STAT 5014 Homework 3

Chaoran Wang

2017-09-19

Fineshed Problem 1 to 2

Problem 4

What I takeaway from these guides is it is very important to learn a good programming style which makes my codes be easy to read, share, and verify. For example, I used "=" more than "<-" before which seems not to be good. And sometimes, I liked to named some regression to be lm1 which is not clear for an object name. Besides, I hope I could be more familiar with "%>%" operator in dplyr package later. At this time, the statement in order seems to be more familiar to me. I am always confused about how to use "%>%" when coding.

Problem 5

I cannot find syntax error using 'lint' function.

```
##                                                                  file line
## 1 D:/STAT_5014/02_data_munging_summarizing_R_git/HW2_Wang_Chaoran.Rmd   NA
##   column message  type
## 1     NA    <NA> error
```

Problem 6

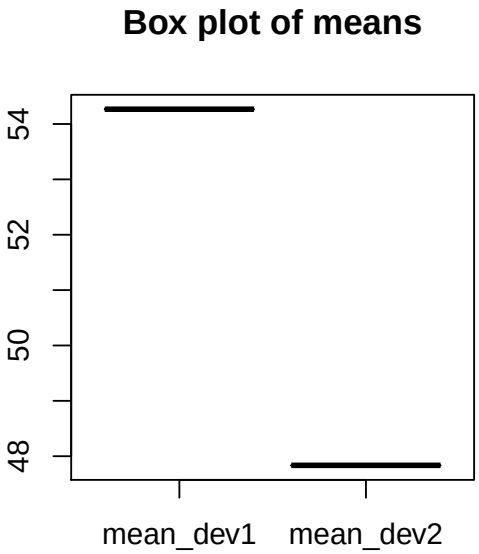
I first create the function which calculate the statistics we need. Then I read in the data and tidy it. Next, I apply the function to the data by group as Observer and print the results we need. The codes are in Appendix.

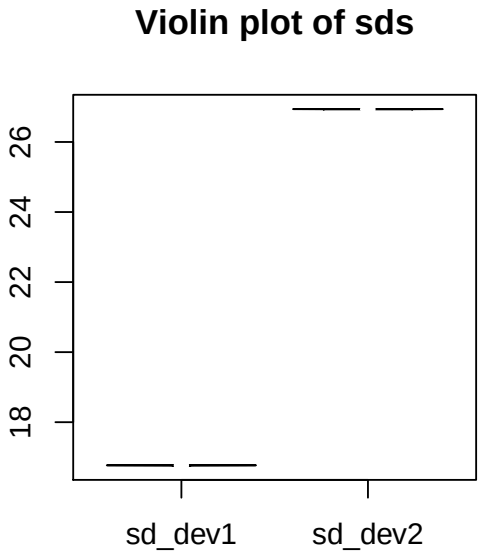
When I try to plot the boxplots and violin plots of dev1 and dev2 together, the plots just show as a straight line. I guess that is because of the ranges of each column are small. The means of dev1 are all about 54.26 while the means of dev2 are all about 47.83. That makes the two ranges lie far away from each other and look like a straight line. The same thing happens to violin plot. Hence, I decide to split them into different plots which look better.

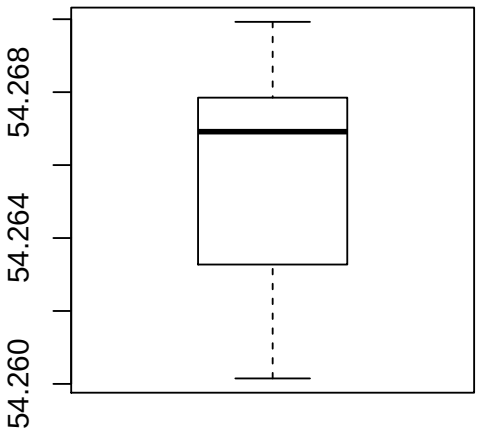
Table 1: Summary Statistics of 13 Observers

