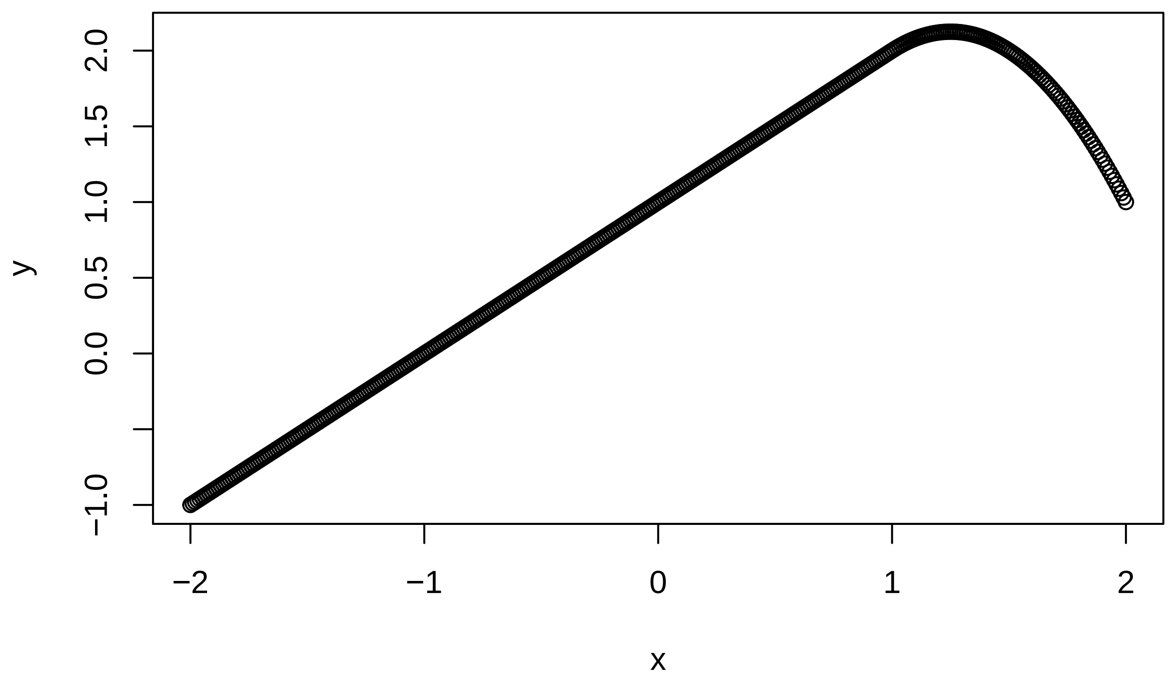
MA679Hw5

Siwei Hu

February 18, 2019

3

```
x = seq(-2,2,0.01)
y = 1 + x + -2 * (x-1)^2 * I(x>1)
plot(x, y)
```



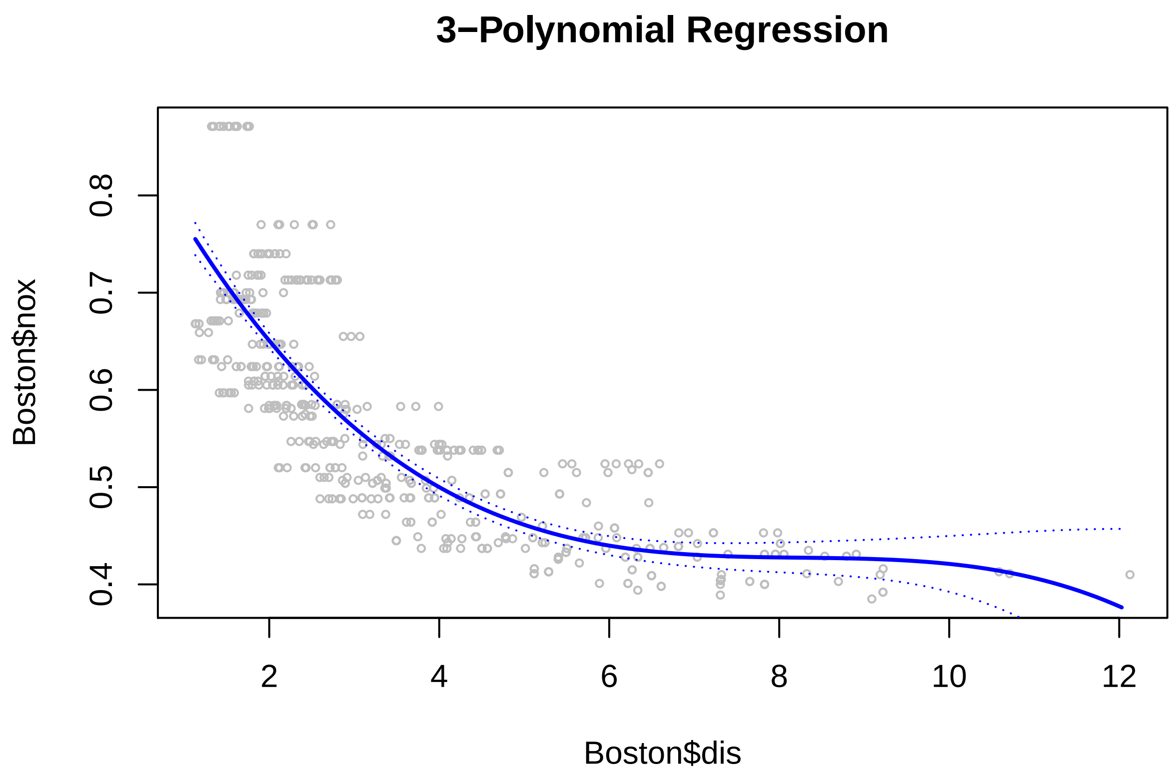
##7.9 ###(a)

```
fit1 <- lm(nox~poly(dis,3),data = Boston)
summary(fit1)
```

```
##
## 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:
```

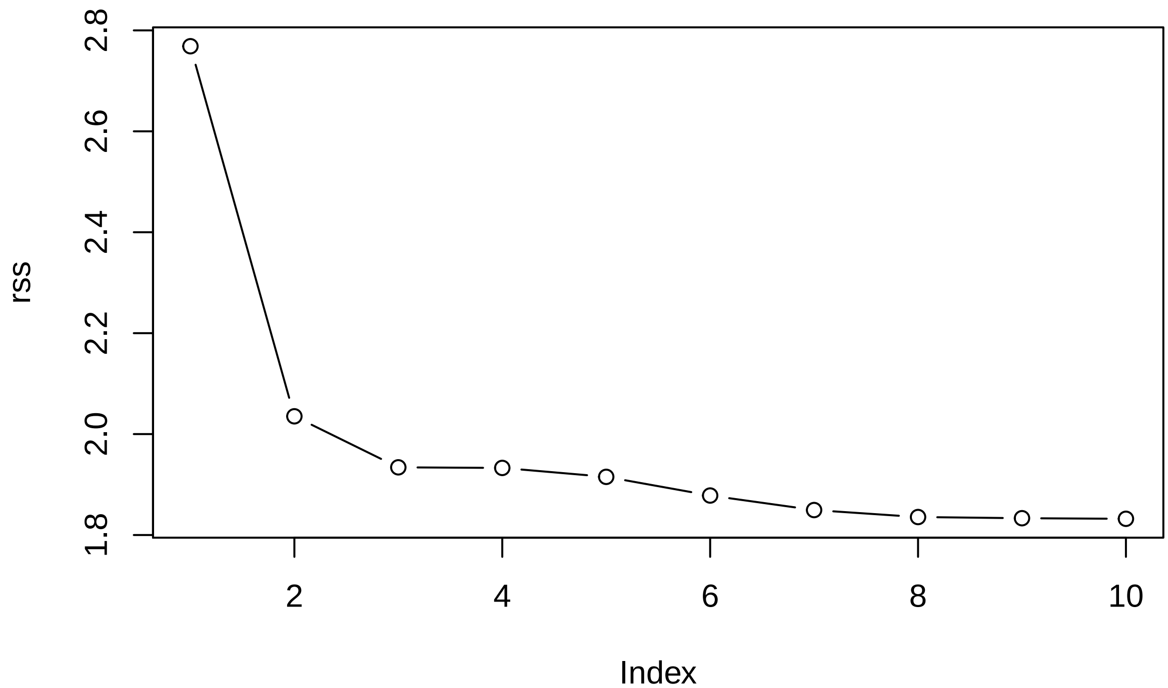
```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
## ---
## 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
```

```
dislims <- range(Boston$dis)
dis.grid <- seq(dislims[1], dislims[2], 0.1)
pred1 <- predict(fit1,newdata = list(dis = dis.grid),se = TRUE)

se.band <-cbind(pred1$fit + 2*pred1$se.fit,pred1$fit - 2*pred1$se.fit)

plot(x = Boston$dis,y = Boston$nox, xlim = dislims, cex = 0.5, col = 'grey')
title("3-Polynomial Regression")
lines(dis.grid, pred1$fit, lwd=2, col="blue")
matlines(dis.grid, se.band, lwd=1, col="blue", lty=3)
```



