compare these results with what we saw in subquestion b).*