Homework 5

Li Zhang

06/30/2021

Reuse

For many of these exercises, you may be able to reuse functions written in prior homework. Include those functions here. You may find that you will need to modify your functions to work correctly for these exercises.

```
StandardError <- function(sd,n){
  sd/sqrt(n)
}

ConfidenceBound<-function(sd, n, alpha=0.05){
  qnorm(1-alpha/2)*StandardError(sd,n)
}

ConfidenceInterval<-function(mean, sd, n, alpha=0.05){
  local.Lower<- mean-ConfidenceBound(sd, n)
  local.Upper<- mean+ConfidenceBound(sd, n)
  return(list(Lower=local.Lower, Upper=local.Upper))
}
```

Starting with R 4.0, the default behavior of `read.table` and related functions has changed. You may wish to include this option for backward compatibility. Note that this is only a short-term solution (see https://developer.r-project.org/Blog/public/2020/02/16/stringsasfactors/)

```
options(stringsAsFactors = TRUE)
```

```
## Warning in options(stringsAsFactors = TRUE): 'options(stringsAsFactors = TRUE)'
## is deprecated and will be disabled
```

Warning Starting with these exercises, I will be restricting the use of external libraries in R, particularly `tidyverse` libraries. Our goal here is to understand the R language and the mechanics of the R system. Much of the tidyverse is a distinct language, implemented in R. You will be allowed to use whatever libraries tickle your fancy in the final project.

Exercise 1

Part a

Go to http://www.itl.nist.gov/div898/strd/anova/SiRstv.html and use the data listed under `Data File in Table Format` (https://www.itl.nist.gov/div898/strd/anova/SiRstvt.dat)

Part b

Edit this into a file (tab delimited, `.csv`, etc,) that can be read into R or SAS, or find an appropriate function that can read the file as-is. You will need to upload the edited file to D2L along with your Rmd/SAS files.

Provide a brief comment on changes you make, or assumptions about the file needed for you file to be read into R/SAS. Read the data into a data frame or data table.

```
Data_path = "https://www.itl.nist.gov/div898/strd/anova/SiRstvt.dat"
readLines(Data_path, n=100)
```

```
##  [1] "NIST/ITL StRD "
##  [2] "Dataset Name:   SiRstv     (SiRstvt.dat)"
##  [3] ""
##  [4] ""
##  [5] "File Format:    ASCII"
##  [6] "                Certified Values   (lines 41 to 47)"
##  [7] "                Data               (lines 61 to 65) "
##  [8] ""
##  [9] ""
## [10] "Procedure:      Analysis of Variance"
## [11] ""
## [12] ""
## [13] "Reference:      Ehrstein, James and Croarkin, M. Carroll."
## [14] "                Unpublished NIST dataset."
## [15] ""
## [16] ""
## [17] "Data:           1 Factor"
## [18] "                5 Treatments"
## [19] "                5  Replicates/Cell"
## [20] "                25 Observations"
## [21] "                3 Constant Leading Digits"
## [22] "                Lower Level of Difficulty"
## [23] "                Observed Data"
## [24] ""
## [25] ""
## [26] "Model:          6 Parameters (mu,tau_1, ... , tau_5)"
## [27] "                y_{ij} = mu + tau_i + epsilon_{ij}"
## [28] ""
## [29] ""
## [30] ""
## [31] ""
## [32] ""
## [33] ""
## [34] ""
## [35] ""
## [36] "Certified Values:"
## [37] ""
## [38] "Source of                 Sums of               Mean               "
## [39] "Variation          df     Squares              Squares             F Statistic"
## [40] ""
## [41] "Between Instrument  4 5.11462616000000E-02 1.27865654000000E-02 1.18046237440255E+00"
## [42] "Within Instrument  20 2.16636560000000E-01 1.08318280000000E-02"
## [43] ""
## [44] "                   Certified R-Squared 1.90999039051129E-01"
## [45] ""
## [46] "                   Certified Residual"
## [47] "                   Standard Deviation  1.04076068334656E-01"
## [48] ""
## [49] ""
```

```
## [50] ""
## [51] ""
## [52] ""
## [53] ""
## [54] ""
## [55] ""
## [56] "Data:"
## [57] "                         Instrument"
## [58] ""
## [59] "    1           2           3           4           5"
## [60] ""
## [61] "196.3052   196.3042   196.1303   196.2795   196.2119"
## [62] "196.1240   196.3825   196.2005   196.1748   196.1051"
## [63] "196.1890   196.1669   196.2889   196.1494   196.1850"
## [64] "196.2569   196.3257   196.0343   196.1485   196.0052"
## [65] "196.3403   196.0422   196.1811   195.9885   196.2090"
```

```
exe_data=read.table(Data_path,skip=58,header=TRUE)
write.table(exe_data, file="exe1_data.csv", sep=',', row.names = FALSE)
#PMC: In order to get the useful data,
#I used function readLines
#and tried to find how many lines we should skip
#PMC:I skipped the first 58 lines to get the useful data.
#and write the data into exe1_data.csv
#then read it using read.csv again.(maybe this is unnaccessary)
exe1_data=read.csv("exe1_data.csv", header=TRUE)
head(exe1_data)
```

```
##         X1       X2       X3       X4       X5
## 1 196.3052 196.3042 196.1303 196.2795 196.2119
## 2 196.1240 196.3825 196.2005 196.1748 196.1051
## 3 196.1890 196.1669 196.2889 196.1494 196.1850
## 4 196.2569 196.3257 196.0343 196.1485 196.0052
## 5 196.3403 196.0422 196.1811 195.9885 196.2090
```

