# HU Extension – Assignment 02 – E-185 Big Data Analytics

## Handed out: 02/15/2013 – Due by 17:25PM on Friday, 02/22/2013

### Problem 1

#### Write an R function that would allow you to generate a matrix (data frame) with m columns and n rows and populate every column of that matrix with randomly generated values that fall between values a and b. Choose some arbitrary values for parameters a and b. Use your function to generate a matrix with 4 columns and a reasonably large value of the number of rows n , e.g. 1000-2000. Plot histograms of data in two columns of your matrix and demonstrate to yourself that data are uniformly distributed, between selected values a and b. We are not performing any statistical tests for “uniformity” of the distribution. Just present data graphically in two different plots or histograms and take a look.

Selecting a as -5 and b as 5, we created the following function

r\_matrix <- function(n, m, a = -50, b = 50) {

matrix(runif(n \* m, min = a, max = b), nrow = n, ncol = m)

}

Creating a matrix with 4 columns and 1500 rows

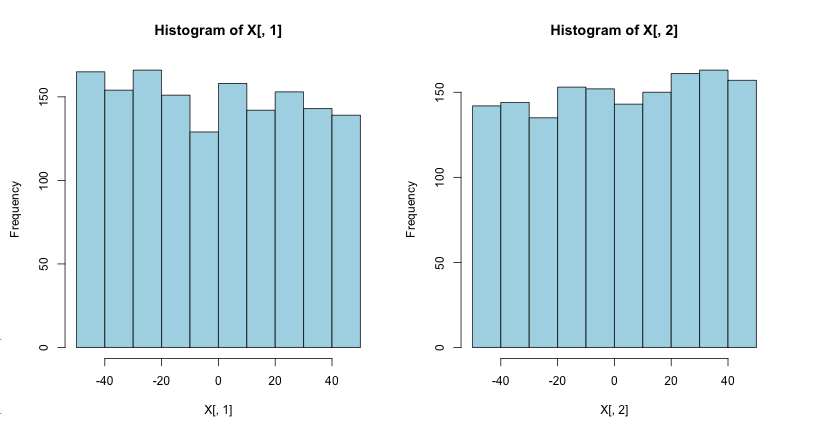
X <- r\_matrix(1500, 4)

Selecting 2 rows to plot the histograms

par(mfrow = c(1, 2))

hist(X[, 1], col = "light blue")

hist(X[, 2], col = "light blue")



### Problem 2

#### Try to find a way to present two distributions contained in any two of columns of your matrix on a single plot. To do that you might want to export the distribution data from two columns into two stand-alone vectors of equal length. Plot one distribution first using a call to plot(x,y1) function, where vector x contains the “predictor” or the parameter vector with values between a and b you selected above. To add the next curve (distribution y2) try invoking function lines(x,y2). To improve your diagram, present two curves in different color and add labels on x and y axis, as well as the title to your graph. Try adding the distribution from the third column to your graph.

We are defining 'x' as the integer breaks of possible values to the variables, and creating a frequency vector of the observer variables by using the functions cut to separate the data into the the integer intervals and table to summarizing the data.

We have explicitly left the last integer value of the interval out as the cut option used does not consider it. The graph is shown below:

x <- -50:49

breaks <- -50:50

y1 = table(cut(X[, 1], breaks, right = FALSE))

plot(x, as.vector(y1), type = "l", col = 2, xlim = c(-50, 50), xlab = "Value",

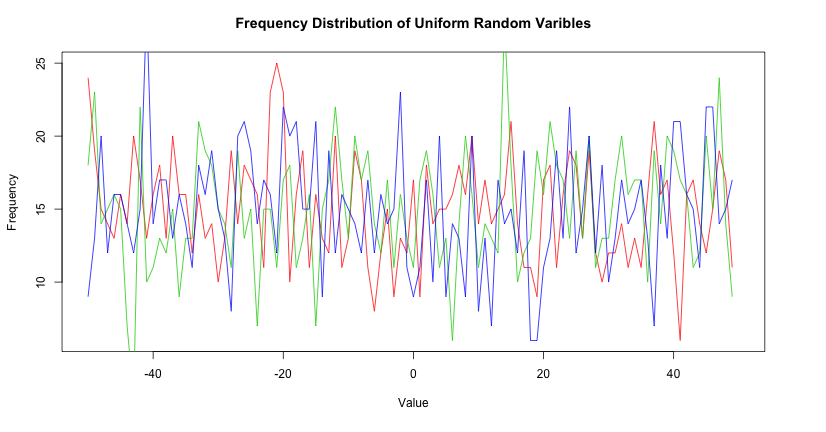
ylab = "Frequency", main = "Frequency Distribution of Uniform Random Varibles")

y2 = table(cut(X[, 2], breaks, right = FALSE))

lines(x, as.vector(y2), type = "l", col = 3)

y3 = table(cut(X[, 3], breaks, right = FALSE))

lines(x, as.vector(y3), type = "l", col = 4)



### Problem 3

#### Write a simple R function that will determine the median of the distribution contained in a vector. Apply your function to a column of your matrix and compare the result with the value obtained with R provided function median().

