HU Extension **Assignment 01** E-63 Big Data Analytics

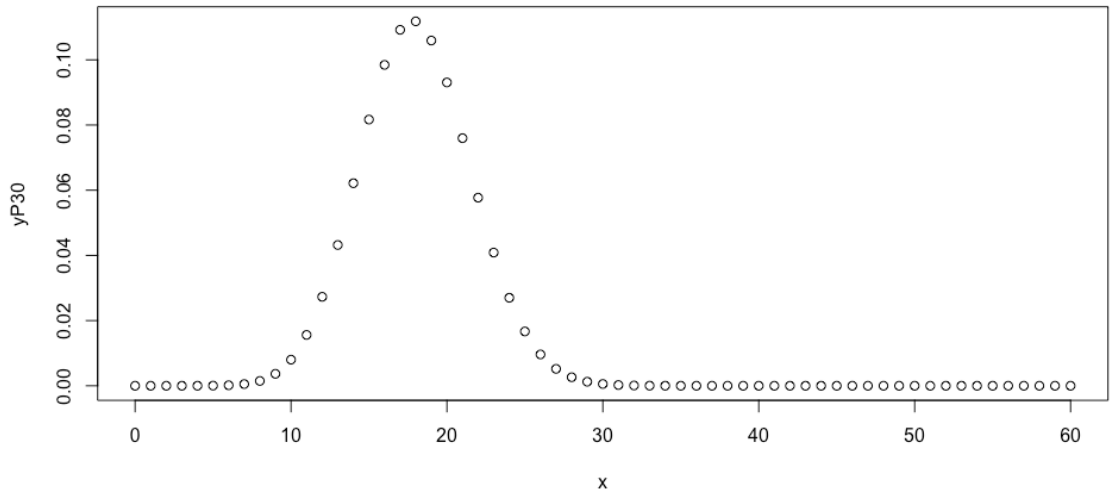
Handed out: 09/01/2017 Due by 11:59AM on Saturday, 09/09/2017

It is recommended that your solution for this assignment is implemented in R. If you insist, you can submit your solution in any language of your choice.

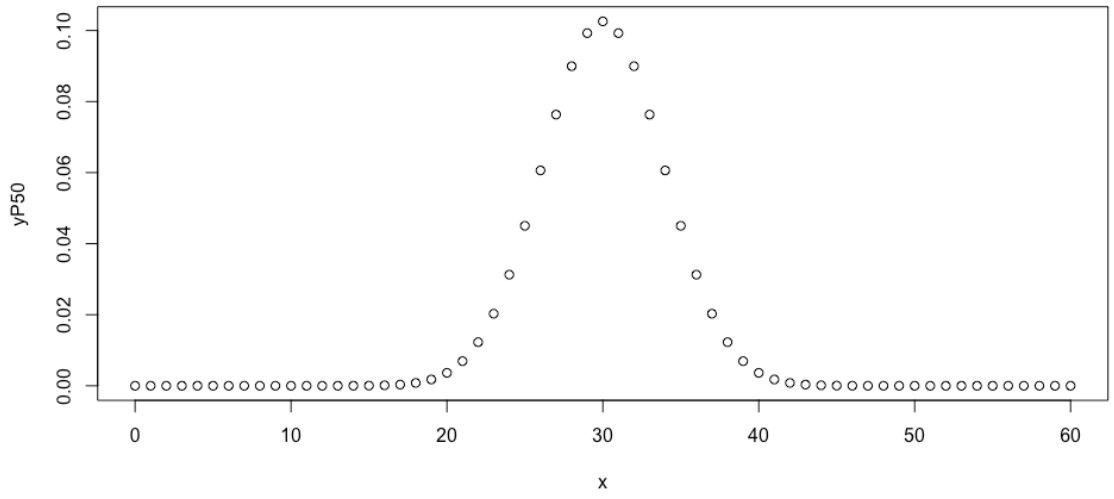
Problem 1.

Binomial distribution describes coin tosses with potentially doctored or altered coins. Value of p is the probability that head comes on top. If both the head and the tail have the same probability, p = 0.5. If the coin is doctored or altered, p could be larger or smaller. Plot on three separate graphs the binomial distribution for p = 0.3, p = 0.5 and p = 0.8 for the total number of trials n = 60 as a function of k, the number of successful (head up) trials.

