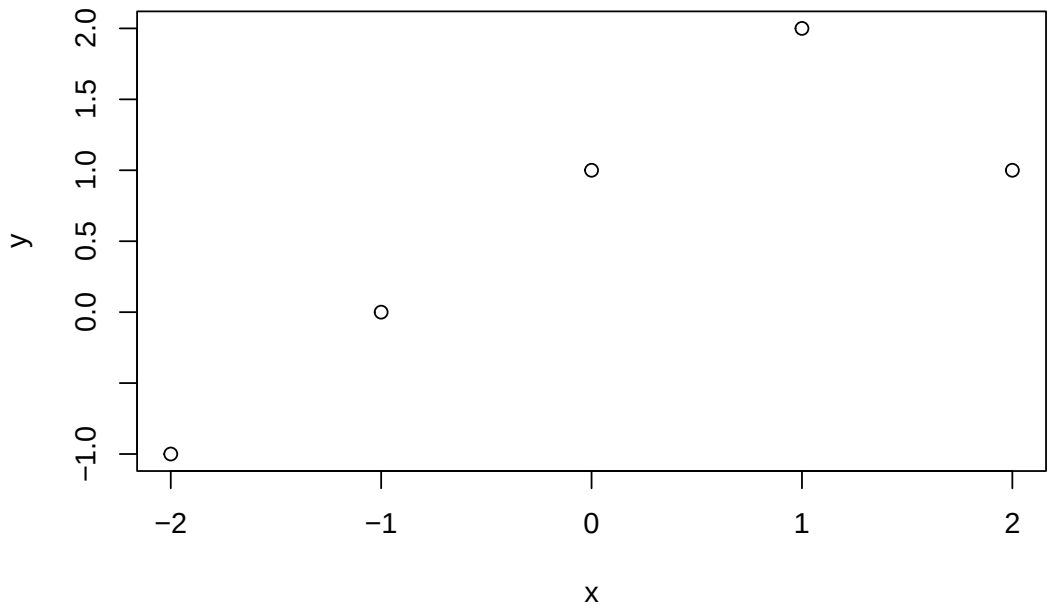
Chapter 7

Tong Sun

2/13/2022

7.3

Suppose we fit a curve with basis functions $b_1(X) = X$, $b_2(X) = (X-1)^2 I(X >= 1)$. (Note that $I(X >= 1)$ equals 1 for X>=1 and 0 otherwise.) We fit the linear regression model$Y = \beta_0 + \beta_1 b_1(X) + \beta_2 b_2(X) + \epsilon$, and obtain coefficient estimates $\hat{\beta}_0 = 1$,$\hat{\beta}_1 = 1$,$\hat{\beta}_2 = -2$. Sketch the estimated curve between X=-2 and X=2. Note the intercepts, slopes and other relevant information.



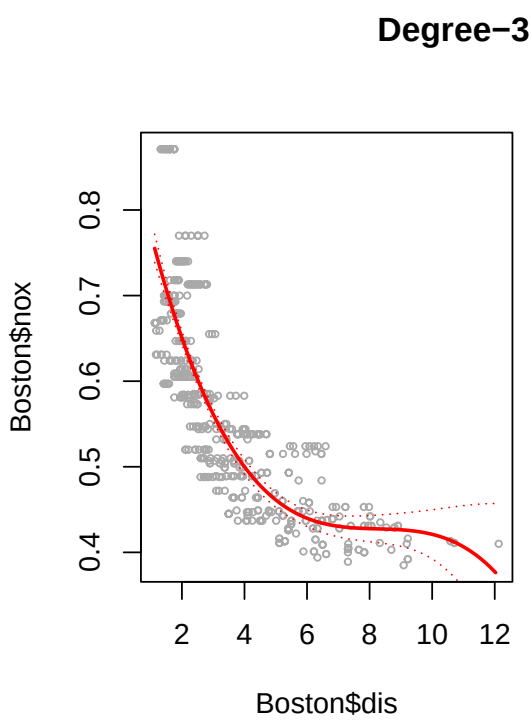
The curve is linear between-2 and 1: $y = 1 + x$ and quadratic between 1 and 2:$y = 1 + x - 2(x-1)^2$.