Since we aim to find the value in the middle of the distribution, we begin by sorting the vector. On an vector of odd length, the answer can be easily found by picking the value in the middle. On an vector of even length, however, the values closest to the middle must be averaged out.

myMedian <- function(X) {

sorted\_x <- sort(X)

if ((length(X)%%2) == 0) {

# even length, must average

retval <- (sorted\_x[length(X)/2] + sorted\_x[length(X)/2 + 1])/2

} else {

# odd number, robust result

retval <- sorted\_x[length(X)/2 + 1]

}

retval

}

Testing my function against the default R implementation median()

myMedian(X[, 4])

## [1] -0.4387

median(X[, 4])

## [1] -0.4387

### Problem 4.

#### Using the function from Problem 1, generate a matrix with 50 columns. Add 5 new columns to your matrix. Let the first additional column contain the sum of the first 10 columns divided by 10. Let the second additional column contain the sum of the first 20 columns divided by 20, and so on until the fifth additional column contains a sum of all 50 original columns divided by 50. Present the distributions of data in those added columns on one plot. I hope that you will see how distributions gradually become less rugged and start approaching a bell curve.

Creating the random matrix and computing the 5 additional columns. They are put in a Y variable for clarity on the next steps.

row\_length <- 1500

X <- r\_matrix(row\_length, 50)

Y <- matrix(0, nrow = row\_length, ncol = 5)

for (j in 1:5) {

for (i in 1:row\_length) {

Y[i, j] <- mean(X[i, 1:(10 \* j)])

}

}

Generating the plots using the same strategy used on **Problem 2.**

x <- -50:49

breaks <- -50:50

y1 = table(cut(Y[, 1], breaks, right = FALSE))

y2 = table(cut(Y[, 2], breaks, right = FALSE))

y3 = table(cut(Y[, 3], breaks, right = FALSE))

y4 = table(cut(Y[, 4], breaks, right = FALSE))

y5 = table(cut(Y[, 5], breaks, right = FALSE))

max\_y <- max(y1, y2, y3, y4, y5) \* 1.1

plot(x, as.vector(y1), type = "l", col = 2, xlim = c(-50, 50), ylim = c(0, max\_y),

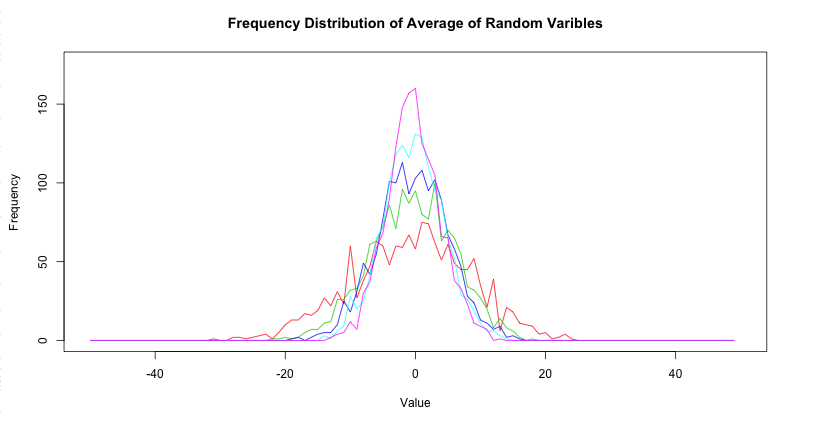
xlab = "Value", ylab = "Frequency", main = "Frequency Distribution of Average of Random Varibles")

lines(x, as.vector(y2), type = "l", col = 3)

lines(x, as.vector(y3), type = "l", col = 4)

lines(x, as.vector(y4), type = "l", col = 5)

lines(x, as.vector(y5), type = "l", col = 6)



### Problem 5.

#### For the distribution contained in the last (5th) additional column use R supplied functions to determine the mean value and the standard deviation. Write a simple R function which would generate a Gaussian curve with provide mean and standard deviation. Plot that Gaussian function on the same plot with the distribution contained in that 5th additional column. You might need to scale (multiply by a factor) the Gaussian in order to get a reasonable overlap of two distributions. Hopefully, you have demonstrated the Central Limit Theorem.

Finding the mean and the standard deviation

mean\_y <- mean(Y[, 5])

mean\_y

## [1] 0.1627

sd\_y <- sd(Y[, 5])

sd\_y

## [1] 4.099

The normal distribution function can be obtained by using the dnorm() function in R. It is being scaled by a factor of 10 times the numerical frequency value for better visual comparison.

