**Statistics Final Project Report**

**Course Code: 201-DDD-05 STATISTICS**

Author:

Hoang Anh, Nguyen

*John Abbott College*

*Montreal, CANADA*

# *Abstract—The objective of this project is to equip the author with the knowledge of using R functional programming language in statistical and probabilistic approach to simulate various hypothetical scenarios and incentivize real-world discussions around the obtained results. One of those scenarios is to model the Most Recent Common Ancestor (MRCA) in accordance to Dr. Joseph T. Chang’s (Yale University) paper published on Nature magazine[[1]](#footnote-2). Though the simulation used in this paper is not exactly novelty as it is based on Chang’s model in the first place, it is an interesting and rewarding process nevertheless in order to replicate what Chang has described in his paper[[2]](#footnote-3).*

# Forewords:

Even though the scripts are coded to satisfy the immediate need for a result, they are strictly coded base on the fact that the user studiously feed in valid inputs. In other words, to be essentially frank, the scripts do not catch unexpected errors such as invalid data type inputs or intential attempts to break the program by injecting range-exceeding inputs (e.g. overflowing the memory buffer, overflowing the range of float numbers).

# Part 1. Installing and Learning R

**Exercise 1. Simulating Probabilities**

In this exercise, I learned how to define a variable (or in R case, a vector) and assign a value to it using various methods, operations on those variables set up a dataset/dataframe, drawing bar graphs and pie charts, and randomly sample a tuple with R’s built-in function. Aside from that, the exercise also implements some basic programming concepts, i.e. data types, loops, or a counter variable. Below are the R scripts for each question in this exercise.

*Codeblock. 1. Exercise 1a*

*Codeblock 2. Exercise 1b.*



The original R scripts can be found on my GitHub repository <https://github.com/hirumaakane-ha/r-project-2019>. Running these scripts yield the following results in the console/terminal/cmd:

* A close up of an umbrella

  Description automatically generatedExercise 1i, ii:

Chart 1i. Simulation of 500 Pseudo-Random Rolls of a 12-sided Die

A close up of an umbrella

Description automatically generated

Chart 1. Simulation of 500 Pseudo-Random Rolls of a 12-sided Die with Side 1 has Triple the Probability of Getting Rolled

* Exercise 1iii:

The estimated probability yielded by running the corresponding scripts once is **0.0701**. It is to be noted that this number varies every time the script is rerun so there is no way to replicate this estimated probability recorded here.

However, we can calculate the exact theoreotical probability of getting sum of 10 when rolling the two dice. That theoretical probability is \_\_\_\_.

Here is the R-base terminal output from running the scripts in exercise 1c:

…

> desiredoutcomes <- sumn()

> estprob <- c(desiredoutcomes/10000)

> estprob

[1] 0.0701

# Exercise 2. Creating Functions in R

In this exercise, I learned how to function are created in R. Now, since I have already had a background in C++, Python, and NodeJs, this process went pretty much smoothly. Part (a) and (b) were done in recursion to make them more exciting (and easier on the printer).

*Codeblock 2. Exercise 2*







Running the R scripts presented above in exercise 2 with regards to test cases yielded the following output in the console/terminal/cmd:

* Exercise 2a. Sum of Integers:

> cat('Sum of integers to n: ', sumInt(8), '\n')

Sum of integers to n: 36

> 1+2+3+4+5+6+7+8

[1] 36

* Exercise 2b. Sum of Squared Integers:

> cat('Sum of Square is: ', sumSquare(9), '\n')

Sum of Square is: 285

* Exercise 2c. Sum of Squared Integers from to :

> cat('Sum is: ', sumSquaredkn(6,9), '\n')

Sum is: 230

* Exercise 2d. Rewriting to call (c) in (b):

> cat("Sum of squared k to n = ", rewrite(8,9), "\n")

Sum of squared k to n = 145

> cat("Sum of squared k to n = ", rewrite(1,9), "\n")

Sum of squared k to n = 285

> cat("Sum of squared k to n = ", rewrite(9,9), "\n")

Sum of squared k to n = 81

**Exercise 3. Probability Distributions**

This exercise deals directly with whatever I have learned throughout the course in the field of probability. The types of distribution I encountered while completing this exercise varied, including but not exclusively, binomial distribution, Poisson distribution, pmf, cdf, pdf, normal distribution, uniform distribution, chi-squared, Fisher’s distribution, etc. This was where the stakes were higher since applying these distributions into the code rquired one to know which parameters were permissible in the R library and the *stats* package. One website that I constantly relied on was the *documentation website for the stats package[[3]](#footnote-4)*. In this section for exercise 3, I analyze the problems in 3 parts: identifying the type of distribution, identifying and calculating (if necessary) the parameters, and choosing the right function to use.

*Exercise 3a. Estimating winning Tim Horton cups in a Kirkland store*

* Identifying the type of distribution:

**Part 2. Finding the Most Recent Common Ancestors through Computational Approach**

1. **Defining Most Recent Common Ancestor**

First, we need to define what a common ancestor (abbrev. ‘CA’) is. A common ancestor

1. http://www.stat.yale.edu/~jtc5/papers/CommonAncestors/NatureCommonAncestors-Article.pdf [↑](#footnote-ref-2)
2. http://www.stat.yale.edu/~jtc5/papers/Ancestors.pdf [↑](#footnote-ref-3)
3. <https://stat.ethz.ch/R-manual/R-patched/library/stats/html/00Index.html>. Accessed on 10/03/2019. This documentation is applicable for R version 3.6.1 from CRAN. [↑](#footnote-ref-4)