Hw6

Abdullah Farouk

2017-11-07

To do this assignment I call upon the following packages

```
#install.packages('np')
#install.packages('OpenMx')  #To create diagnol matrices
library('OpenMx')
```

```
## OpenMx is not compiled to take advantage of computers with multiple cores.
```

```
library('ggplot2')
library('np')
```

```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-3)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

```
data("cps71")
```

Question 2

Here I specify my initial conditions

```
age <- cps71$age   #Values of x
wage <- cps71$logwage
n <-length(age)    #Determine the number of rows in z_x
h <- nrow(cps71)^(-1/5)
x <- c(25, 35, 50)
```

I have written code for each of of the estimators in seperate portions. (P = 0) corresponds to fitted values obtained from the local linear estimator, whilst (p = 1) from the local linear and (p = 2) from the local quadratic estimator respectively.

```
#When P = 0
# Create  Z matrix
z_x <- matrix(nrow = n, ncol = 1)
z_x[,1] <- rep(1, n)
betas_0 <- c()


#Calculate weights
for ( i in x) {
  deviation <- c(age - i)
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0 <- as.numeric((solve((t(z_x)%*%w_x%*%z_x))%*%t(z_x)%*%w_x%*%wage))
  betas_0 <- c(betas_0, b_hat_0)
}
```

```r
#When P = 1
# Create  Z matrix
z_x_ll <- matrix(nrow = n, ncol = 2)
z_x_ll[,1] <- rep(1, n)
betas_0_ll <- c()
beta_hat_1 <- c()
e1 <- c(1,0)

for ( i in x) {
  deviation <- c(age - i)
  z_x_ll[,2] <- deviation
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0_ll <- as.numeric(e1 %*% (solve((t(z_x_ll) %*%
                w_x %*% z_x_ll)) %*% t(z_x_ll) %*% w_x %*% wage))
  betas_0_ll <- c(betas_0_ll, b_hat_0_ll)
  beta_hat_1 <- c(beta_hat_1, as.numeric((solve((t(z_x_ll) %*% w_x %*%
                z_x_ll)) %*% t(z_x_ll) %*% w_x %*% wage)))
}

#P = 2
 # Create  Z matrix
z_x_lq <- matrix(nrow = n, ncol = 3)
z_x_lq[,1] <- rep(1, n)
betas_0_lq <- c()
e1q <- c(1,0,0)
beta_hat_2 <- c()

for ( i in x) {
  deviation <- c(age - i)
  z_x_lq[,2] <- deviation
  z_x_lq[,3] <- deviation^2
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0_lq <- as.numeric(e1q %*% (solve((t(z_x_lq) %*%
                w_x %*% z_x_lq)) %*% t(z_x_lq) %*% w_x %*% wage))
  betas_0_lq <- c(betas_0_lq, b_hat_0_lq)
  beta_hat_2 <- c(beta_hat_2, as.numeric((solve((t(z_x_lq) %*%
                w_x %*% z_x_lq)) %*% t(z_x_lq) %*% w_x %*% wage)))
}

#Reporting Betas
Loc_kernel <- as.data.frame(betas_0)   #Betas when p = 0
Loc_linear <- as.data.frame(beta_hat_1, row.names = c("Beta 0 when x = 25",
"Beta 1 when x = 25", "Beta 0 when x = 35", "Beta 1 when x = 35",
"Beta 0 when x = 50", "Beta 1 when x = 50"))  #Betas when p = 1
Loc_quadratic <- as.data.frame(beta_hat_2, row.names = c("Beta 0 when x = 25",
"Beta 1 when x = 25", "Beta 2 when x = 25", "Beta 0 when x = 35",
"Beta 1 when x = 35", "Beta 2 when x = 35", "Beta 0 when x = 50",
"Beta 1 when x = 50", "Beta 2 when x = 50"))  #Betas when p = 1
Loc_kernel
```