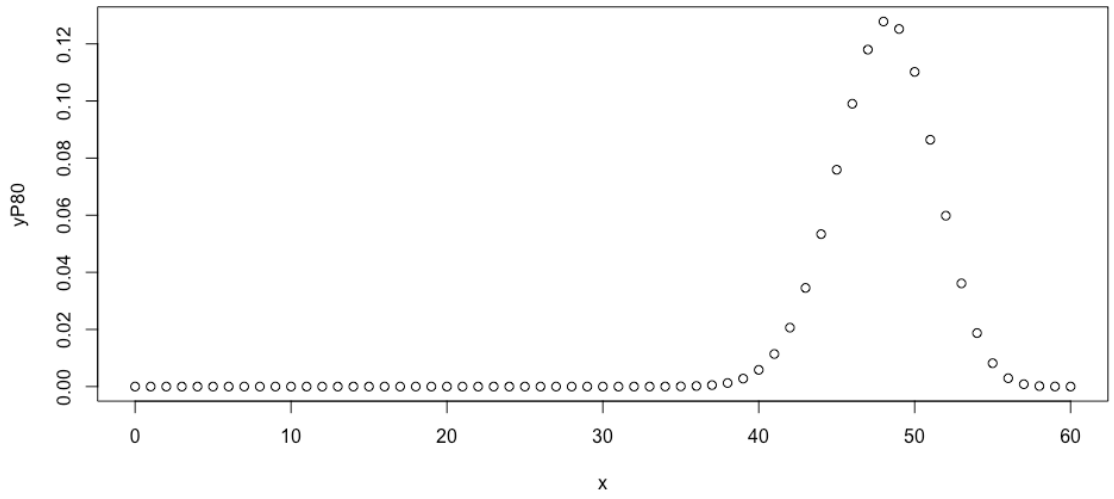
```
x <- seq(0, 60, by=1)
yP30 <- dbinom(x, 60, 0.3)
yP50 <- dbinom(x, 60, 0.5)
yP80 <- dbinom(x, 60, 0.8)

plot(x, yP30)
```



plot(x, yP50)



plot(x, yP80)

