# 5 Data Tables

07-03-2020

## Instructions

There are six exercises below. You are required to provide five solutions, with the same options for choosing languages as with the last exercise. You can provide solutions in two languages for one exercise only (for example, Ex. 1,2,3,5 in R and Ex. 1 in SAS is acceptable, Ex. 1,2,3 in SAS and Ex. 1,2 in R is not).

*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 midterm and final projects.

### 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.

I'm also including data vectors that can be used in some exercises.

```
CaloriesPerServingMean <- c(268.1, 271.1, 280.9, 294.7, 285.6, 288.6, 384.4)
CaloriesPerServingSD <- c(124.8, 124.2, 116.2, 117.7, 118.3, 122.0, 168.3)
Year <- c(1936, 1946, 1951, 1963, 1975, 1997, 2006)

cohen.d <- function(m_1,m_2,s_1,s_2) {
  s.pooled <- sqrt((s_1^2 + s_2^2)/2)
  return(abs(m_1-m_2)/s.pooled)
}

required.replicates <- function(cv,dif,alpha = 0.05, beta = 0.20) {
  z_alpha <- qnorm(1-alpha/2)
  z_beta <- qnorm(1-beta)
  return(ceiling(2*((cv/dif)^2)*((z_alpha + z_beta)^2)))
}


rule.thumb <- function(CV,Diff){
  return(ceiling(16/((Diff/CV)^2)))
}
```

```r
required.replicates.org <- function(m_1,s_1,m_2,s_2,alpha = 0.05, beta =
0.20) {
  s.pooled <- sqrt((s_1^2 + s_2^2)/2)
  z_alpha <- qnorm(1-alpha/2)
  z_beta <- qnorm(1-beta)
  cv <- s.pooled/((m_1 + m_2)/2)
  dif <- (m_1 - m_2)/((m_1 + m_2)/2)
  return(ceiling(2*((cv/dif)^2)*((z_alpha + z_beta)^2)))
}
```

## Warning

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/)

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

## Exercise 1.

This exercise will repeat Exercise 1 from Homework 4, but using a data table.

### Part a.

Create a data table or frame with 4 columns:

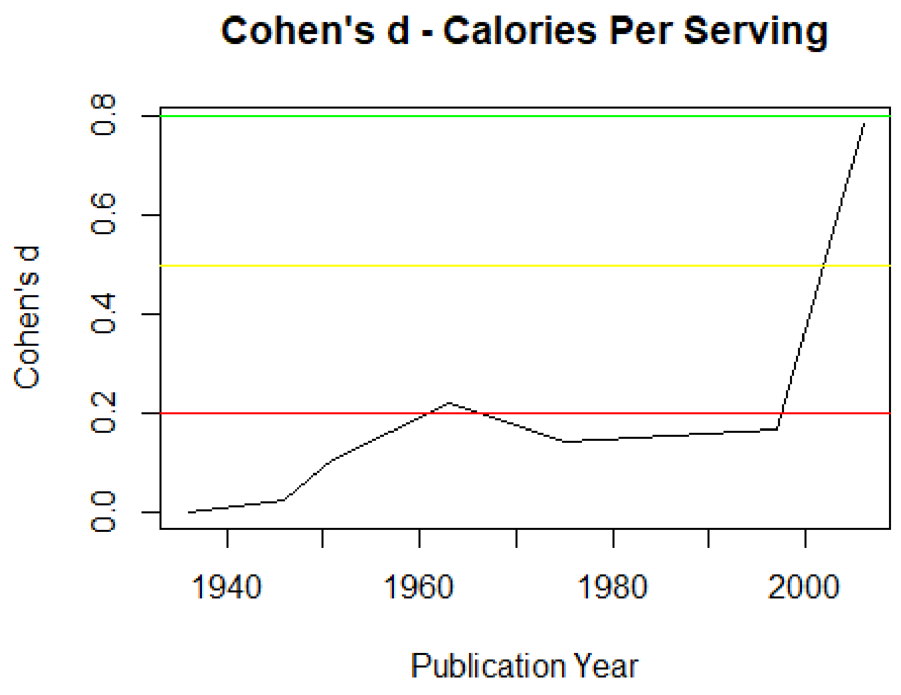
- Define `M1` to be the 7 means for Calories per Serving from Wansink Table 1
- Define `M2` be the mean for Calories per Serving, 1936
- Define `S1` to be the 7 standard deviations from Wansink Table 1
- Define `S2` be the standard deviation for Calories per Serving, 1936

Calculate Cohen's $d$ for each `M1` vs `M2` using the data columns from your table as arguments and append this to your data data as `D`. Add an additional table column, `Year` for the publication years 1936,1946, … ,2006. Plot `D` as the dependent variable and `Year` as the independent variable.