(b)

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

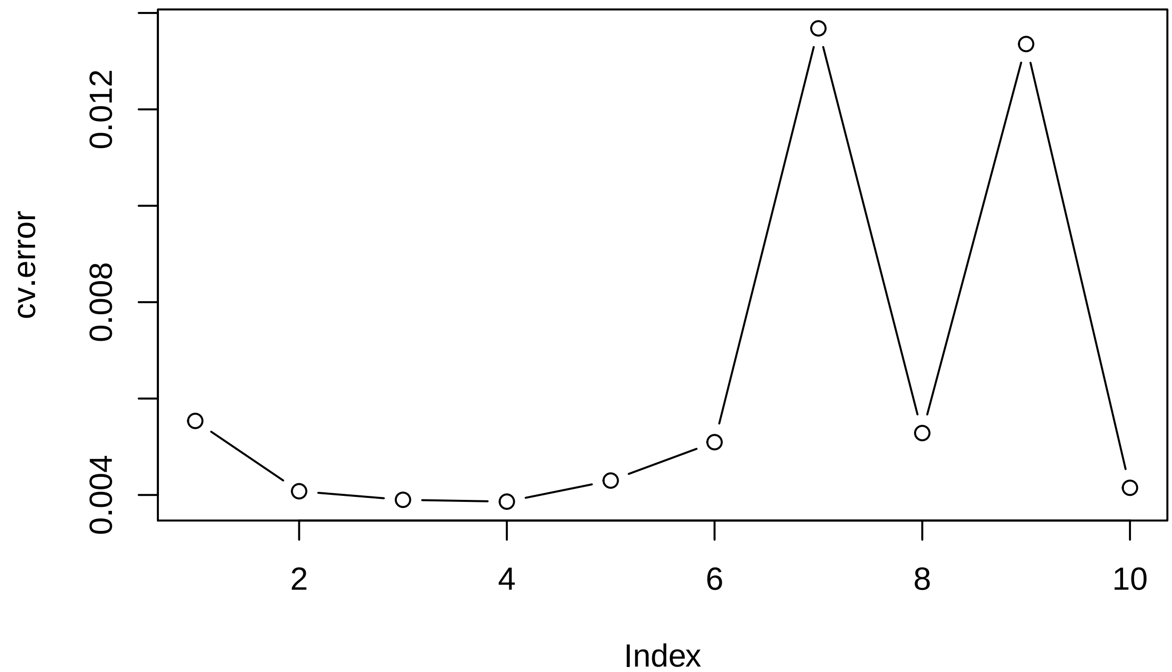


(C)

```r
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]
}
cv.error
```

```
##  [1] 0.005536329 0.004077147 0.003899587 0.003862127 0.004298590
##  [6] 0.005095283 0.013680327 0.005284520 0.013355413 0.004148996
```

```r
plot(cv.error, type = 'b')
```



It's better to choose degree = 4, ACCORDING to cross-validation, the 4-polynomial model has the lowest average RSS than others. So we choose 4th degree.