Part c

There are 5 columns in these data. Calculate mean and sd and sample size for each column in this data, using column summary functions. Print the results below

```
exe1_mean<-round(apply(exe1_data, 2, mean),4)
exe1_sd<-round(apply(exe1_data, 2, sd),4)
sample_size<-apply(exe1_data, 2, length)
exe1_info<-data.frame(exe1_mean,exe1_sd,sample_size)
exe1_info
```

```
##    exe1_mean exe1_sd sample_size
## X1  196.2431  0.0875           5
## X2  196.2443  0.1380           5
## X3  196.1670  0.0937           5
## X4  196.1481  0.1042           5
## X5  196.1432  0.0884           5
```

```
#I created a data frame to represent mean, sd and sample size for each x.
#I also creatde the matrix,
#truns out creating data frame is more convenient than creating matrix.
```

```r
#exe1_info<-matrix(c(exe1_mean, exe1_sd, sample_size), nrow=3, byrow=TRUE)
#rownames(exe1_info)<-c("Mean", "SD", "n")
#colnames(exe1_info)<-c("X1","X2","X3","X4","X5")
#print("[<-"(exe1_info, as.character(exe1_info)), quote=FALSE)
```

Reuse your `ConfidenceInterval` function to compute confidence intervals for the means in this data set. Note, you can do this with one function call if you use vectors.

```r
CI_X1<-ConfidenceInterval(exe1_info[1, 1],exe1_info[1, 2], exe1_info[1, 3])
CI_X1
```

```
## $Lower
## [1] 196.1664
##
## $Upper
## [1] 196.3198
```

```r
CI_X2<-ConfidenceInterval(exe1_info[2, 1],exe1_info[2, 2], exe1_info[2, 3])
CI_X2
```

```
## $Lower
## [1] 196.1233
##
## $Upper
## [1] 196.3653
```

```r
CI_X3<-ConfidenceInterval(exe1_info[3, 1],exe1_info[3, 2], exe1_info[3, 3])
CI_X3
```

```
## $Lower
## [1] 196.0849
##
## $Upper
## [1] 196.2491
```

```r
CI_X4<-ConfidenceInterval(exe1_info[4, 1],exe1_info[4, 2], exe1_info[4, 3])
CI_X4
```

```
## $Lower
## [1] 196.0568
##
## $Upper
## [1] 196.2394
```

```r
CI_X5<-ConfidenceInterval(exe1_info[5, 1],exe1_info[5, 2], exe1_info[5, 3])
CI_X5
```

```
## $Lower
## [1] 196.0657
##
## $Upper
## [1] 196.2207
```

Exercise 2

We will use data from https://acsess.onlinelibrary.wiley.com/doi/abs/10.2134/jeq2007.0099, Table 1. The original paper is also available on D2L.

Download the file `Khan.csv` from D2L and read the file into a data frame. Print a summary of the table.

```
exe2_table=read.csv("Khan.csv", header=TRUE)
summary(exe2_table)
```

```
##    Rotation          Fertilizer           Depth               Mean55     
##  Length:27          Length:27          Length:27          Min.   :1.020  
##  Class :character   Class :character   Class :character   1st Qu.:1.434  
##  Mode  :character   Mode  :character   Mode  :character   Median :1.487  
##                                                           Mean   :1.538  
##                                                           3rd Qu.:1.661  
##                                                           Max.   :2.109  
##      Mean05           SD05               Diff        
##  Min.   :0.751   Min.   :0.004000   Min.   :-0.5020  
##  1st Qu.:1.141   1st Qu.:0.006500   1st Qu.:-0.2970  
##  Median :1.312   Median :0.008000   Median :-0.2090  
##  Mean   :1.325   Mean   :0.008519   Mean   :-0.2126  
##  3rd Qu.:1.474   3rd Qu.:0.010000   3rd Qu.:-0.1105  
##  Max.   :1.887   Max.   :0.014000   Max.   : 0.0130  
```

To show that the data was read correctly, create three plots. Plot

1. Rotation vs Fertilizer
2. Mean55 vs Fertilizer
3. Mean55 vs Mean05

`Mean05` and `Mean55` are the amount of soil organic carbon measured in crop land experimental units in 2005 and 1955 respectively. `Rotation` is the crop rotation plan (i.e. corn followed by soybeans followed by corn) for the respective plots, and `Fertilizer` is the type of fertilizer applied to the plots over the period from 1955-2005.