Add to this plot three horizontal lines, one at $d = 0.2$, one at $d = 0.5$ and one at $d = 0.8$. You should use different colors or different styles for each line. Should any of the effect sizes be considered *large*?

```r
CaloriesPerServing.dat <- data.frame(
   M1 = CaloriesPerServingMean,
   M2 = CaloriesPerServingMean[1],
   S1 = CaloriesPerServingSD,
   S2 = CaloriesPerServingSD[1]
)
CaloriesPerServing.dat$D <-
```

```r
cohen.d(CaloriesPerServing.dat$M1,CaloriesPerServing.dat$M2,
             CaloriesPerServing.dat$S1,CaloriesPerServing.dat$S2)

CaloriesPerServing.dat$Year <- Year

plot(CaloriesPerServing.dat$Year, CaloriesPerServing.dat$D, main="Cohen's d - 
Calories Per Serving", type="l",
     ylab = "Cohen's d", xlab = 'Publication Year')
abline(a = 0.2,b = 0, col = 'red')
abline(a = 0.5, b = 0, col = 'yellow')
abline(a = 0.8, b = 0, col = 'green')
```



**Comment** - The Cohen's D for 1936 vs. 2006 is very close to large, but not quite to the 0.8 mark.

## Exercise 2

### Part a.

You will repeat the calculations from Homework 4, Ex 2, but this time, using a data table. However, instead of a $5 \times 6$ matrix, the result with be a table with 30 rows, each corresponding to a unique combination of CV from $8, 12, \ldots, 28$ and Diff from $5, 10, \ldots, 25$.

The table should look something like

$$\left\{\begin{array}{cc} CV & Diff \\ 8 & 5 \\ 8 & 10 \\ 8 & 15 \\ \vdots & \vdots \\ 12 & 5 \\ 12 & 10 \\ 12 & 15 \\ \vdots & \vdots \\ 28 & 5 \\ 28 & 10 \\ 28 & 15 \end{array}\right\}$$

## Part b.

Add to the table a column `D` by calculating Cohen's $d$ for each row of the table. Also calculate for each row a required replicates using the $z$-score formula and name this `RR`. Finally, calculate the required replicates using the rule of thumb for each row and name this `Thumb`.

**Do not print this table in the typeset document**. Instead, we will examine graphs below.

If you choose SAS, you can use the framework code from the first exercise.

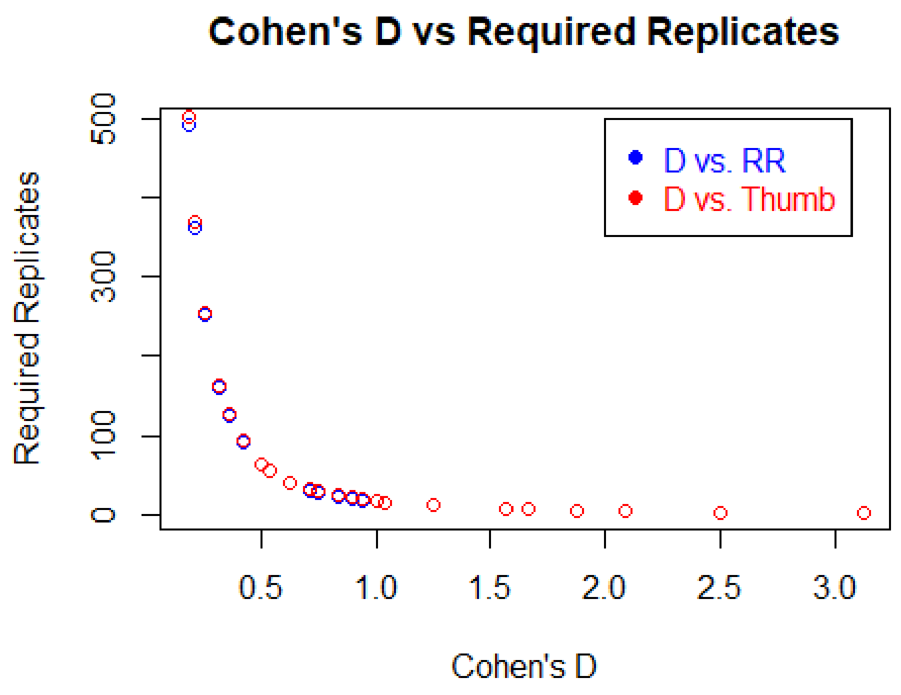
```
Ex2.dat <- data.frame(
CV = matrix(seq(8,28,by=4),nrow=30),
Diff = matrix(seq(5,25,by=5),nrow=30)
)
Ex2.dat <- Ex2.dat[order(Ex2.dat$CV,Ex2.dat$Diff),]

Ex2.dat$D <- abs(Ex2.dat$Diff)/Ex2.dat$CV

Ex2.dat$RR <- required.replicates(Ex2.dat$CV,Ex2.dat$Diff)

Ex2.dat$Thumb <- rule.thumb(Ex2.dat$CV,Ex2.dat$Diff)
```