```
##    betas_0
## 1 13.38221
```

```
## 2 13.91700
## 3 13.97354
```

```r
Loc_linear
```

```
##                       beta_hat_1
## Beta 0 when x = 25 13.3837636
## Beta 1 when x = 25  0.2144434
## Beta 0 when x = 35 13.9170023
## Beta 1 when x = 35  0.1111487
## Beta 0 when x = 50 13.9732874
## Beta 1 when x = 50 -0.1199316
```

```r
Loc_quadratic
```

```
##                       beta_hat_2
## Beta 0 when x = 25 13.3868399
## Beta 1 when x = 25  0.1837910
## Beta 2 when x = 25 -0.1195553
## Beta 0 when x = 35 13.9217498
## Beta 1 when x = 35  0.1111471
## Beta 2 when x = 35 -0.2431643
## Beta 0 when x = 50 13.9743428
## Beta 1 when x = 50 -0.1817528
## Beta 2 when x = 50 -0.1865389
```

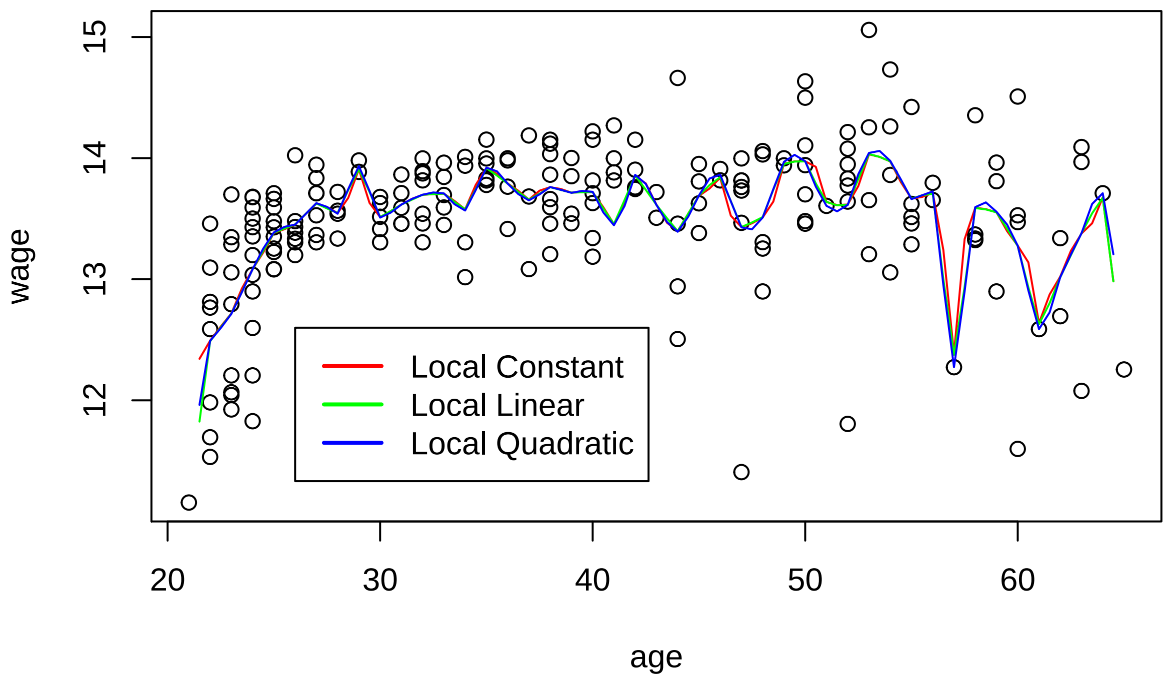
```r
#Reporting fitted values
fitted_values <- rbind(betas_0, betas_0_ll, betas_0_lq)
colnames(fitted_values) <- x
rownames(fitted_values) <- c('Local Constant Estimator', 'Local Linear Estimator',
                             'Local Quadratic Estimator')
fitted_values
```