(D)

```
fit2 <-lm(nox ~ bs(dis,df = 4),data = Boston)
summary(fit2)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, df = 4), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.124622 -0.039259 -0.008514  0.020850  0.193891
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.73447    0.01460  50.306  < 2e-16 ***
## bs(dis, df = 4)1 -0.05810    0.02186  -2.658  0.00812 **
## bs(dis, df = 4)2 -0.46356    0.02366 -19.596  < 2e-16 ***
## bs(dis, df = 4)3 -0.19979    0.04311  -4.634 4.58e-06 ***
## bs(dis, df = 4)4 -0.38881    0.04551  -8.544  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.06195 on 501 degrees of freedom
## Multiple R-squared:  0.7164, Adjusted R-squared:  0.7142
## F-statistic: 316.5 on 4 and 501 DF,  p-value: < 2.2e-16
```

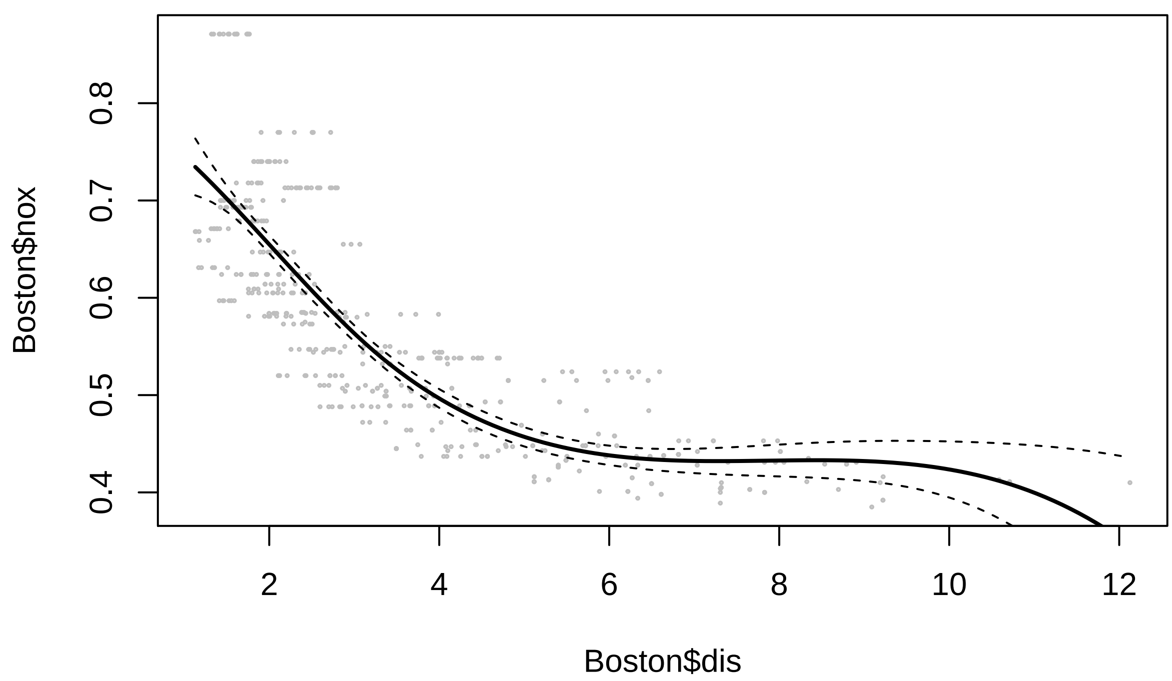
```
dim(bs(Boston$dis,df = 4))
```

```
## [1] 506   4
```

```
attr(bs(Boston$dis,df = 4),"knots")
```