## Part c.

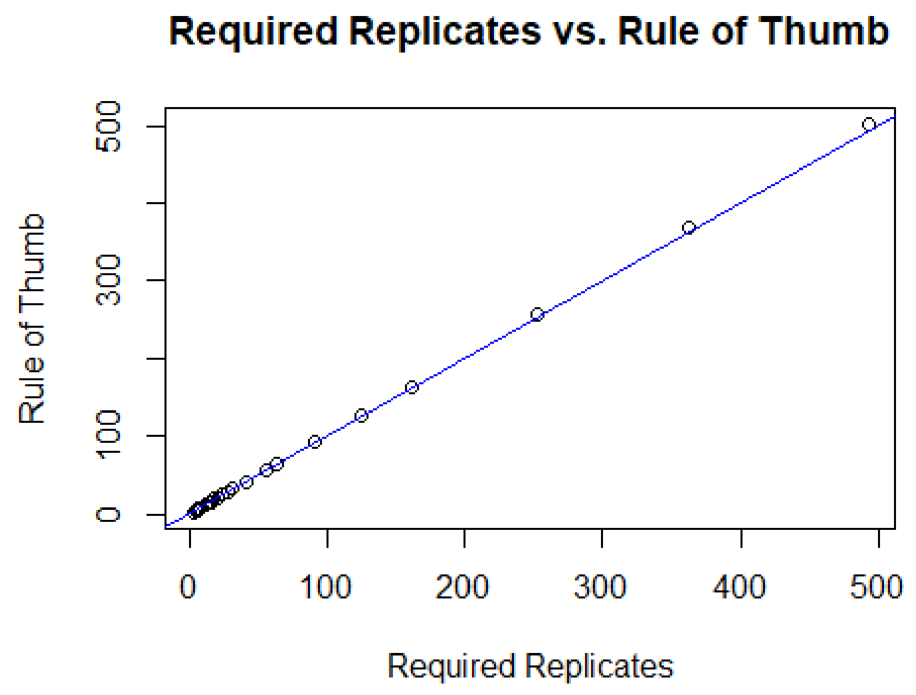
Produce one graph showing the relationship between Cohen's $d$ and required replicates. Plot `D` as the independent variable and `RR` as the dependent variable. Add to this graph an additional plot with Plot `D` as the independent variable and `Thumb` as the dependent variable. Use different colors, points or lines for each plot.

Produce a second graph showing the relationship between the two formula for determining required replicates. Plot `RR` as the independent variable and `Thumb` as the dependent variable. Add a line with intercept = 0 and slope = 1 to indicate an exact linear relationship between the two values.

```r
plot(Ex2.dat$D,Ex2.dat$RR,type = 'p',main = "Cohen's D vs Required
Replicates",
   xlab = "Cohen's D",ylab = 'Required Replicates', col = 'blue')
lines(Ex2.dat$D,Ex2.dat$Thumb, type = 'p', col = 'red')
legend(2,500,legend = c("D vs. RR", "D vs. Thumb"),col = c('blue','red'),
       pch = c(19,19),text.col = c('blue','red'))
```



```r
plot(Ex2.dat$RR,Ex2.dat$Thumb,type = 'p',main = 'Required Replicates vs. Rule
of Thumb',
   xlab = 'Required Replicates',ylab = 'Rule of Thumb')
abline(a = 0, b = 1, col = 'blue')
```



## Exercise 3

We will be working with data from Table 1 and Table 2, https://peerj.com/articles/4428/.