```
##                                 25       35       50
## Local Constant Estimator  13.38221 13.91700 13.97354
## Local Linear Estimator    13.38376 13.91700 13.97329
## Local Quadratic Estimator 13.38684 13.92175 13.97434
```

Over here I compute and plot the fitted values for all ages for all three estimators.

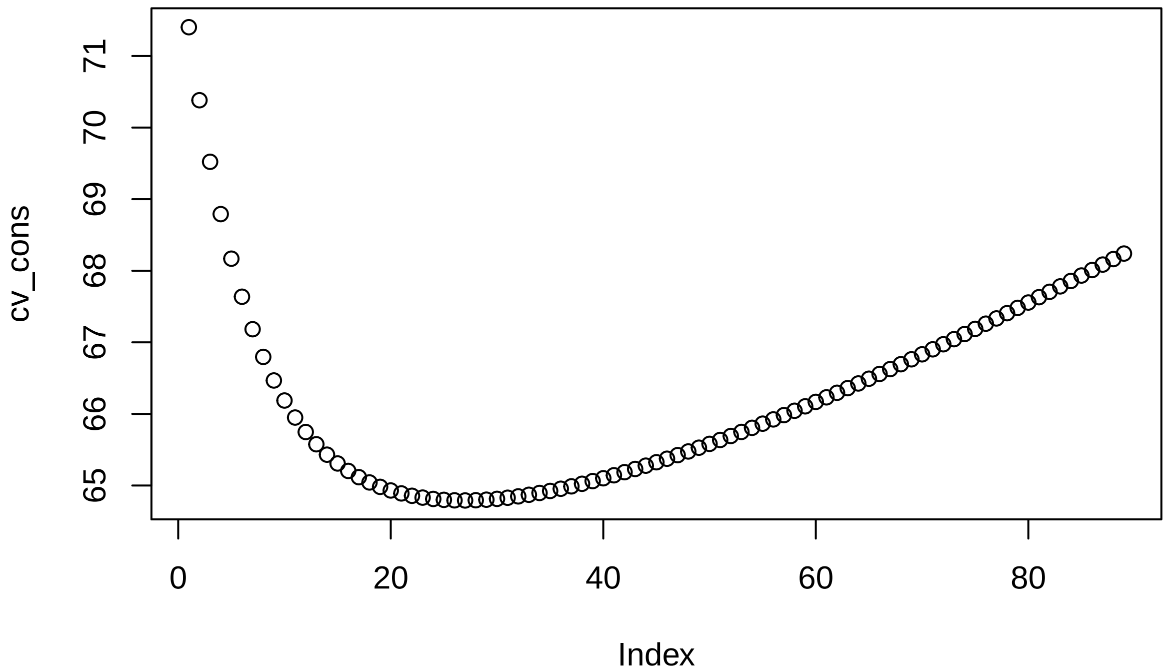
```r
#Plot for all x's when P = 0
betas_0_1 <-c()
x_1 <- seq(min(age) + 0.5, max(age) - 0.5, by = 0.5)

for ( i in x_1) {
  deviation <- c(age - i)
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0_1 <- as.numeric((solve((t(z_x)%*%w_x%*%z_x))%*%t(z_x)%*%w_x%*%wage))
  betas_0_1 <- c(betas_0_1, b_hat_0_1)
}


#Plot for all x when P = 1
betas_0_ll_1 <- c()

for ( i in x_1) {
```

```
  deviation <- c(age - i)
  z_x_ll[,2] <- deviation
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0_ll_1 <- as.numeric(e1 %*% (solve((t(z_x_ll) %*% w_x %*%
                z_x_ll)) %*% t(z_x_ll) %*% w_x %*% wage))
  betas_0_ll_1 <- c(betas_0_ll_1, b_hat_0_ll_1)
}

 #Plot for all x when P = 2
betas_0_lq_1 <- c()

 for (i in x_1) {
  deviation <- c(age - i)
  z_x_lq[,2] <- deviation
  z_x_lq[,3] <- deviation^2
  weights <- (1/h)*dnorm(x = (deviation/h), mean = 0, sd = 1)
  w_x <- vec2diag(x = weights)
  b_hat_0_lq_1 <- as.numeric(e1q %*% (solve((t(z_x_lq) %*% w_x %*%
                z_x_lq)) %*% t(z_x_lq) %*% w_x %*% wage))
  betas_0_lq_1 <- c(betas_0_lq_1, b_hat_0_lq_1)
}