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.

```
##
## 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
```

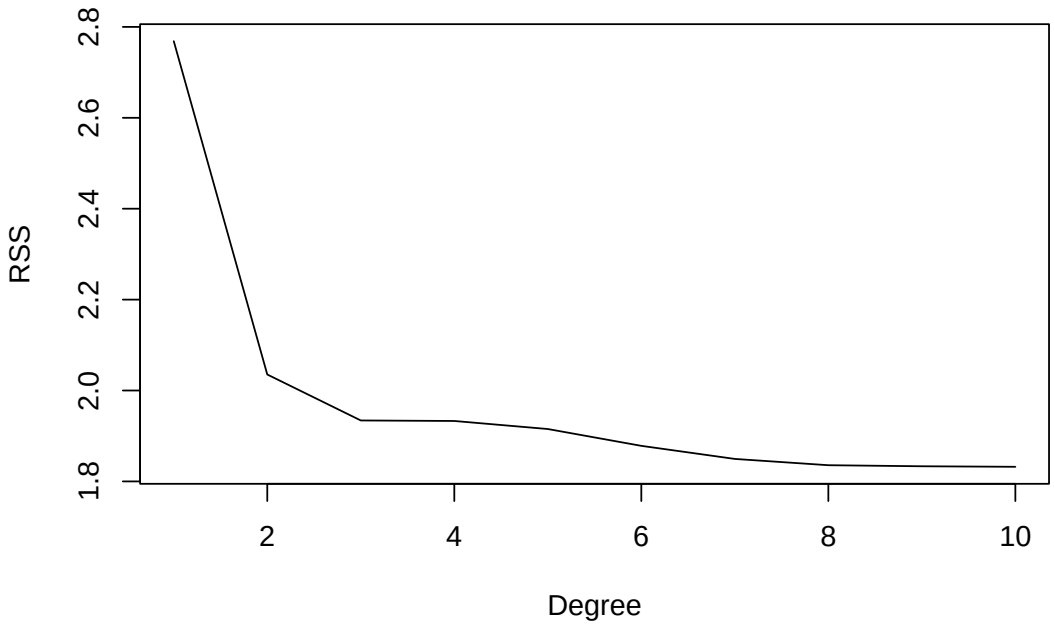
Here the poly() command allows us to avoid having to write out a long formula with powers of dis. The function returns a matrix whose columns are a basis of orthogonal polynomials, which essentially means that each column is a linear combination of the variables dis, dis^2 and dis^3.



Here I create a grid of values for dis at which I want predictions. And then I plot the data and add the fit from the degree-3 polynomial. Here the *mar* and *oma* arguments to *par*() allow me to control the margins

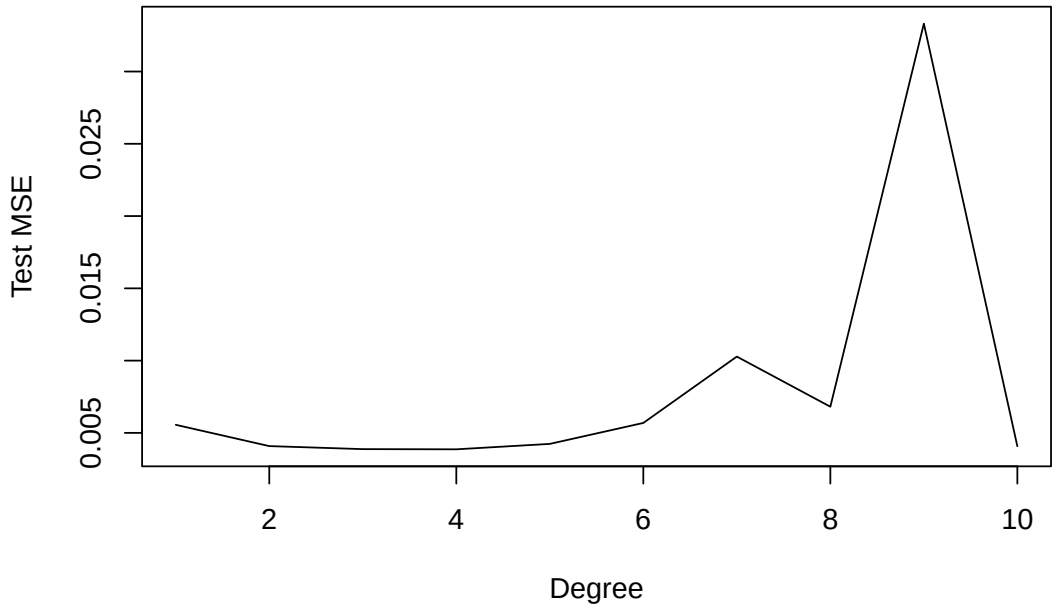
of the plot, and the *title*() function creates a figure title that spans both subplots. We may conclude that all polynomial terms are significant.