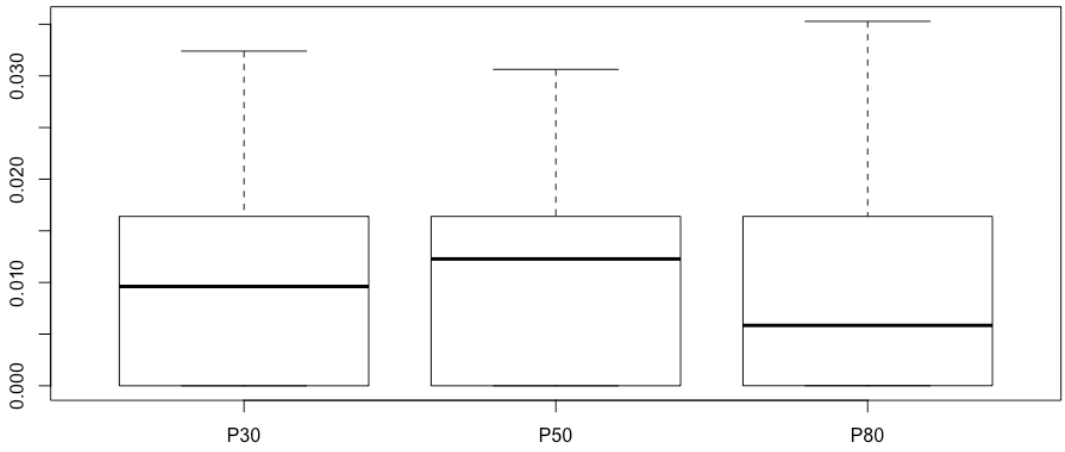


Subsequently, place all three curves on the same graph. For each value of p, determine 1st Quartile, median, mean, standard deviation and the 3rd Quartile. Present those values as a vertical box plot with the probability p on the horizontal axis.

```
df <- data.frame(P30=numeric(),
                 P50=numeric(),
                 P80=numeric(),
                 stringsAsFactors=FALSE)

df["1st_quar",]   = c(quantile(yP30)[2]["25%"], quantile(yP50)[2], quantile(yP80)[2])
df["median",]     = c(quantile(yP30), quantile(yP50), quantile(yP80))
df["mean",]       = c(mean(yP30), mean(yP50), mean(yP80))
df["stdev",]      = c(sd(yP30), sd(yP50), sd(yP80))
df["3rd_quar",]   = c(quantile(yP30)[4], quantile(yP50)[4], quantile(yP80)[4])
print(df)
```

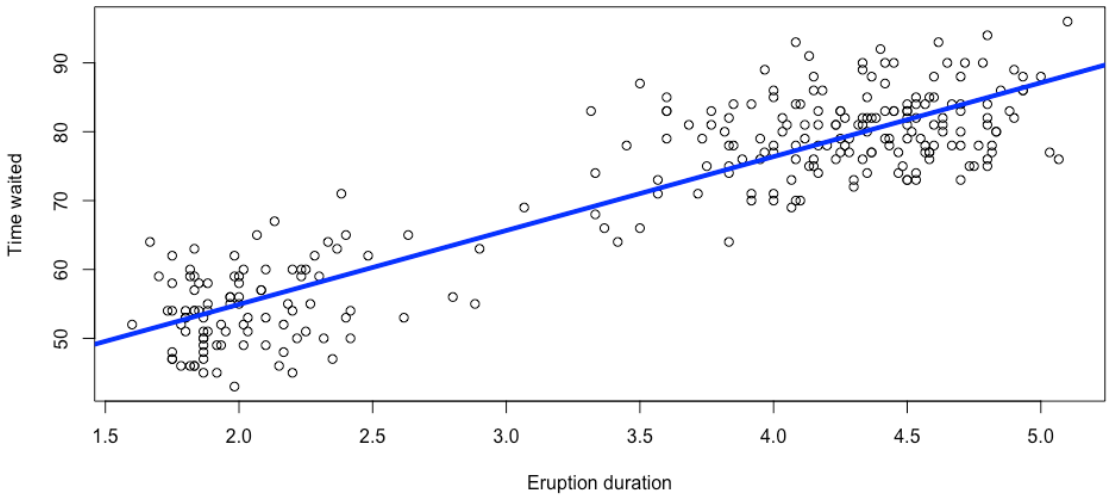
```
                                 P30              P50
1st_quar 0.000000000000746088661148297355282 8 0.000000000334981
median   0.000000000000000000000000000000424 0.000000000000746
mean     0.016393442622950820525273840644331 4 0.016393442622951
stdev    0.032397550132792343746679364357987 7 0.030629924444124
3rd_quar 0.009613404177267951059016048986904 9 0.012276881030422
                          P80
1st_quar 0.00000000000000000659
median   0.000008357379610580218 6
mean     0.016393442622950817055 8
stdev    0.035279806883377069770 5
3rd_quar 0.005842578514739280894 1
```

```
boxplot(df, horizontal = FALSE)
```



Problem 2.

```
library(MASS)
head(faithful)

# 1. We first find the range of eruption durations.

duration = faithful$eruptions;
range(duration)

# 2. Break the range into non-overlapping intervals.

breaks = seq(1.5, 5.5, by=0.5);
breaks

# 3. Classify the eruption durations according to which interval they fall into.

duration.cut = cut(duration, breaks, right=FALSE)
duration.freq

duration.freq = table(duration.cut);
duration.freq

duration.freq = cbind(duration.freq)
duration.freq

# 4. "Compute the frequency of eruptions in each interval" or count the number of
#    eruption durations in each interval.

duration.freq = table(duration.cut)
duration.relfreq = duration.freq / nrow(faithful);
duration.relfreq
duration.cut

old = options(digits=3);
cbind(duration.freq, duration.relfreq)

duration = faithful$eruptions; # the eruption
waiting = faithful$waiting; # the waiting interval
head(cbind(duration, waiting))

model = lm(waiting ~ duration, data = faithful)
print(model)
plot(duration, waiting, xlab="Eruption duration", ylab="Time waited")
abline(model, col='blue', lwd = 4)
```