#Plot all of the estimators on the same graph
{plot(age, wage)
lines(x=x_1, y=betas_0_1, col="red")
lines(x=x_1, y=betas_0_ll_1, col="green")
lines(x=x_1, y=betas_0_lq_1, col="blue")
legend(x = 26, y = 12.6, col = c("red", "green", "blue"), lwd = c(2,
2, 2), c("Local Constant", "Local Linear", "Local Quadratic"),
bg = "white")}
```

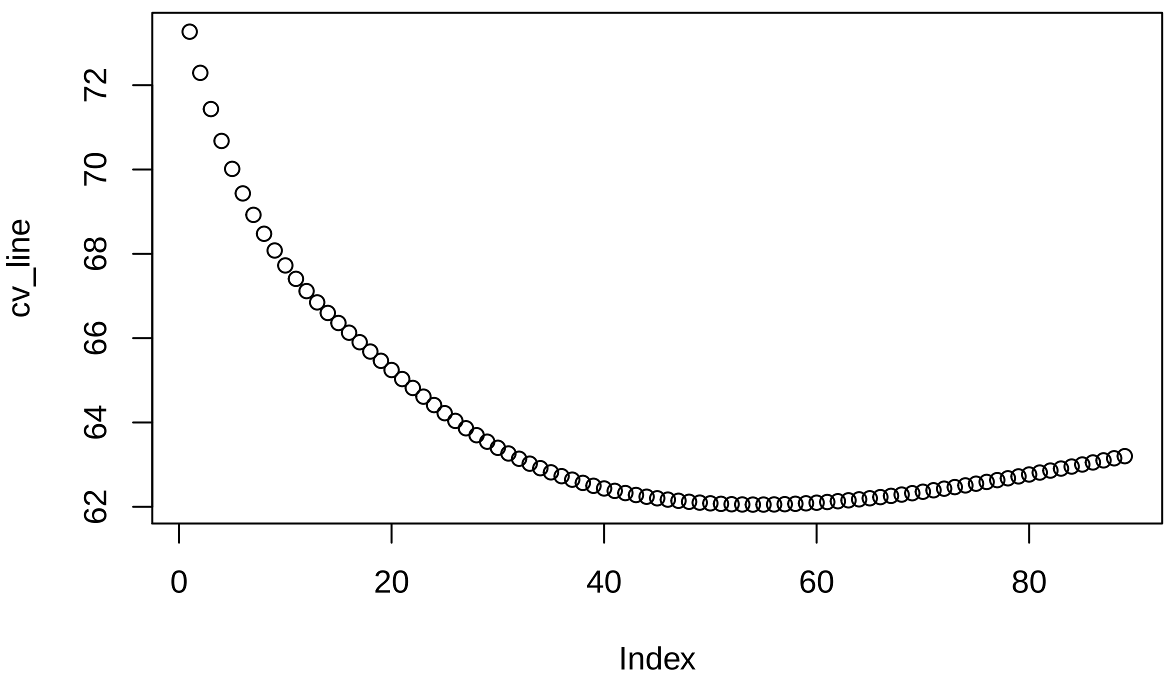


Question 3

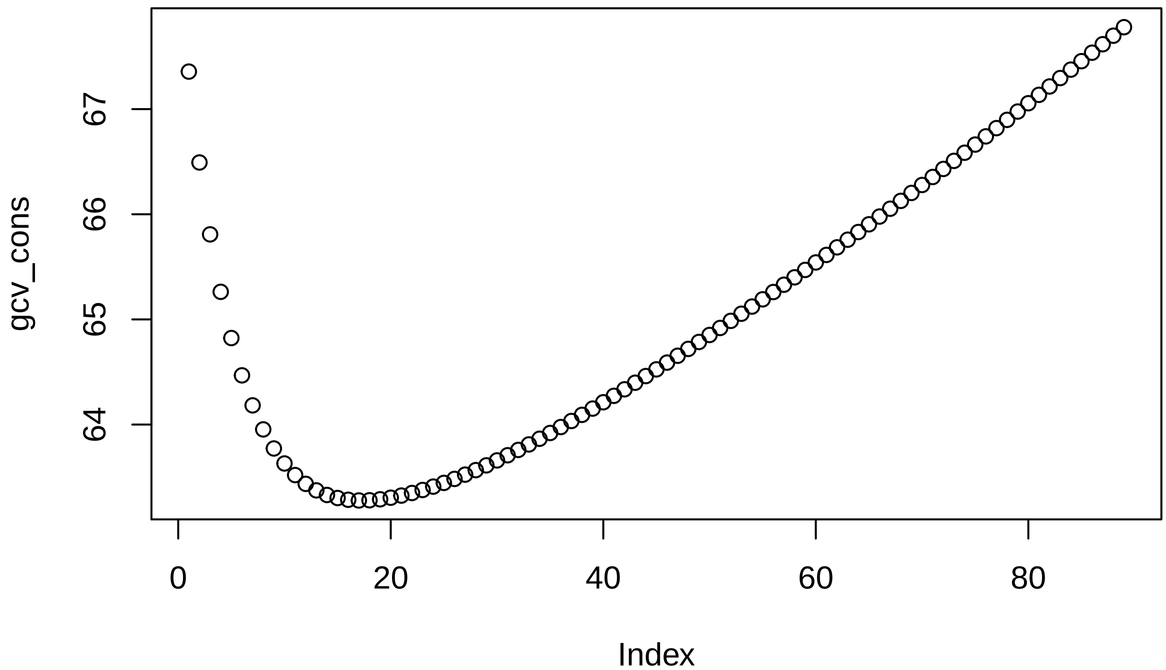
```
#initialize matrices
library(ggplot2)
z_x_lc <- matrix(nrow = n, ncol = 1)
z_x_lc[,1] <- rep(1, n)
z_x_ll <- matrix(nrow = n, ncol = 2)
z_x_ll[,1] <- rep(1, n)

