Chapter 07: Moving Beyond Linearity

Solutions to Exercises

February 17, 2023

CONCEPTUAL

EXERCISE 1:

Part a)

[... *will come back to this. maybe.*]

Part b)

[... *will come back to this. maybe.*]

Part c)

[... *will come back to this. maybe.*]

Part d)

[... *will come back to this. maybe.*]

Part e)

[... *will come back to this. maybe.*]

EXERCISE 2:

Part a)

When $\lambda = \infty$, first term does not matter. $g^{(0)} = g = 0$ means $\hat{g}$ must be 0

Part b)

When $\lambda = \infty$, first term does not matter. $g^{(1)} = g' = 0$ means $\hat{g}$ must be constant (horizontal line).

Part c)

When $\lambda = \infty$, first term does not matter. $g^{(2)} = g'' = 0$ means $\hat{g}$ must be a straight line like $3x + 2$.

Part d)

When $\lambda = \infty$, first term does not matter. $g^{(3)} = g''' = 0$ means $\hat{g}$ must be a smooth quadratic curve like x^2.

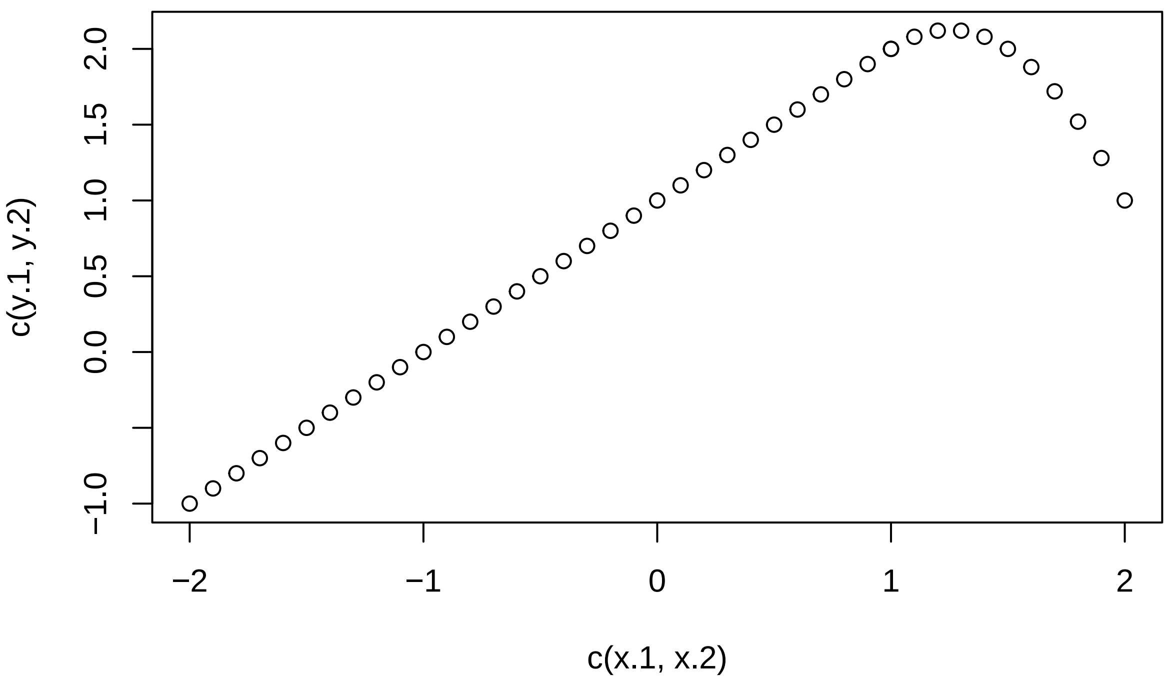
Part e)

When $\lambda = \infty$, second term does not matter and $\hat{g}$ becomes a linear regression least squares fit. $\hat{g}$ can make many forms (e.g. $3x + 5$).

EXERCISE 3:

- $X < 1 : Y = 1 + X$
- $X \geq 1 : Y = 1 + X - 2(X-1)^2$

```
x.1 <- seq(-2,1,0.1)   # X<1
x.2 <- seq(1,2,0.1)    # X>=1
y.1 <- 1 + x.1
y.2 <- 1 + x.2 - 2*(x.2-1)^2
plot(c(x.1,x.2),c(y.1,y.2))
```



EXERCISE 4:

Plugging in the coefficients, $\hat{Y} = 1 + b_1(X) + 3b_2(X)$

- $X < 0 : Y = 1 + (0) + 3(0) = 1$
- $0 \leq X < 1 : 1 + (1) + 3(0) = 2$
- $1 \leq X \leq 2 : 1 + (1 - (X-1)) + 3(0) = 3 - X$
- $2 < X < 3 : 1 + (0) + 3(0) = 1$
- $3 \leq X \leq 4 : 1 + (0) + 3(X-3) = 3X - 8$
- $4 < X \leq 5 : 1 + (0) + 3(1) = 4$
- $X > 5 : 1 + (0) + 3(0) = 1$

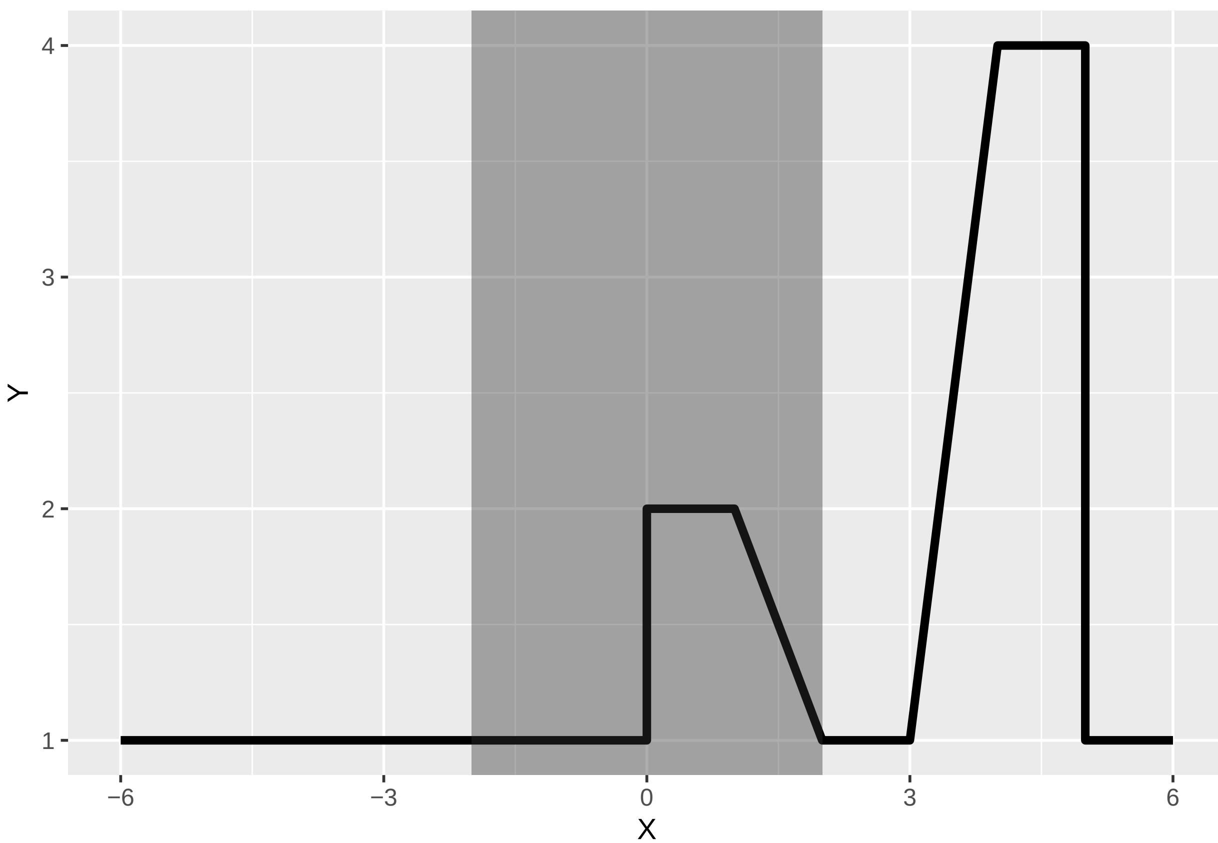
```
require(ggplot2)
```

```
##           : ggplot2
```

```r
x.1 <- seq(-6, 0, 0.1)  # [-6,0)
x.2 <- seq(0, 1, 0.1)   # [0,1)
x.3 <- seq(1, 2, 0.1)   # [1,2]
x.4 <- seq(2, 3, 0.1)   # (2,3)
x.5 <- seq(3, 4, 0.1)   # [3,4]
x.6 <- seq(4, 5, 0.1)   # (4,5]
x.7 <- seq(5, 6, 0.1)   # (5,6)
y.1 <- rep(1, length(x.1))
y.2 <- rep(2, length(x.2))
y.3 <- 3 - x.3
y.4 <- rep(1, length(x.4))
y.5 <- 3*x.5 - 8
y.6 <- rep(4, length(x.6))
y.7 <- rep(1, length(x.7))
df <- data.frame(X = c(x.1,x.2,x.3,x.4,x.5,x.6,x.7),
                 Y = c(y.1,y.2,y.3,y.4,y.5,y.6,y.7))
p <- ggplot(df, aes(x=X,y=Y)) + geom_line(size=1.5)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```

```r
rect <- data.frame(xmin=-2, xmax=2, ymin=-Inf, ymax=Inf)
p + geom_rect(data=rect, aes(xmin=xmin, xmax=xmax, ymin=ymin, ymax=ymax),
              fill="grey20",
              alpha=0.4,
              inherit.aes = FALSE)
```



EXERCISE 5:

Part a)

Because $g^{(3)}$ is more stringent on its smoothness requirements then $g^{(4)}$, we'd expect $\hat{g}_2$ to be more flexible and be able to have a better fit to the training data and thus a smaller training RSS.

Part b)

Hard to say. Depends on true form of y. If $\hat{g}_2$ overfits the data because of its increased flexibility, then $\hat{g}_1$ will likely have a better test RSS.

Part c)

When $\lambda = 0$, only the first term matters, which is the same for both $\hat{g}_1$ and $\hat{g}_2$. The two equations become the same and they would have the same training and test RSS.

APPLIED

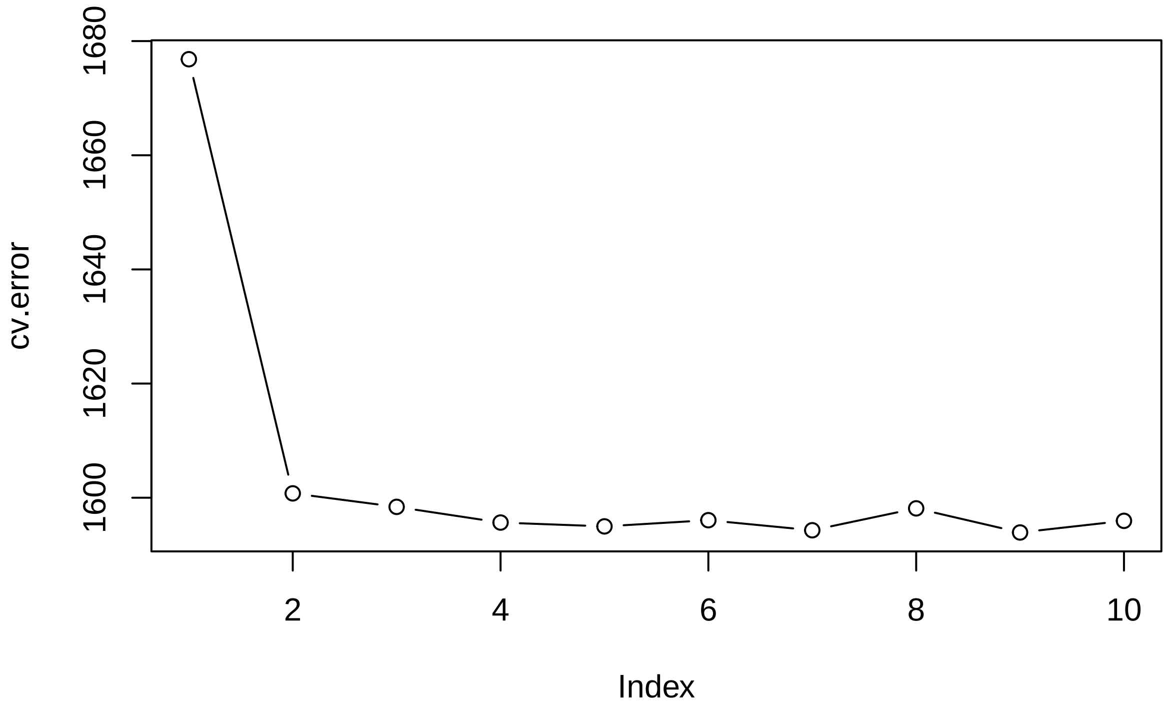
EXERCISE 6:

Part a)