##(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.



It seems that the RSS decreases with the degree of the polynomial, and so is minimum for a polynomial of degree 10.

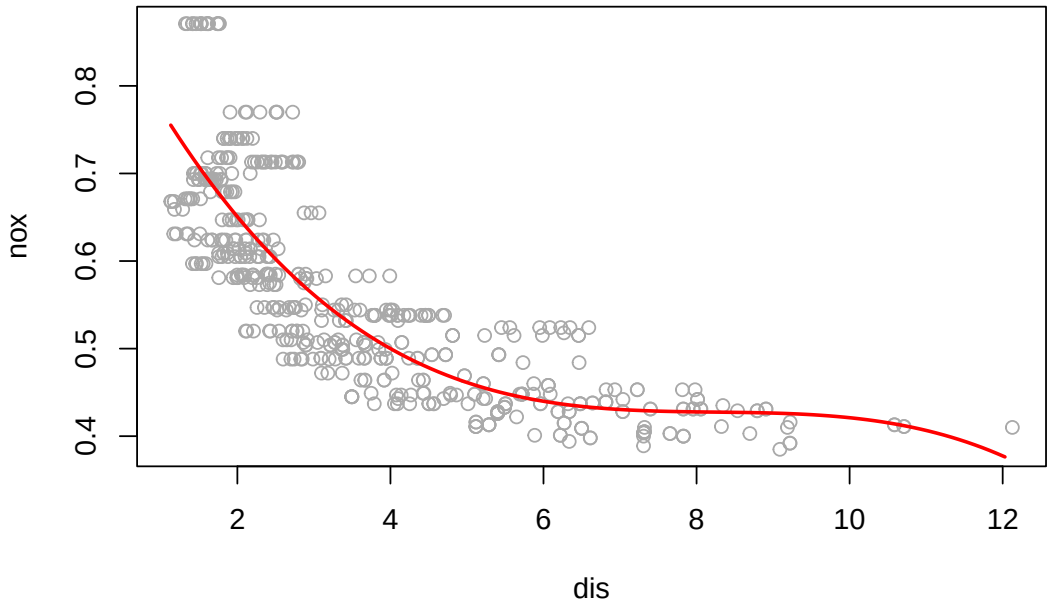
##(c) Perform cross-validation or another approach to select the optimal degree for the polynomial, and explain your results.



We may see that a polynomial of degree 4 minimizes the test MSE.

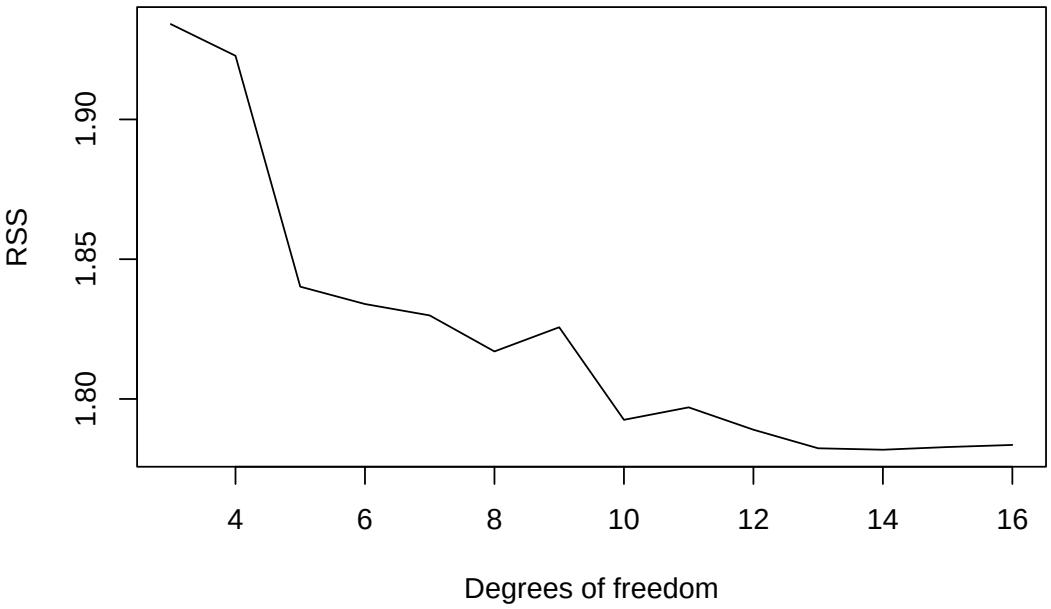
##(d) Use the bs() function to fit a regression spline to predict nox using dis. Report the output for the fit using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

```
##
## 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
```



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.



