STA237H1F Assignment #1 (Fall 2023) - Introduction to RStudio and Estimating Probabilities via Simulation

Kevin (Qifan Hu) (1008866817 LEC 0101)

2023-09-26

Assignment #1 (both .Rmd & .pdf) - Due on Quercus 5:00pm, Fri Oct 6, 2023

Direct link to assignment - https://q.utoronto.ca/courses/316967/assignments/1163488

Graded out of 30 marks & worth 7.5% of your STA237H1F grade

NOTE: you must export *both* your completed R Markdown (i.e., rmd) file and your pdf file of your answers from U of T JupyterHub and save on your machine; then upload to Quercus.

NOTE - Save a copy of this rmd file as STA237A1yourname.rmd before you start editing it.

This is an R Markdown document. Markdown is a syntax for authoring documents that are a mix of text and R code and output. For more details on using R Markdown see http://rmarkdown.rstudio.com. When you click the **Knit** button above, a document will be generated that includes both the text content as well as the output of any embedded R code chunks within your R Markdown file. When you submit STA237H1F Assignments this term, you will need to submit your R Markdown (.rmd) file AND the pdf file you generate by clicking on **Knit > Knit to PDF** on the top left of this RStudio window (save that file as something meaningful such as STA237A1yourname.pdf). To save on your own machine to upload to Quercus, check the boxes next to each of the files in the files tab on the right, then click **More > Export** and download to your own machine. Then, upload and submit both files (double-check they are the correct files!) from your machine in the Quercus Assignment.

The best way to learn R is to try coding yourself and to ask for support if and when needed. In this assignment, you will gain hands-on experience with RStudio and a reproducible workflow, and use R to simulate random experiments and estimate and compute probabilities. This assignment must be completed ***independently*** *so you will gain these skills and have the preparation to succeed in STA237H1F and later courses. You are strongly encouraged to start this assignment early and visit the instructor and/or TA office hours for support well before the deadline.*

INTRODUCTION

The greyed-out section below is an R code chunk. Any R code you type in an R code chunk may be run in R Markdown by clicking on the 'play' (triangle) button in the top right corner of the chunk. All R code chunks will be compiled when you knit your rmd file to produce your pdf file.

For example, we could use the following code to simulate flipping a fair coin in R:

```r
# Any comments you include in an R code chunk must be proceeded with "#"

# save possible outcomes in an R vector (called outcomes here)
outcomes <- c("H", "T")
```

```r
# flip fair coin once
set.seed(1)
sample(outcomes, 1)
```

```
## [1] "H"
```

```r
# flip fair coin three times (need to set argument replace=TRUE to sample with replacement)
set.seed(1)
sample(outcomes, 3, replace = TRUE)
```

```
## [1] "H" "T" "H"
```

```r
# could also define possible outcomes in terms of numbers - e.g., 0 for Tails and 1 for Heads
# these can be saved in an R object like above - e.g., outcomes<-0:1 or written as the first
# argument in your sample function
set.seed(1)
sample(0:1, 1)
```

```
## [1] 0
```

```r
set.seed(1)
sample(0:1, 3, replace = TRUE)
```

```
## [1] 0 1 0
```

```r
# may wish to store results in a new R object
results <- sample(0:1, 3, replace = TRUE)

# type the name of the R object to see its contents
results
```

```
## [1] 0 1 0
```

```r
# can use sum() and mean() functions on results
sum(results) # since 1 represents heads here and 0 is tails, this is the number of heads
```

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

```r
# since 1 represents heads here and 0 is tails,
# this is the proportion or relative frequency of heads:
mean(results)
```

```
## [1] 0.3333333
```

```r
# we could also write mean(results==1); it will give the same result here:
mean(results == 1)
```

```
## [1] 0.3333333
```

[*Note: If you type really long lines of R code or comments in the R code chunks, they may "overflow" outside the grey box in your pdf file when you knit it so you won't be able to see everything you typed. A way to avoid this is to truncate your comments/code in the code chunk (i.e., spread over multiple lines) as we did in lines 38-40 in the above R code chunk so they don't "overflow". Review your knitted pdf document to make sure your R code/comments in your R code chunks are showing the way you wish them to show.*]

Try running the code above several times by clicking on the play triangle on the top right of the R code chunk. Notice that you are not ending up with the same simulated flips each time.