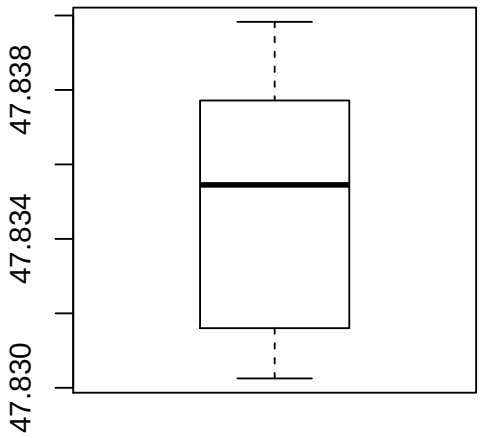
Observer	mean_dev1	mean_dev2	sd_dev1	sd_dev2	cor_dev1dev2
1	54.26610	47.83472	16.76983	26.93974	-0.0641284
2	54.26873	47.83082	16.76924	26.93573	-0.0685864
3	54.26732	47.83772	16.76001	26.93004	-0.0683434
4	54.26327	47.83225	16.76514	26.93540	-0.0644719
5	54.26030	47.83983	16.76774	26.93019	-0.0603414
6	54.26144	47.83025	16.76590	26.93988	-0.0617148
7	54.26881	47.83545	16.76670	26.94000	-0.0685042
8	54.26785	47.83590	16.76676	26.93610	-0.0689797
9	54.26588	47.83150	16.76885	26.93861	-0.0686092
10	54.26734	47.83955	16.76896	26.93027	-0.0629611

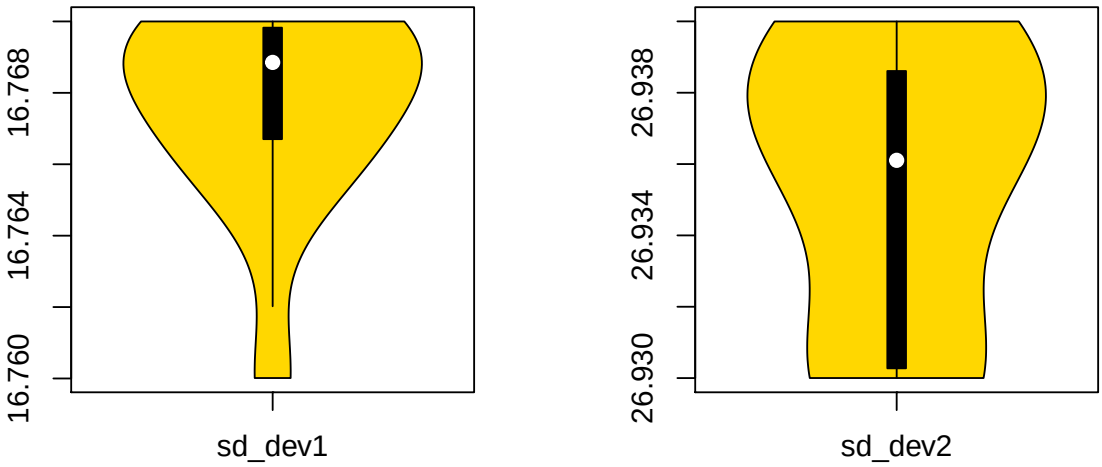
Observer	mean_dev1	mean_dev2	sd_dev1	sd_dev2	cor_dev1dev2
11	54.26993	47.83699	16.76996	26.93768	-0.0694456
12	54.26692	47.83160	16.77000	26.93790	-0.0665752
13	54.26015	47.83972	16.76996	26.93000	-0.0655833











Problem 7

First, I read in and create a tidy dataset. To tidy the dataset, I first removed the reduplicated "Day" columns and the first row with names. Then I rename the columns. I then using gather() function to tidy all readings by devices or doctors. Then I assign 1 to 6 to three devices and three doctors to be indicators of different reading methods.

After tidying, a summary is in Table 1. Since I am not sure what the experiment is actually, I cannot analyze the data based on the information I have. It looks not like a simple linear model. The codes are in Appendix.

