Homework 5

Anbin Rhee

11/16/2021

Problem 1

I created R markdown.

Problem 2

(a)

When I try bootstrapping with this code, the code does not match the name of variable.

(b)

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(reshape)
```

```
##
## Attaching package: 'reshape'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```
## The following object is masked from 'package:dplyr':
##
##     rename
```

```
data <- read.delim("https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat", header = FALSE
data1 <- data.frame(cbind(data[1,], data[2,], data[3,]))
for (i in 2:10) {
  data1[i,] <- data.frame(cbind(data[3*i-2,], data[3*i-1,], data[3*i,]))
}
data1 <- data1[,colSums(is.na(data1))<nrow(data1)]
colnames(data1) <- c("item","1-1","1-2","1-3","2-1","2-2","2-3","3-1","3-2","3-3","4-1","4-2","4-3","5-
```

```
data1 <- melt(data1, id.vars = "item")
data1 <- separate(data = data1, col = 'variable',
                  into = c("operator", "TimeOfMeasurement"))
data1$operator <- as.numeric(data1$operator)
operators <- data1$operator
measurevalue <- data1$value
f <- function(boot_samplesize){
    b0 <- c(NA)
    b1 <- c(NA)
    for (i in 1:100){
      data_boot <- data1[sample(1:boot_samplesize, boot_samplesize, replace = TRUE),]
      b0[i] <- lm(value~operator, data=data_boot)$coefficients[1]
      b1[i] <- lm(value~operator, data=data_boot)$coefficients[2]
      }
    print(paste("Intercept Estimate: ", round(mean(b0),2)))
    print(paste("Operator Estimate: ", round(mean(b1),2)))
}
system.time(
  {
    set.seed(1)
    f(150)
  }
)
```

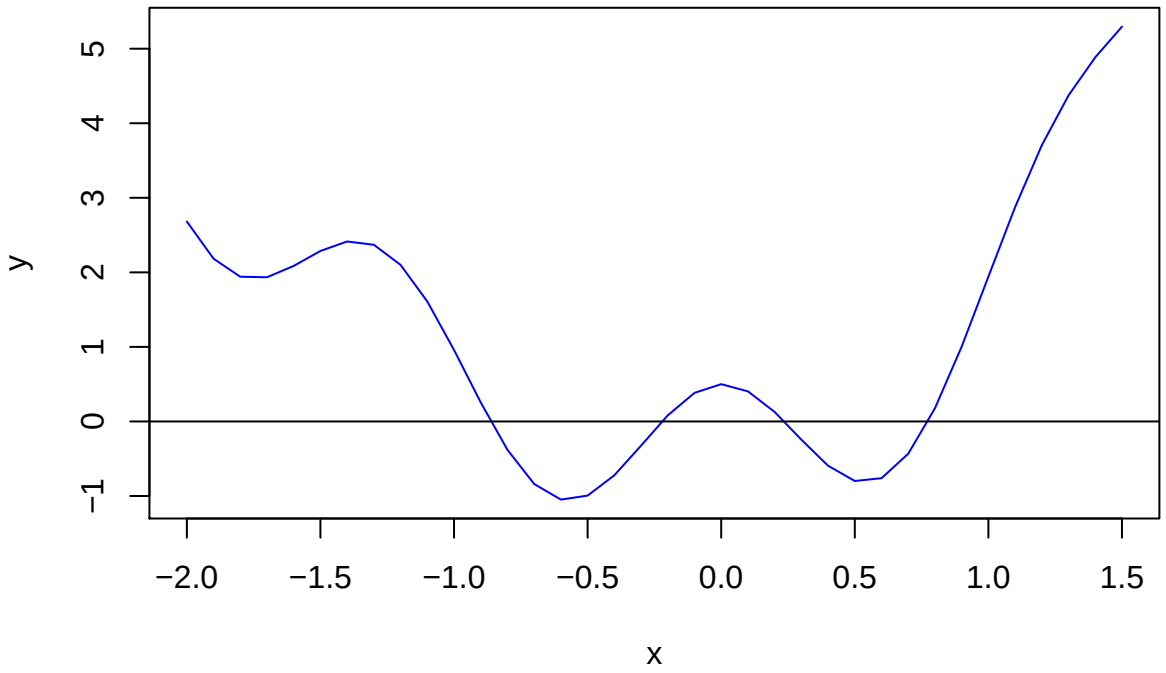
```
## [1] "Intercept Estimate:  4.55"
## [1] "Operator Estimate:  0.04"
```