### Part a

Download the file `lacanne2018.csv` from D2L and read the file into a data frame. Print a summary of the table. This file was exported from the raw data file linked at https://doi.org/10.7717/peerj.4428/supp-1

```
lac18 = 'lacanne2018.csv'
lacanne2018.dat <- read.csv(lac18,header = TRUE)
summary(lacanne2018.dat)
```

```
##      Order                 Town    State       Lat             Lon
##  Min.   : 1.000   Bismarck :4   MN:1   Min.   :40.32   Min.   :-100.60
##  1st Qu.: 4.750   Arlington:2   ND:4   1st Qu.:43.34   1st Qu.: -98.98
##  Median : 8.500   Bladen   :2   NE:4   Median :44.43   Median : -97.37
##  Mean   : 9.375   White    :2   SD:7   Mean   :44.11   Mean   : -85.95
##  3rd Qu.:13.750   York     :2          3rd Qu.:45.31   3rd Qu.: -96.60
##  Max.   :19.000   Gary     :1          Max.   :47.14   Max.   :  96.36
##                   (Other)  :3
##  Organic Cover Insecticide Pesticide Tillage Grazed   Composite
##  N:15    N:8   N: 6        N: 2      N:9     N:12   Min.   :0.000
##  Y: 1    Y:8   Y:10        Y:14      Y:7     Y: 4   1st Qu.:0.000
##                                                                      Median :2.000
##                                                                      Mean   :1.938
##                                                                      3rd Qu.:3.250
##                                                                      Max.   :5.000
##
##       POM          Study.Year
##  Min.   :3.850   Min.   :2015
##  1st Qu.:4.702   1st Qu.:2015
##  Median :6.015   Median :2015
##  Mean   :6.026   Mean   :2015
##  3rd Qu.:7.402   3rd Qu.:2016
##  Max.   :8.180   Max.   :2016
##
```

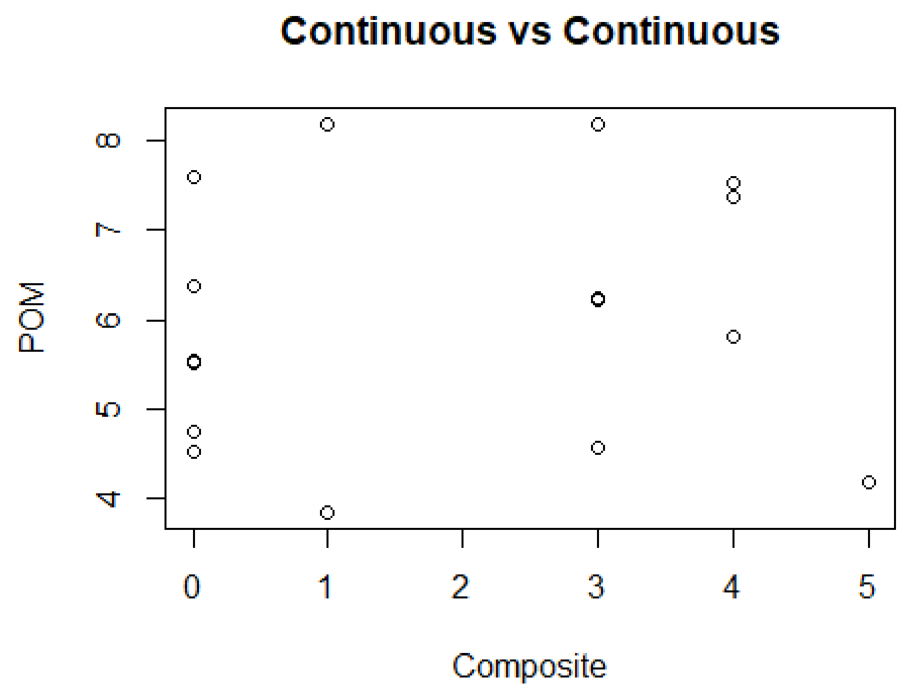
## Part b

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

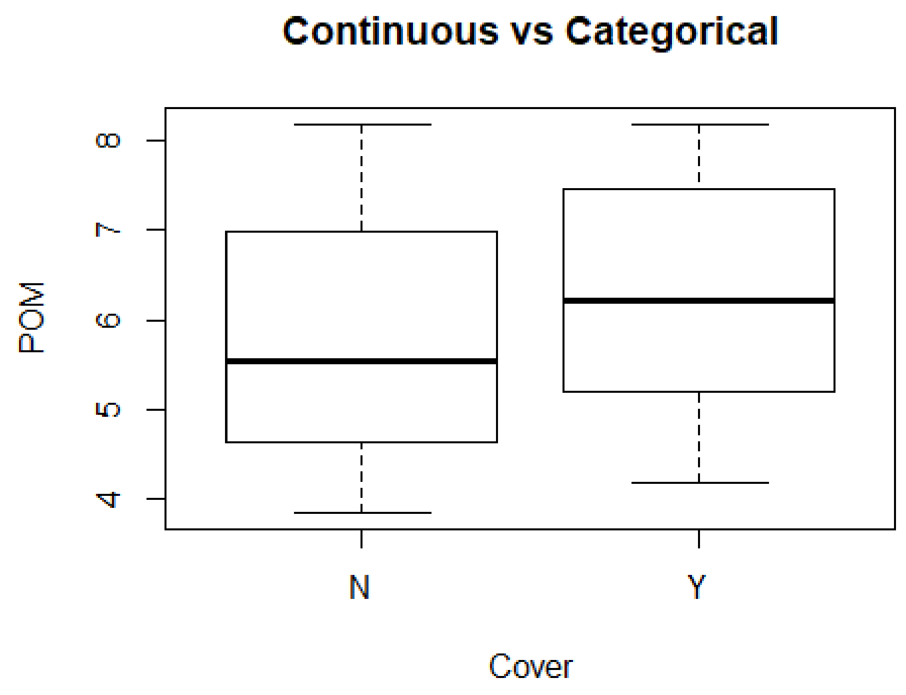
1. POM vs Composite
2. POM vs Cover
3. Cover vs State

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.