Table 2: Blood Pressure Data Summary

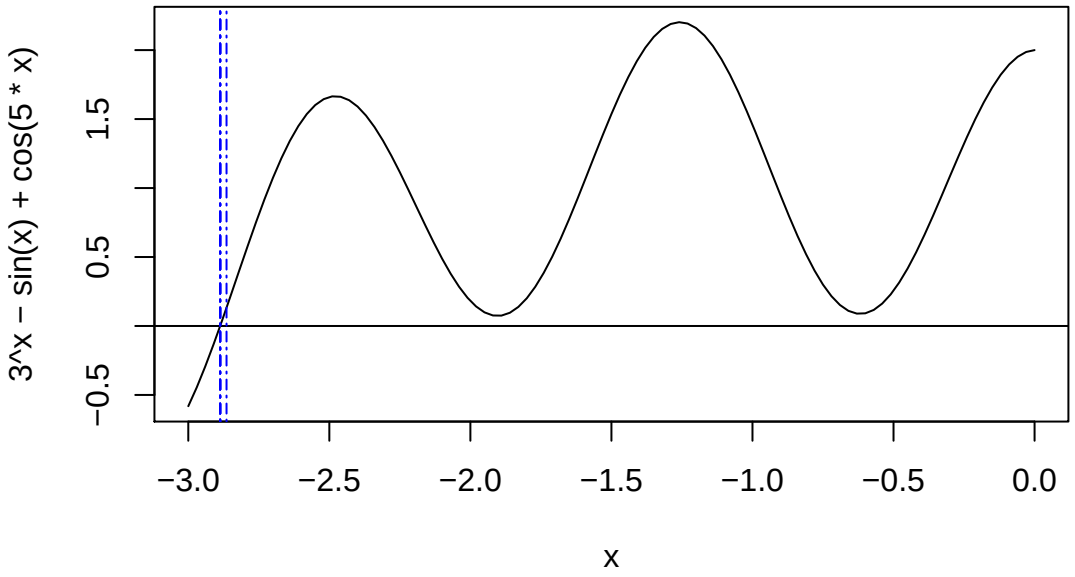
Day	Method	ID	value
Min. : 1	Length:90	Min. :1	Min. :110.8
1st Qu.: 4	Class :character	1st Qu.:1	1st Qu.:125.5
Median : 8	Mode :character	Median :2	Median :130.4
Mean : 8	NA	Mean :2	Mean :129.0
3rd Qu.:12	NA	3rd Qu.:3	3rd Qu.:134.3
Max. :15	NA	Max. :3	Max. :139.6