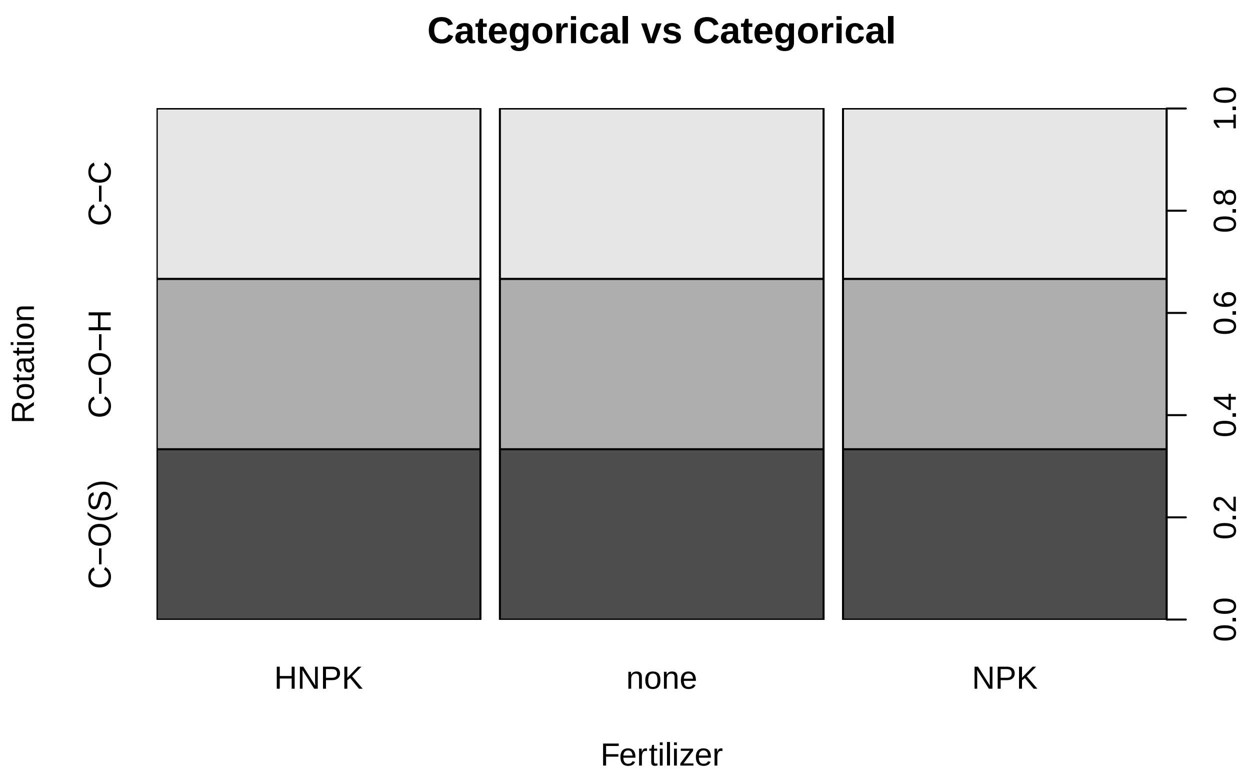
These three plots should reproduce the three types of plots shown in the `RegressionEtcPlots` video, **Categorical vs Categorical**, **Continuous vs Continuous** and **Continuous vs Categorical**. Add these as titles to your plots, as appropriate.

Do you notice anything unusual about the data?