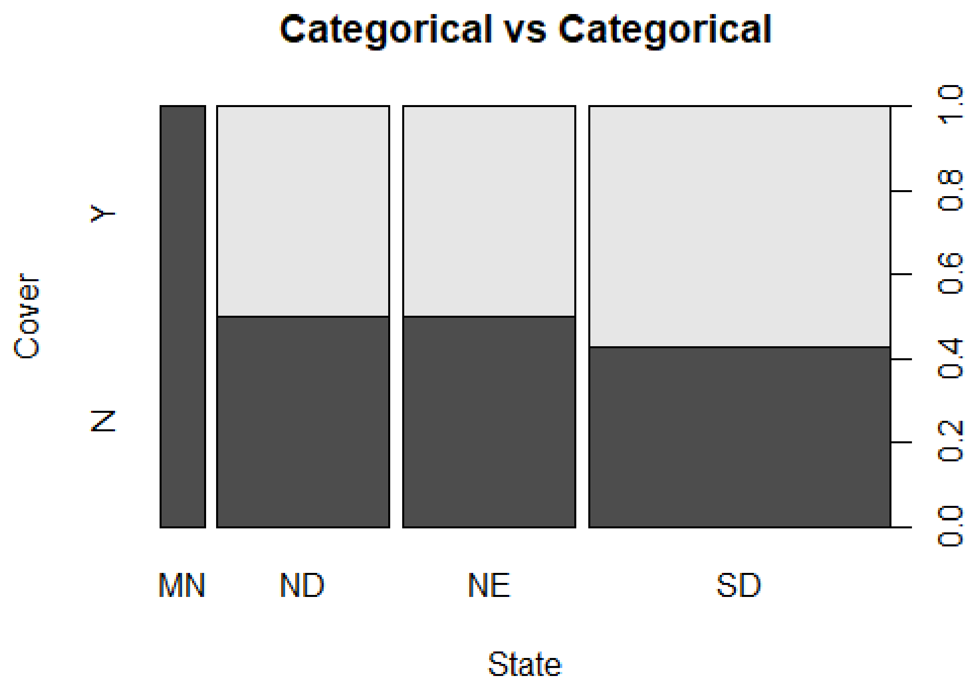
```
plot(POM ~ Composite,data = lacanne2018.dat,main = 'Continuous vs
Continuous')
```



```
plot(POM ~ Cover,data = lacanne2018.dat, main = 'Continuous vs Categorical')
```



```
plot(Cover ~ State,data = lacanne2018.dat, main = 'Categorical vs
Categorical')
```



## Exercise 4

Calculate a one-way analysis of variance from the data in Exercise 3.

## Option A

Let $y$ be the `POM`. Let the $k$ treatments be `Composite`. Let $T_i$ be the `POM` total for `Composite` $i$ and let $r_i$ be the number of observations for `Composite` $i$. Denote the total number of observations $N = \sum r_i$.

### Part a

Find the treatment totals

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

and replicates per treatment

$$\mathbf{r} = r_1 \dots r_k$$

from the Lacanne data, using group summary functions and compute a grand total $G$ for `POM`. Print $\mathbf{T}$, $\mathbf{r}$ 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).

### Part b

Calculate sums of squares as

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

and calculate $MSB = (\text{Treatments SS})/(k-1)$ and $MSW = (\text{Residual SS})/(N-k)$.

### Part c.

Calculate an F-ratio and a $p$ for this $F$, using the $F$ distribution with $k-1$ and $N-k$ degrees of freedom. Use $\alpha = 0.05$.

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

```r
#Evaluate this chunk by setting eval=TRUE above.
summary(aov(POM ~ factor(Composite), data=lacanne.dat))
```

## Option B

You may reuse code from Exercise 6, Homework 4. Use group summary functions to calculate means, standard deviations and replicates from the pumpkin data, then calculate $MSW$ and $MSB$ as previously. Report the F-ratio and $p$ value as above.