```
##    user  system elapsed
##   0.098   0.001   0.099
```

Problem 3

(a)

```
x <- seq(from = -2, to = 1.5, by = 0.1)
y <- 3^x - sin(x) + cos(5*x) + x^2 - 1.5
plot(x,y,type = 'l',col="blue")
abline(h=0)
```



We can see that there are 4 roots in the plot.

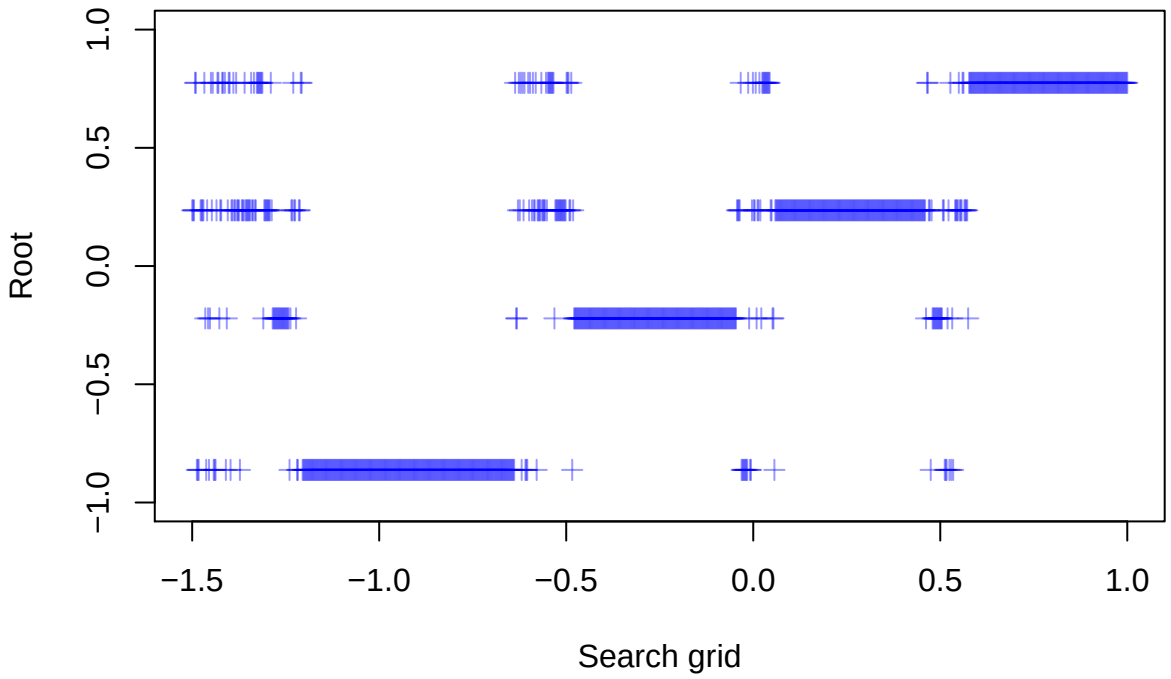
```
newton <- function(x0) {
  f <- function(x){3^x - sin(x) + cos(5*x) + x^2 - 1.5}
  h <- 0.001
  i <- 1
  x1 <- x0
  p <- numeric(100)
  while (i <= 100) {
    df.dx <- (f(x0+h)-f(x0))/h
    x1 <- (x0 - (f(x0)/df.dx))
    p[i] <- x1
    i <- i + 1
    c <- abs(x1-x0)
    x0 <- x1
    if(c < 0.00001){
      break
    }
  }
  return(p[i-1])
}
```

(b)