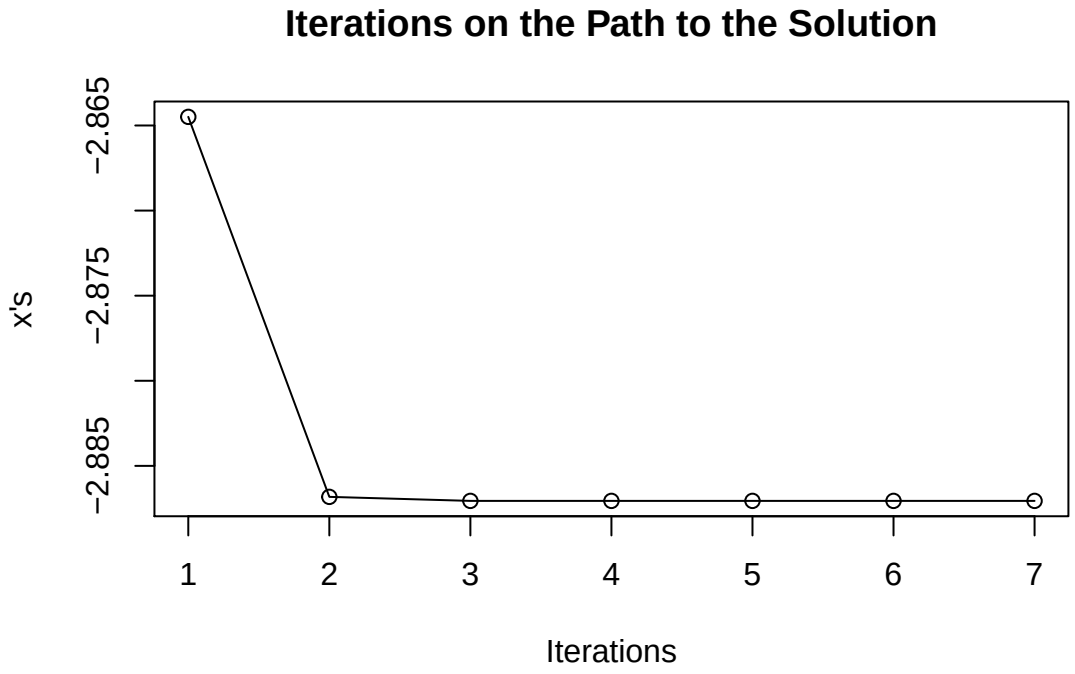
Problem 8

I first create a function based on my math knowledge about Newton's Method. The codes are in Appendix.

Since the function has sin and cos part, the numbers of roots of this function should be infinite if we do not assign a domian to them. Hence, I assign $x \in [-3, 0]$ here to find the root within this domian. Solving the function by my Newton's method function, I can get a root to be $x = -2.887058$. I draw the function in domain and add the line $f(x) = 0$. I also draw the iterations of x's. We can see the last x (the root we got) intersect the line $f(x) = 0$ and the function $f(x)$. The codes are in Appendix.

```
## [1] -2.864494 -2.886821 -2.887058 -2.887058 -2.887058 -2.887058 -2.887058
```





Repeating this step in different domains and assigning different but resonable initial value of x_0, we are able to get many other roots. Since the steps are similar, I omit them in this homework.

Problem 9

For this problem, we are trying to munge some large datasets and give a briefly summary. Since the datasets are very large, it might be a good idea to read some of them (first 100 rows here) in R and finish the necessary work of columns first. The files are local on my computer. The directory might be different.

[1] "Gebreken" Gebrek.identificatie Ingangsdatum.gebrek Einddatum.gebrek "character" "integer" "integer" Gebrek.paragraaf.nummer Gebrek.artikel.nummer Gebrek.omschrijving "integer" "character" "character" [1] "Geconstat" Kenteken Soort.erkenning.keuringsinstantie "character" "character" Meld.datum.door.keuringsinstantie Meld.tijd.door.keuringsinstantie "integer" "integer" Gebrek.identificatie Soort.erkenning.omschrijving "character" "character" Aantal.gebreken.geconstateerd "integer" [1] "Personen" Kenteken Voertuigsoort "character" "character" Merk Handelsbenaming "character" "character" Datum.tenaamstelling Bruto.BPM "character" "integer" Cilinderinhoud Massa.ledig.voertuig "integer" "integer" Toegestane.maximum.massa.voertuig Datum.eerste.toelating "integer" "character" Datum.eerste.afgifte.Nederland Catalogusprijs "character" "integer" WAM.verzekerd "character"

What columns they have are not shown in English which makes me confused. But never mind, I use Google Translate to understand what the columns mean. Continuing with the datasets, I figured that it was unrealistic to read all data in and analyze them because there are so much information in these three datasets. That might be better to focus just one aspect, maybe the data in one year, to be our interest. So I choose rows in 2017 to be my interest just as Dr. Settlage required. And for the columns, I also just choose few of them. For "Gebreken", it has defect code information and description. For "Geconstat", it has inspection date and defect code in respective to license plates. For "Personen", it has make and model of vehicle in respective to license plates. Using Google Translate, I figured which columns I need, first and sixth columns of "Gebreken", first, third, and fifth columns of "Geconstat" (Defects), and first, third, and forth columns of "Person". Thanks to the work from Dr. Settlage, I learned that "fread" is a function which is similar to read.table but much faster and convienient. I use it then to read the specfic data I am interested in.

Next, I use merge function to merge three datasets by plates and defect code. During cleaning data, I translate the columns into English and remove NAs. Then I use unique() function to find out how many different makes and models of cars are reported with defects in 2017. I use sort() function to get which the 5 most frequent defects and the top makes/models having that defects in 2017.