(Hint - show that)

$$\frac{\sum y^2 - \sum \frac{T_i^2}{r_i}}{N-k} = MSW = \frac{\sum_i (n_i - 1)s_i^2}{N-k}$$

## Exercise 5

### 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.

```
temp <- read.table('https://www.itl.nist.gov/div898/strd/anova/SiRstvt.dat',
                   header = FALSE, skip = 60, nrows = 5)
colnames(temp) <- c('col1','col2','col3','col4','col5')
write.csv(temp,'instrument.csv',row.names = FALSE)
Ex5.dat <- read.csv('instrument.csv')
```

**Comment** - I used the *read.table* function to import just the data and ignore the comment information at the top. I did this by skipping to row 61 and only importing 5 rows. In addition, I added column names as col1 … col5. At this point, I could have just used the *temp* data frame for analysis.

## 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

```
instr_mean <- mapply(mean,Ex5.dat)
instr_sd <- mapply(sd,Ex5.dat)
instr_length <- mapply(length,Ex5.dat)
instr_summary.dat <- data.frame(Means = instr_mean,Std_Dev = instr_sd,
                                Sample_Size = instr_length,
                                row.names =
c('col1','col2','col3','col4','col5')
                               )
instr_summary.dat

##          Means    Std_Dev Sample_Size
## col1 196.2431 0.08747329           5
## col2 196.2443 0.13797498           5
## col3 196.1670 0.09372413           5
## col4 196.1481 0.10422674           5
## col5 196.1432 0.08844797           5
```

Determine the largest and smallest means, and their corresponding standard deviations, and calculate an effect size and required replicates to experimentally detect this effect.

If you defined functions in the previous exercises, you should be able to call them here.

```
cat('Max Mean =',max(instr_summary.dat$Means),'\n')

## Max Mean = 196.2443

cat('Min Mean =',min(instr_summary.dat$Means),'\n')
```

```
## Min Mean = 196.1432
m_1 <- instr_summary.dat[2,1]
s_1 <- instr_summary.dat[2,2]
m_2 <- instr_summary.dat[5,1]
s_2 <- instr_summary.dat[5,2]

effect_size <- cohen.d(m_1,m_2,s_1,s_2)
RR <- required.replicates.org(m_1,s_1,m_2,s_2)

cat('Effect size =',effect_size,'\n')
## Effect size = 0.8720476
cat('Required Replicates =',RR,'\n')
## Required Replicates = 21
```

## Exercise 6

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).

```
elo.dat <- read.csv('elo.csv',row.names = 1)
summary(elo.dat)
##      Weight        Conference      ELO        ActualFinish
ExpectedFinish
##  Min.   :125.0   Big Ten:87   Min.   :1228   AA     :80    E[AA]     :80
##  1st Qu.:141.0   EIWA   :55   1st Qu.:1342   cons 12:40    E[cons 12]:36
##  Median :157.0   Big 12 :52   Median :1372   cons 16:40    E[cons 16]:36
##  Mean   :170.9   ACC    :40   Mean   :1379   cons 24:79    E[cons 24]:66
##  3rd Qu.:184.0   MAC    :34   3rd Qu.:1410   cons 32:80    E[cons 32]:46
##  Max.   :285.0   Pac 12 :25   Max.   :1584   cons 33:10    E[NQ]     :65
##                  (Other):36
```

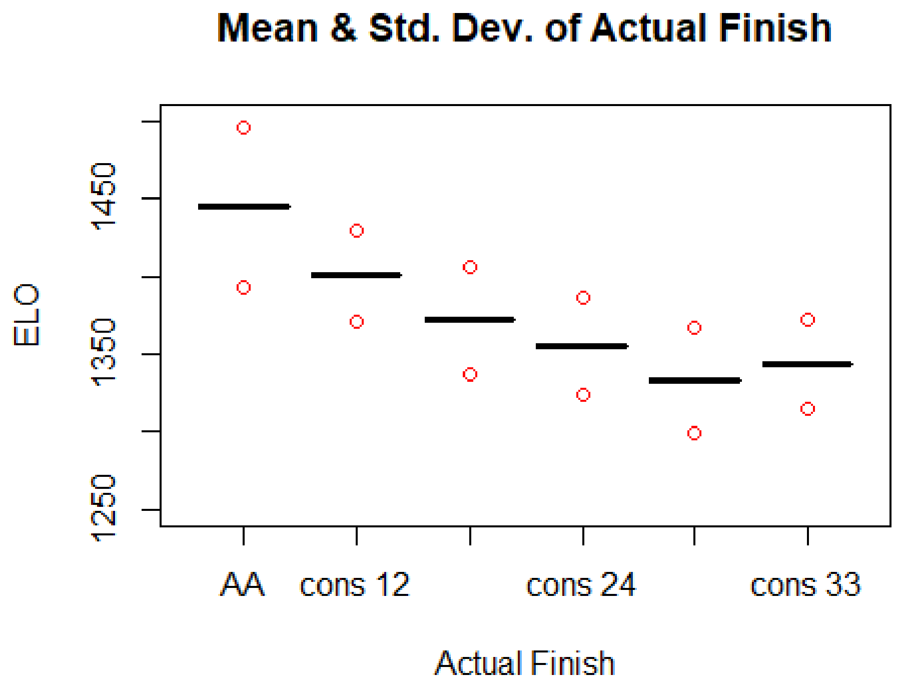
Each row corresponds to an individual wrestler, his weight class and collegiate conference. The WrestleStat ELO score is listed, along with his tournament finish round (i.e. AA = 1-8 place, `cons  12` = lost in the final consolation round, etc.). I calculated an expected finish based on his ELO ranking 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 Expected Finish and by Actual Finish?