```
gridvec <- seq(from = -1.5, to = 1, length.out = 1000)
system.time({
  sapply(X = gridvec[-363], FUN = newton)
})
```

```
##    user  system elapsed
##   0.025   0.000   0.025
```

```r
plot(x = gridvec[-363],
     y = sapply(X = gridvec[-363], FUN = newton),
     ylim = c(-1,1), pch = 3,
     col = rgb(0,0,1, alpha = 0.4),
     xlab = 'Search grid',
     ylab = 'Root')
```



I drew a scatter plot for a summary.

Problem 4

(a)

```r
f <- function(c, m) {
  yhat <- m * operators + c
  MSE <- sum((measurevalue - yhat) ^ 2) / 150
  converged = F
  iterations = 0
  while(converged == F) {
    m1 <- m - 0.001 * ((1 / 150) * (sum((yhat - measurevalue) * operators)))
    c1 <- c - 0.001 * ((1 / 150) * (sum(yhat - measurevalue)))
    m <- m1
    c <- c1
    yhat <- m * operators + c
    MSE1 <- sum((measurevalue - yhat) ^ 2) / 150
    if(MSE - MSE1 <= 0.0001) {
      converged = T
      return(c(c, m))
    }
    iterations = iterations + 1
    if(iterations > 1000) {
      converged = T
```

```
      return(c(c, m))
    }
  }
}
f(c = runif(1, 3, 5), m = runif(1, 0, 1))
```