R "randomly" generates values using a pseudo-random number generator each time the code is run. Even though we perceive these values as random, R actually generates them based on a pseudo-random number generator algorithm. We can use R function `set.seed()` before each use of the `sample()` function to initialize the pseudo-random number generator from the same point each time. This will generate the same results every time we run the code from the same seed, and will ensure our results are reproducible. Without `set.seed()`, every time we generate random values in R, we end up with different numbers which can make it difficult to identify errors in our processes.

Try adding R code `set.seed(1)` above the each of the `sample()` lines in the previous R code chunk and running the R code chunk a couple of times. Are you ending up with the same results each time? Then try changing the `set.seed()` parameter (i.e., just include any positive integer in the brackets). Did your coin flip simulation results change again?

For this assignment, use `set.seed(type your student number)` ahead of each use of the sample function so you can answer the questions based on your results, and your simulated results will remain the same when you knit your assignment #1 rmd file to pdf.

You may wish to include special characters like greek letters (like omega) or mathematical formulas in your assignment answers. If you copy and paste these symbols from another program (e.g., Word), they will appear in your rmd file, but your document will *not* knit (you will receive an uninformative error and knitting will fail) so you will not be able to generate a pdf file from your rmd file. Instead, if you would like to include special characters you should use LaTeX (a typsetting language) code that, when knitted, will show the special characters in your pdf file. For instance, omega can be typed as Ω, the union symbol as $\cup$, the intersection symbol as $\cap$ and the subset symbol as $\subseteq$. These symbols will appear in the knitted pdf document, but the LaTeX code for them are in the rmd file. Other mathematical relation symbols can be written as $=$, $<$, $>$, $\leq$, $\geq$, $\neq$. A longer list of LaTeX code for typing mathematical symbols is available at https://www.caam.rice.edu/~heinken/latex/symbols.pdf. Here is another useful overview of RMarkdown formatting with some examples of typing equations and special characters - http://www.math.mcgill.ca/yyang/regression/RMarkdown/example.html.

One last comment about R - please note that R can also be used as a calculator. Run the R code below for examples (including $n!$ and $\left(\begin{array}{c} n \\ p \end{array}\right)$). Try more of your own!

```r
1 + 1
```

```
## [1] 2
```

```r
5 - 4
```

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

```r
2 * (1 + 3)
```

```
## [1] 8
```

```r
1 / 4
```

```
## [1] 0.25
```

```r
# here are a couple of built-in functions that may be useful for counting problems
factorial(5)
```

```
## [1] 120
```

```r
choose(5, 2)
```

```
## [1] 10
```

STA237H1F ASSIGNMENT 1 QUESTIONS (Fall 2023)

Answer each of the following questions with R code chunks and/or text, as appropriate.

QUESTION 1 (4 marks)

Consider rolling *one* fair six-sided die (singular of 'dice') *once*.

(a) (1 mark) What is the sample space? Type your answer to 1a here: $\Omega = \{1, 2, 3, 4, 5, 6\}$

(b) (3 marks) Write R code to simulate rolling *one* fair six-sided die *once* and type your simulated outcome below the R code chunk. Remember to set the seed with your student number at the top of your R code chunk.

```r
# Type R code for your answer to 1b in this section
set.seed(1008866817)
dice_outcomes <- 1:6
sample(dice_outcomes, 1)
```

```
## [1] 5
```

Type your simulated outcome for 1b: 5

QUESTION 2 (5 marks)

Simulating more rolls of a *one* fair six-sided die.

(a) (3 marks) Write R code to simulate rolling *one* fair six-sided die *1000 times* and store your results in a R object called `rolls`. Then, estimate the probability of rolling a *2* on one roll of a fair six-sided die based on your simulated results. Remember to set the seed with your student number at the top of your R code chunk.

```r
# Type R code for your answer to 2a in this section
set.seed(1008866817)
dice_outcomes <- 1:6
rolls <- sample(dice_outcomes, 1000, replace = TRUE)
mean(rolls == 2)
```

```
## [1] 0.185
```

Estimated probability of rolling a 2: 0.185

(b) (2 marks) What is the theoretical probability of rolling a *2* when you roll a fair six-sided die? Is this the same as your estimated probability in part 2a? Why or why not.

Type your answer to 2b here: The theoretical probability of rolling a 2 is 1/6 (0.1667). My estimated probability is not the same because 1000 does not provide the accuracy needed here, the more times the rolls are simulated, the more accurate it becomes (i.e. closer to the theoretical probability).