```
require(ISLR2)
require(boot)
data(Wage)
set.seed(1)

# cross-validation
cv.error <- rep(0,10)
for (i in 1:10) {
  glm.fit <- glm(wage~poly(age,i), data=Wage)
  cv.error[i] <- cv.glm(Wage, glm.fit, K=10)$delta[1]  # [1]:std, [2]:bias-corrected
}
cv.error
```

```
## [1] 1676.826 1600.763 1598.399 1595.651 1594.977 1596.061 1594.298 1598.134
## [9] 1593.913 1595.950
```

```
plot(cv.error, type="b")  # 4th degree looks good!
```

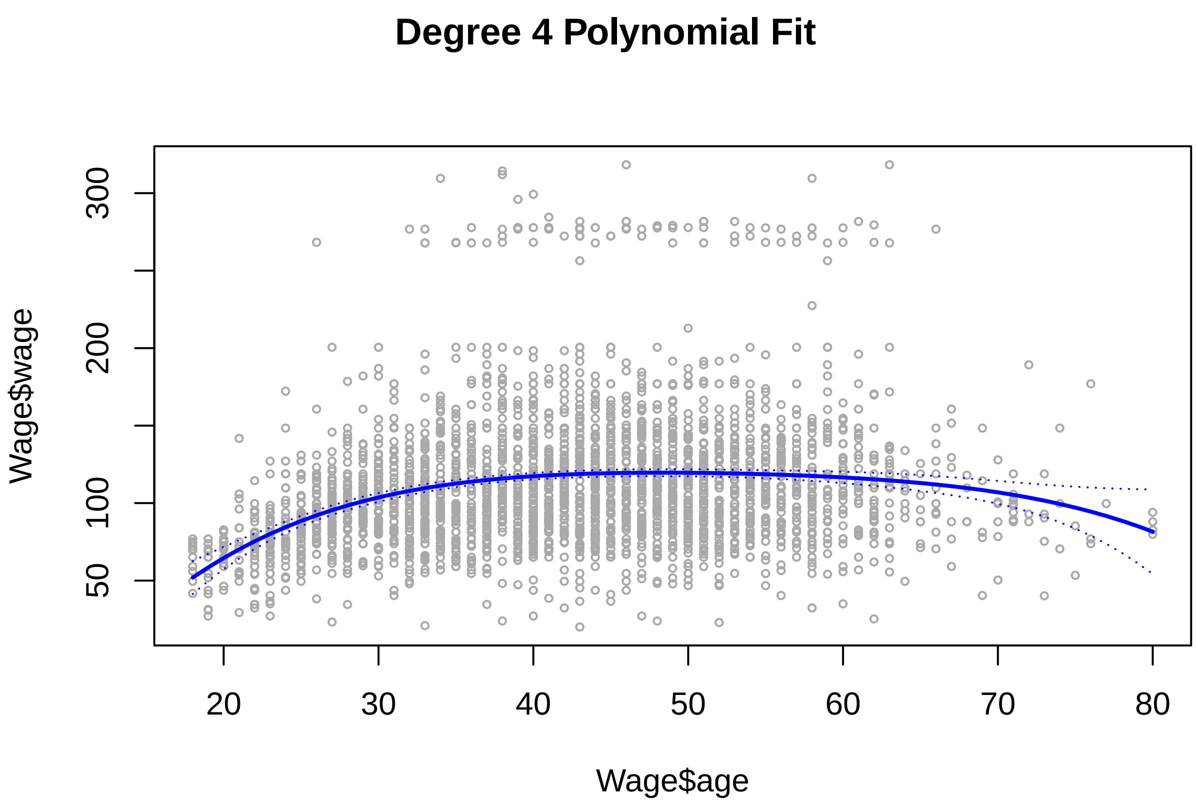



```
# ANOVA
fit.01 <- lm(wage~age, data=Wage)
fit.02 <- lm(wage~poly(age,2), data=Wage)
fit.03 <- lm(wage~poly(age,3), data=Wage)
fit.04 <- lm(wage~poly(age,4), data=Wage)
fit.05 <- lm(wage~poly(age,5), data=Wage)
fit.06 <- lm(wage~poly(age,6), data=Wage)
fit.07 <- lm(wage~poly(age,7), data=Wage)
fit.08 <- lm(wage~poly(age,8), data=Wage)
fit.09 <- lm(wage~poly(age,9), data=Wage)
fit.10 <- lm(wage~poly(age,10), data=Wage)
anova(fit.01,fit.02,fit.03,fit.04,fit.05,fit.06,fit.07,fit.08,fit.09,fit.10)
```

```
## Analysis of Variance Table
##
## Model  1: wage ~ age
## Model  2: wage ~ poly(age, 2)
## Model  3: wage ~ poly(age, 3)
## Model  4: wage ~ poly(age, 4)
## Model  5: wage ~ poly(age, 5)
## Model  6: wage ~ poly(age, 6)
## Model  7: wage ~ poly(age, 7)
## Model  8: wage ~ poly(age, 8)
## Model  9: wage ~ poly(age, 9)
## Model 10: wage ~ poly(age, 10)
##    Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1    2998 5022216
## 2    2997 4793430  1    228786 143.7638 < 2.2e-16 ***
## 3    2996 4777674  1     15756   9.9005  0.001669 **
## 4    2995 4771604  1      6070   3.8143  0.050909 .
## 5    2994 4770322  1      1283   0.8059  0.369398
```

```
## 6     2993 4766389  1       3932   2.4709  0.116074
## 7     2992 4763834  1       2555   1.6057  0.205199
## 8     2991 4763707  1        127   0.0796  0.777865
## 9     2990 4756703  1       7004   4.4014  0.035994 *
## 10    2989 4756701  1          3   0.0017  0.967529
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# 3rd or 4th degrees look best based on ANOVA test
# let's go with 4th degree fit
agelims <- range(Wage$age)
age.grid <- seq(agelims[1], agelims[2])
preds <- predict(fit.04, newdata=list(age=age.grid), se=TRUE)
se.bands <- preds$fit + cbind(2*preds$se.fit, -2*preds$se.fit)
par(mfrow=c(1,1), mar=c(4.5,4.5,1,1), oma=c(0,0,4,0))
plot(Wage$age, Wage$wage, xlim=agelims, cex=0.5, col="darkgrey")
title("Degree 4 Polynomial Fit", outer=TRUE)
lines(age.grid, preds$fit, lwd=2, col="blue")
matlines(age.grid, se.bands, lwd=1, col="blue", lty=3)
```



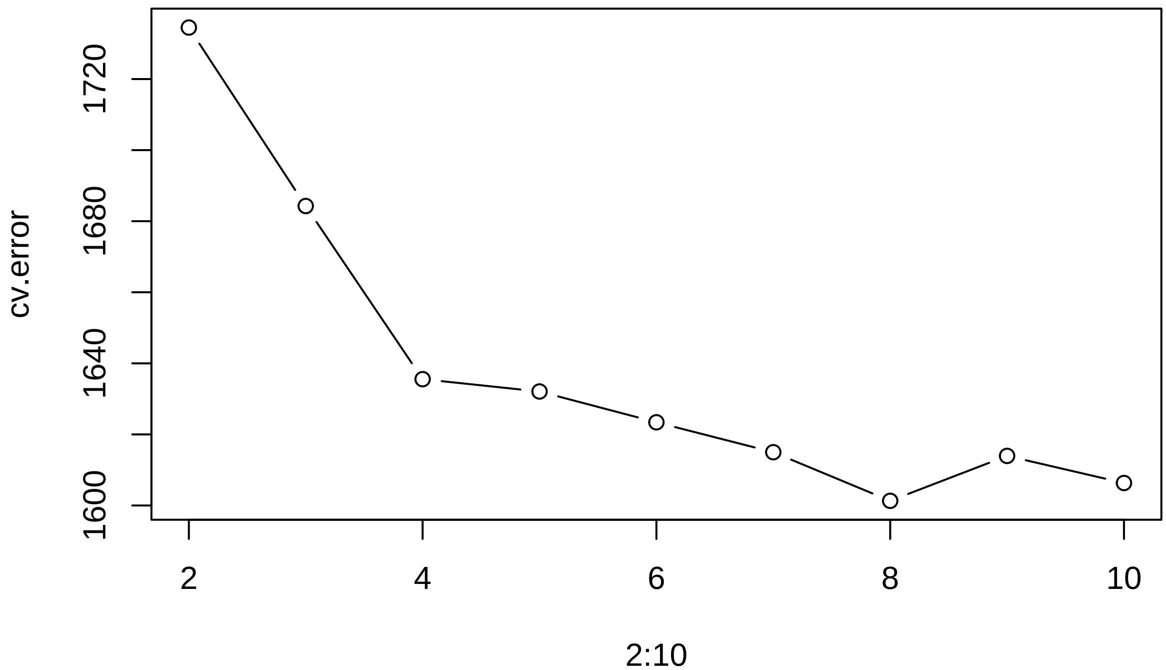
Part b)

```
set.seed(1)

# cross-validation
cv.error <- rep(0,9)
for (i in 2:10) {
  Wage$age.cut <- cut(Wage$age,i)
```

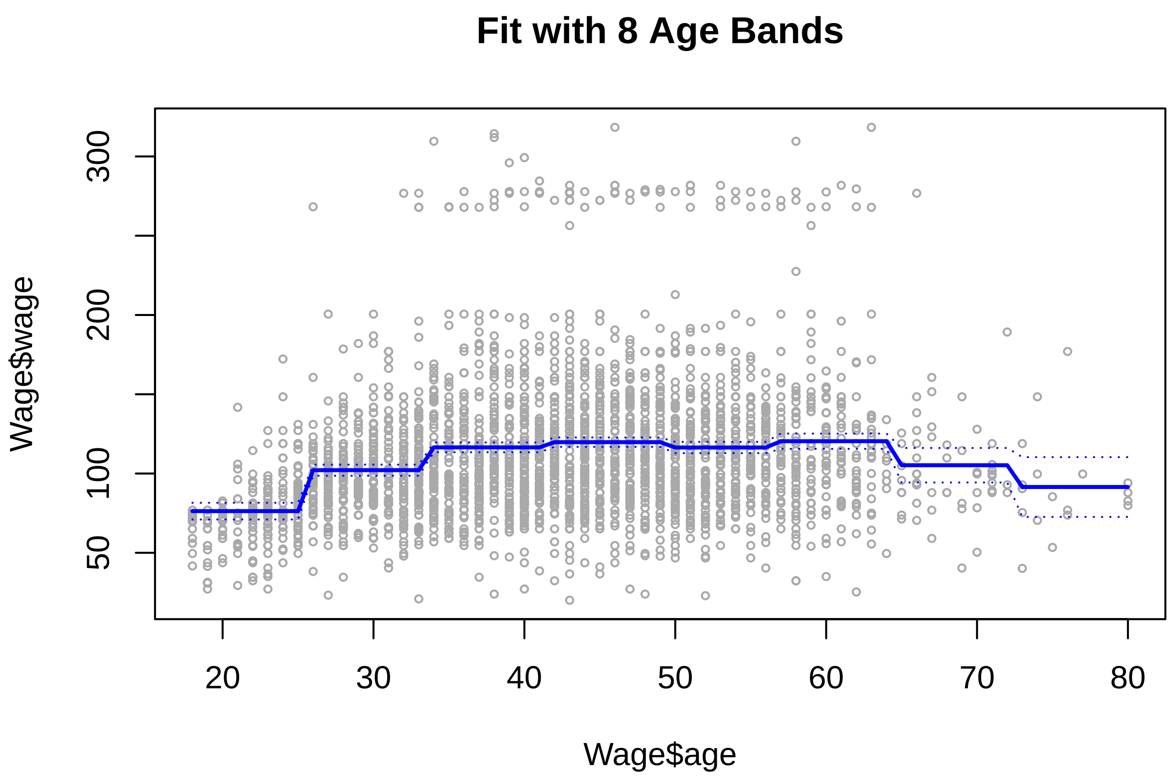
```r
  glm.fit <- glm(wage~age.cut, data=Wage)
  cv.error[i-1] <- cv.glm(Wage, glm.fit, K=10)$delta[1]  # [1]:std, [2]:bias-corrected
}
cv.error
```