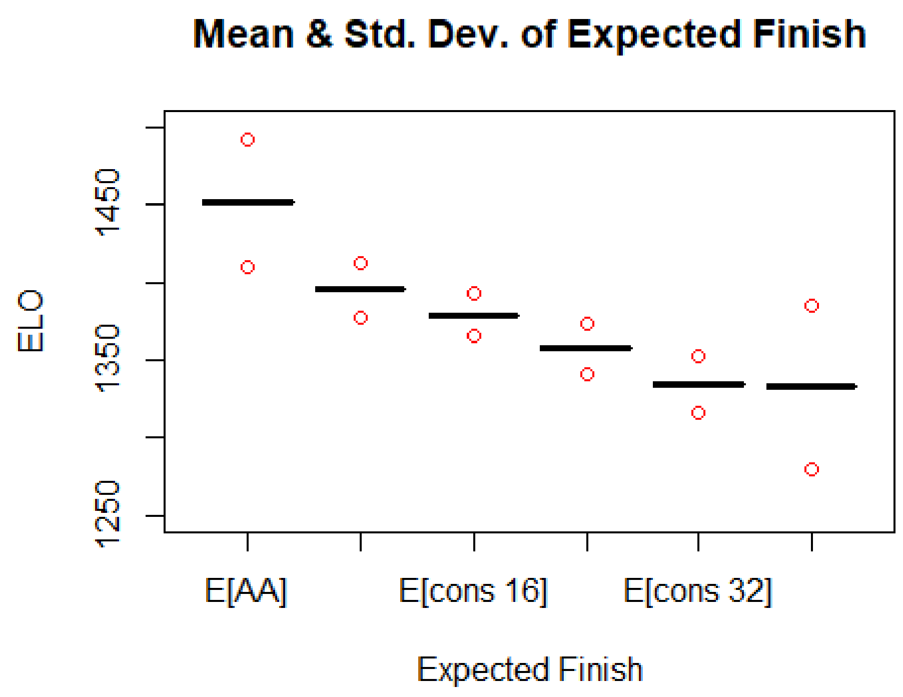
```
print('Means of Actual Finish')

## [1] "Means of Actual Finish"

af_mean <- aggregate(ELO ~ ActualFinish, data = elo.dat, 'mean', na.rm=TRUE)
af_mean

##   ActualFinish      ELO
## 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

print('Standard Deviations of Actual Finish')

## [1] "Standard Deviations of Actual Finish"

af_sd <- aggregate(ELO ~ ActualFinish, data = elo.dat, 'sd', na.rm=TRUE)
af_sd

##   ActualFinish      ELO
## 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

print('Means of Expected Finish')

## [1] "Means of Expected Finish"

ef_mean <- aggregate(ELO ~ ExpectedFinish, data = elo.dat, 'mean',
na.rm=TRUE)
ef_mean

##   ExpectedFinish      ELO
## 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
```

```r
print('Standard Deviations of Expected Finish')

## [1] "Standard Deviations of Expected Finish"

ef_sd <- aggregate(ELO ~ ExpectedFinish, data = elo.dat, 'sd', na.rm=TRUE)
ef_sd

##   ExpectedFinish      ELO
## 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

plot(af_mean$ELO ~ af_mean$ActualFinish,main = "Mean & Std. Dev. of Actual
Finish",
   xlab = "Actual Finish",ylab = 'ELO',ylim = c(1250,1500))
lines(af_sd$ActualFinish,af_mean$ELO+af_sd$ELO,type = 'p',lty = 3,col =
'red')
lines(af_sd$ActualFinish,af_mean$ELO-af_sd$ELO,type = 'p',lty = 3,col =
'red')
```