QUESTION 3 (5 marks)

Consider rolling a *weighted* six-sided die; one with sides still marked as 1, 2, 3, 4, 5, and 6, but one where the probability of rolling a 1 is 0.4, and all the other sides are equally likely.

(a) (1 mark) What is the theoretical probability of rolling a *2* with this weighted die? Show your steps.

Type your answer to 3a here: $P(2) = 1/(0.4 + 1 \times 5) = 1/5.4 = 0.1852$

(b) (2 marks) Write R code to simulate rolling this weighted six-sided die *5000 times* and estimate the probability of rolling a *2* on one roll based on your simulated results. Remember to set the seed with your student number at the top of your R code chunk. [Hint: Copy and paste the code you wrote for question 2a; then modify accordingly. Note you will need to modify an argument in `sample()` - if you type `help(sample)` at the > prompt in the R Console on the bottom left RStudio window and hit enter, help documentation on `sample()` will open in the Help tab on the bottom right window.]

```
# Type R code for your answer to 3b in this section
set.seed(1008866817)
dice_outcomes <- 1:6
dice_prob <- c(0.4, 1, 1, 1, 1, 1)
rolls <- sample(dice_outcomes, 5000, replace = TRUE, prob = dice_prob)
mean(rolls == 2)
```

```
## [1] 0.183
```

Estimated probability of rolling a 2: 0.183

(c) (2 marks) Is your estimated probability of rolling a 2 with this weighted die from 3(b) close to the theoretical probability from part 3(a)? Is this surprising? Justify your answer.

Type your answer to 3c here: My estimation of rolling 2 is close to the theoretical probability from (a). Since we are running through more sample simulations, the overall result should get closer compared to question 2, where we were only going through 1000 samples instead of 5000.

QUESTION 4 (5 marks)

Conditional probabilities can be estimated via simulation as well. For example, consider flipping a fair coin *3* times. Let event A = flip at least one head, and event B = flip exactly two heads, and suppose we would like to estimate $P(B|A)$. The following code simulates 1000 repetitions of this experiment and estimates the probability of $P(B|A)$. Try running it and compare it to the theoretical value of $P(B|A)$.

```
set.seed(1008866817)
reps <- 1000
num_heads <- numeric(reps)          # initializes vector to store number of heads for all reps
for (i in 1:reps) {                # for loop simulates the experiment 'reps' times
  flips <- sample(0:1, 3, replace = TRUE) # temporarily store current rep in flips
  num_heads[i] <- sum(flips)          # store number of heads in ith entry of vector num_heads
}

# after the above code runs, vector num_heads contains number of heads for each repetition
P_BandA <- mean(num_heads == 2 & num_heads >= 1)
P_A <- mean(num_heads >= 1)
P_BandA / P_A
```

```
## [1] 0.4363636
```

```
#another way that focuses on when A occurs only
nA <- 0
nBinA <- 0
for (i in 1:reps) {
  if (num_heads[i] >= 1) {
    nA <- nA + 1                              # track number of repetitions where A occurs
    if (num_heads[i] == 2) nBinA <- nBinA + 1   # track number of repetitions B occurs when A occurs
  }
}
nBinA / nA
```

```
## [1] 0.4363636
```

(a) (3 marks) Let's consider a *different* experiment for this question. Consider rolling *two* fair six-sided dice and determining the sum of the dots showing on the two dice. Let event C = roll exactly one 1, and event D = sum of two dice is less than 6, and suppose we would like to estimate $P(D|C)$.

Modify the code below (which is one of the versions that have been copied and pasted from the above R code chunk) to estimate $P(D|C)$ based on *5000* repetitions of the experiment. Remember to set the seed with your student number at the top of your R code chunk, and to report your estimate of $P(D|C)$ under the R code chunk.

```r
set.seed(1008866817)

# Modify this code to answer Question 4:
dice_outcomes <- 1:6
reps <- 5000
nC <- 0
nDinC <- 0
for (i in 1:reps) {
  temp_rolls <- sample(dice_outcomes, 2, replace = TRUE)
  # temp_rolls[0]
  # temp_rolls[1]
  if (((temp_rolls[1] == 1) & (temp_rolls[2] != 1)) | ((temp_rolls[1] != 1) & (temp_rolls[2] == 1))) {
    nC <- nC + 1
    if (sum(temp_rolls) < 6) {
      nDinC <- nDinC + 1
    }
  }
}
nDinC / nC
```

```
## [1] 0.5983322
```

Estimate of $P(D|C)$: 0.5983322

(b) (2 marks) What is the theoretical value of $P(D|C)$? Show your steps. How does this value compare to your estimated probability in part 4a? Is this surprising? Why or why not?