```
## [1] 1734.489 1684.271 1635.552 1632.080 1623.415 1614.996 1601.318 1613.954
## [9] 1606.331
```

```r
plot(2:10, cv.error, type="b")  # 7 or 8 cuts look optimal
```

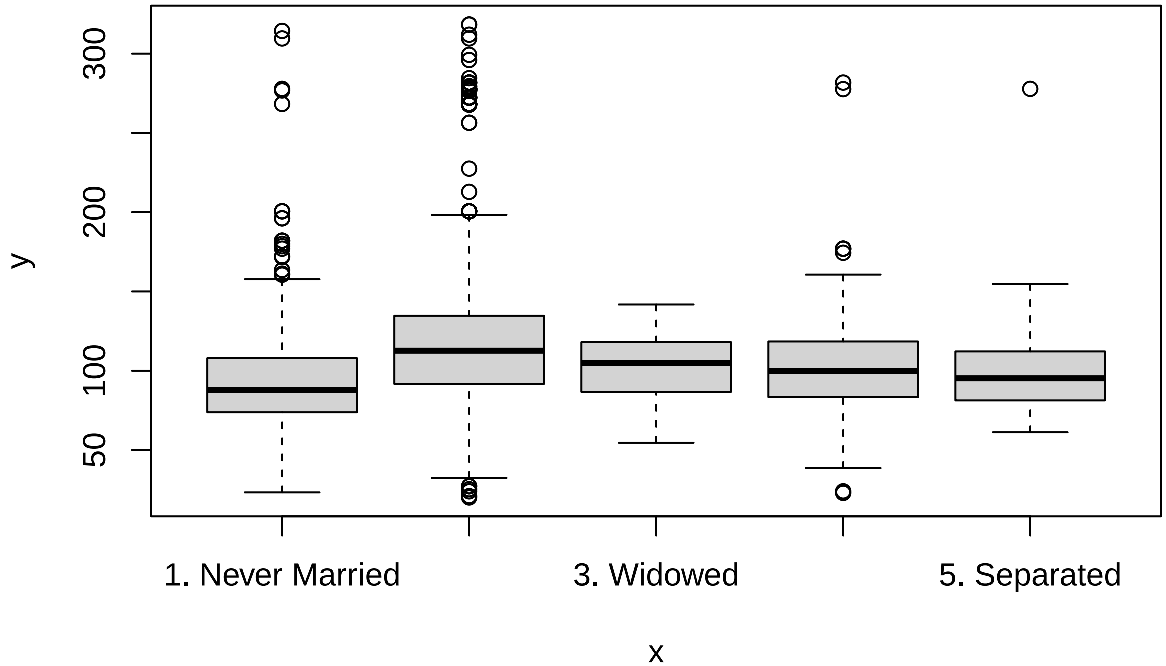



```r
# going with 8 cuts
cut.fit <- glm(wage~cut(age,8), data=Wage)
preds <- predict(cut.fit, newdata=list(age=age.grid), se=TRUE)
se.bands <- preds$fit + cbind(2*preds$se.fit, -2*preds$se.fit)
plot(Wage$age, Wage$wage, xlim=agelims, cex=0.5, col="darkgrey")
title("Fit with 8 Age Bands")
lines(age.grid, preds$fit, lwd=2, col="blue")
matlines(age.grid, se.bands, lwd=1, col="blue", lty=3)
```

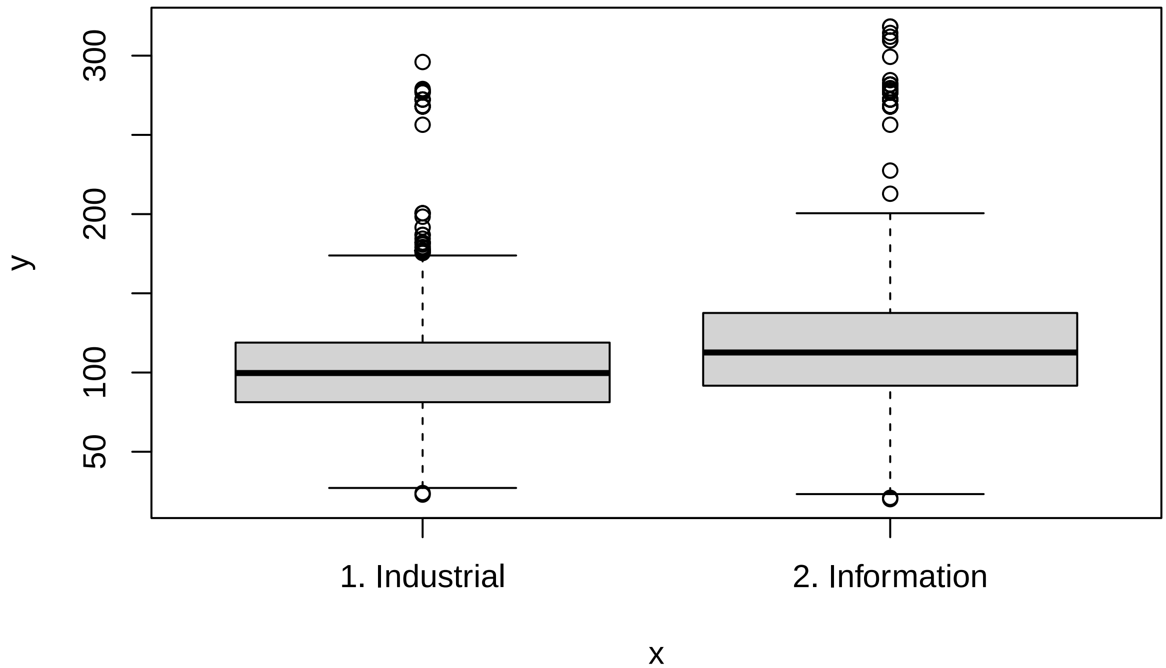


EXERCISE 7:

```
plot(Wage$maritl, Wage$wage)
```



```r
plot(Wage$jobclass, Wage$wage)
```



Both marital status and job class are categorical variables. It seems that on a univariate basis, wages for `jobclass=Information` are higher than `jobclass=Industrial`. For marital status, married seems to have the highest wages, though this is probably confounded by age.

```r
require(gam)
gam.fit1 <- gam(wage~ns(age,5), data=Wage)
gam.fit2.1 <- gam(wage~ns(age,5)+maritl, data=Wage)
gam.fit2.2 <- gam(wage~ns(age,5)+jobclass, data=Wage)
gam.fit3 <- gam(wage~ns(age,5)+maritl+jobclass, data=Wage)
anova(gam.fit1, gam.fit2.1, gam.fit3)
```

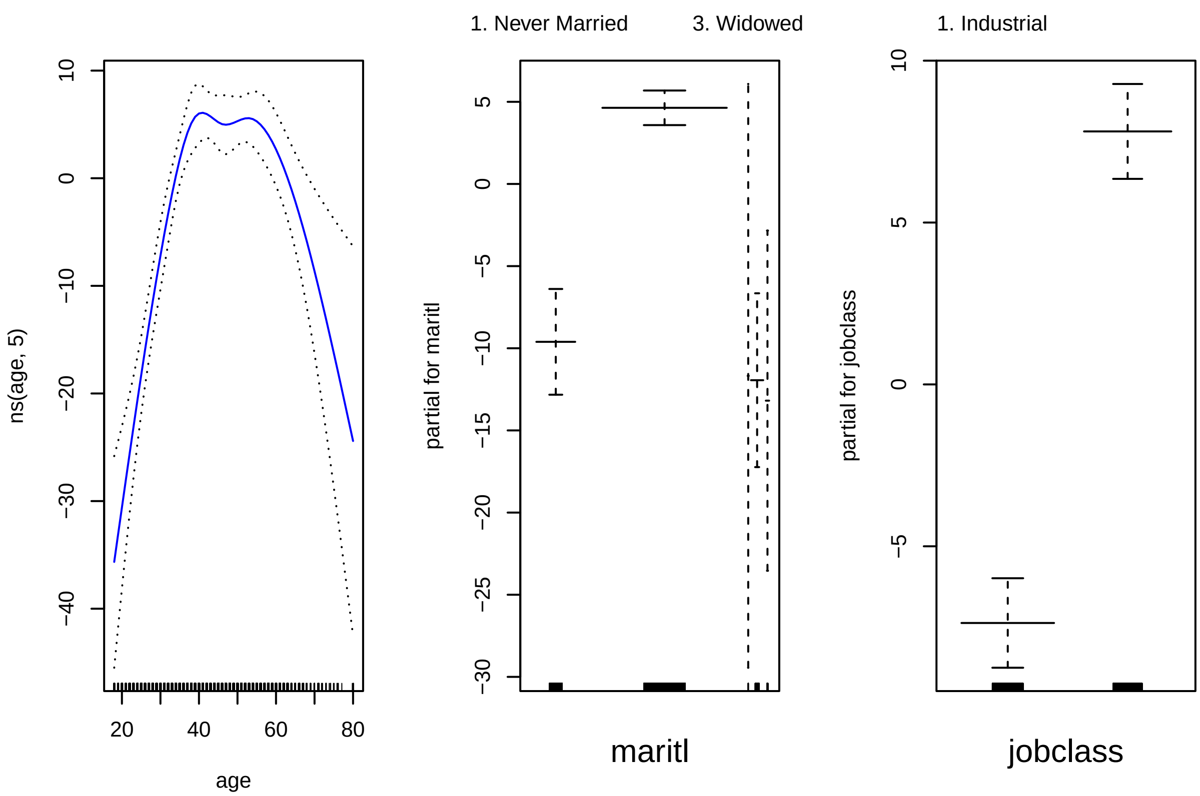
```
## Analysis of Deviance Table
##
## Model 1: wage ~ ns(age, 5)
## Model 2: wage ~ ns(age, 5) + maritl
## Model 3: wage ~ ns(age, 5) + maritl + jobclass
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      2994    4768634
## 2      2990    4647371  4   121263 < 2.2e-16 ***
## 3      2989    4477023  1   170348 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(gam.fit1, gam.fit2.2, gam.fit3)
```

```
## Analysis of Deviance Table
##
## Model 1: wage ~ ns(age, 5)
```

```
## Model 2: wage ~ ns(age, 5) + jobclass
## Model 3: wage ~ ns(age, 5) + maritl + jobclass
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      2994    4768634
## 2      2993    4601881  1   166752 < 2.2e-16 ***
## 3      2989    4477023  4   124858 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

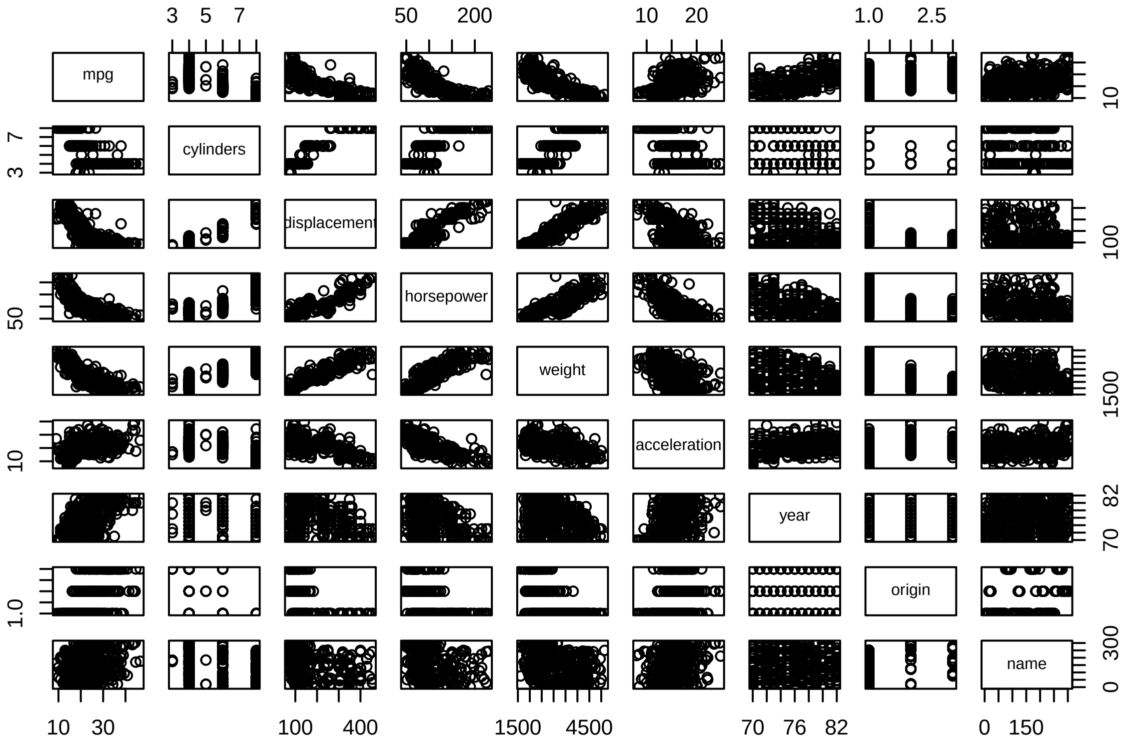
```
# both marital status and job class are significant even with age included
par(mfrow=c(1,3))
plot(gam.fit3, se=TRUE, col="blue")
```



EXERCISE 8:

Assume we are interested in predicting `mpg`.