L_lc <- matrix(nrow = n, ncol = n)
L_ll <- matrix(nrow = n, ncol = n)
cv_lc <- c()
cv_ll <- c()
e1 <- c(1,0)
h_1 <- seq(0.6,5, by = 0.05)
cv_cons <- c()
cv_line <- c()
gcv_cons <- c()
gcv_line <- c()

for (j in 1 : length(h_1)) {

  for (i in 1 : n) {
    z_x_ll[,2] <- c(age - age[i])
    deviation_1 <- c(age - age[i])
    weights_1 <- (1/h_1[j])*dnorm(x = (deviation_1/h_1[j]), mean = 0, sd = 1)
    w_x_1 <- vec2diag(x = weights_1)
    L_lc[i, ] <- as.numeric((solve((t(z_x_lc) %*% w_x_1 %*%
                 z_x_lc)) %*% t(z_x_lc) %*% w_x_1))
    L_ll[i, ] <- as.numeric(e1 %*% (solve((t(z_x_ll) %*%
                w_x_1 %*% z_x_ll)) %*% t(z_x_ll) %*% w_x_1))
    cv_lc[i] <- c(((wage[i] - (L_lc %*% wage)[i])/(1 - L_lc[i,i]))^2)
    cv_ll[i] <- c(((wage[i] - (L_ll %*% wage)[i])/(1 - L_ll[i,i]))^2)
  }
  cv_cons[j] <- sum(cv_lc)
  cv_line[j] <- sum(cv_ll)
  v_lc <- sum(diag(L_lc))
  v_ll <- sum(diag(L_ll))
  gcv_cons[j] <- c(sum(((wage - (L_lc%*%wage))/(1-(v_lc/n)))^2))
  gcv_line[j] <- c(sum(((wage - (L_ll%*%wage))/(1-(v_ll/n)))^2))

