PSTAT 10 Worksheet 5

Due 7/12/22

A Tricky Dice Problem

In this worksheet, we will investigate the following question:

You roll a fair die a **very** large number of times. What is the probability you will have more low (123) than high (456) numbers AND ALSO have more odd (135) than even (246) numbers?

You may be tempted to say 1/4, but this is incorrect; odd numbers are more likely to be low numbers. In the language of probability, these events are *not independent*. In fact, if you think about it, the true probability should be slightly higher than 1/4 purely because odd numbers are more likely to be low numbers. But how much higher?

The answer, which can be derived theoretically using something called the *multivariate central limit theorem*, is a bit surprising:

$$p = \frac{1}{4} + \frac{\arcsin(1/3)}{2\pi}. \tag{$*$}$$

Without using sophisticated tools from probability theory, we can easily approximate the value using simulation.

Step 1.

One replication of the experiment consists of rolling a die a **very** large number of times. Create a function `roll(nroll)` that rolls a fair die `nroll` times and returns TRUE if there are more low than high numbers AND ALSO there are more odd than even numbers, and returns FALSE otherwise.

We will consider 10,000 large enough and use `nroll=10000`.

Step 2.

The `roll` function just constructed gives us a single replication of the experiment. To find the *probability*, we must replicate the experiment a large number of times and find the proportion of TRUE replications.

Replicate the experiment 500 times, with 10,000 rolls each time. The result should be a logical vector of length 500. Assign this result to a variable called `replicates`.

Step 3.

What is the estimated answer to the dice problem? Comment on how easy this was compared to learning the *multivariate central limit theorem*.

Step 4.

We can visualize our approximation by plotting the *running mean*; we should expect that our approximation gets better with more replicates.

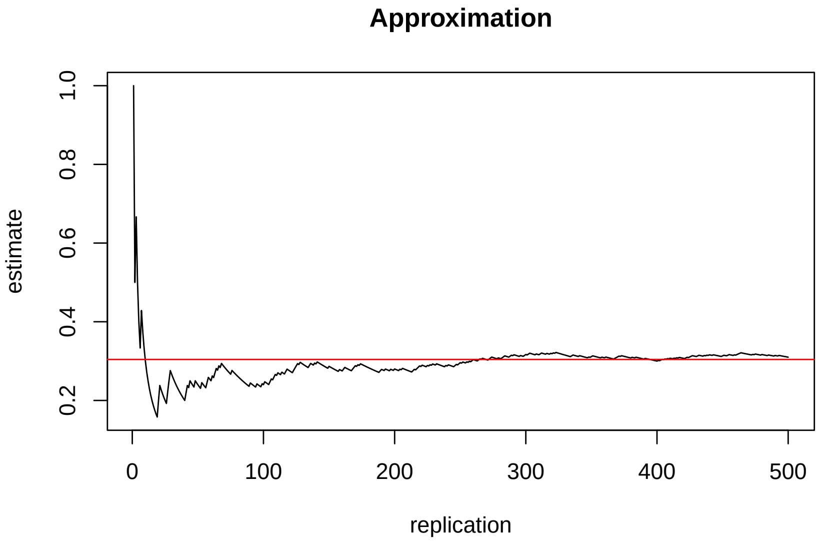
First, create a function `running_mean(m)` that gives the average of the first `m` elements of `replicates`. Check that `running_mean(500)` gives your answer from Step 3.

Next, find the running mean with the following line of code:

```
running <- sapply(1:500, running_mean)
```

`running` now contains the running mean over all 500 replicates.

Finally, make the following plot:



The red horizontal line is the true value given in (∗). You can find the true value with `asin`, which is the arcsin function.

Your plot will of course be different due to randomness. Remember to set a seed so that your results are replicable.