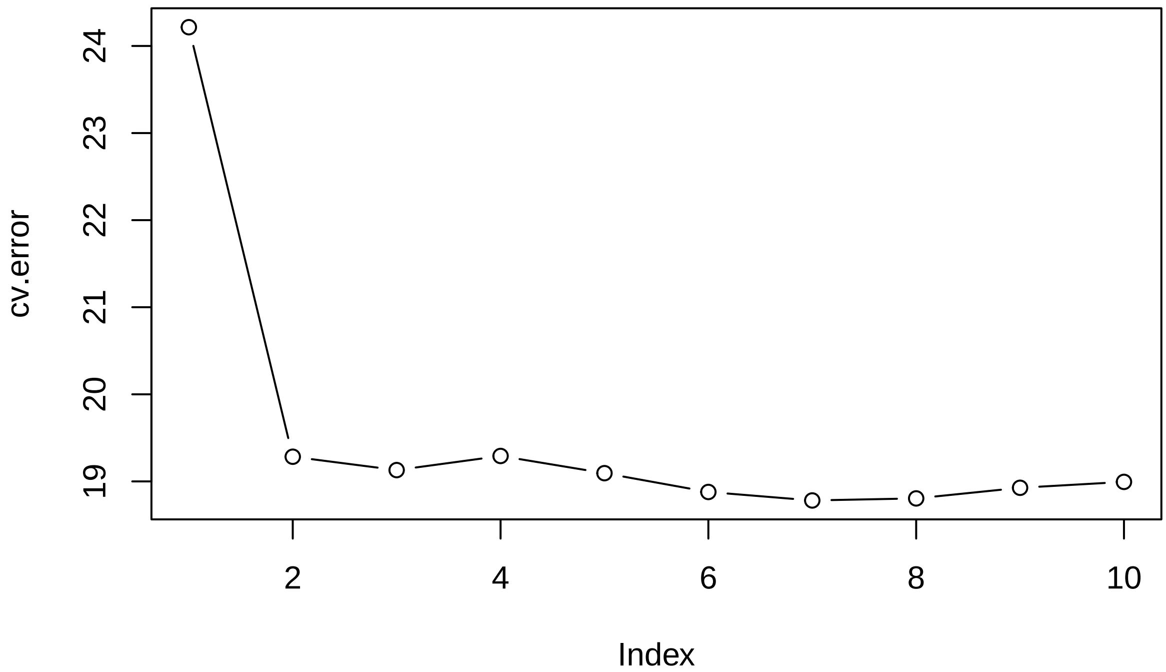
```
require(ISLR2)
require(boot)
require(gam)
data(Auto)
set.seed(1)

# a few quick plots to look at data
pairs(Auto)
```




```
# `displacement`, `horsepower`, `weight`, `acceleration` may have nonlinear relationships

# polynomial fit with cross-validation on `horsepower`
cv.error <- rep(0,10)
for (i in 1:10) {
  glm.fit <- glm(mpg~poly(horsepower,i), data=Auto)
  cv.error[i] <- cv.glm(Auto, glm.fit, K=10)$delta[1]   # [1]:std, [2]:bias-corrected
}
cv.error
```

```
##  [1] 24.21538 19.28327 19.12998 19.29201 19.09471 18.87874 18.78127 18.80484
##  [9] 18.92676 18.99439
```

```
plot(cv.error, type="b")  # 1st degree definitely not enough, 2nd looks good
```




```r
# gam fit with `horsepower`, `weight` and `cylinders`
Auto$cylinders <- factor(Auto$cylinders)  # turn into factor variable
gam.fit1 <- gam(mpg~poly(horsepower,2), data=Auto)
gam.fit2.1 <- gam(mpg~poly(horsepower,2)+weight, data=Auto)
gam.fit2.2 <- gam(mpg~poly(horsepower,2)+cylinders, data=Auto)
gam.fit3 <- gam(mpg~poly(horsepower,2)+weight+cylinders, data=Auto)
anova(gam.fit1, gam.fit2.1, gam.fit3)
```

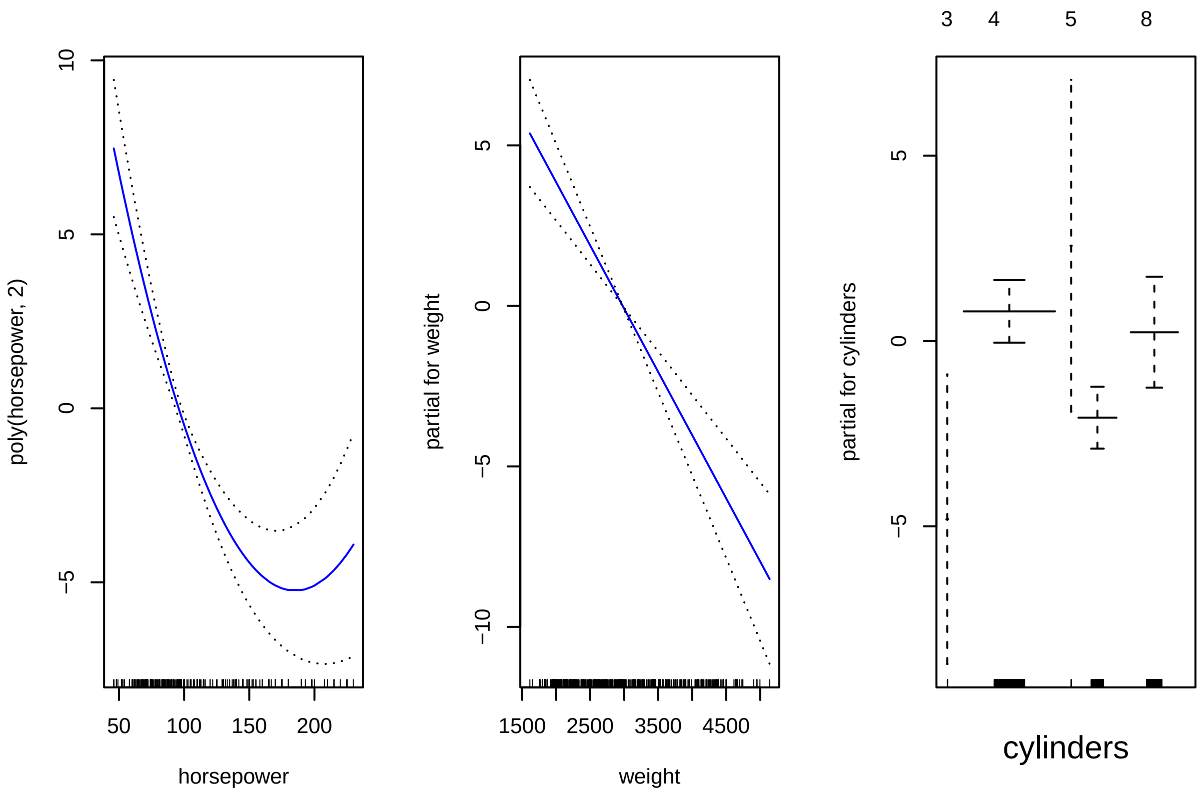
```
## Analysis of Deviance Table
##
## Model 1: mpg ~ poly(horsepower, 2)
## Model 2: mpg ~ poly(horsepower, 2) + weight
## Model 3: mpg ~ poly(horsepower, 2) + weight + cylinders
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       389     7442.0
## 2       388     6201.6  1  1240.42 < 2.2e-16 ***
## 3       384     5699.7  4   501.93 8.129e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(gam.fit1, gam.fit2.2, gam.fit3)
```

```
## Analysis of Deviance Table
##
## Model 1: mpg ~ poly(horsepower, 2)
## Model 2: mpg ~ poly(horsepower, 2) + cylinders
## Model 3: mpg ~ poly(horsepower, 2) + weight + cylinders
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       389     7442.0
## 2       385     6315.5  4  1126.48 1.289e-15 ***
## 3       384     5699.7  1   615.86 1.183e-10 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

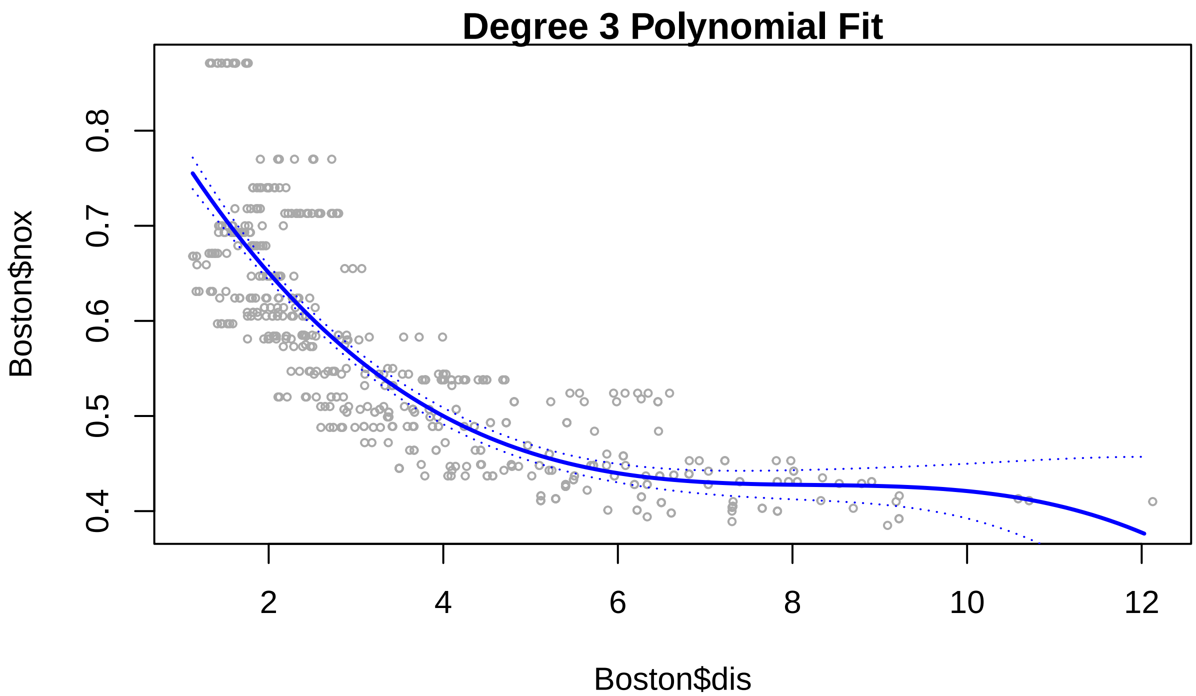
```
# both `weight` and `cylinders` are significant even with `horsepower` included
par(mfrow=c(1,3))
plot(gam.fit3, se=TRUE, col="blue")
```



EXERCISE 9:

Part a)

```
require(MASS)
data(Boston)
set.seed(1)
fit.03 <- lm(nox~poly(dis,3), data=Boston)
dislims <- range(Boston$dis)
dis.grid <- seq(dislims[1], dislims[2], 0.1)
preds <- predict(fit.03, newdata=list(dis=dis.grid), se=TRUE)
se.bands <- preds$fit + cbind(2*preds$se.fit, -2*preds$se.fit)
par(mfrow=c(1,1), mar=c(4.5,4.5,1,1), oma=c(0,0,4,0))
plot(Boston$dis, Boston$nox, xlim=dislims, cex=0.5, col="darkgrey")
title("Degree 3 Polynomial Fit")
lines(dis.grid, preds$fit, lwd=2, col="blue")
matlines(dis.grid, se.bands, lwd=1, col="blue", lty=3)
```