}

h_scores <- as.data.frame(rbind(cv_cons, cv_line, gcv_cons, gcv_line))

plot(cv_cons)
```



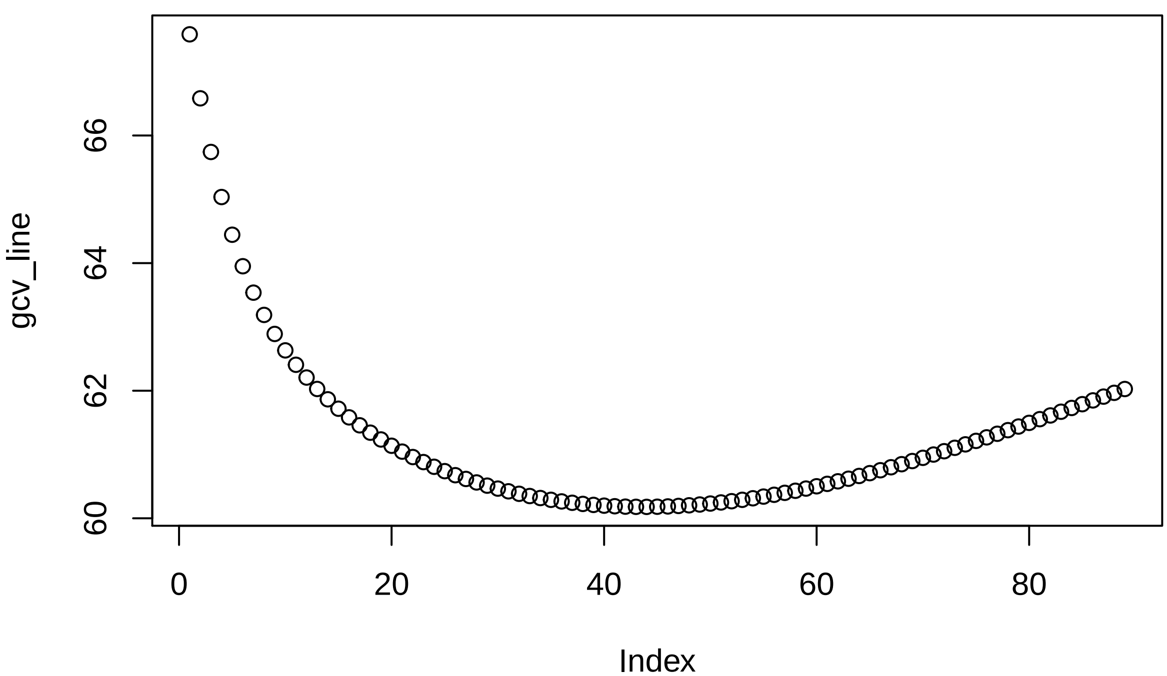

```
plot(cv_line)
```




```
plot(gcv_cons)
```




```
plot(gcv_line)
```

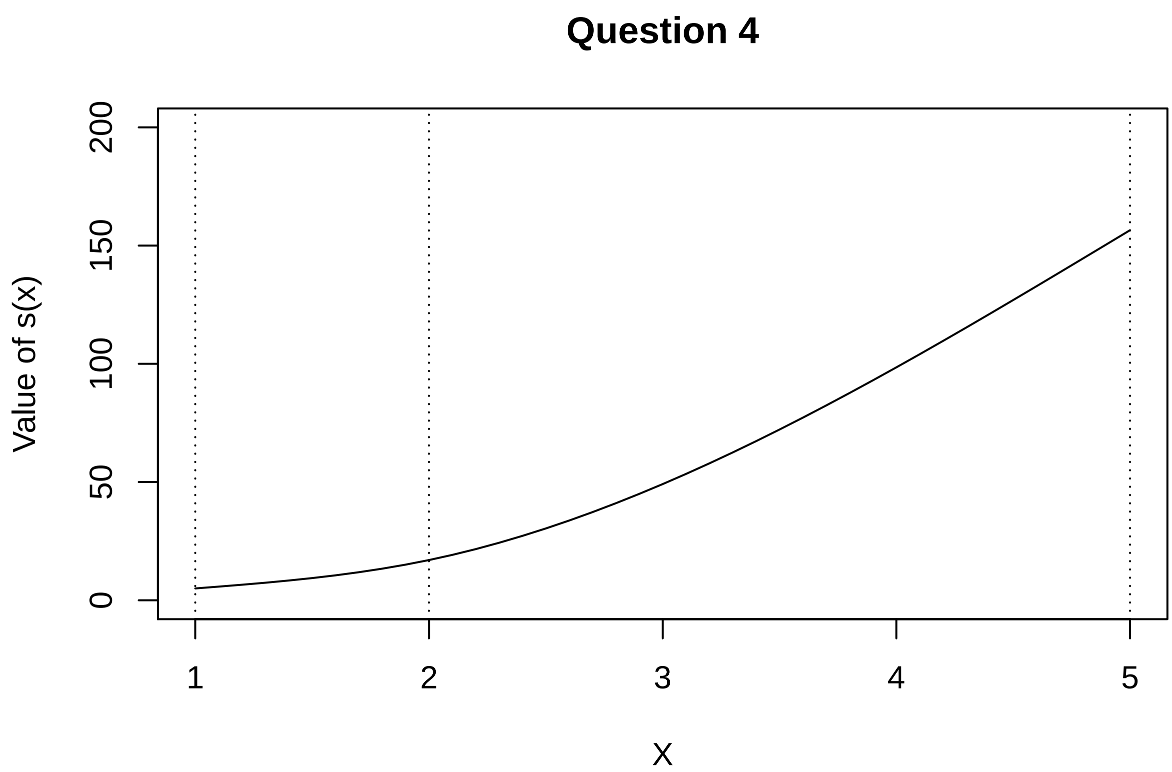



```
#Values of h that minimize each criterion
hh <- cbind(h_1[which.min(cv_cons)],h_1[which.min(cv_line)],
           h_1[which.min(gcv_cons)], h_1[which.min(gcv_line)])
colnames(hh) <- c('CV P = 0','CV P = 1',
                'GCV P = 0', 'GCV P = 1')
hh
```

```
##      CV P = 0 CV P = 1 GCV P = 0 GCV P = 1
## [1,]      1.9     3.25       1.4      2.75
```

Question 4 - Please refer to my written solutions for more details

```r
#Solving for coefficients. Collaborated with Eric Sanders to solve this.
gg <- c(5,5,17,17,156.5,156.5, 0,0,0,0,0,0)

s1 <- c(1,1,0,0,0,0,0,0,0,0,0,0)
s2 <- c(0,0,1,1,1,1,0,0,0,0,0,0)
s3 <- c(0,0,1,2,4,8,0,0,0,0,0,0)
s4 <- c(0,0,0,0,0,0,1,2,4,8,0,0)
s5 <- c(0,0,0,0,0,0,1,5,25,125,0,0)