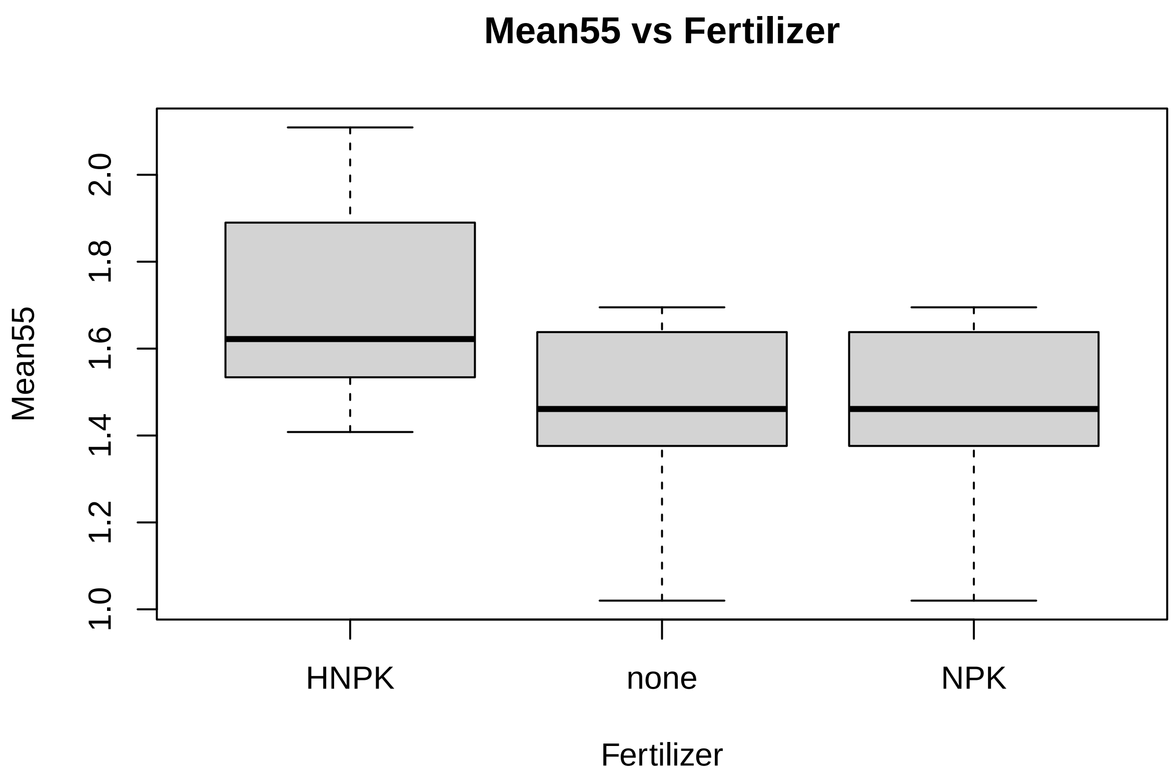
```
#Categorical vs Categorical mosaic plot    Rotation vs Fertilizer
table(exe2_table$Rotation, exe2_table$Fertilizer)
```

```
##          
##           HNPK none NPK
##   C-C        3    3   3
##   C-O-H      3    3   3
##   C-O(S)     3    3   3
```

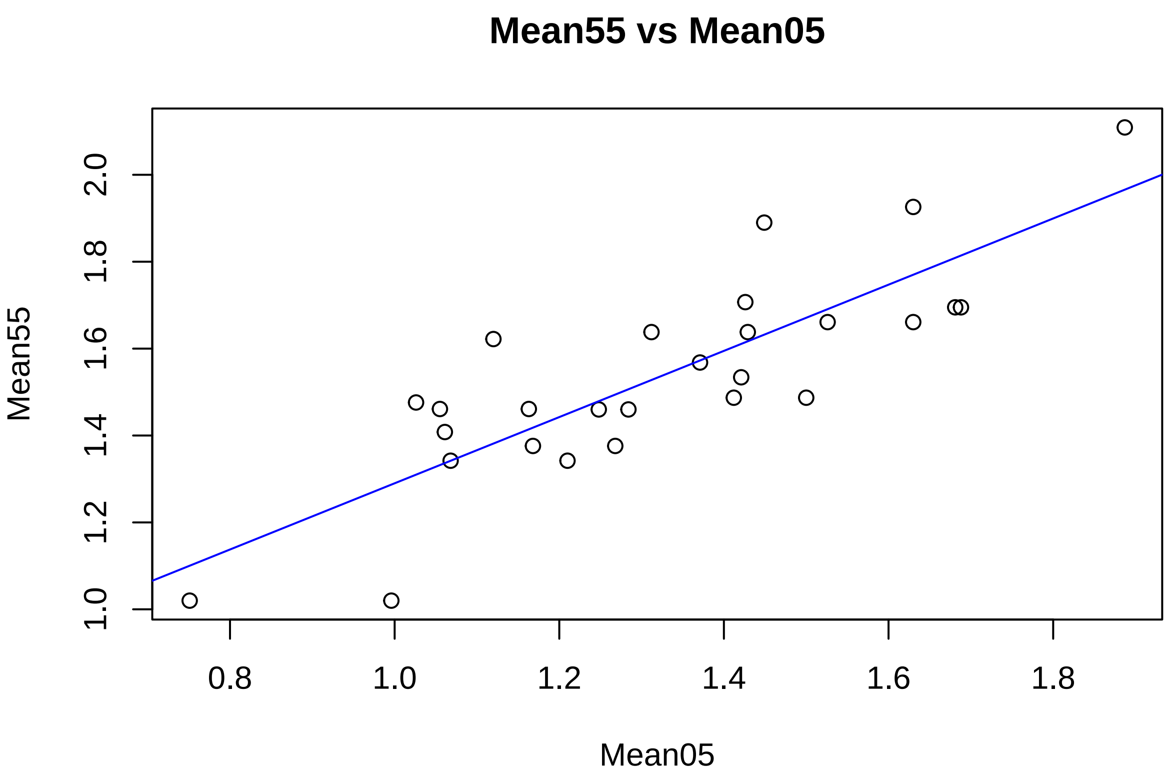
```
exe2_table$Rotation<-as.factor(exe2_table$Rotation)
exe2_table$Fertilizer<-as.factor(exe2_table$Fertilizer)
plot(Rotation~Fertilizer, data=exe2_table, main="Categorical vs Categorical")
```




```
#Continuous vs Categorical box-whisker plot   Mean55 vs Fertilizer

#boxplot( Mean55 ~ Fertilizer, data=exe2_table,
#main="Mean55 vs Fertilizer",
#xlab="Fertilizer", ylab="Mean55")

exe2_table$Fertilizer<-as.factor(exe2_table$Fertilizer)
plot(Mean55~Fertilizer, data=exe2_table,main="Mean55 vs Fertilizer" )
```




```r
#Continuous vs Continuous regression plot     Mean55 vs Mean05

plot(exe2_table$Mean05, exe2_table$Mean55,
     xlab="Mean05", ylab="Mean55", main="Mean55 vs Mean05")
exe2_3.lm <-lm(exe2_table$Mean55~exe2_table$Mean05)
abline(reg=exe2_3.lm,col="blue")
```




```
#unusual about the data: there is a copy-and-paste error
#in one of the tables from the original paper.
#and the data in this exercise comes from that table.
#Before publishing any paper, we should carefully examine our data
#and make sure every data we use is correct.
```

