Assignment 4

EMATM0061: Statistical Computing and Empirical Methods, TB1, 2022

Dr. Rihuan Ke

Introduction

The submission deadline for this assignment is 23:59, 24 October 2022. Note that this assignment will not count towards your final grade. However, it is recommended that you try to answer the questions to gain a better understanding of the concepts and submit your work before the deadline to receive feedback.

Create an R Markdown for the assignment

It is a good practice to use R Markdown to organize your code and results. You can start with the template called `Assignment04_Template.Rmd` which can be downloaded via Blackboard.

You may also want to use R Markdown to organize your solutions so that you can submit them later. If you are considering submitting your solutions, please generate a PDF file. For example, you can choose the "PDF" option when creating the R Markdown file (note that this option may require Tex to be installed on your computer), or use R Markdown to output an HTML and print it as a PDF file in a browser. We can only accept PDF files in the submission of this assignment. To submit the assignment, please visit the "Assignment" tab on the Blackboard page, where you downloaded the assignment.

Load packages

Some of the questions in this assignment require the tidyverse package. If It hasn't been installed on your computer, please use `install.packages()` to install them first.

To road the tidyverse package:

```
library(tidyverse)
```

1. Probability theory

In this section we consider some of the concepts introduced in Lecture 9.

Recall that we have introduced the three key rules of probability. Given a sample space Ω along with a well-behaved collection of events $\mathcal{E}$, a probability $\mathbb{P}$ is a function which assigns a number $\mathbb{P}(A)$ to each event $A \in \mathcal{E}$, and satisfies rules 1, 2, and 3:

Rule 1: $\mathbb{P}(A) \geq 0$ for any event $A \in \mathcal{E}$

Rule 2: $\mathbb{P}(\Omega) = 1$ for sample space Ω

Rule 3: For pairwise disjoint events $A_1, A_2, \cdots$ in $\mathcal{E}$, we have

$$\mathbb{P}(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$$

1.1 Rules of probability

(Q1) Construct probability based on the Rules of probability

Consider a sample space $\Omega = \{a, b, c\}$ and a set of events $\mathcal{E} = \{A \subseteq B\}$. Based on the rules of probability, find an example of probability $\mathbb{P} : \mathcal{E} \to [0, 1]$ such that $\mathbb{P}$ satisfies

$$\mathbb{P}(\{a, b\}) = 0.6 \quad \text{and } \mathbb{P}(\{b, c\}) = 0.5.$$

In your example, you need to define $\mathbb{P}$ clearly. Make sure that your function $\mathbb{P}$ satisfies the three rules, but you don't need to write down the proof (that it satisfies the three rules).

(Q2) Verify that the following probability space satisfies the rules of probability.

Consider a setting in which the sample space $\Omega = \{0, 1\}$, and $\mathcal{E} = \{A \subseteq \Omega\} = \{\emptyset, \{0\}, \{1\}, \{0, 1\}\}$. For a fixed $q \in [0, 1]$, define a function $\mathbb{P} : \mathcal{E} \to [0, 1]$ by

$$\mathbb{P}(\emptyset) = 0,\ \mathbb{P}(\{0\}) = 1 - q,\ \mathbb{P}(\{1\}) = q,\ \mathbb{P}(\{0, 1\}) = 1.$$

Show that the probability space $(\Omega, \mathcal{E}, \mathbb{P})$ satisfies the three rules of probability.

1.2 Deriving new properties from the rules of probability

(Q1) Union of a finite sequence of disjoint events.

Recall that in Rule 3, we have

$$\mathbb{P}(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$$

for an infinite sequence of pairwise disjoint events $A_1, A_2, \cdots$. Show that for a finite sequence of disjoint events $A_1, A_2, \cdots A_n$, for any integer n bigger than 1, the above equality holds as a consequence of Rule 3, i.e.,

$$\mathbb{P}(\cup_{i=1}^{n} A_i) = \sum_{i=1}^{n} \mathbb{P}(A_i)$$

(Q2) Probability of a complement.