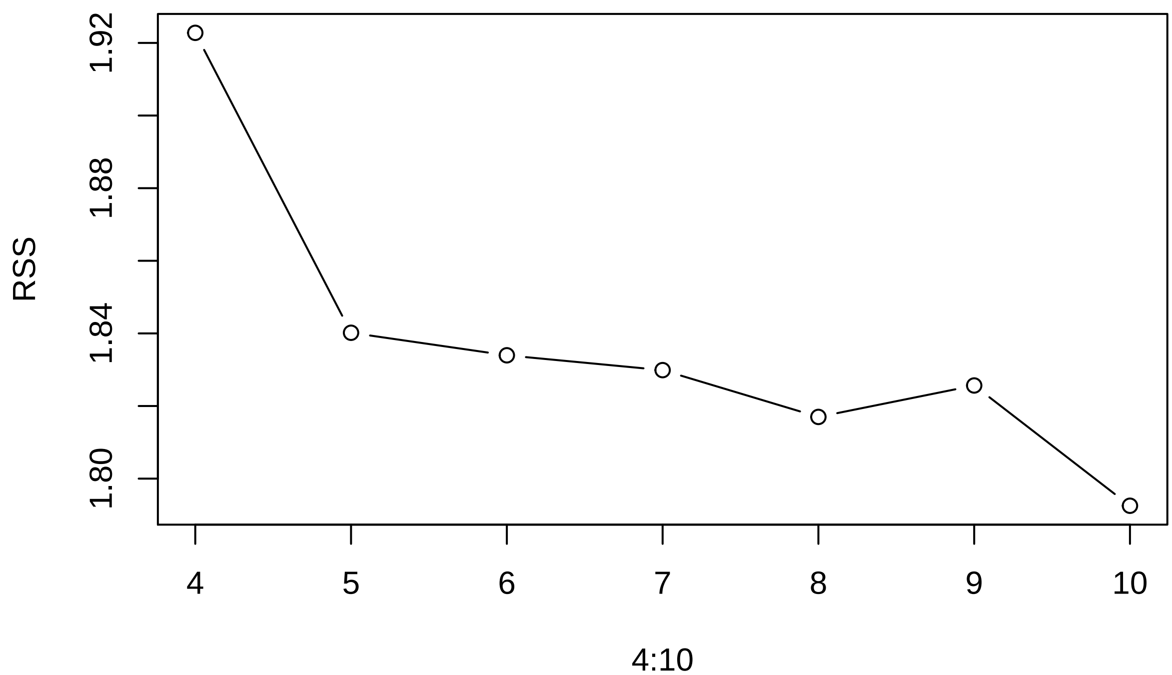
```
##     50%
## 3.20745
```

```
pred2 <- predict(fit2, newdata = list(dis = dis.grid), se = T)
plot(x = Boston$dis, y = Boston$nox, cex = 0.2, col = "grey")
lines(dis.grid,pred2$fit, lwd = 2)
lines(dis.grid, pred2$fit + 2*pred2$se, lty = "dashed")
lines(dis.grid,pred2$fit - 2*pred2$se,lty = "dashed")
```



(E)

```
RSS <- rep(0,7)
for (i in 4:10){
  glm.bs <- lm(nox~bs(dis,i),data = Boston)
  RSS[i-3] <- sum(glm.bs$residuals^2)
}

plot(4:10,RSS, type = "b")
```



(F)

```
set.seed(1)
cv.df <- rep(NA,7)
for(i in 4:10){
  glm.bs <- lm(nox~ bs(dis,df = i),data = Boston)
  cv <- cv.glm(Boston,glm.bs, K = 10)
  cv.df <- cv$delta[2]
}

#plot(x = 4:10, y = cv.df, type = "b")
```

10

(a)

```
train <- sample(1:nrow(College), nrow(College)/2)
train.c <- College[train,]
test.c <- College[-train,]

fitreg.fwd <- regsubsets(Outstate~., data = train.c, nvmax = 17, method = "forward")
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 406 linear dependencies found
```

```
## Reordering variables and trying again:
```

```r
fwd.summary <- summary(fitreg.fwd)
```

```r
reg.fit = regsubsets(Outstate ~ ., data = College, method = "forward")
```