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

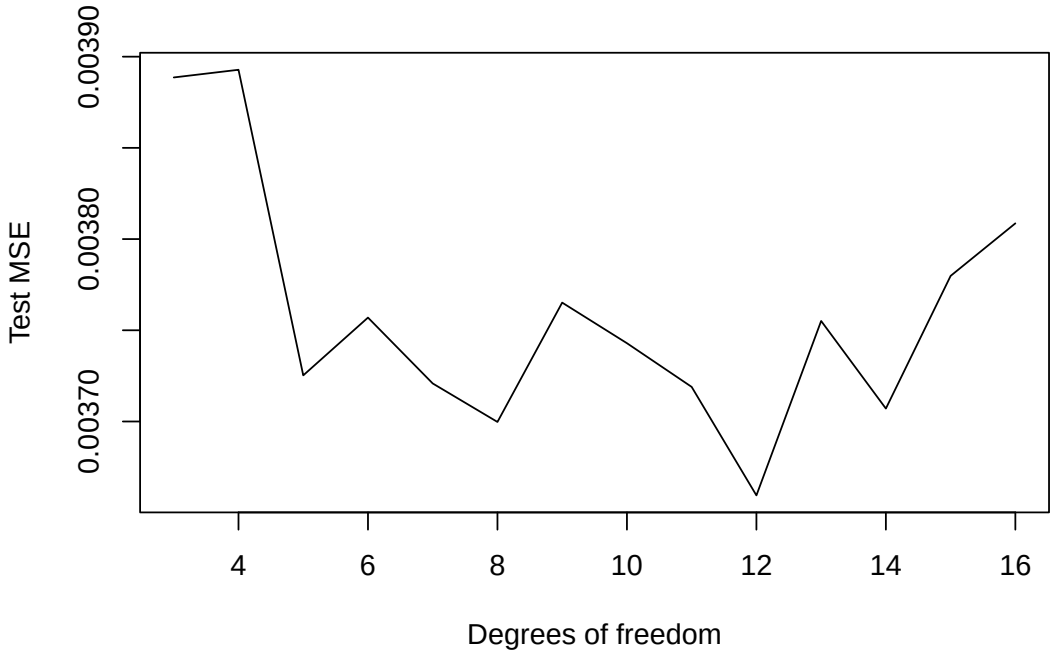
##(f) Perform cross-validation or another approach in order to select the best degrees of freedom for a regression spline on this data.Describe your results.

```
## 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
```

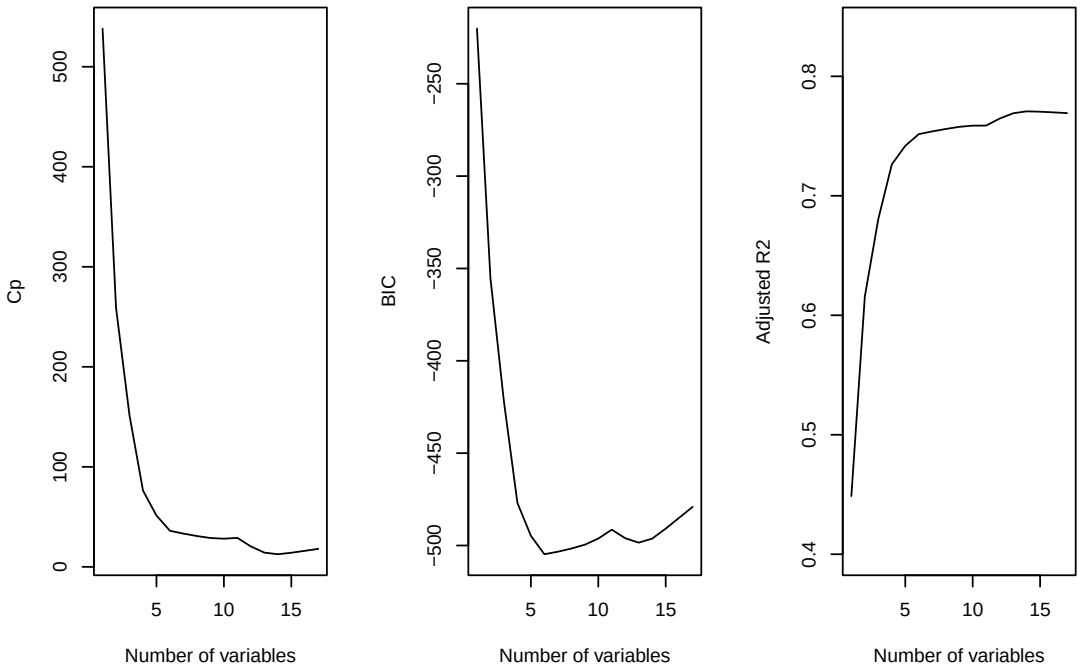
```
## Warning in bs(dis, degree = 3L, knots = c(‘7.692308%‘ = 1.5741, ‘15.38462%‘ =
## 1.8209, : some ’x’ values beyond boundary knots may cause ill-conditioned bases

## 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
```