X <- -50:49

breaks <- -50:50

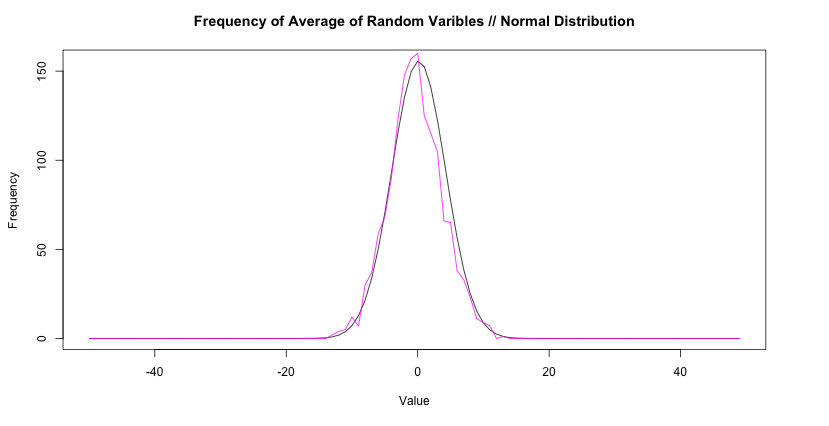
y5 = table(cut(Y[, 5], breaks, right = FALSE))

scale <- max(y5)

plot(X, scale \* 10 \* dnorm(X, mean\_y, sd\_y), type = "l", xlab = "Value", ylab = "Frequency",

main = "Frequency of Average of Random Varibles // Normal Distribution")

lines(x, as.vector(y5), type = "l", col = 6)



It would seem the Central Limit Theorem is true. :)

### Problem 6.

#### Plot the binomial distribution for p = 0.3, p = 0.5 and p = 0.7 and the total number of trials n = 60 as a function of k the number of successful trials. 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.

We begin by assigning the k range and calculating the distribution for the requested values of p:

k <- 0:60

dbin\_03 <- dbinom(k, 60, 0.3)

dbin\_05 <- dbinom(k, 60, 0.5)

dbin\_07 <- dbinom(k, 60, 0.7)

Once they are calculated, we can create the plots.

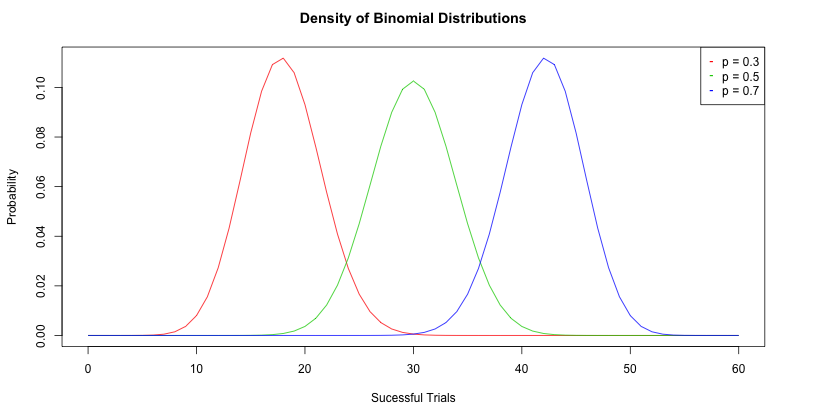
plot(k, dbin\_03, col = 2, type = "l", ylab = "Probability", xlab = "Sucessful Trials",

main = "Density of Binomial Distributions")

lines(k, dbin\_05, col = 3)

lines(k, dbin\_07, col = 4)

legend("topright", c("p = 0.3", "p = 0.5", "p = 0.7"), pch = "-", cex = 1, col = 2:4)



Calculating the summary and the standard deviation of the density

s\_03 <- summary(dbin\_03)

s\_03

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.00000 0.00000 0.00001 0.01640 0.00961 0.11200

s\_05 <- summary(dbin\_05)

s\_05

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.00000 0.00000 0.00005 0.01640 0.01230 0.10300

s\_07 <- summary(dbin\_07)

s\_07

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.00000 0.00000 0.00001 0.01640 0.00961 0.11200

sd\_03 <- sd(dbin\_03)

sd\_03

## [1] 0.0324

sd\_05 <- sd(dbin\_05)

sd\_05

## [1] 0.03063

sd\_07 <- sd(dbin\_07)

sd\_07

## [1] 0.0324

Finally, we draw the boxplot with the calculated values

boxplot(dbin\_03, dbin\_05, dbin\_07, col = 2:4, ylab = "Distribution Density",

xlab = "Trial Probability", main = "Boxplot of Trial Probability Distribution",

names = c("p = 0.3", "p = 0.5", "p = 0.7"))