Exercise 3

Calculate a one-way analysis of variance from the data in Exercise 1. First, compute a difference in soil organic carbon between 1955 and 2005 (`Mean05-Mean55`). Call this `CarbonLoss`

Let y_{ij} be the `CarbonLoss`. Let the k treatments be `Fertilizer`. Let $T_i = \sum_{j=1}^{n_i} y_{ij}$ be the `CarbonLoss` total for `Fertilizer` i and let n_i be the number of observations for `Fertilizer` i. Denote the total number of observations $N = \sum n_i$.

Part a

Find the treatment (`Fertilizer`) totals

$$\mathbf{T} = T_1 \dots T_k$$

and observations per treatment

$$\mathbf{n} = n_1 \dots n_k$$

from the Khan data, using group summary functions and compute a grand (overall) total

$$G = \sum_i \sum_j y_{ij}$$

for `CarbonLoss`. Print **T**, **r** and G below. In SAS, you can use `proc summary` or `proc means` to compute T and r and output a summary table. I find the rest is easier in IML (see `use` to access data tables in IML).

```
carbon_difference<-exe2_table$Mean05-exe2_table$Mean55
carbon_difference
```

```
##  [1] -0.208 -0.274 -0.024 -0.108 -0.132 -0.269 -0.113 -0.197 -0.450 -0.075
## [11] -0.326 -0.176  0.013 -0.209 -0.212 -0.441 -0.281 -0.347 -0.007 -0.135
## [21] -0.406 -0.014 -0.031 -0.298 -0.222 -0.296 -0.502
```

```
#since the Khan.csv file already has diff column,
#I will use it to answer the rest of the question.
n<-aggregate(Depth~Fertilizer, data=exe2_table, FUN=length)
names(n)[2]<-"n(observation)"
y=aggregate(Diff~Fertilizer, data=exe2_table, FUN=sum)
names(y)[2]<-"CarbonLoss"
exe3_table_a<-merge(n,y, by="Fertilizer")
exe3_table_a
```

```
##   Fertilizer n(observation) CarbonLoss
## 1       HNPK              9     -2.849
## 2       none              9     -1.631
## 3        NPK              9     -1.260
```

```
G<-sum(exe3_table_a[,3])
G
```

```
## [1] -5.74
```

Part b

Calculate sums of squares as

$$
\begin{aligned}
\text{Correction Factor}: C &= \frac{G^2}{N} \\
\text{Total SS}: &= \sum y_{ij}^2 - C \\
\text{Treatments SS}: &= \sum \frac{T_i^2}{n_i} - C \\
\text{Residual SS}: &= \text{Total SS} - \text{Treatments SS}
\end{aligned}
$$