[1] 503 [1] 33470

```
## Warning: package 'bindrcpp' was built under R version 3.3.3
```

Table 3: The 5 Most Frequent Defects in 2017

Defect_Code	Defect_Count	Make	Top_Make_Count
AC1	385621	VOLKSWAGEN	43489
K04	271259	PEUGEOT	27702
RA2	223142	OPEL	29085
205	171244	VOLKSWAGEN	18391
497	139980	PEUGEOT	16318

```
## Warning in anova.lmlist(object, ...): models with response '"NULL"' removed
## because response differs from model 1
```

```
## Warning in anova.lm(object): ANOVA F-tests on an essentially perfect fit
## are unreliable
```

Table 4: Defects Data Summary

Make	Num_Def
A.C.L. : 1	Min. : 1.0

Make	Num_Def
A.M.C. : 1	1st Qu.: 3.0
ACCUBUILT: 1	Median : 9.0
ACM : 1	Mean : 8727.3
ACURA : 1	3rd Qu.: 60.5
ADRIA : 1	Max. :466979.0
(Other) :497	NA

```
## Warning in anova.lm(lm_defect_make): ANOVA F-tests on an essentially
## perfect fit are unreliable
```

Table 5: ANOVA of fitting Number of defects with Make

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
defect_make$Make	502	1.102845e+12	2196902116	NaN	NaN
Residuals	0	0.000000e+00	NaN	NA	NA

Hence, there are 503 different makes and 33470 different models reported defects in 2017. "AC1 K04 RA2 205 497" are the 5 most frequent defects in 2017. And the regression summary and ANOVA table are shown. For the lm and anova analysis, I finished it for Make but keep getting error "cannot allocate vector of size 8.3 Gb" when doing same thing for Model. I guess that might because of the size of dataframe is to large for lm function. Hope you would not mark me wrong. Besides the output of lm summary is too long, so I do not show it here. The codes are in Appendix.

In order to make the codes work more computationally efficient, I think that might be better to read the data of 2017 only at first.