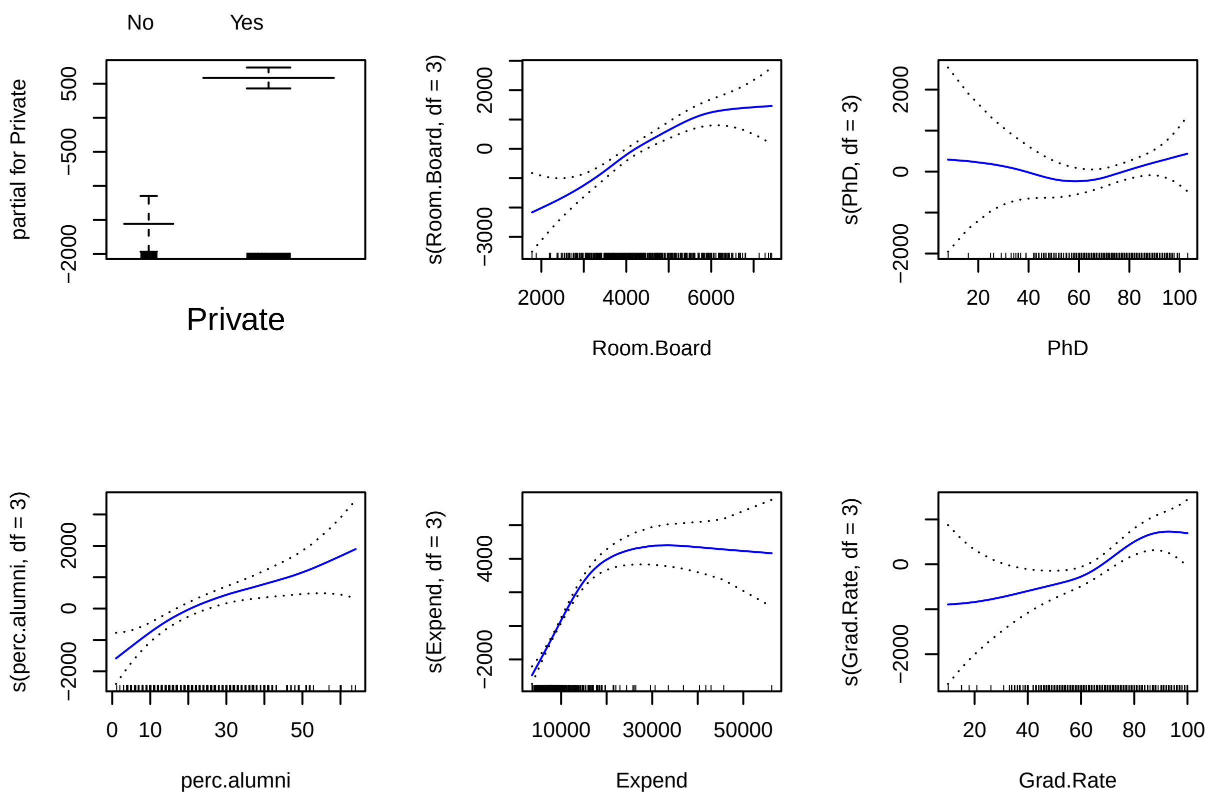
```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 17 linear dependencies found
```

```r
coefi = coef(reg.fit, id = 6)
names(coefi)
```

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

(b)

```r
gam.fit = gam(Outstate ~ Private + s(Room.Board, df = 3) + s(PhD, df = 3) +
    s(perc.alumni, df = 3) + s(Expend, df = 3) + s(Grad.Rate, df = 3), data = train.c)
par(mfrow = c(2, 3))
plot(gam.fit, se = T, col = "blue")
```




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

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 3) + s(PhD,
##     df = 3) + s(perc.alumni, df = 3) + s(Expend, df = 3) + s(Grad.Rate,
##     df = 3), data = train.c)
## Deviance Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -6314.65 -1338.76   -27.08  1275.75  7274.90
##
## (Dispersion Parameter for gaussian family taken to be 3830647)
##
##     Null Deviance: 6831988270 on 387 degrees of freedom
## Residual Deviance: 1421168136 on 370.9995 degrees of freedom
## AIC: 7001.228
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                         Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                  1 1702407412 1702407412 444.418 < 2.2e-16 ***
## s(Room.Board, df = 3)    1 1147025080 1147025080 299.434 < 2.2e-16 ***
## s(PhD, df = 3)           1  383687698  383687698 100.163 < 2.2e-16 ***
## s(perc.alumni, df = 3)   1  325348817  325348817  84.933 < 2.2e-16 ***
## s(Expend, df = 3)        1  625156051  625156051 163.199 < 2.2e-16 ***
## s(Grad.Rate, df = 3)     1   47208476   47208476  12.324 0.0005022 ***
## Residuals              371 1421168136    3830647
## ---
## 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 = 3)        2  2.662 0.07116 .
## s(PhD, df = 3)               2  1.250 0.28780
## s(perc.alumni, df = 3)       2  1.486 0.22773
## s(Expend, df = 3)            2 43.135 < 2e-16 ***
## s(Grad.Rate, df = 3)         2  2.329 0.09881 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(c)

```r
pred.gam <- predict(gam.fit, newdata = test.c)
err.gam <- mean((test.c$Outstate - pred.gam)^2)

SS.tot <- mean((test.c$Outstate - mean(test.c$Outstate))^2)
rss <- 1- err.gam/SS.tot
rss
```

```
## [1] 0.7744624
```