Prove that if Ω is a sample space, $S \subseteq \Omega$ is an event and $S^c := \Omega \backslash S$ is its complement, then we have

$$\mathbb{P}(S^c) = 1 - \mathbb{P}(S).$$

(Q3) The union bound

In Rule 3, for pairwise disjoint events $A_1, A_2, \cdots$, we have

$$\mathbb{P}(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$$

Recall that we have also shown the union bound as a consequence of the rules of probability: for a sequence of events $S_1, S_2, \cdots$, we have $\mathbb{P}(\cup_{i=1}^{\infty} S_i) \leq \sum_{i=1}^{\infty} \mathbb{P}(S_i)$.

Given an example of a sequence of sets $S_1, S_2, \cdots$, such that $\mathbb{P}(\cup_{i=1}^{\infty} S_i) \neq \sum_{i=1}^{\infty} \mathbb{P}(S_i)$.

(Q4) Probability of union and intersection of events.

Show that for events $A \subseteq \Omega$ and $B \subseteq \Omega$, we have

$$\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B)$$

2. Finite probability spaces

Now, let's move on to consider probability on finite sample spaces, which is covered in Lecture 10.

2.1 Sampling with replacement

Recall that for positive integers n and k, $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ gives the number of subsets of size k from a set of n objects.

You can compute this number straightforwardly within `R` via the choose function `choose()`. For example, if we want to compute the number of different subsets of size 3 from a collection of size 8 we would compute:

```
choose(8,3)
```

```
## [1] 56
```

Suppose we have a bag containing 10 spheres. This includes 3 red spheres and 7 blue spheres.

Let's suppose that we draw a sphere at random from the bag (all spheres have equal probability of being drawn). We record its colour and then return the sphere to the bag. This process is repeated 22 times. This is an example of **sampling with replacement** since the spheres are replaced after each draw.

(Q1) Write down a mathematical expression for the probability that z out of the 22 selections were red spheres (here $z \in \{0, 1, \cdots, 22\}$).

You can try writing down your mathematical expression using `LaTex` code in your R markdown file, making use of the LaTex functions `binom` and `frac`. For information about writing mathematical formulae in Rmarkdown documents visit: https://bookdown.org/yihui/bookdown/markdown-extensions-by-bookdown.html#equations

(Q2) Next write an R function called `prob_red_spheres()` which takes z as an argument and computes the probability that z out of a total of the 22 balls selected are red.

Test your function as follows:

```
prob_red_spheres(10)
```

```
## [1] 0.05285129
```

(Q3) Generate a data frame called `prob_by_num_reds` with two columns `num_reds` and `prob`. The `num_reds` column should contain numbers 1 through 22 and the `prob` column should give the associated probability of selecting that many reds out of a total number of 22 selections.