Appendix 1: R code

```
########################### Problem6_func Function to create a summary statistic
sum_stat <- function(x) {
    dev1 <- x[, 1]
    dev2 <- x[, 2]
    mean_dev1 <- mean(dev1)
    mean_dev2 <- mean(dev2)
    sd_dev1 <- sd(dev1)
    sd_dev2 <- sd(dev2)
    cor_dev1dev2 <- cor(dev1, dev2)
    result <- data.frame(mean_dev1, mean_dev2, sd_dev1, sd_dev2, cor_dev1dev2)
    return(result)
}
########################### Problem6_data Read data and tidy data
data_pro6 <- readRDS(file = "D:/STAT_5014/03_good_programming_R_functions/HW3_data.rds") %>%
    arrange(Observer)
# Collecting the summary statistics via sum_stat function Using by function
# here which is easiser than looping
sum_list <- by(data_pro6[, 2:3], data_pro6[, 1], sum_stat)
# Convert the list to data.frame
sum_result <- data.frame(matrix(unlist(sum_list), nrow = 13, byrow = T))
Observer <- c(1:13)
sum_result <- cbind(Observer, sum_result)
colnames(sum_result) <- c("Observer", "mean_dev1", "mean_dev2", "sd_dev1", "sd_dev2",
    "cor_dev1dev2")
knitr::kable(sum_result, caption = "Summary Statistics of 13 Observers")
par(mfrow = c(1, 2))
boxplot(sum_result[, 2:3])
title("Box plot of means")
vioplot(sum_result[, 4], sum_result[, 5], names = c("sd_dev1", "sd_dev2"), col = "gold")
title("Violin plot of sds")
par(mfrow = c(1, 2))
boxplot(sum_result[, 2])
boxplot(sum_result[, 3])
par(mfrow = c(1, 2))
vioplot(sum_result[, 4], names = "sd_dev1", col = "gold")
vioplot(sum_result[, 5], names = "sd_dev2", col = "gold")
########################### Problem7_BP_analysis get data
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BloodPressure.dat"
BP_raw <- read.table(url, header = F, skip = 1, fill = T, stringsAsFactors = F)
BP_tidy <- BP_raw[-1, -5]
colnames(BP_tidy) <- c("Day", "Device_1", "Device_2", "Device_3", "Doctor_1",
    "Doctor_2", "Doctor_3")
BP_tidy$Day <- as.numeric(as.character(BP_tidy$Day))
BP_tidy <- gather(BP_tidy, Method, value, Device_1:Doctor_3) %>% separate(Method,
    c("Method", "ID"), "_") %>% arrange(Day)
BP_tidy$value <- as.numeric(as.character(BP_tidy$value))
BP_tidy$ID <- as.numeric(as.character(BP_tidy$ID))
########################### Problem8_func Function of Newton's Method
newton <- function(fun, tol = 1e-20, x0, N) {
    h <- 0.001
    i <- 1
```

```
    x1 <- x0
    z <- numeric(N)
    while (i <= N) {
        df_dx <- (fun(x0 + h) - fun(x0))/h
        error <- fun(x0)/df_dx
        x1 <- (x0 - error)
        z[i] <- x1
        i <- i + 1
        if (abs(x1 - x0) < tol)
            break
        x0 <- x1
    }
    return(z[1:(i - 1)])
}
############################ Problem8_sol Solve the function
fun <- function(x) {
    3^x - sin(x) + cos(5 * x)
}
z <- newton(fun, x0 = -3, N = 100)
print(z)
curve(3^x - sin(x) + cos(5 * x), -3, 0)
abline(h = 0)
abline(v = z[1:length(z)], lty = 6, col = "blue")
plot(1:length(z), z, xlab = "Iterations", ylab = "x's", main = "Iterations on the Path to the Solution")
lines(1:length(z), z)
############################ Problem9 munge

# defect code and description
Car_Gebreken_select <- fread(input = "D:/Open_Data_RDW__Gebreken.csv", header = T,
    select = c(1, 6), showProgress = F)
# license plate, inspection date and defect code
Car_Geconstat_select <- fread(input = "D:/Open_Data_RDW__Geconstateerde_Gebreken.csv",
    header = T, select = c(1, 3, 5), showProgress = F)
# license plate, make and model of vehicle
Car_Person_select <- fread(input = "D:/Personenauto_basisdata.csv", header = T,
    showProgress = F, select = c(1, 3, 4))

Car_Geconstat_select_2017 <- Car_Geconstat_select[grep("2017", Car_Geconstat_select$"Meld datum door ke
    ]
# merge datasets
Car_plates <- merge(Car_Geconstat_select_2017, Car_Person_select, by = "Kenteken")
Car_defect <- merge(Car_Gebreken_select, Car_plates, by = "Gebrek identificatie")
# Clean data
Car_defect <- Car_defect[complete.cases(Car_defect), ]
# Translate
colnames(Car_defect) <- c("Defect_Code", "Description", "License_Plate", "Report_Date",
    "Make", "Model")
num_makes <- length(unique(Car_defect$Make))
print(num_makes)
num_models <- length(unique(Car_defect$Model))
print(num_models)
# most frequent defects top5_defects <- sort(table(Car_defect$Defect_Code),
# decreasing=TRUE)[1:5]
```

```r
Top5_defects <- Car_defect %>% group_by(Defect_Code, Make) %>% summarise(Make_Count = n()) %>%
    mutate(Defect_Count = sum(Make_Count)) %>% arrange(desc(Defect_Count)) %>%
    mutate(Top_Make_Count = max(Make_Count)) %>% filter(Make_Count %in% Top_Make_Count) %>%
    select(Defect_Code, Defect_Count, Make, Top_Make_Count)
knitr::kable(Top5_defects[1:5, ], caption = "The 5 Most Frequent Defects in 2017")
# number of defects vs. make
defect_make <- as.data.frame(table(Car_defect$Make))
colnames(defect_make) <- c("Make", "Num_Def")
lm_defect_make <- lm(defect_make$Num_Def ~ defect_make$Make)
ano_defect_make <- anova(lm_defect_make, data = defect_make)
# summary(lm_defect_make)
knitr::kable(summary(defect_make), caption = "Defects Data Summary")
knitr::kable(anova(lm_defect_make), caption = "ANOVA of fitting Number of defects with Make")
# number of defects vs. model
defect_model <- as.data.frame(table(Car_defect$Model))
colnames(defect_model) <- c("Model", "Num_Def")
# lm_defect_model <- lm(defect_model$Num_Def~defect_model$Model)
# ano_defect_make <- anova(lm_defect_make, data=defect_make)
# summary(lm_defect_make) summary(ano_defect_make)
```