Test MSE is minimum for 10 degrees of freedom.

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.



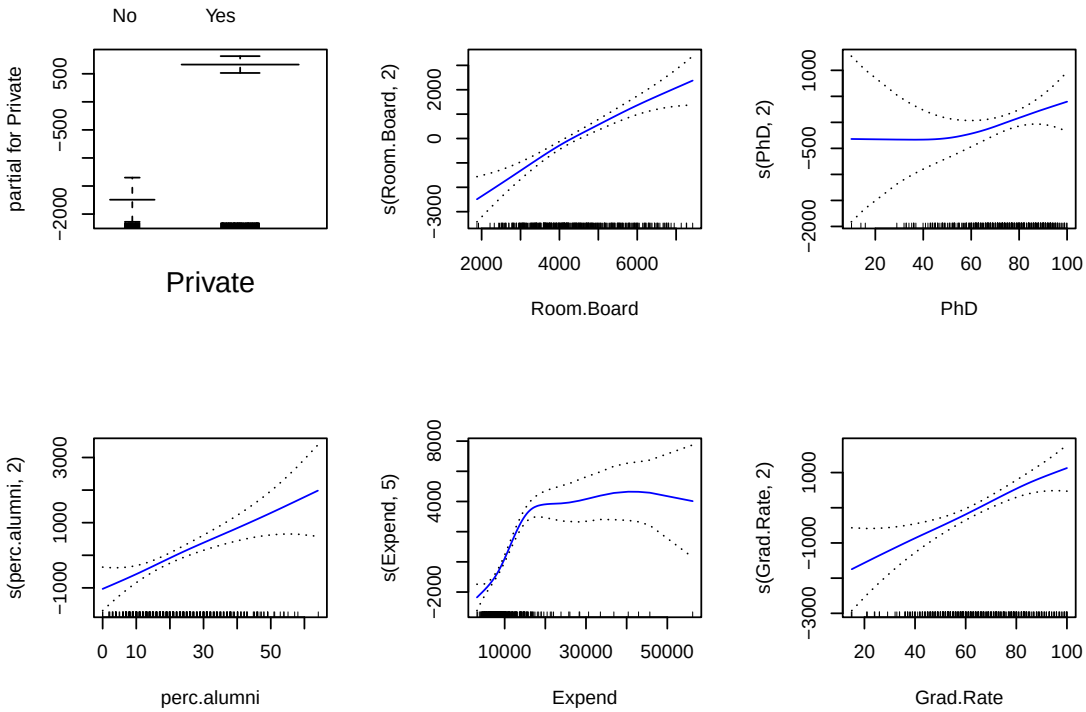
From the plots above, I think size six is the minimum size for the subset.

```
## [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.

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

```
## Loaded gam 1.20
```



I specify that the function of "Room.Board" should have two degrees of freedom, the function of "PhD" should have two degrees of freedom, the function of "perc.alumni" should have two degrees of freedom, the function of "Expend" should have five degrees of freedom and the function of "Grad.Rat" should have five degrees of freedom. Since "Private" is qualitative, I leave it as is and it is converted into dummy variables.For the College data, plots of the relationship between each feature and the response, out of state tuition, in the fitted model. Each plot displays the fitted function and pointwise standard errors.Each panel indicates that holding all the other predictors (without the x-axis one) fixed, the relationship between out-of-state tuition and predictor.For the "Room.Board", "PhD", "perc.alumni", "Expend" and "Grad.Rat" predictors, with holding other five variables fixed, tuition tends to increase with each predictor. And for "Private" variable, holding other variables fixed, those with answer "Yes" tends to pay more tuition than those saying "No".

##(c) Evaluate the model obtained on the test set, and explain the results obtained.