Type your answer to 4b here: According to the property of conditional probability, $P(D|C) = \frac{P(D \cap C)}{P(C)}$

We will first calculate P(C), which is the probability of *rolling exactly one 1*. The number of total possible outcomes is $6^2 = 36$.

For C to occur, we will have to either roll 1 on the first or second dice, not both, so we can split this into two cases:

- Case 1: we *roll 1 on the first dice and not the second dice.* Possible outcomes in this case are: (1,2), (1,3), (1,4), (1,5), (1,6). So there are 5 possible ways to satisfy C in this case.
- Case 2: we *roll 1 on the second dice and not the first dice.* Possible outcomes in this case are: (2,1), (3,1), (4,1), (5,1), (6,1). So there are 5 possible ways to satisfy C in this case.

Thus, there are $5 + 5 = 10$ total ways for C to occur, so $P(C) = \frac{10}{36} = \frac{5}{18}$.

For *sum of two dice to be less than six* at the same time as *rolling exactly one 1*, we are able to select ones that has sum less than six from possible outcomes of C: (1,2), (1,3), (1,4), (2,1), (3,1), (4,1).

We can see that 6 possible outcomes satisfies this requirement, which means that $P(D \cap C) = \frac{6}{36} = \frac{1}{6}$.

As a result, we can now calculate

$$P(D|C) = \frac{P(D \cap C)}{P(C)} = \frac{\frac{1}{6}}{\frac{5}{18}} = \frac{18}{30} = \frac{3}{5} = 0.6$$

This theoretical value of $P(D|C)$ is very close to my estimated probability in part 4a. This is not surprising because the more we simulate the events in code, the closer we get to the theoretical outcome, and 5000 times should be plenty to get around the theoretical probability.

QUESTION 5 (6 marks)

Consider the digits 1, 2, 3, 4, 5, and 6.

(a) (2 marks) i. Use R to compute the number of possible rearrangements of these *6 digits*. Then, ii. use R code to simulate one repetition of this experiment. Remember to set the seed with your student number before each use of the `sample()` function.

```
# 5ai. count
factorial(6)
```

```
## [1] 720
```

```
# 5aii. simulate
digits <- 1:6
set.seed(1008866817)
sample(digits, 6, replace = FALSE)
```

```
## [1] 5 3 6 4 1 2
```

(b) (2 marks) i. Use R to compute the number of ways *3 digits* can be randomly selected from the *6 digits* one at a time without replacement. Then, ii. use R code to simulate one repetition of this experiment. Remember to set the seed with your student number before each use of the `sample()`function.

```
# 5bi. count
choose(6, 3)
```

```
## [1] 20
```

```
# 5bii. simulate
digits <- 1:6
set.seed(1008866817)
sample(digits, 3, replace = FALSE)
```

```
## [1] 5 3 6
```

(c) (2 marks) i. Use R to compute the number of ways sets of *2 digits* can be selected at random from the *6 digits*. Then, ii. use R code to simulate one repetition of this experiment. Remember to set the seed with your student number before each use of the `sample()`function.

```
# 5ci. count
6^2 # with replacement
```

```
## [1] 36
```

```
# choose(6, 2) # without replacement (Alternative)

# 5cii. simulate
digits <- 1:6
set.seed(1008866817)
sample(digits, 2, replace = TRUE) # with replacement
```

```
## [1] 5 3
```

```
# set.seed(1008866817)
# sample(digits, 2, replace = FALSE) # without replacement (Alternative)
```

ASSIGNMENT REPRODUCIBILITY (5 marks)

Your assignment #1 file submission in the Quercus Assignment must include *both* the rmd file with your assignment #1 answers that was compiled (or *knitted*) to produce a pdf file of your assignment #1 answers. # Rubric:

- 0/5 marks - submitted rmd file did not produce submitted pdf file
- 1/5 marks - no rmd file submitted
- 1/5 marks - number other than your student number used in 'set.seed()' or seed not set in R code chunk(s)
- 3/5 marks - submission includes rmd file only (i.e., no pdf)
- 5/5 marks - both your pdf file and rmd file used to produce your pdf file submitted

---

THIS IS THE END OF STA237H1F ASSIGNMENT #1

```
## [1] 0.9509014
```

```
## [1] "Fri Oct  6 19:44:24 2023"
```