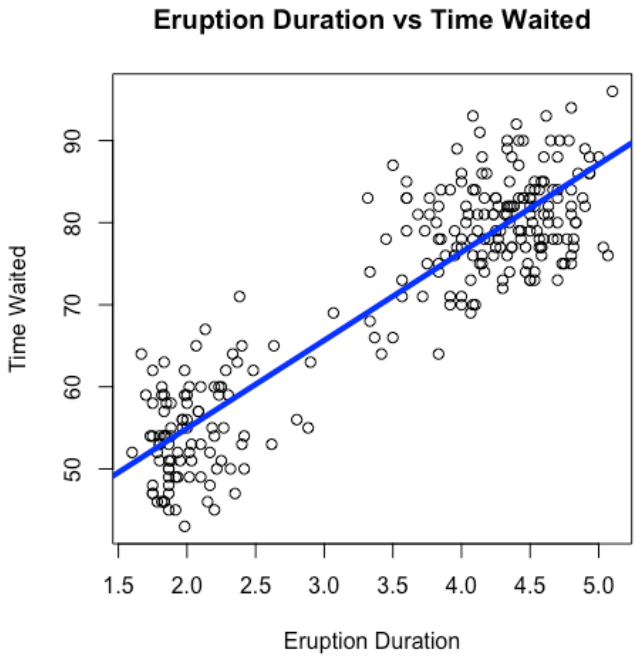
Problem 3. **Calculate the covariance matrix of the `faithful` data. Determine the eigenvalues and eigenvectors of that matrix. Demonstrate that two eigenvectors are mutually orthogonal. Examine whether the eigenvector with the larger eigenvalue is parallel with line discovered by `lm()` function it the previous problem.**

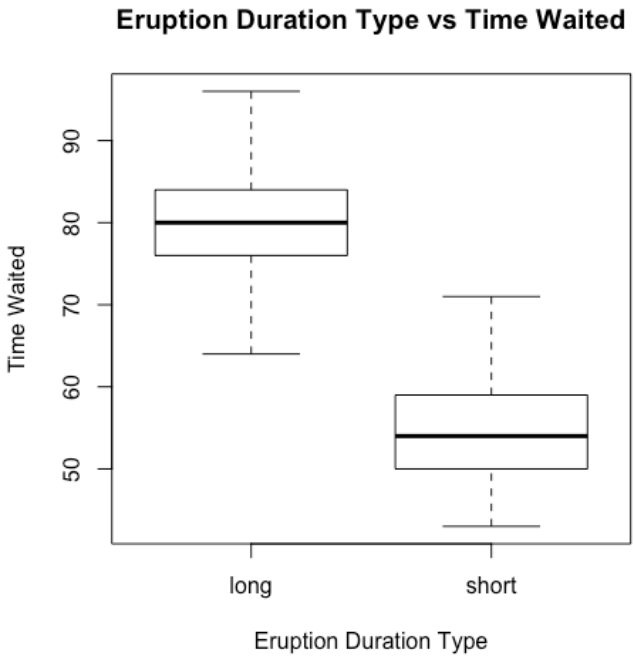
```
# By using the function eigen the eigenvalues and eigenvectors of the covariance matrix are computed
Eigenvalues <- eigen(cov(faithful))$values
print(Eigenvalues)
Eigenvectors <- eigen(cov(faithful))$vectors
print(Eigenvectors)

> Eigenvalues <- eigen(cov(faithful))$values
> print(Eigenvalues)
[1] 185.882   0.244

# Prove that two eigin vectors are Orthogonal you multiply them using a Scalar Product
# http://hyperphysics.phy-astr.gsu.edu/hbase/vsca.html
# http://www.purplemath.com/modules/mtrxmult.htm

# scalar product of two values should result in 0 (from the lecture)
# (A_x * B_x) + (A_y * B_y)
#       [,1]    [,2]
# [1,] 0.0755 -0.9971
# [2,] 0.9971  0.0755

# (0.0755 * 0.9971) + (-0.9971 * 0.0755) = 0
```

Problem 4.