```
summary(fit.03)
```

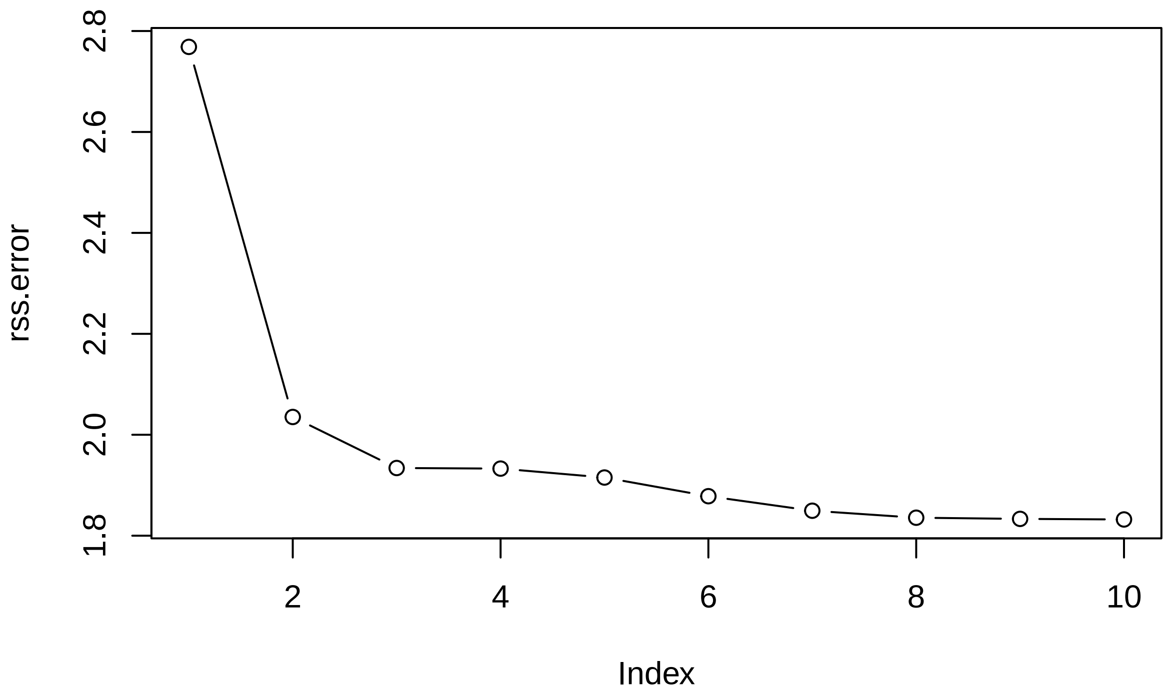
```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 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
```

Part b)

```
rss.error <- rep(0,10)
for (i in 1:10) {
  lm.fit <- lm(nox~poly(dis,i), data=Boston)
  rss.error[i] <- sum(lm.fit$residuals^2)
}
rss.error
```

```
##  [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484 1.835630
##  [9] 1.833331 1.832171
```

```
plot(rss.error, type="b")
```

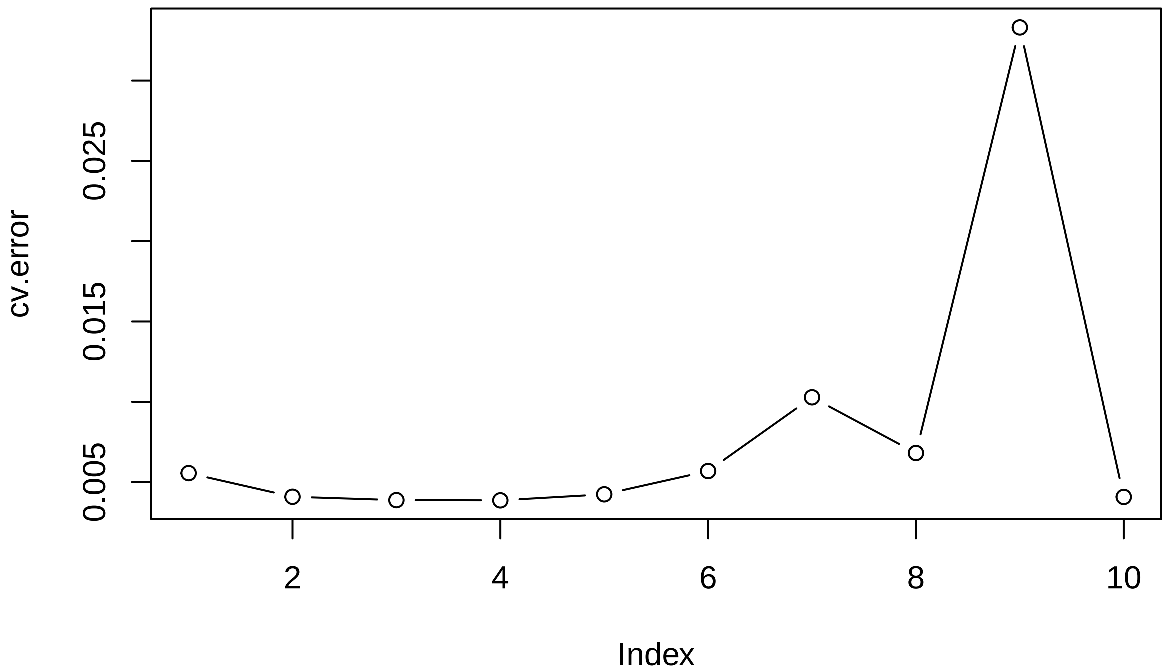


Part c)

```
require(boot)
set.seed(1)
cv.error <- rep(0,10)
for (i in 1:10) {
  glm.fit <- glm(nox~poly(dis,i), data=Boston)
  cv.error[i] <- cv.glm(Boston, glm.fit, K=10)$delta[1]  # [1]:std, [2]:bias-corrected
}
cv.error
```

```
##  [1] 0.005558263 0.004085706 0.003876521 0.003863342 0.004237452 0.005686862
##  [7] 0.010278897 0.006810868 0.033308607 0.004075599
```

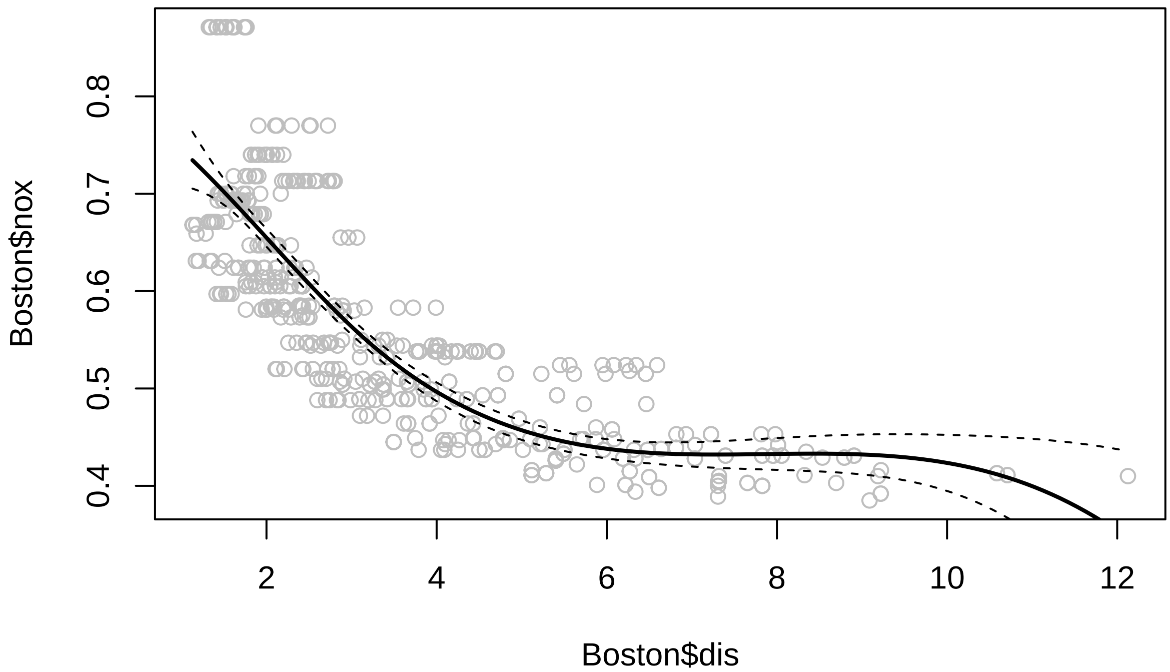
```
plot(cv.error, type="b")  # woah!
```



The optimal fit seems to be with a 4th degree polynomial, though the 2nd degree fit is not much worse. Crazy things happen with 7th and 9th degree fits.

Part d)

```
require(splines)
fit.sp <- lm(nox~bs(dis, df=4), data=Boston)
pred <- predict(fit.sp, newdata=list(dis=dis.grid), se=T)
plot(Boston$dis, Boston$nox, col="gray")
lines(dis.grid, pred$fit, lwd=2)
lines(dis.grid, pred$fit+2*pred$se, lty="dashed")
lines(dis.grid, pred$fit-2*pred$se, lty="dashed")
```




```
# set df to select knots at uniform quantiles of `dis`
attr(bs(Boston$dis,df=4),"knots")  # only 1 knot at 50th percentile
```

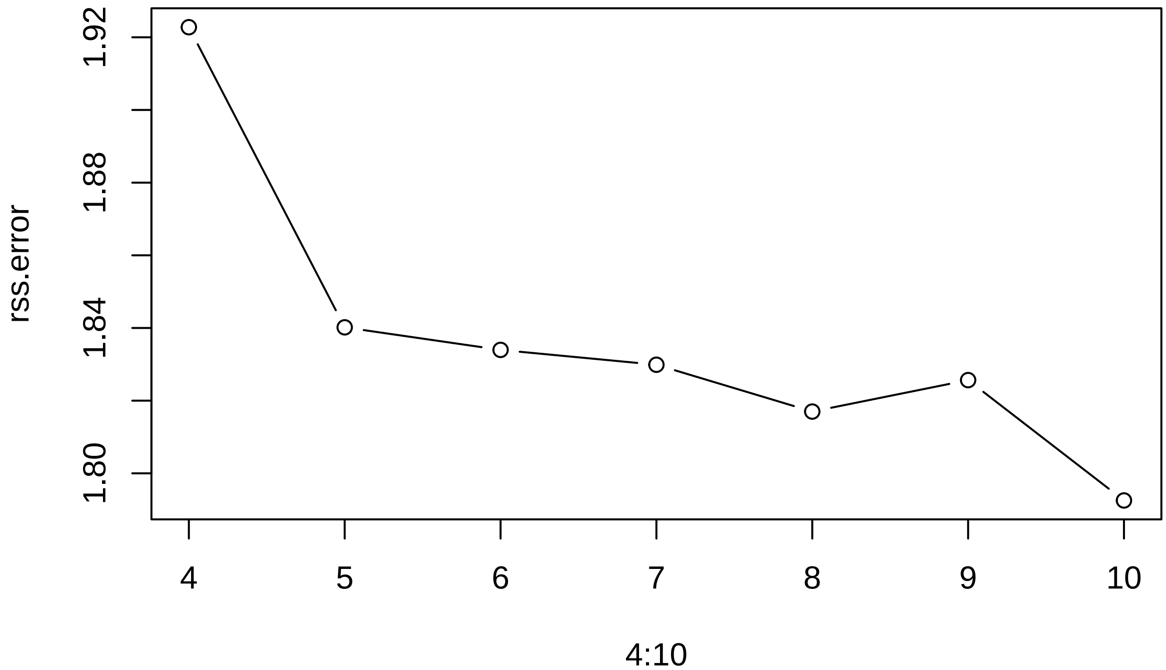
```
##     50%
## 3.20745
```

Part e)

```
require(splines)
set.seed(1)
rss.error <- rep(0,7)
for (i in 4:10) {
  fit.sp <- lm(nox~bs(dis, df=i), data=Boston)
  rss.error[i-3] <- sum(fit.sp$residuals^2)
}
rss.error
```