(D)

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

```
##
## Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 3) + s(PhD,
##     df = 3) + s(perc.alumni, df = 3) + s(Expend, df = 3) + s(Grad.Rate,
##     df = 3), data = train.c)
```

```
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -6314.65 -1338.76   -27.08  1275.75  7274.90
##
## (Dispersion Parameter for gaussian family taken to be 3830647)
##
##     Null Deviance: 6831988270 on 387 degrees of freedom
## Residual Deviance: 1421168136 on 370.9995 degrees of freedom
## AIC: 7001.228
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                         Df     Sum Sq    Mean Sq F value    Pr(>F)
## Private                  1 1702407412 1702407412 444.418 < 2.2e-16 ***
## s(Room.Board, df = 3)    1 1147025080 1147025080 299.434 < 2.2e-16 ***
## s(PhD, df = 3)           1  383687698  383687698 100.163 < 2.2e-16 ***
## s(perc.alumni, df = 3)   1  325348817  325348817  84.933 < 2.2e-16 ***
## s(Expend, df = 3)        1  625156051  625156051 163.199 < 2.2e-16 ***
## s(Grad.Rate, df = 3)     1   47208476   47208476  12.324 0.0005022 ***
## Residuals              371 1421168136    3830647
## ---
## 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 = 3)        2  2.662 0.07116 .
## s(PhD, df = 3)               2  1.250 0.28780
## s(perc.alumni, df = 3)       2  1.486 0.22773
## s(Expend, df = 3)            2 43.135 < 2e-16 ***
## s(Grad.Rate, df = 3)         2  2.329 0.09881 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From summary of gam.fit, we do anova for nonparametric Effects to compare about five different predictor's non-linear relationship with response. From p-value, we know there is strong non-linear relationship between Expend and response. And phd and response have moderately non linear relationship.

11

(a)

```r
x1 <- rnorm(100)
x2 <- rnorm(100)
eps <- rnorm(100,sd = 0.1)

Y = 5 + 4*x1 + 3*x2 + eps
```

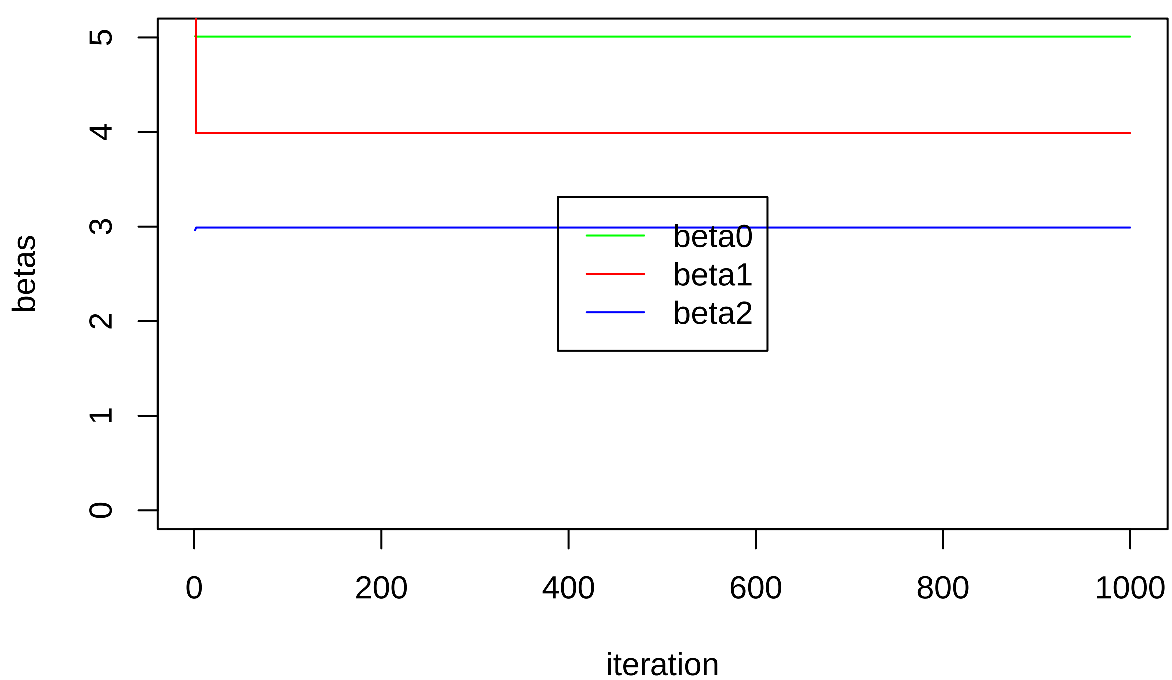
(b)