```
## [1] 3349290
```

```
## [1] 0.7660016
```

We obtain a test R^2 of 0.77 using GAM with six predictors.

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

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, 2) + s(PhD,
##     2) + s(perc.alumni, 2) + s(Expend, 5) + s(Grad.Rate, 2),
##     data = College.train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -7402.89 -1114.45   -12.67  1282.69  7470.60
```

```
##
## (Dispersion Parameter for gaussian family taken to be 3711182)
##
##     Null Deviance: 6989966760 on 387 degrees of freedom
## Residual Deviance: 1384271126 on 373 degrees of freedom
## AIC: 6987.021
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                     Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private              1 1778718277 1778718277 479.286 < 2.2e-16 ***
## s(Room.Board, 2)     1 1577115244 1577115244 424.963 < 2.2e-16 ***
## s(PhD, 2)            1  322431195  322431195  86.881 < 2.2e-16 ***
## s(perc.alumni, 2)    1  336869281  336869281  90.771 < 2.2e-16 ***
## s(Expend, 5)         1  530538753  530538753 142.957 < 2.2e-16 ***
## s(Grad.Rate, 2)      1   86504998   86504998  23.309 2.016e-06 ***
## Residuals          373 1384271126    3711182
## ---
## 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, 2)         1  1.9157    0.1672
## s(PhD, 2)                1  0.9699    0.3253
## s(perc.alumni, 2)        1  0.1859    0.6666
## s(Expend, 5)             4 20.5075 2.665e-15 ***
## s(Grad.Rate, 2)          1  0.5702    0.4506
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The "Anova for Parametric Effects" p-values clearly demonstrate that "Private", "Room.Board","PhD", "perc.alumni","Expend" and "Grad.Rate" are all highly statistically significant, even when only assuming a linear relationship. Alternatively, the "Anova for Nonparametric Effects" p-values for predictors correspond to a null hypothesis of a linear relationship versus the alternative of a non-linear relationship. The large p-value for "Room.Board", "PhD", "perc.alumni" and "Grad.Rate" reinforces the conclusion from the Anova test that a linear function is adequate for these terms. However, there is very clear evidence that a non-linear term is required for "Expend".

7.11