```
## [1] 1.922775 1.840173 1.833966 1.829884 1.816995 1.825653 1.792535
```

```
plot(4:10, rss.error, type="b")  # RSS decreases on train set w more flexible fit
```

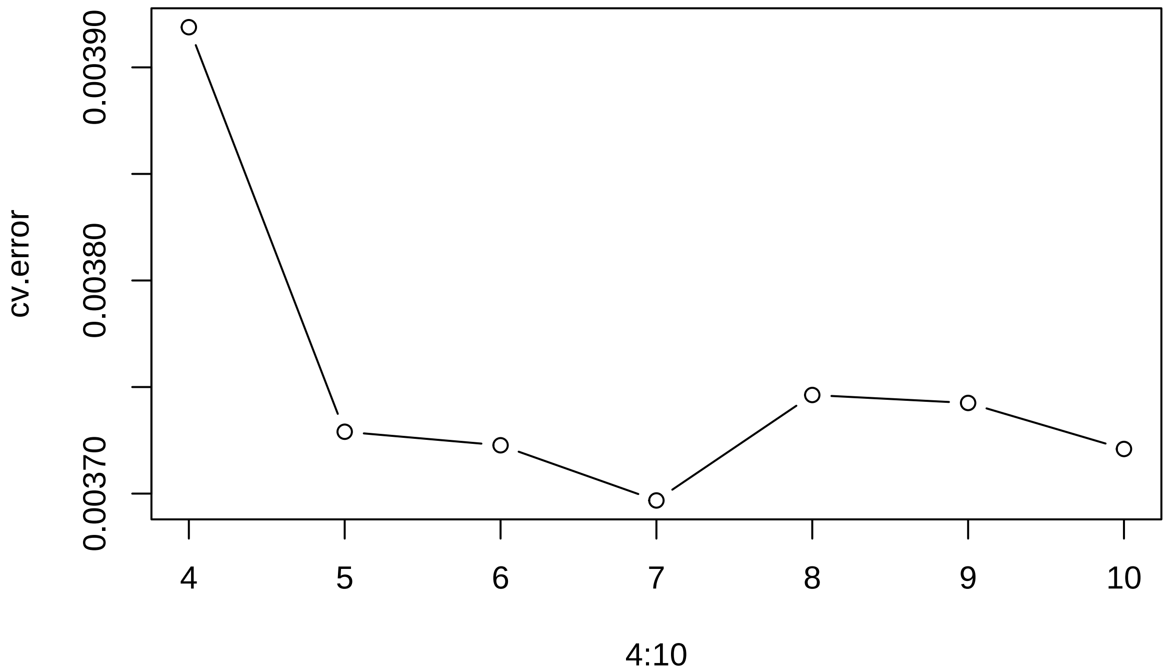


Part f)

```
require(splines)
require(boot)
set.seed(1)
cv.error <- rep(0,7)
for (i in 4:10) {
  glm.fit <- glm(nox~bs(dis, df=i), data=Boston)
  cv.error[i-3] <- cv.glm(Boston, glm.fit, K=10)$delta[1]
}
cv.error
```

```
## [1] 0.003918838 0.003729024 0.003722683 0.003696789 0.003746270 0.003742534
## [7] 0.003720942
```

```
plot(4:10, cv.error, type="b")  # should use at least df=5
```



EXERCISE 10:

Part a)

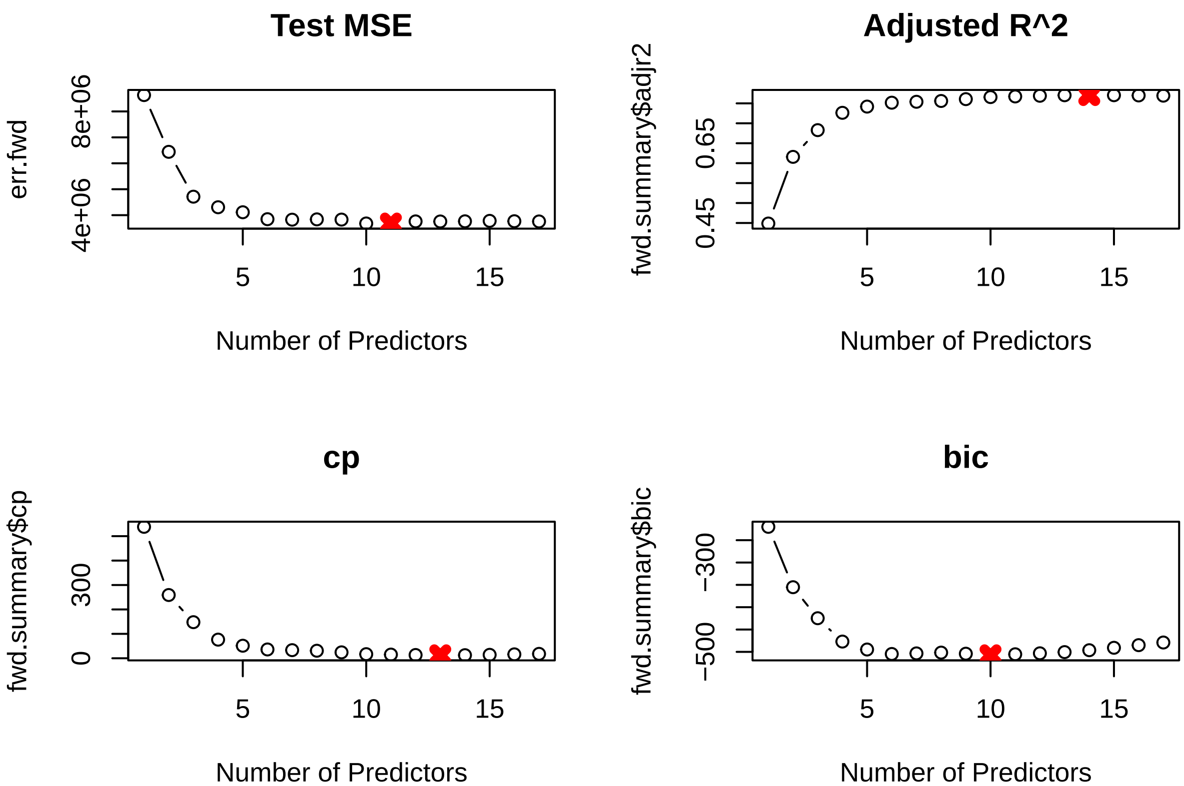
```
require(ISLR2)
require(leaps)
data(College)
set.seed(1)

# split data into train and test sets
trainid <- sample(1:nrow(College), nrow(College)/2)
train <- College[trainid,]
test <- College[-trainid,]

# predict function from chapter 6 labs
predict.regsubsets <- function(object, newdata, id, ...){
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi <- coef(object, id=id)
  xvars <- names(coefi)
  mat[,xvars]%*%coefi
}

# forward selection
fit.fwd <- regsubsets(Outstate~., data=train, nvmax=ncol(College)-1)
(fwd.summary <- summary(fit.fwd))
```

```
## Subset selection object
## Call: regsubsets.formula(Outstate ~ ., data = train, nvmax = ncol(College) -
##     1)
```

```
## 17 Variables  (and intercept)
##             Forced in Forced out
## PrivateYes      FALSE      FALSE
## Apps            FALSE      FALSE
## Accept          FALSE      FALSE
## Enroll          FALSE      FALSE
## Top10perc       FALSE      FALSE
## Top25perc       FALSE      FALSE
## F.Undergrad     FALSE      FALSE
## P.Undergrad     FALSE      FALSE
## Room.Board      FALSE      FALSE
## Books           FALSE      FALSE
## Personal        FALSE      FALSE
## PhD             FALSE      FALSE
## Terminal        FALSE      FALSE
## S.F.Ratio       FALSE      FALSE
## perc.alumni     FALSE      FALSE
## Expend          FALSE      FALSE
## Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 17
## Selection Algorithm: exhaustive
##           PrivateYes Apps Accept Enroll Top10perc Top25perc F.Undergrad
## 1  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 2  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 3  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 4  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 5  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 6  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 7  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 8  ( 1 )  "*"        " "  " "    " "    "*"       " "       " "
## 9  ( 1 )  "*"        "*"  "*"    "*"    "*"       " "       " "
## 10  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "
## 11  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "
## 12  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       " "
## 13  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       " "
## 14  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       " "
## 15  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       " "
## 16  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
## 17  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
##           P.Undergrad Room.Board Books Personal PhD Terminal S.F.Ratio
## 1  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 2  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 3  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 4  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 5  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 6  ( 1 )  " "         "*"        " "   " "      " " "*"      " "
## 7  ( 1 )  " "         "*"        " "   "*"      " " "*"      " "
## 8  ( 1 )  " "         "*"        " "   "*"      " " "*"      " "
## 9  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 10  ( 1 ) " "         "*"        " "   " "      " " "*"      " "
## 11  ( 1 ) " "         "*"        " "   "*"      " " "*"      " "
## 12  ( 1 ) " "         "*"        " "   "*"      " " "*"      " "
## 13  ( 1 ) " "         "*"        " "   "*"      " " "*"      "*"
## 14  ( 1 ) " "         "*"        "*"   "*"      " " "*"      "*"
```

```
## 15  ( 1 ) "*"          "*"         "*"    "*"       " " "*"       "*"
## 16  ( 1 ) "*"          "*"         "*"    "*"       " " "*"       "*"
## 17  ( 1 ) "*"          "*"         "*"    "*"       "*" "*"       "*"
##           perc.alumni Expend Grad.Rate
## 1  ( 1 )  " "         " "    " "
## 2  ( 1 )  "*"         " "    " "
## 3  ( 1 )  " "         "*"    " "
## 4  ( 1 )  "*"         "*"    " "
## 5  ( 1 )  "*"         "*"    "*"
## 6  ( 1 )  "*"         "*"    "*"
## 7  ( 1 )  "*"         "*"    "*"
## 8  ( 1 )  "*"         "*"    "*"
## 9  ( 1 )  "*"         "*"    "*"
## 10  ( 1 ) "*"         "*"    "*"
## 11  ( 1 ) "*"         "*"    "*"
## 12  ( 1 ) "*"         "*"    "*"
## 13  ( 1 ) "*"         "*"    "*"
## 14  ( 1 ) "*"         "*"    "*"
## 15  ( 1 ) "*"         "*"    "*"
## 16  ( 1 ) "*"         "*"    "*"
## 17  ( 1 ) "*"         "*"    "*"
```

```r
err.fwd <- rep(NA, ncol(College)-1)
for(i in 1:(ncol(College)-1)) {
  pred.fwd <- predict(fit.fwd, test, id=i)
  err.fwd[i] <- mean((test$Outstate - pred.fwd)^2)
}
par(mfrow=c(2,2))
plot(err.fwd, type="b", main="Test MSE", xlab="Number of Predictors")
min.mse <- which.min(err.fwd)
points(min.mse, err.fwd[min.mse], col="red", pch=4, lwd=5)
plot(fwd.summary$adjr2, type="b", main="Adjusted R^2", xlab="Number of Predictors")
max.adjr2 <- which.max(fwd.summary$adjr2)
points(max.adjr2, fwd.summary$adjr2[max.adjr2], col="red", pch=4, lwd=5)
plot(fwd.summary$cp, type="b", main="cp", xlab="Number of Predictors")
min.cp <- which.min(fwd.summary$cp)
points(min.cp, fwd.summary$cp[min.cp], col="red", pch=4, lwd=5)
plot(fwd.summary$bic, type="b", main="bic", xlab="Number of Predictors")
min.bic <- which.min(fwd.summary$bic)
points(min.bic, fwd.summary$bic[min.bic], col="red", pch=4, lwd=5)
```

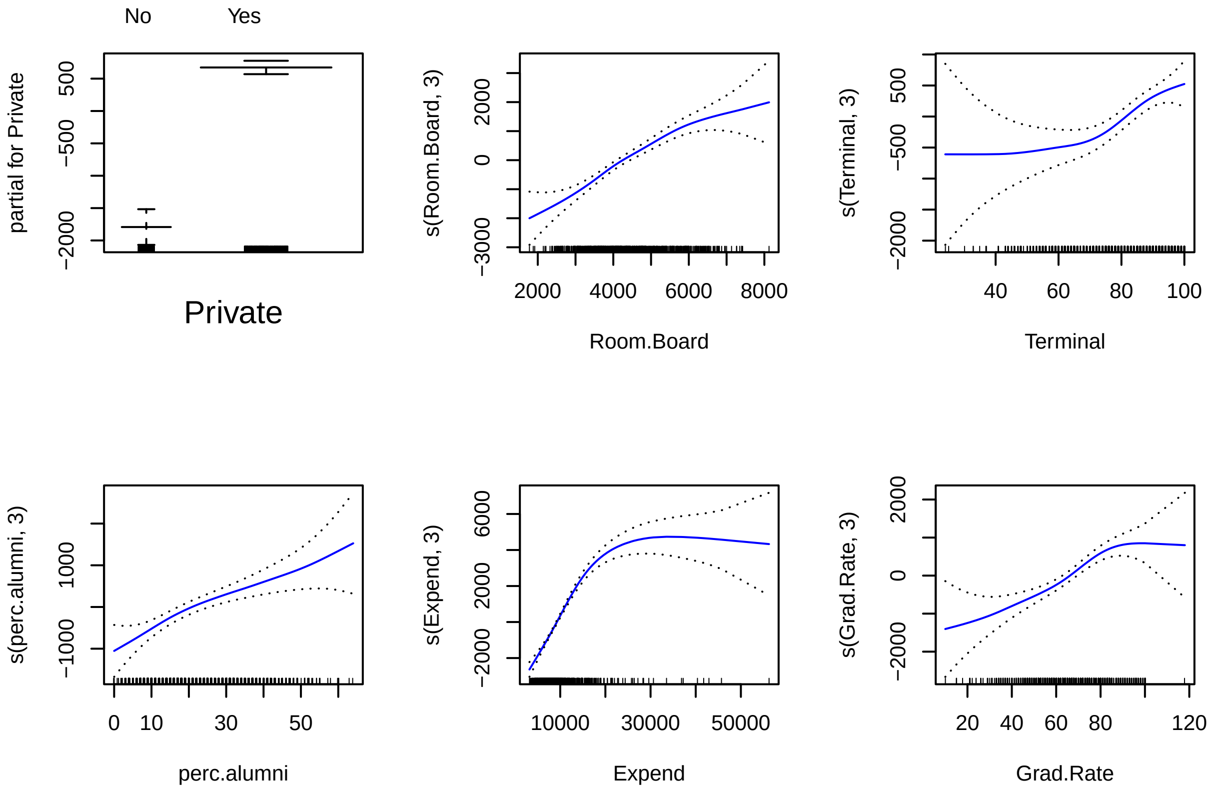