Print the calculated sums of squares below.

```
N<-sum(exe3_table_a[,2])
C<-((G)^2)/N
C
```

```
## [1] 1.220281
```

```
Total_sum<-sum((exe2_table[,7])^2)
Total_SS<-Total_sum-C
Total_SS
```

```
## [1] 0.5416585
```

```
#alternative cumbersome sway to calculate Total sums of squares
#mean_1<-sum(exe3_table$CarbonLoss)/N
#Total_SS1<-(exe2_table[1,7]-mean_1)^2+(exe2_table[2,7]-mean_1)^2+(exe2_table[3,7]-mean_1)^2+(exe2_tabl
#Total_SS1
```

```r
#Total_SS<-sum((exe3_table_a[1,3])^2,(exe3_table_a[2,3])^2,(exe3_table_a[3,3])^2)-C

Treat_SS<-sum((((exe3_table_a[1,3])^2)/exe3_table_a[1,2]),
              (((exe3_table_a[2,3])^2)/exe3_table_a[2,2]),
              (((exe3_table_a[3,3])^2)/exe3_table_a[3,2]))-C
Treat_SS
```

```
## [1] 0.1535587
```

```r
Residual_SS<-Total_SS-Treat_SS
Residual_SS
```

```
## [1] 0.3880998
```

Part c.

Calculate MSB = (Treatments SS)/$(k-1)$ and MSW = (Residual SS)/$(N-k)$. Calculate an F-ratio (MSB/MSW) and a p for this F, using the F (`pf`) distribution with $k-1$ and $N-k$ degrees of freedom. Use $\alpha = 0.05$ and `lower.tail = FALSE`.

```r
k<-length(exe3_table_a[,2])
k
```

```
## [1] 3
```

```r
N<-sum(exe3_table_a[,2])
N
```

```
## [1] 27
```

```r
exe3_MSB=(Treat_SS)/(k-1)
exe3_MSW=Residual_SS/(N-k)
F_ratio=exe3_MSB/exe3_MSW
F_ratio
```

```
## [1] 4.748018
```

```r
p_value=pf(F_ratio, (k-1), (N-k), lower.tail = FALSE)
p_value
```

```
## [1] 0.01830686
```

To check your work, use `aov` as illustrated in the chunk below:

```r
#Evaluate this chunk by setting eval=TRUE above.
summary(aov(CarbonLoss ~ factor(Fertilizer), data=exe3_table_a))
```

```
##                    Df Sum Sq Mean Sq
## factor(Fertilizer)  2  1.382   0.691
```

The press release associated with this paper (https://aces.illinois.edu/news/study-reveals-nitrogen-fertilizers-deplete-soil-organic-carbon) claims that "Study Reveals that Nitrogen Fertilizers Deplete Soil Organic Carbon". Do these data support that claim? Consider the commentary at https://acsess.onlinelibrary.wiley.com/doi/10.2134/jeq2008.0001le and https://acsess.onlinelibrary.wiley.com/doi/full/10.2134/jeq2010.0001le .

Exercise 4

There is a web site (https://www.wrestlestat.com/rankings/starters) that ranks college wrestlers using an ELO scoring system (https://en.wikipedia.org/wiki/Elo_rating_system). I was curious how well the rankings predicted performance, so I gathered data from the 2018 NCAA Wrestling Championships (https://i.turner.ncaa.com/sites/default/files/external/gametool/brackets/wrestling_d1_2018.pdf). Part of the data are on D2L in the file `elo.csv`. You will need to download the file to your computer for this exercise.

Read the data below and print a summary. The data were created by writing a data frame from R to csv (`write.csv`), so the first column is row number and does not have a header entry (the header name is an empty string).

```
exe4_table=read.csv("elo.csv", header=TRUE)
summary(exe4_table)
```

```
##        X             Weight       Conference             ELO
##  Min.   :  3.0   Min.   :125.0   Length:329         Min.   :1228
##  1st Qu.:180.0   1st Qu.:141.0   Class :character   1st Qu.:1342
##  Median :376.0   Median :157.0   Mode  :character   Median :1372
##  Mean   :377.3   Mean   :170.9                      Mean   :1379
##  3rd Qu.:567.0   3rd Qu.:184.0                      3rd Qu.:1410
##  Max.   :761.0   Max.   :285.0                      Max.   :1584
##  ActualFinish       ExpectedFinish
##  Length:329         Length:329
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

Each row corresponds to an individual wrestler, his weight class and collegiate conference. The WrestleStat ELO score is listed, along with his tournament finish round (i.e. `AA` = 1-8 place, `cons 12` = lost in the final consolation round, etc.). I calculated an expected finish based on his ELO ranking within the weight class, where `E[AA]` = top 8 ranked, expected to finish as AA, etc.

Produce group summaries or plots to answer the following:

- What are the mean and standard deviations of `ELO` by `ExpectedFinish` and by `ActualFinish`?