You noticed that eruptions clearly fall into two categories, short and long. Let us say that short eruptions are all which have duration shorter than 3.1 minute. Add a new column to data frame `faithful` called `type`, which would have value `'short'` for all short eruptions and value `'long'` for all long eruptions. Next use `boxplot()` function to provide your readers with some basic statistical measures for waiting. In a separate plot present the box plot for duration times. Please note that `boxplot()` function also accepts as its first argument a formula such as `waiting ~ type`, where `waiting` is the numeric vector of data values to be split in groups according to the grouping variable `type`. The second argument of function `boxplot()` is called `data`, which in our case will take the name of our dataset, i.e. `faithful`. Find a way to add meaningful legends to your graphs. Subsequently, present both boxplots on one graph.

```
# https://stackoverflow.com/questions/39165340/dataframe-create-new-column-based-on-other-columns
df <- transform(faithful, type= ifelse(eruptions <= 3.1, "short", "long"))

# Checking Results of new column are correct
head(df)

# Making new vectors
waiting = df$waiting
type = df$type

# Creating a new boxplot
par(mfrow=c(1,2))
plot(duration, waiting, xlab="Eruption Duration", ylab="Time Waited", main="Eruption Duration vs Time Waited")
abline(model, col='blue', lwd = 4)
boxplot(waiting ~ type, horizontal = FALSE, xlab="Eruption Duration Type",
ylab="Time Waited", main="Eruption Duration Type vs Time Waited")
```

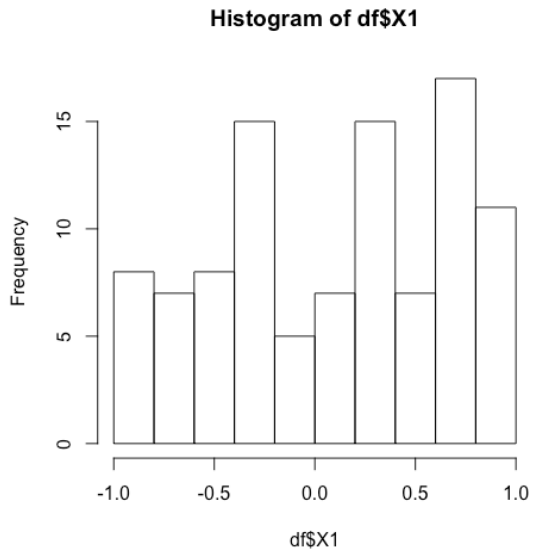


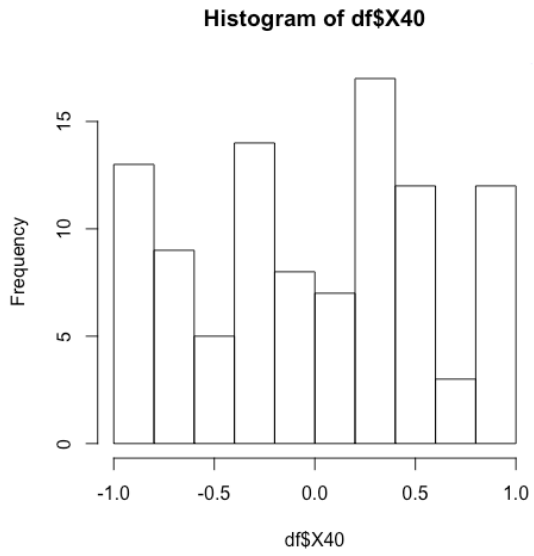


Problem 5.