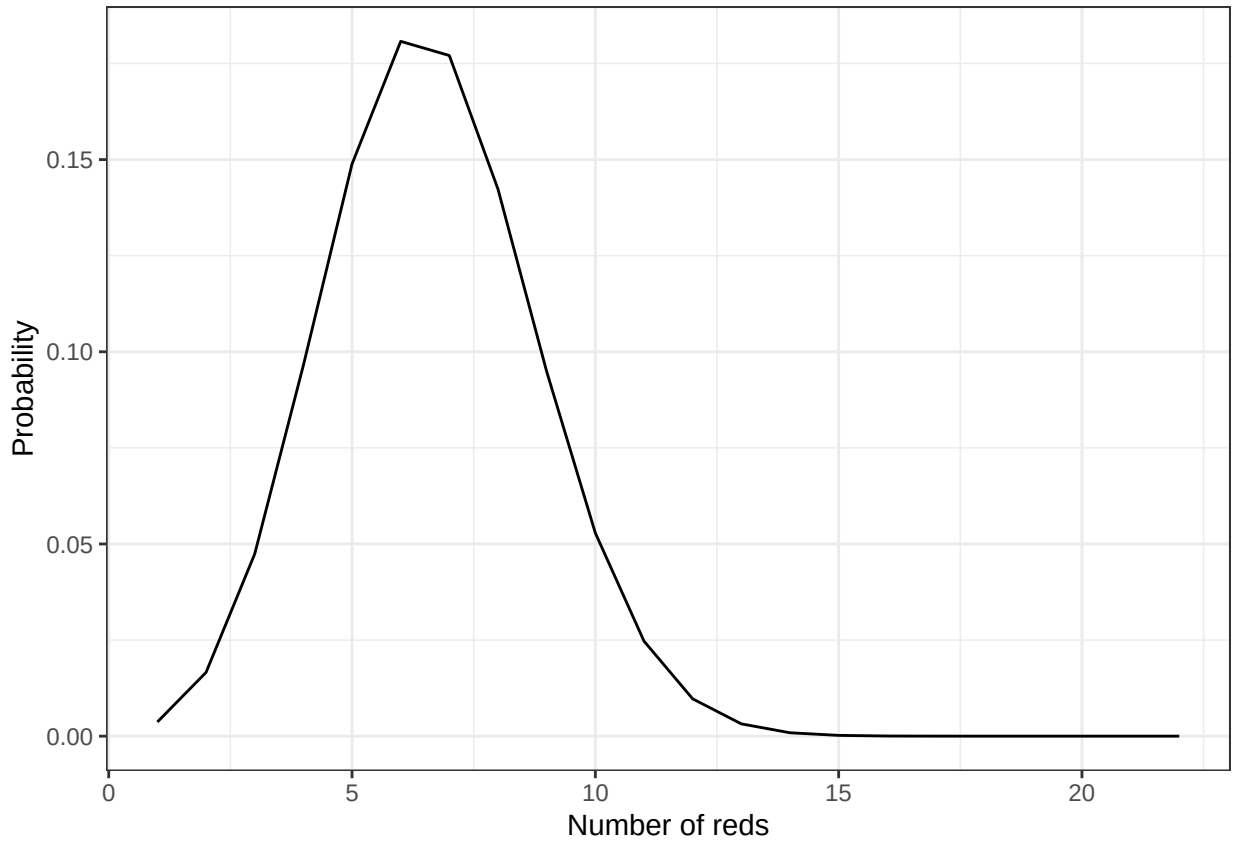
Display the first 3 rows of your data frame:

```
prob_by_num_reds %>% head(3)
```

```
##   num_reds        prob
## 1        1 0.003686403
## 2        2 0.016588812
## 3        3 0.047396606
```

(Q4) Now use the `geom_line()` function within the `ggplot2` library, in conjunction with your data frame to display a plot of the probability as a function of the number of reds.

Your plot should look as follows:



(Q5)

Next we shall explore the `sample()` function within R. Let's suppose we want to simulate a random experiment in which we sample with replacement from a collection of 10 objects, and repeat this process 22 times.

We can do this by calling:

```
sample(10, 22, replace=TRUE)
```

```
##  [1]  7  3  1  3 10  5 10  3  5  1  8  4  8  4  9  1  1  6  7  5  2  4
```

Try this out for yourself. The output should be a vector of length 22 consisting entirely of numbers between 1 and 10. Since this is sampling with replacements and the number of samples exceeds the number of elements, there will be repetitions.

Try rerunning the function. You probably get a different sample. This is to be expected, and even desirable, since the function simulates a random sample. However, the fact that we get a different answer every time we run the code is problematic from the perspective of reproducibility. To avoid this process we can set a random seed via the function set.seed(). By doing so we should get the same output every time. Try the following out for yourself:

```
## Setting the random seed just once
set.seed(0)
for(i in 1:5){
  print(sample(100,5,replace=FALSE))
  # The result may well differ every time
}
## Resetting the random seed every time
```

```
for(i in 1:5){
  set.seed(1)
  print(sample(100,5,replace=FALSE))
  # The result should not change
}
```

We shall now use the `sample()` to construct a simulation study to explore the probability of selecting z red balls from a bag of size 10, with 3 red and 7 blue balls, when sampling 22 balls with replacement.