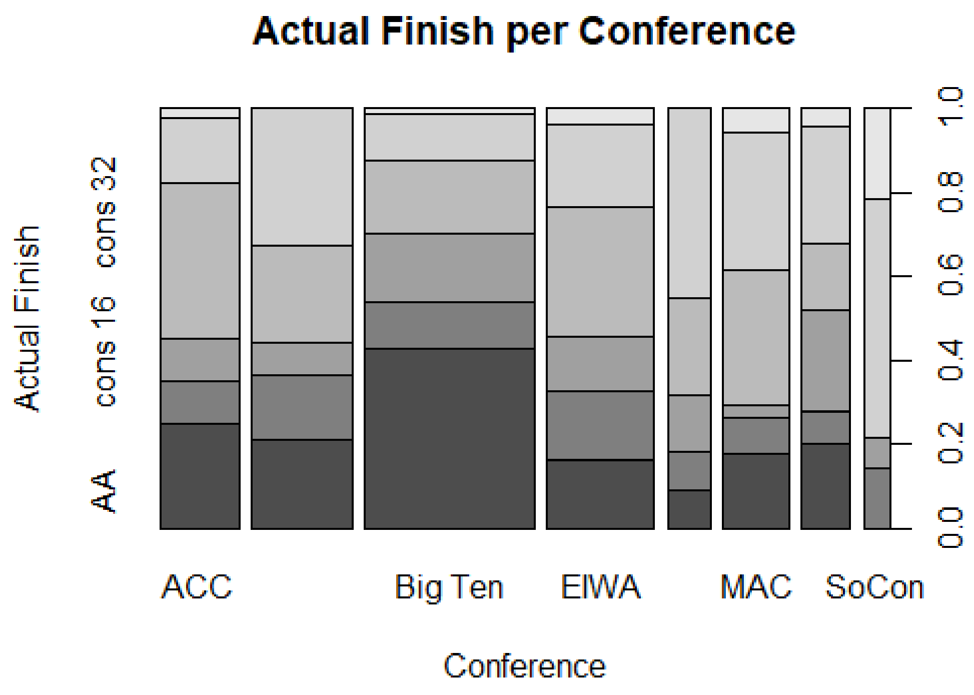
```r
plot(ef_mean$ELO ~ ef_mean$ExpectedFinish,main = "Mean & Std. Dev. of
Expected Finish",
   xlab = "Expected Finish",ylab = 'ELO',ylim = c(1250,1500))
lines(ef_sd$ExpectedFinish,ef_mean$ELO+ef_sd$ELO,type = 'p',lty = 3,col =
'red')
```

```
lines(ef_sd$ExpectedFinish,ef_mean$ELO-ef_sd$ELO,type = 'p',lty = 3,col =
'red')
```



- Do all conferences have similar quality, or might we suspect one or more conferences have better wrestlers than the rest? (You don't need to perform an analysis, just argue, based on the summary, if a deeper analysis is warranted).

```
print('Means of Conference')

## [1] "Means of Conference"

af_mean <- aggregate(ELO ~ Conference, data = elo.dat, 'mean', na.rm=TRUE)
af_mean

##   Conference      ELO
## 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

plot(elo.dat$Conference,elo.dat$ActualFinish,main = 'Actual Finish per
Conference',
     xlab = 'Conference', ylab = 'Actual Finish')
```



**Comment** - I believe a deeper analysis would be needed to determine if all conferences have similar quality. Looking at the number of wrestlers per conference, the Big Ten has 87 wrestlers vs the second highest conference of 55, which is 58% larger.

Looking at the mean of Elo score for each conference we see that the Big Ten has the 3rd highest elo score out of all the conferences. Looking at a Mosiac, we can see that the Big Ten has the largest number of AA actual finishes. This may suggest that the number of wrestlers is influencing the number of AA acutal finishes, but a deeper analysis is required.

- How well does ELO predict finish? Use a contingency table or mosaic plot to show how often, say, and `AA` finish corresponds to an `E[AA]` finish.

```
with(elo.dat, table(ActualFinish, ExpectedFinish))

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

**Comment** - Looking at the above contingency table, we can see that Elo does a fairly good job of predicting the AA Actual Finish with being correct 71% of the time. However, it does break down in the other finishes. For example, the cons 12 only predicts the correct finish

13% of the time. It does get better at the other extreme, however with the cons 32 predicting a correct result 48% of the time.

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

```
aggregate(ELO ~ Weight, data = elo.dat, length)
```

```
##    Weight ELO
## 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
```

**Comment** There is nothing labeled as non-qualifiers, unlike the categories for the expected finish.

Looking at the count of wrestlers for each weight class, there are only 32 wrestlers in the 197 weight class while all other weight classes have 33. This would suggest that non-qualifiers are not included in this dataset.