Create a matrix with 40 columns and 100 rows. Populate each column with random variable of the uniform distribution with values between -1 and 1 (symmetric around zero). Let the distribution for each column appear like the one on slide 92 of the lecture note, except centered around zero. Present two distributions contained in any two randomly selected columns of your matrix on two separate plots. Convince yourself that generated distributions are (close to) uniform.

```
df <- data.frame(replicate(40, runif(100, min = -1, max = 1)))
hist(df$X1)
hist(df$X40)
```

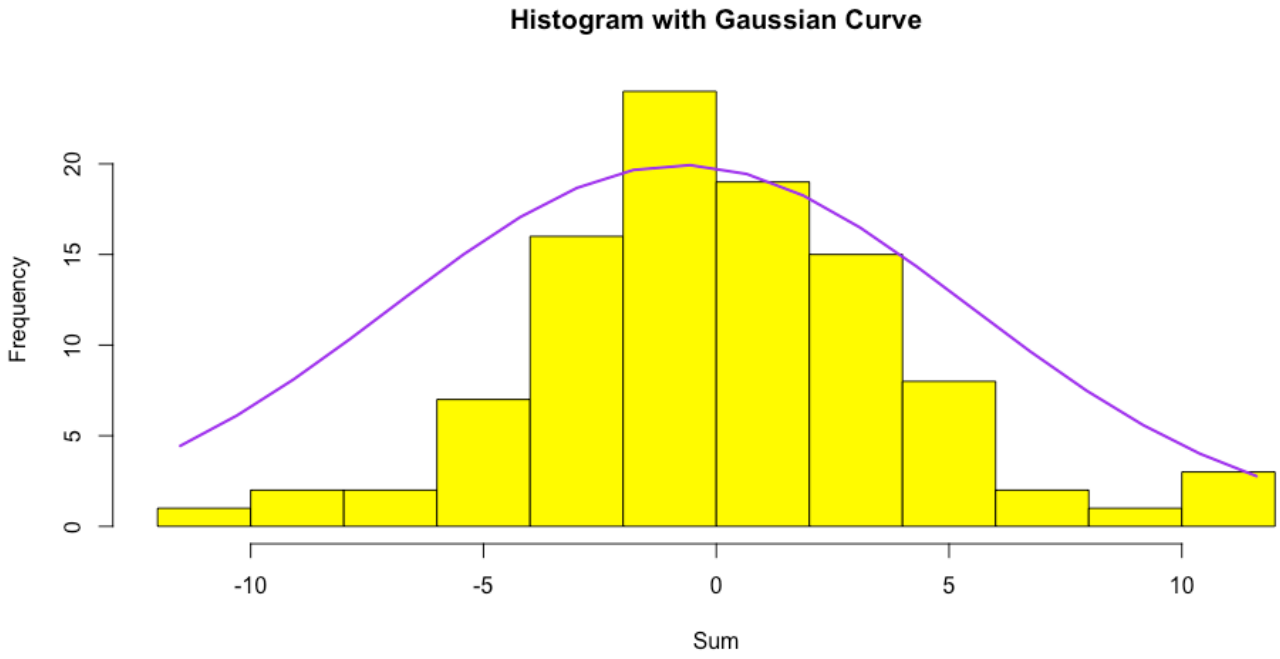




Problem 6.

Start with your matrix from problem 5. Add yet another column to that matrix and populate that column with the sum of original 40 columns. Create a histogram of values in the new column showing that the distribution resembles the Gaussian curve. Add a true, calculated, Gaussian curve to that diagram with the parameters you expect from the sum of 40 random variables of uniform distribution **(15%)**

```
df$sum <- rowSums(df)
print(df)

x_norm <- seq(min(df$sum), max(df$sum), length=20)
y_norm <- (dnorm(xfit, mean=mean(df$sum), sd=sd(df$sum))) *
(diff(h$mids[1:2]*length(df$sum)))
lines(x_norm, y_norm, col="blue", lwd=2)

h <- hist(df$sum, breaks = 10, col="yellow", main = "Histogram with Gaussian Curve",
xlab="Sum", ylab="Sum")
lines(x_norm, y_norm, col="purple", lwd=2)
```



SUBMISSION INSTRUCTIONS:

Your main submission should be an MS Word document containing your code, results produced by that code and brief textual descriptions of what you did and why. Typically, you copy important snippets of your code and the results into this Word document. Describe the purpose of every code snippet and the significance of the results. Start with the text of this homework assignment as the template. Please add any other files that you might have used or generated. Please do not provide ZIP or RAR or any other archives. Canvas cannot open them and they turn into a nuisance for us.