First set a random seed. Then create a data frame called `sampling_with_replacement_simulation` consisting of two columns. The first is called `trial` and contains numbers 1 through 1000. The second is called `sample_balls` and corresponds to random samples of size 22 from a bag of size 10, with replacement. We can do this as follows:

```
num_trials<-1000 # set the number of trials
set.seed(0) # set the random seed
sampling_with_replacement_simulation<-data.frame(trial=1:num_trials) %>%
  mutate(sample_balls = map(.x=trial, ~sample(10,22, replace = TRUE)))
# generate collection of num_trials simulations
```

Now add a new column called `num_reds` such that, for each row, `num_reds` contains an integer which gives the number of items within the sample for that row (the entry in the `sample_balls` column) which are less than or equal to three (assuming that the three red balls are labelled by $\{1, 2, 3\}$). For example, suppose that in some row of the data frame, the `sample_balls` column contains the following list:

4 10 4 10 8 3 5 5 5 5 10 7 1 2 1 10 5 6 5 7 1 10

Then the corresponding row of the `num_reds` column should contain the number 5, since 5 of these values are less than equal to 3. You may want to use the functions `mutate()`, `map_dbl` and `sum()`. After performing this step, the data frame `sampling_with_replacement_simulation` should contain one column called `num_reds`.

(Q6)

Next we shall add a new column called `predicted_prob` to our existing data frame `prob_by_num_reds` which gives the number of times (that we observed the corresponding number of reds within our simulation) divided by the total number of observations. This aims to estimate the probability of the event that z out of a total of the 22 balls selected are red (for $z = 1, 2, \cdots, 22$). We can do this as follows:

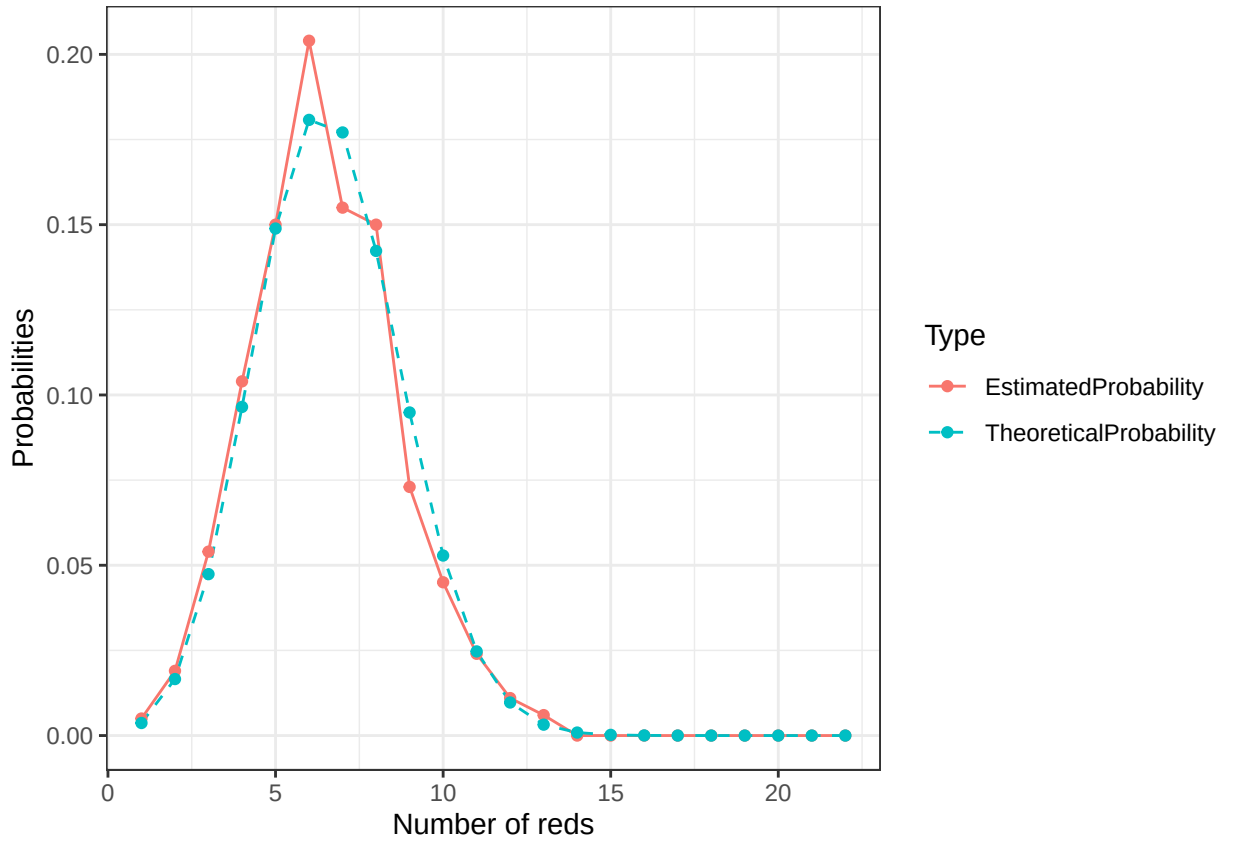
```
num_reds_in_simulation<-sampling_with_replacement_simulation %>%
  pull(num_reds)
# we extract a vector corresponding to the number of reds in each trial
prob_by_num_reds<-prob_by_num_reds %>%
  mutate(predicted_prob=map_dbl(.x=num_reds,~sum(num_reds_in_simulation==.x))/num_trials)
# add a column which gives the number of trials with a given number of reds
```