s6 <- c(0,0,0,0,0,0,0,0,0,0,1,5)
s7 <- c(0,1,0,-1,-2,-3,0,0,0,0,0,0)
s8 <- c(0,0,0,1,4,12,0,-1,-4,-12,0,0)
s9 <- c(0,0,0,0,0,0,0,1,10,75,0,-1)
s10 <- c(0,0,0,0,2,12,0,0,-2,-12,0,0)
s11 <- c(0,0,0,0,0,0,0,0,2,30,0,0)
s12 <- c(0,0,0,0,2,6,0,0,0,0,0,0)

dq <- as.matrix(rbind(s1,s2,s3,s4,s5,s6,s7,s8, s9, s10, s11,s12))
beta_values <- solve(dq, gg)


#Plot graph
{plot(x = 2, y = 2, col = 'transparent' ,xlim = c(1, 5), ylim = c(0,200),
      xlab = 'X', ylab = "Value of s(x)", main = "Question 4")
lines(x = seq(1, 2, by = 0.1), y = beta_values[3] +
      beta_values[4] * seq(1,2, by = 0.1) +
        beta_values[5] * seq(1, 2, by = 0.1)^2 +
        beta_values[6] * seq(1, 2, by = 0.1)^3)
lines(x = seq(2, 5, by = 0.1), y = beta_values[7] +
      beta_values[8] * seq(2,5, by = 0.1) +
        beta_values[9] * seq(2, 5, by = 0.1)^2 +
        beta_values[10] * seq(2, 5, by = 0.1)^3)
abline(v = c(1, 2, 5), lty = 3)}
```


Question 4
Value of s(x)
0
50
100
150
200
1
2
3
4
5
X