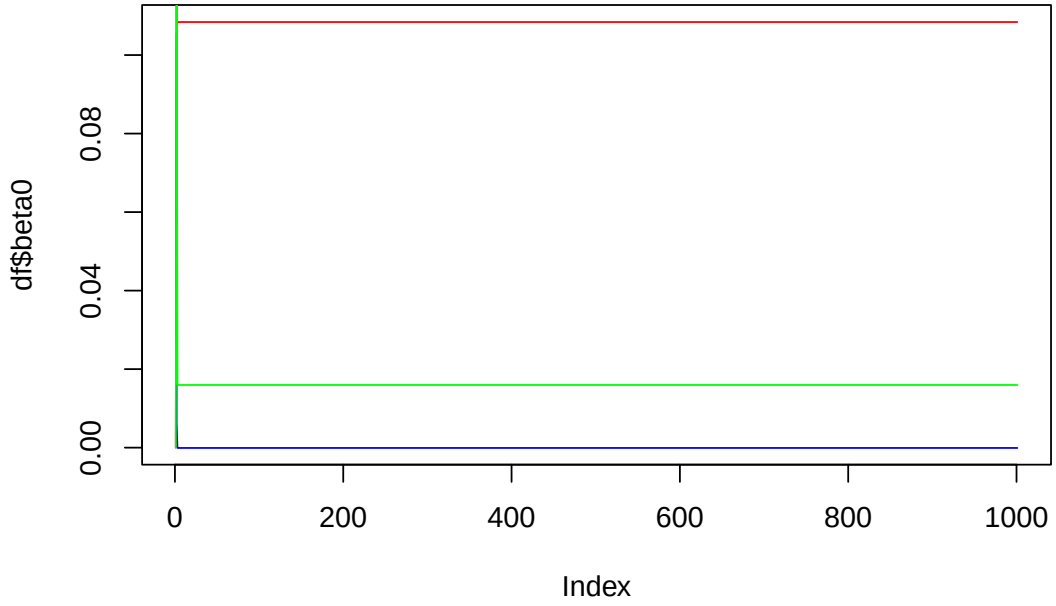
In Section 7.7, it was mentioned that GAMs are generally fit using a backfitting approach. The idea behind backfitting is actually quite simple. We will now explore backfitting in the context of multiple linear regression. Suppose that we would like to perform multiple linear regression, but we do not have software to do so. Instead, we only have software to perform simple linear regression. Therefore, we take the following iterative approach: we repeatedly hold all but one coefficient estimate fixed at its current value, and update only that coefficient estimate using a simple linear regression. The process is continued until convergence—that is, until the coefficient estimates stop changing. We now try this out on a toy example. ##(a) Generate a response Y and two predictors X_1 and X_2, with n = 100.

##(b) Initialize $\hat{\beta}_1$ to take on a value of your choice. It does not matter what value you choose.

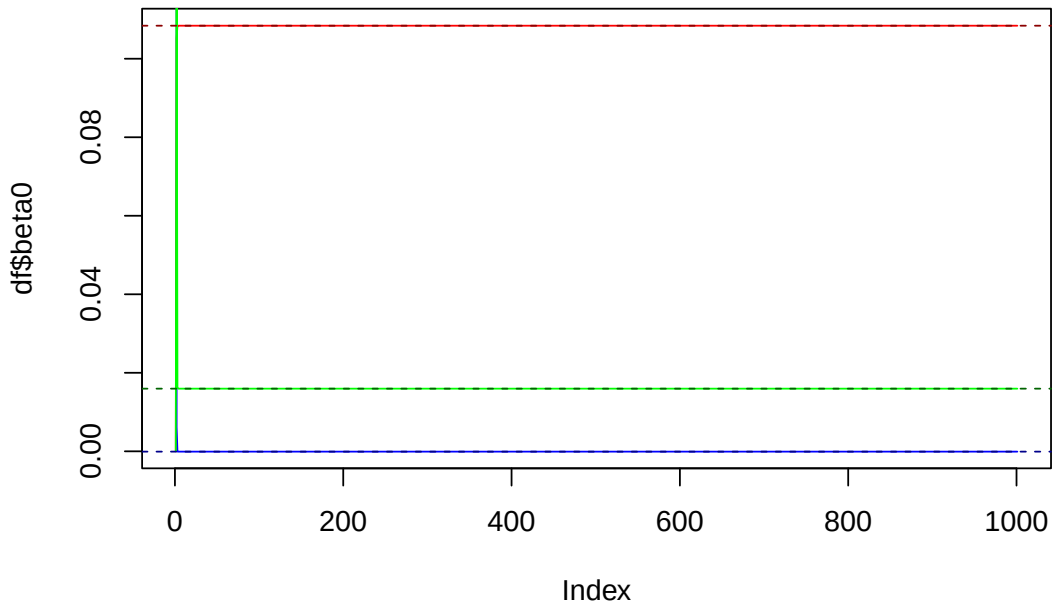
##(c) Keeping $\hat{\beta}_1$ fixed, fit the model: $Y - \hat{\beta}_1 X_1 = \beta_0 + \beta_2 X_2 + \epsilon$

##(d) Keeping $\hat{\beta}_2$ fixed, fit the model: $Y - \hat{\beta}_2 X_2 = \beta_0 + \beta_1 X_1 + \epsilon$

##(e) Write a for loop to repeat (c) and (d) 1,000 times. Report the estimates of $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\beta}_2$ at each iteration of the for loop. Create a plot in which each of these values is displayed, with $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\beta}_2$ each shown in a different color.



##(f) Compare your answer in (e) to the results of simply performing multiple linear regression to predict Y using X_1 and X_2. Use the abline() function to overlay those multiple linear regression coefficient estimates on the plot obtained in (e).



The coefficients from multiple linear regression and iterations are almost the same.

##(g) On this data set, how many backfitting iterations were required in order to obtain a "good" approximation to the multiple regression coefficient estimates?

From the for loop in 1000 times,it is difficult to see the exactly backfitting iterations. So I run a for loop in 10 times again, I find that for the 10-times loop, three iterations are enough to converge.