(Q7) Finally, create a plot which compares the results of your simulation with your probability formula.

Your result should look something like the plot below. Of course, since this is a random simulation, your result may well look slightly different.

```
prob_by_num_reds %>%
  rename(TheoreticalProbability=prob, EstimatedProbability=predicted_prob) %>%
  pivot_longer(cols=c("EstimatedProbability","TheoreticalProbability"),
               names_to="Type",values_to="count") %>%
  ggplot(aes(num_reds,count)) +
  geom_line(aes(linetype=Type, color=Type)) + geom_point(aes(color=Type)) +
```

```
scale_linetype_manual(values = c("solid", "dashed"))+
theme_bw() + xlab("Number of reds") + ylab("Probabilities")
```



Try to make sense of each line in the above code.

Notice the close relationship between probabilities and long-run frequency behaviour. We shall explore this connection over the next few lectures.

2.2 Sampling without replacement

Let's suppose we have a large bag containing 100 spheres. There are 50 red spheres, 30 blue spheres and 20 green spheres. Suppose that we sample 10 spheres from the bag **without** replacement.

(Q1) What is the probability that one or more colours are missing from your selection? First aim to answer this question via a simulation study using ideas from the previous question.

You may want to use the following steps:

1. First set a random seed;
2. Next set a number of trials (e.g., 10 or 1000. Try this initially with a small number of simulations. Increase your number of simulations to about a relatively large number to get a more accurate answer, once everything seems to be working well), and a sample size (10);
3. Now use a combination of the functions `sample()`, `mutate()` and `map()` to generate your samples. Here you are creating a sample of size 10 from a collection of 100 balls - the sampling is done **without** replacement;
4. Now compute the number of "reds", "greens" and "blues" in your sample using the `map_dbl()` and `mutate()` functions.

5. Compute the minimum of the three counts using the `pmin()` function. When this minimum is zero, then one of the three colours is missing. It is recommended that you look up the difference between `pmin()` and `min()` here;
6. Compute the proportion of rows for which the minimum number of the three counts is zero.

(Q2) (*Optional) This question is optional and may be omitted if you are short on time.

Let's derive a mathematical expression for the probability that we considered in **(Q1)**: You can try and use "combinations" with $\binom{n}{k}$ to compute the probability directly. First aim to compute the number of subsets of size 10 from 100 which either entirely miss out one of the subsets Red $= \{1, \cdots, 50\}$, Blues $= \{51, \cdots, 80\}$, Greens $= \{81, \cdots, 100\}$. Then compute the number of subsets of size 10 from 100 which miss out two of the subsets Red, Blues, Green. Be careful not to double count some of these subsets! Once you have computed all such subsets, combine them with the formula for the total number of subsets of size 10 from a set of 100, to compute the probability of missing a colour.

Once you have the mathematical expression for the probability, you can check how close the probability computed in **(Q1)** to the theoretical values.