```
#EF_mean=aggregate(ELO~ExpectedFinish, data=exe4_table, FUN=mean)
EF_mean=aggregate(exe4_table[,4], by=list(exe4_table$ExpectedFinish), FUN=mean,na.rm=TRUE)
names(EF_mean)[2]<-"mean_of_ELO(by ExpectedFinish)"
EF_mean
```

```
##      Group.1 mean_of_ELO(by ExpectedFinish)
## 1      E[AA]                       1451.336
## 2 E[cons 12]                       1395.442
## 3 E[cons 16]                       1379.404
## 4 E[cons 24]                       1357.369
## 5 E[cons 32]                       1334.704
## 6      E[NQ]                       1332.821
```

```
#EF_sd=aggregate(ELO~ExpectedFinish, data=exe4_table, FUN=sd)
EF_sd=aggregate(exe4_table[,4], by=list(exe4_table$ExpectedFinish), FUN=sd,na.rm=TRUE)
names(EF_sd)[2]<-"sd_of_ELO(by ExpectedFinish)"
EF_sd
```

```
##      Group.1 sd_of_ELO(by ExpectedFinish)
## 1      E[AA]                     41.04978
## 2 E[cons 12]                     17.77768
## 3 E[cons 16]                     13.11593
```

```
## 4 E[cons 24]                     16.02282
## 5 E[cons 32]                     18.02051
## 6      E[NQ]                     52.69272
```

```
#AF_mean=aggregate(ELO~ActualFinish, data=exe4_table, FUN=mean)
AF_mean=aggregate(exe4_table[,4], by=list(exe4_table$ActualFinish), FUN=mean,na.rm=TRUE)
names(AF_mean)[2]<-"mean_of_ELO(by ActualFinish)"
AF_mean
```

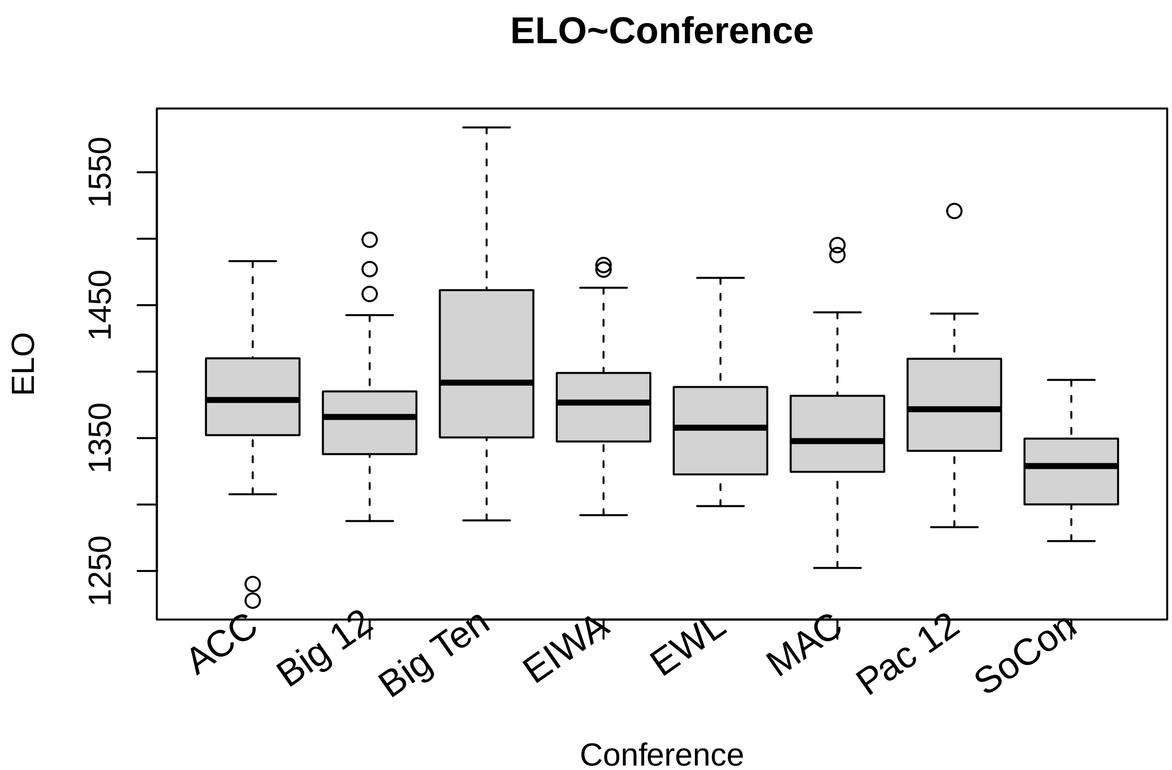
```
##   Group.1 mean_of_ELO(by ActualFinish)
## 1      AA                     1444.556
## 2 cons 12                     1400.708
## 3 cons 16                     1371.745
## 4 cons 24                     1355.130
## 5 cons 32                     1333.270
## 6 cons 33                     1343.795
```