```
# model metrics do not improve much after 6 predictors
coef(fit.fwd, 6)
```

```
##   (Intercept)    PrivateYes    Room.Board      Terminal   perc.alumni
## -4726.8810613  2717.7019276     1.1032433    36.9990286    59.0863753
##        Expend     Grad.Rate
##     0.1930814    33.8303314
```

Part b)

```
require(gam)
gam.fit <- gam(Outstate ~
                 Private +   # categorical variable
                 s(Room.Board,3) +
                 s(Terminal,3) +
                 s(perc.alumni,3) +
                 s(Expend,3) +
                 s(Grad.Rate,3),
               data=College)
par(mfrow=c(2,3))
plot(gam.fit, se=TRUE, col="blue")
```



Part c)

```
pred <- predict(gam.fit, test)
(mse.error <- mean((test$Outstate - pred)^2))
```

```
## [1] 3141785
```

```
err.fwd[6]   # significantly better than linear fit
```

```
## [1] 3844857
```

Part d)

```
summary(gam.fit)
```

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, 3) + s(Terminal,
##     3) + s(perc.alumni, 3) + s(Expend, 3) + s(Grad.Rate, 3),
##     data = College)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -7110.16 -1137.02    50.44  1285.38  8278.86
##
## (Dispersion Parameter for gaussian family taken to be 3520187)
##
##     Null Deviance: 12559297426 on 776 degrees of freedom
```

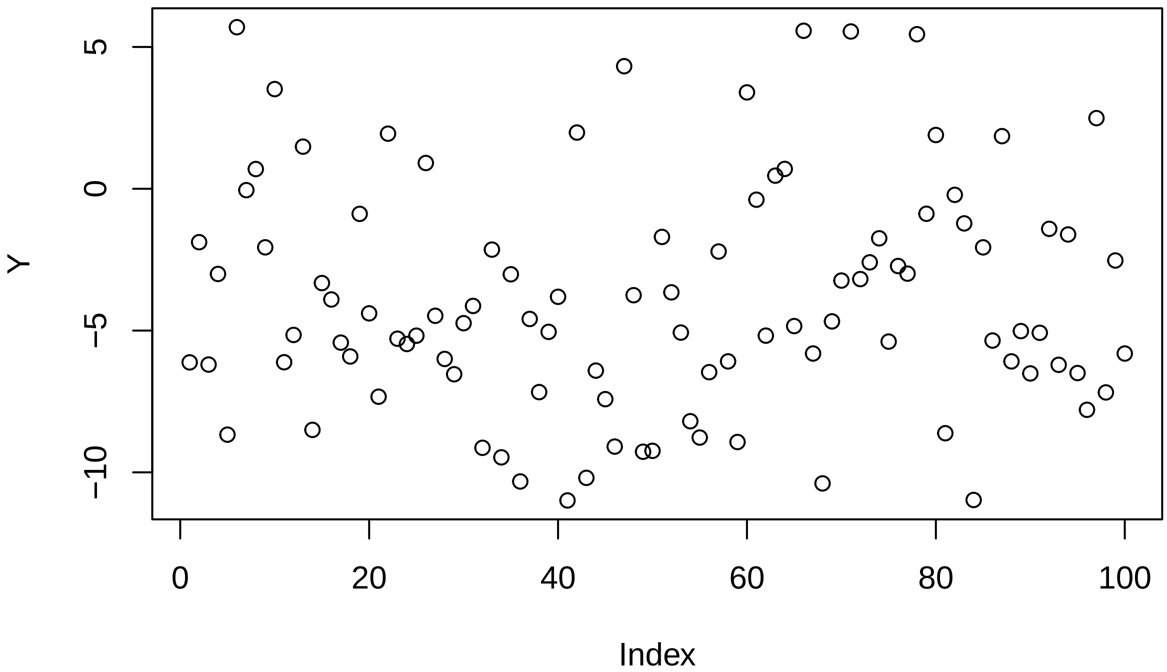
```
## Residual Deviance: 2675342725 on 760.0001 degrees of freedom
## AIC: 13936.36
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                      Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private               1 3366732308 3366732308 956.407 < 2.2e-16 ***
## s(Room.Board, 3)      1 2549088628 2549088628 724.134 < 2.2e-16 ***
## s(Terminal, 3)        1  802254341  802254341 227.901 < 2.2e-16 ***
## s(perc.alumni, 3)     1  525154274  525154274 149.184 < 2.2e-16 ***
## s(Expend, 3)          1 1022010841 1022010841 290.329 < 2.2e-16 ***
## s(Grad.Rate, 3)       1  151344060  151344060  42.993 1.014e-10 ***
## Residuals           760 2675342725    3520187
## ---
## 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, 3)          2  2.591 0.07557 .
## s(Terminal, 3)            2  2.558 0.07815 .
## s(perc.alumni, 3)         2  0.835 0.43446
## s(Expend, 3)              2 56.179 < 2e-16 ***
## s(Grad.Rate, 3)           2  3.363 0.03515 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Strong evidence of non-linear effects for `Expend`, some evidence for `Room.Board`, `Terminal` and `Grad.Rate`, and no evidence for `perc.alumni`.

EXERCISE 11:

Part a)

```r
set.seed(1)
X1 <- rnorm(100)
X2 <- rnorm(100)
beta_0 <- -3.8
beta_1 <- 0.3
beta_2 <- 4.1
eps <- rnorm(100, sd = 1)
Y <- beta_0 + beta_1*X1 + beta_2*X2 + eps
par(mfrow=c(1,1))
plot(Y)
```



Part b)

```
# initialize beta hat 1
bhat_1 <- 1
```

Part c)

```
a <- Y - bhat_1*X1
(bhat_2 <- lm(a~X2)$coef[2])
```

```
##       X2
## 4.047166
```

Part d)

```
a <- Y - bhat_2*X2
(bhat_1 <- lm(a~X1)$coef[2])
```

```
##        X1
## 0.3211108
```

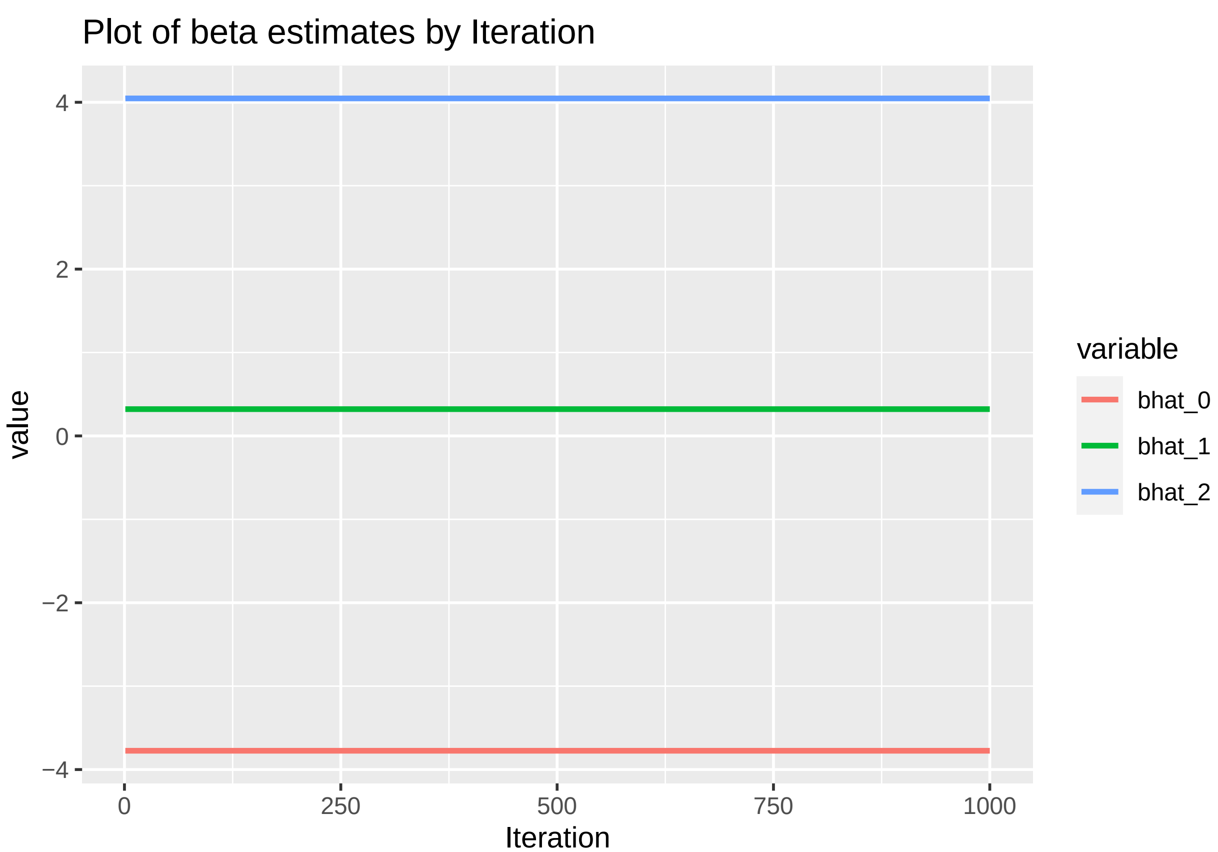
Part e)

```
bhat_0 <- bhat_1 <- bhat_2 <- rep(0, 1000)
for (i in 1:1000) {
  a <- Y - bhat_1[i] * X1
  bhat_2[i] <- lm(a ~ X2)$coef[2]
  a <- Y - bhat_2[i] * X2
  bhat_0[i] <- lm(a ~ X1)$coef[1]
  # bhat_1 will end up with 1001 terms
```

