HW5_arflowers

Anna Flowers

11/8/2021

Homework 5

Problem 2

```
library(tidyr)
setwd("~/Desktop/VT/StatProgPackages")
sensory <- read.csv('Sensory.dat', sep = "", header = FALSE)
```

Part A

The original code supplied does not work because when you use the cbind command on a data frame, it does not automatically make the column names the same as the original vector names. This means that when using the lm command in the loop and calling the logapple08 and logrm08 variables, they are both the original values instead of the resampled ones. So every iteration will give an estimate the same as the original one, since the command is always on the original data.

Part B

```
sensory[-c(2,3,6,9,12,15,18,21,24,27,30), 2:6] <- sensory[-c(2,3,6,9,12,15,18,21,24,27,30), 1:5]
sensory <- sensory[,2:6]
sensory <- sensory[-c(1,2),]
colnames(sensory) <- c('1','2','3','4','5')
sensory <- gather(sensory, key = "Operator", value = "y", 1:5)
sensory$y <- as.numeric(sensory$y)

set.seed(1834)
n <- 100
beta.boot <- matrix(0, n, 5)
system.time({
for(i in 1:n) {
  samp1 <- sample(seq(1, 30, by = 1), 30, replace = TRUE)
  samp2 <- sample(seq(31, 60, by = 1), 30, replace = TRUE)
  samp3 <- sample(seq(61, 90, by = 1), 30, replace = TRUE)
  samp4 <- sample(seq(91, 120, by = 1), 30, replace = TRUE)
  samp5 <- sample(seq(121, 150, by = 1), 30, replace = TRUE)
  sens.boot <- sensory[c(samp1,samp2,samp3,samp4,samp5),]
  fit <- lm(y ~ Operator, sens.boot)
  beta.boot[i,] <- fit$coefficients
}})
```

```
##    user  system elapsed
##   0.079   0.001   0.080
```

```
mean(beta.boot[,1])
```

```
## [1] 4.531833
```

```
mean(beta.boot[,2])
```

```
## [1] 0.5080667
```

```
mean(beta.boot[,3])
```

```
## [1] -0.4377
```

```
mean(beta.boot[,4])
```

```
## [1] 0.6866667
```

```
mean(beta.boot[,5])
```

```
## [1] -0.2438667
```

Problem 3

Part A

From the graph it appears there are 4 roots.

```
f <- function(x) {
  y <- x - (3^(x) - sin(x) + cos(5*x) + x^2 - 1.5) / (log(3)*3^(x) - cos(x) - 5*sin(5*x) + 2*x)
  n <- 0
  while(abs(x - y) > .00001 && n < 100) {
    x <- y
    y <- x - (3^x - sin(x) + cos(5*x) + x^2 - 1.5) / (log(3)*3^x - cos(x) - 5*sin(5*x) + 2*x)
    n <- n + 1
  }
  if(n < 100) {return(y)}
  return(NA)
}
```

If the function does not converge in 100 iterations, it returns a value of NA.

Part B

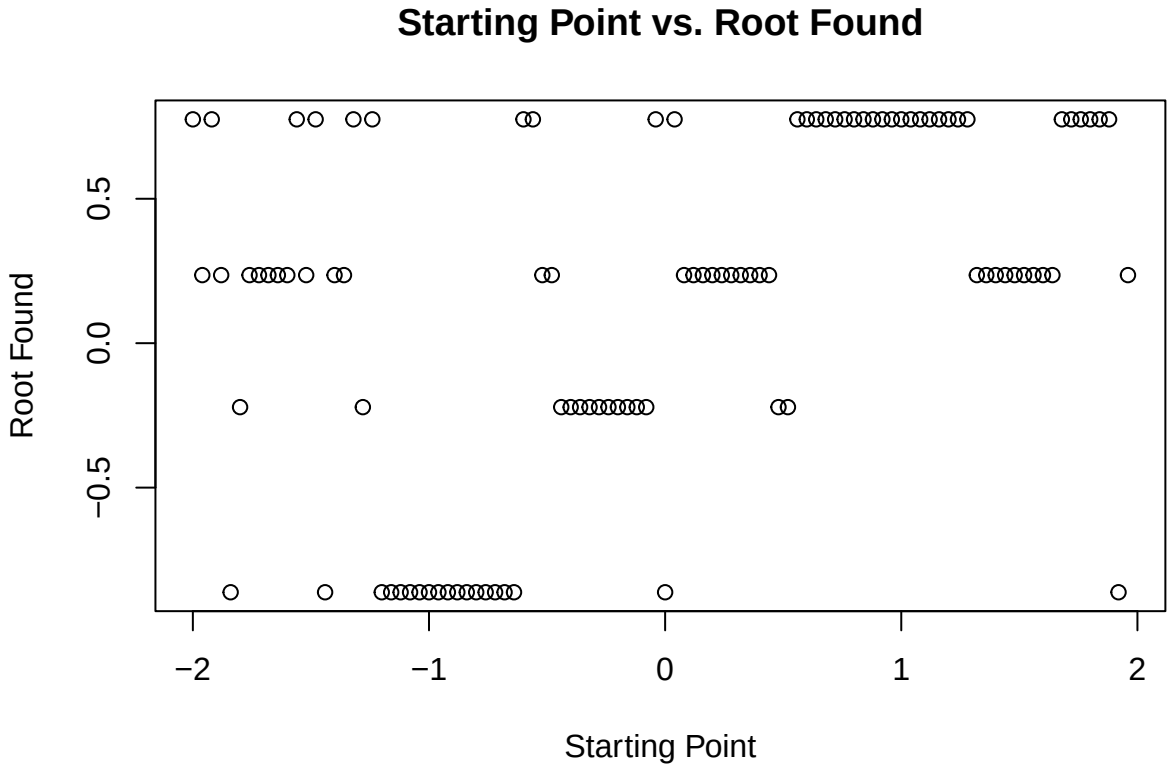
```
vec <- cbind(seq(-2, 1.96, by = 0.04))
length(vec)
```

```
## [1] 100
```

```
time <- system.time({solutions <- apply(vec, 1, f)})
time
```

```
##    user  system elapsed
##   0.016   0.000   0.016
```

```
plot(vec, solutions, xlab = "Starting Point", ylab = "Root Found", main = "Starting Point vs. Root Found
```



Problem 4

Part A

Part B

Part C

Part D