```
## [1] 3.2370216 0.3959257
```

(b)

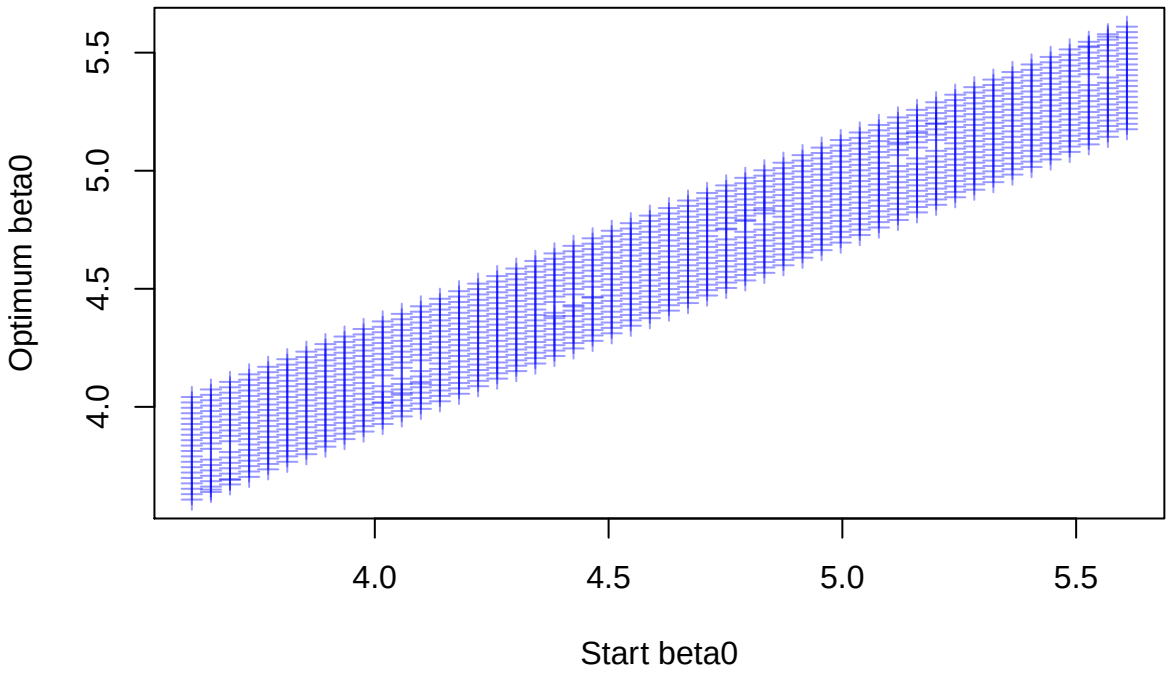
The maximum iterated set is 1000 and stopping rule is 0.0001.

(c), (d)

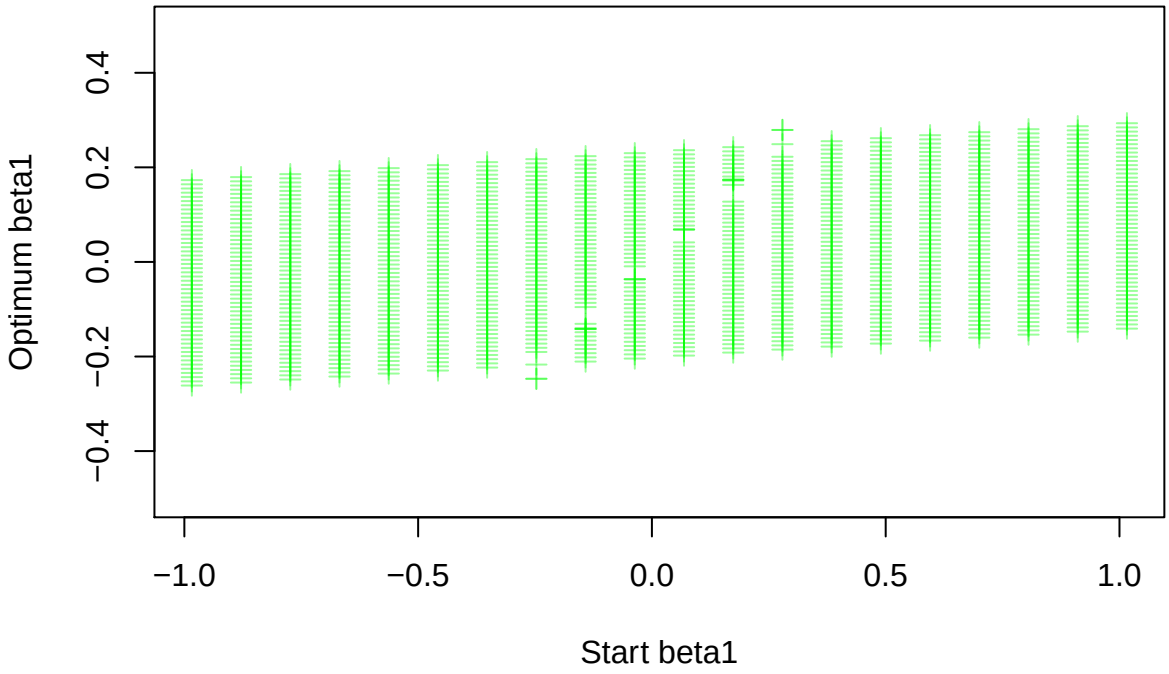
```
trueb0 <- lm(value~operator, data=data1)$coefficients[1]
trueb1 <- lm(value~operator, data=data1)$coefficients[2]
gridb0 <- seq(trueb0 - 1, trueb0 + 1, length.out=50)
gridb1 <- seq(trueb1 - 1, trueb1 + 1, length.out=20)
grid_b <- expand.grid(gridb0, gridb1)
colnames(grid_b) <- c('c', 'm')
system.time(
  {
    opt <- mapply(FUN = f, c=grid_b$c, m=grid_b$m)
  }
)
```

```
##    user  system elapsed
##  43.656   0.298  44.028
```

```
b0_opt <- opt[1,]
b1_opt <- opt[2,]
plot(x=grid_b$c, y=b0_opt, pch = 3,
     xlab = 'Start beta0', ylab = 'Optimum beta0',
     col = rgb(0,0,1, alpha = 0.4))
```




```
plot(x=grid_b$m, y=b1_opt, pch = 3,
     xlab = 'Start beta1', ylab = 'Optimum beta1',
     col = rgb(0,1,0, alpha = 0.4),
     ylim = c(-0.5,0.5))
```



I create a scatter plot for each β_0, β_1 plotting start vs optimum.

Problem 5

I knitted this document as PDF.