```r
  bhat_1[i+1] <- lm(a ~ X1)$coef[2]
}

# make plots
require(ggplot2)
require(reshape2)
```

```
##          : reshape2
```

```r
mydf <- data.frame(Iteration=1:1000, bhat_0, bhat_1=bhat_1[-1], bhat_2)
mmydf <- melt(mydf, id.vars="Iteration")
ggplot(mmydf, aes(x=Iteration, y=value, group=variable, col=variable)) +
  geom_line(size=1) + ggtitle("Plot of beta estimates by Iteration")
```



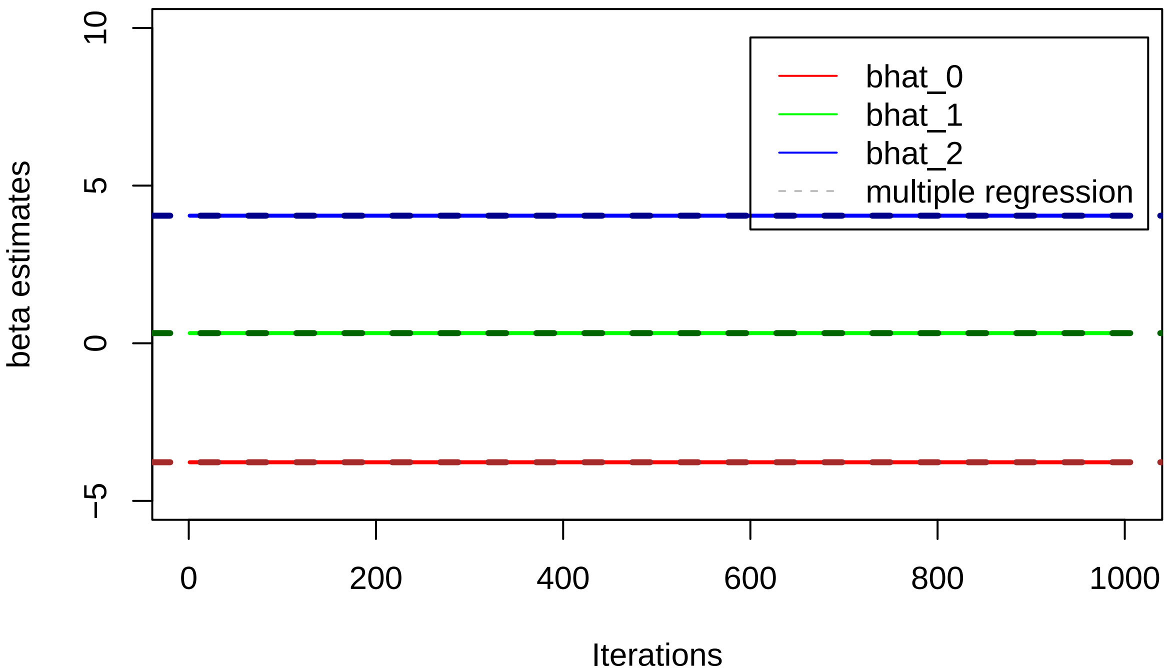
Part f)

```r
fit.lm <- lm(Y ~ X1 + X2)
coef(fit.lm)
```

```
## (Intercept)          X1          X2
##  -3.7746466   0.3211102   4.0465332
```

```r
plot(bhat_0, type="l", col="red", lwd=2, xlab="Iterations",
     ylab="beta estimates", ylim=c(-5,10))
lines(bhat_1[-1], col="green", lwd=2)
```

```
lines(bhat_2, col="blue", lwd=2)
abline(h=coef(fit.lm)[1], lty="dashed", lwd=3, col="brown")
abline(h=coef(fit.lm)[2], lty="dashed", lwd=3, col="darkgreen")
abline(h=coef(fit.lm)[3], lty="dashed", lwd=3, col="darkblue")
legend(x=600,y=9.7, c("bhat_0", "bhat_1", "bhat_2", "multiple regression"),
       lty = c(1,1,1,2),
       col = c("red","green","blue","gray"))
```



Part g)

```
head(mydf)
```

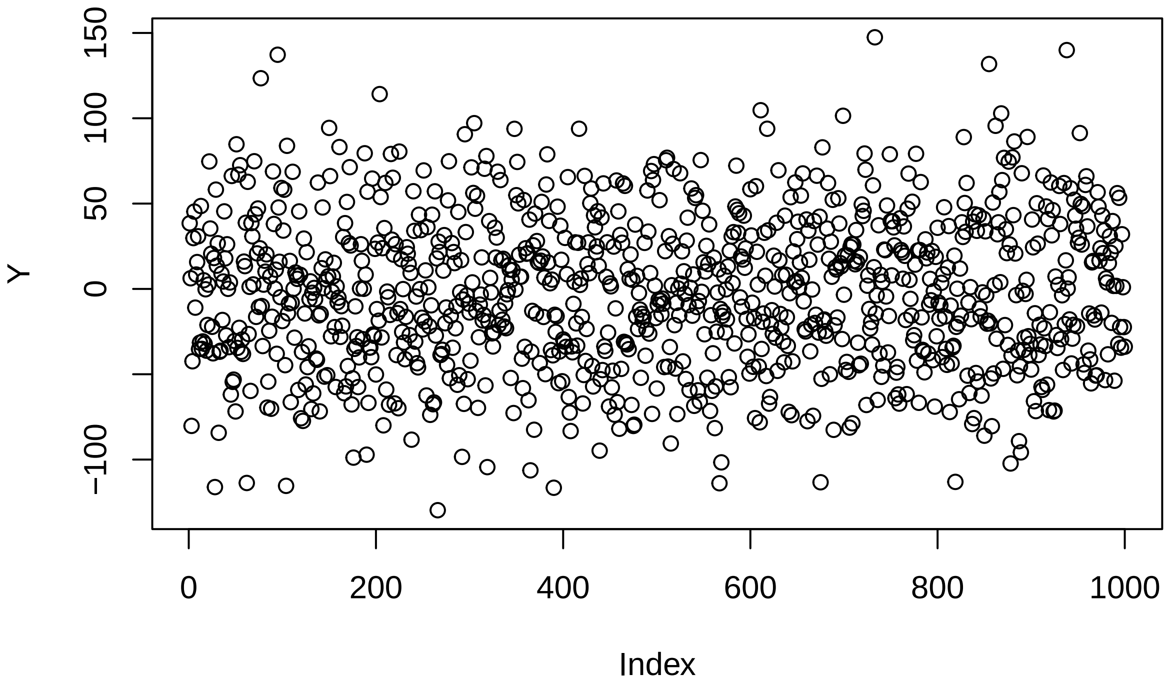
```
##   Iteration    bhat_0    bhat_1   bhat_2
## 1         1 -3.774658 0.3211098 4.046234
## 2         2 -3.774647 0.3211102 4.046533
## 3         3 -3.774647 0.3211102 4.046533
## 4         4 -3.774647 0.3211102 4.046533
## 5         5 -3.774647 0.3211102 4.046533
## 6         6 -3.774647 0.3211102 4.046533
```

One iteration seemed to be enough to get a decent fit. After iteration 2, the beta estimates already converged.

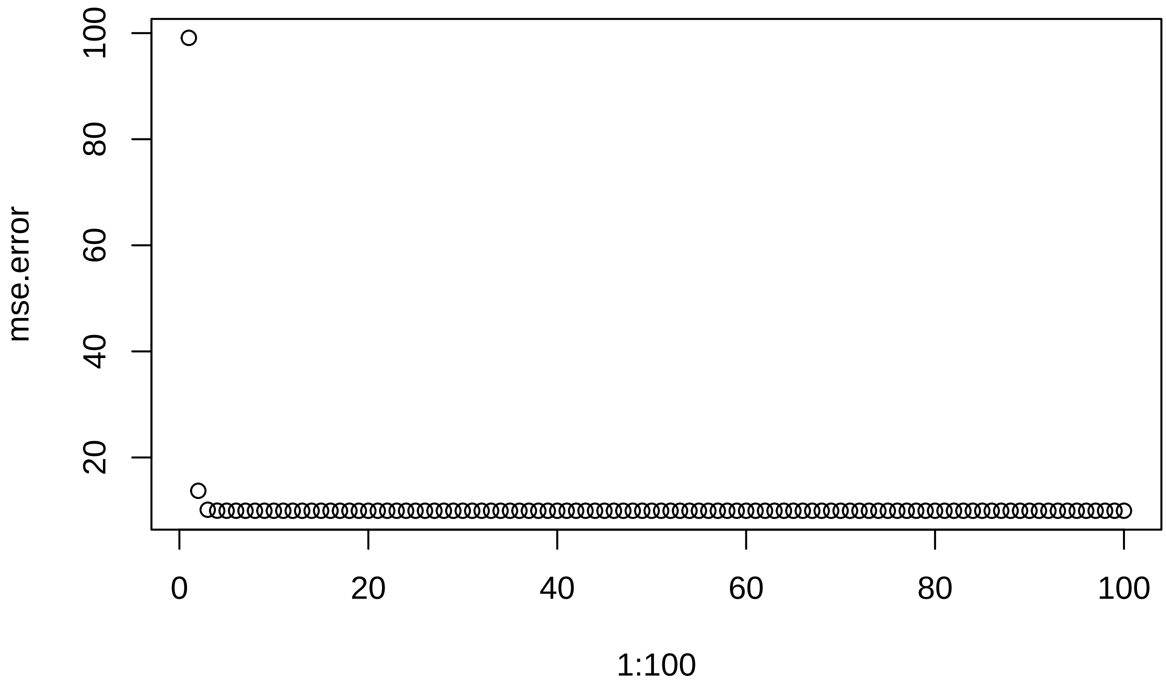
EXERCISE 12:

```
set.seed(1)

# create toy example with 100 predictors
p <- 100    # number of true predictors
```

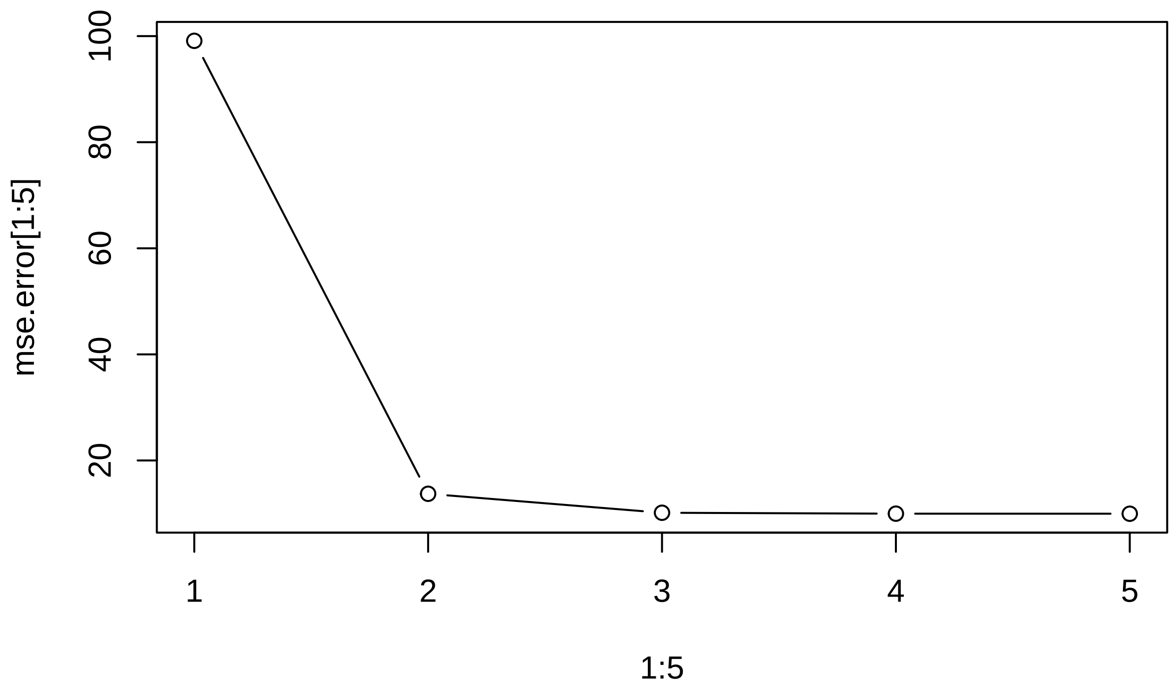
```
n <- 1000  # number of observations
betas <- rnorm(p+1)*5  # extra 1 for beta_0
X <- matrix(rnorm(n*p), ncol=p, nrow=n)
eps <- rnorm(n, sd=0.5)
Y <- betas[1] + (X %*% betas[-1]) + eps  # betas will repeat n times
par(mfrow=c(1,1))
plot(Y)
```




```
# find coef estimates with multiple regression
fit.lm <- lm(Y~X)
bhats.lm <- coef(fit.lm)

# run backfitting with 100 iterations
bhats <- matrix(0, ncol=p, nrow=100)
mse.error <- rep(0, 100)
for (i in 1:100) {
  for (k in 1:p) {
    a = Y - (X[,-k] %*% bhats[i,-k])
    bhats[i:100,k] = lm(a ~ X[,k])$coef[2]
  }
  mse.error[i] <- mean((Y - (X %*% bhats[i,]))^2)
}
plot(1:100, mse.error)
```




```
plot(1:5, mse.error[1:5], type="b")
```




```
# second iteration results were very close to multiple regression
```