```r
beta0 <- rep(NA,1000)
beta1 <- rep(NA,1000)
```

```
beta2 <- rep(NA,1000)

beta1[1] <- 9
```

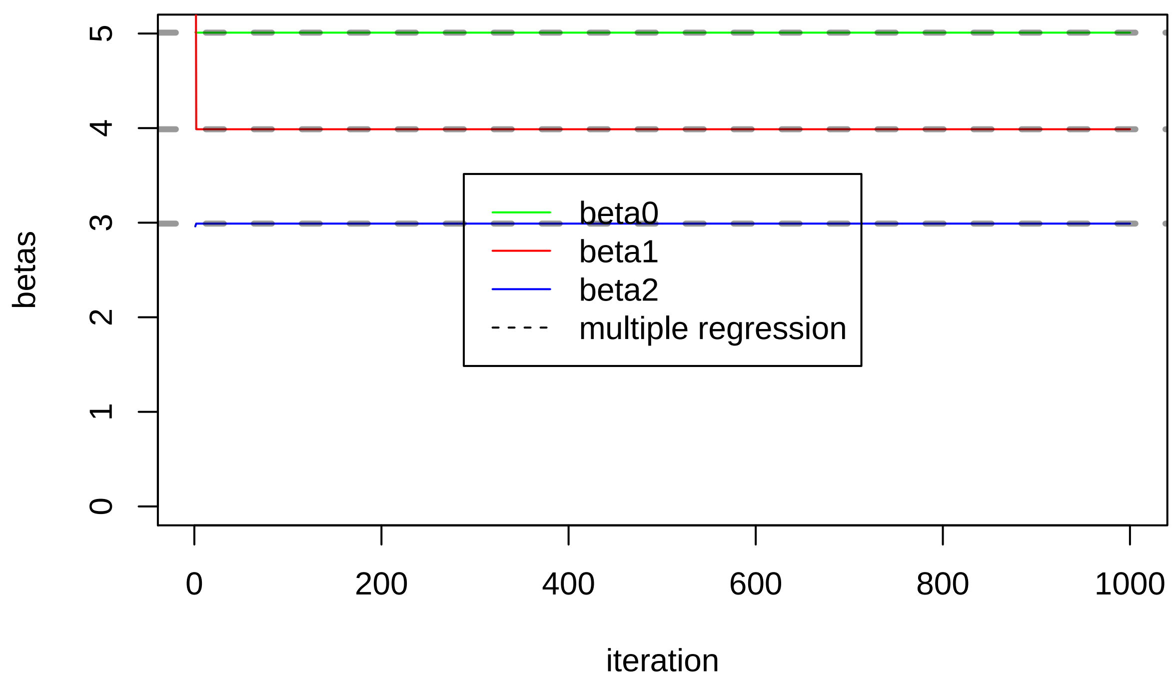
(c)(d)(e)

```
for(i in 1:1000){
  a <- Y - beta1[i]*x1
  fit.lm1<- lm(a~x2)
  beta2[i] <- fit.lm1$coeff[2]
  b <- Y - beta2[i]*x2
  fit.lm2 <- lm(b~x1)
  if(i < 1000){
  beta1[i+1] <- fit.lm2$coef[2]
  }
  beta0[i] <- fit.lm2$coef[1]
}
plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(0, 5), col = "green")
lines(1:1000, beta1, col = "red")
lines(1:1000, beta2, col = "blue")
legend("center", c("beta0", "beta1", "beta2"), lty = 1, col = c("green", "red", "blue"))
```



###(f) Dotted lines show that the estimated multiple regression coefficients match exactly with the coefficients obtained using backfitting.

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



(g)

When the relationship between Y and X's is linear, one iteration is sufficient to attain a good approximation of true regression coefficients.