```
#AF_sd=aggregate(ELO~ActualFinish, data=exe4_table, FUN=sd)
AF_sd=aggregate(exe4_table[,4], by=list(exe4_table$ActualFinish), FUN=sd,na.rm=TRUE)
names(AF_sd)[2]<-"sd_of_ELO(by ActualFinish)"
AF_sd
```

```
##   Group.1 sd_of_ELO(by ActualFinish)
## 1      AA                   50.93285
## 2 cons 12                   29.22633
## 3 cons 16                   34.28861
## 4 cons 24                   30.95125
## 5 cons 32                   34.08563
## 6 cons 33                   28.30588
```

- Do all conferences have similar quality, or might we suspect one or more conferences have better wrestlers than the rest? That is, what is the relationship between `Conference` and `ELO`?

```
#exe4_table$Conference<-as.factor(exe4_table$Conference)
#plot(ELO~Conference, data=exe4_table,main="ELO~Conference" )
table1=aggregate(ELO~Conference, exe4_table, max, na.rm=TRUE)
boxplot(ELO~Conference, data=exe4_table, main="ELO~Conference",
        xlab="Conference", ylab="ELO", xaxt="n")
axis(side =1, labels=FALSE)
n<-length(table1[,1])
text(x=(1:n),
y=par("usr")[3]-0.45,
labels=c("ACC", "Big 12", "Big Ten", "EIWA", "EWL", "MAC", "Pac 12", "SoCon"),
xpd=NA,
srt=35,
cex=1.2,
adj=1)
```




```
mean_elo<-aggregate(ELO~Conference, exe4_table, mean, na.rm=TRUE)
names(mean_elo)[2]<-"Mean"
mean_elo
```

```
##   Conference     Mean
## 1        ACC 1377.845
## 2     Big 12 1368.431
## 3    Big Ten 1407.672
## 4       EIWA 1377.503
## 5        EWL 1361.572
## 6        MAC 1358.482
## 7     Pac 12 1375.496
## 8      SoCon 1327.481
```

```
#No, not all conferences have the same quality.
#and I would like to suspect that
#there is one or more conference have better wrestlers
#than the rest based on my box plot,
#I think Conference Big Ten has better quality than the rest.
#it has the greatest wrestler which has the largest ELO;
#and the mean of ELO for Conference Big Ten is the largest.
```

- How well does ELO predict finish? That is, what is the relationship between `ExpectedFinish` and `ActualFinish`? Use a contingency table or mosaic plot to show how often, say, and `E[AA]` finish corresponds to an `AA` finish.

```
exe4_table3=table(exe4_table$ExpectedFinish,exe4_table$ActualFinish )
exe4_table3
```

```
##
##                AA cons 12 cons 16 cons 24 cons 32 cons 33
```

```
##   E[AA]       57      13       7       3       0       0
##   E[cons 12]  9       13       4       8       2       0
##   E[cons 16]  6        5       7      11       6       1
##   E[cons 24]  1        6       8      29      17       5
##   E[cons 32]  1        0       6      16      22       1
##   E[NQ]       6        3       8      12      33       3
```

```
#exe4_table$ExpectedFinish<-as.factor(exe4_table$ExpectedFinish)
#exe4_table$ActualFinish<-as.factor(exe4_table$ActualFinish)
#plot(ExpectedFinish~ActualFinish, exe4_table, main="Categorical vs Categorical")

#Based on the contingency table and the percentage,
#ELO predict finish did a good job.
#As we can see,
#almost all the numbers at the diagonal of the contingency table
#are larger than the numbers on the same row.
#the numbers at the diagonal of the contingency table
#means it predicted right.
```

- Does this data set include non-qualifiers? (The NCAA tournament only allows 33 wrestlers per weight class).

```
exe4_table4=aggregate(ELO~Weight, exe4_table, length)
names(exe4_table4)[2]<-"Count"
exe4_table4
```

```
##    Weight Count
## 1     125    33
## 2     133    33
## 3     141    33
## 4     149    33
## 5     157    33
## 6     165    33
## 7     174    33
## 8     184    33
## 9     197    32
## 10    285    33
```

```
#This data set doesn't include non-qualifiers,
#because it has ELO score and
#the number of wrestlers for each weight class
#is not greater than 33.

x1<-c(1,2,3)
x2<-c(7,8,9)
sum<-apply
m<-matrix(c(x1,x2), nrow=2, byrow=TRUE)
m
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    7    8    9
```

```
rownames(m)<-c("x1", "x2")
colnames(m)<-c("n1","n2","n3")
m
```

```
##    n1 n2 n3
## x1  1  2  3
## x2  7  8  9
```

```r
d<-data.frame(x1,x2)
d
```

```
##   x1 x2
## 1  1  7
## 2  2  8
## 3  3  9
```