MATH 1060 Lab 3 - Numerical Experiments with R

In this lab, we are doing some programming. I recommend you read Sections 3.4 Variables, 3.5 Functions, and 3.7 Vectors in "Learning Statistics with R" (it's referred as LSWR below).

Law of Large Numbers

1) Consider coin toss experiments. Let us assign integers 1 and 0 to heads and tails, respectively. The law of large numbers states that the expected value for the outcome (proportion of heads) of many trials is 1/2. Copy the following code to your script file. Select line 2 through 10, and click "Run". The graph shows that the proportion of heads approaches 0.5 as the number of trials n increases.

```
1  #Coin toss:  Probability of getting heads
2  n <- 1:1000
3  P<-rep(0, times=length(n))
4  for(m in n){
5    x <- sample(c(0,1), m, replace=TRUE)
6    P[m] <- sum( x )/ m
7  }
8  plot(n, P)
9  P[length(n)]
10
```

Let us go through this code line by line.

Line 2: "`1:1000`" creates a vector whose components are the sequence 1, 2, 3, … , 999, 1000. This vector is assigned to the variable `n`. So, `n` is a vector. Try creating different sequences using ":" in the console pane. For example, type `10:14` at the bottom of the console pane and hit Enter. The output should look like `10 11 12 13 14`.

Line 3: `length(n)` gives the length (or the number of components) of vector `n`. In this particular case `length(n) is 1000`. The command rep(0, times=1000) repeats 0 1000 times, i.e., creates a vector of length 1000 whose components are all zeros. The result is stored in vector P. Try, for example, typing `rep(1, times=10)` in the console pane. Can you guess what `rep(c(0,1), times=5)` does? Note that c(0,1) creates a vector 0 1. (So, for example, c(1, 0, 6, 0) creates a vector 1 0 6 0)

Line 4: `for(m in n){ some commands }` is called a *for-loop.* (`n` is a vector here.) It execute the commands inside the braces {}, `length(n)` times. For each cycle a component of n is stored in m, which may be used by the commands inside {}. What do you think the following statement will do?
`for( i in c(3, 1, 4, 1, 5, 9) ) print(i)`

Note that {} are not necessary here because there is only one command to be repeated.

Line 5: `sample(c(0,1), m, replace=TRUE)` randomly samples a number from the vector c(0,1) repeated m times with replacement. For example,

```
sample(c(0,1), 5, replace=TRUE) □ 1 0 0 1 0
```

How about `sample(1:10, 100, replace=TRUE)`?

Line 6: `sum(x)` sums all the components of vector x. (How would you compute the sum of the sequence of numbers from 1 to 100?) `sum(x)/m` gives the proportion of the number of heads to the total number of coin flips. This value is then assigned to `P[m]` which is the m^{th} component of vector P.

Line 8: `plot(n, P)` command plots a *scatter graph* of points

```
{(n[1], P[1]), (n[2], P[2]), (n[3], P[3]), …},
```

where `(n[i], P[i])` are x- and y-coordinates of a point. Try

```
plot(1:5, c(0,1,-1,2,-2))
```

Line 9: What does `P[length(P)]` mean? If you don't get it, study

```
v <- 1050:1060
v[length(v)]
```

Now you know what each line is doing. Go ahead and modify here and there, and see what happens. Don't be afraid. You are not going to break the computer. You need to be aware of one thing, though. Whenever you have loops in your code (e.g. a for-loop), the computation time might become too long or it gets trapped in an infinite loop. If that happens, you can click on the emergency STOP button that appears on the upper right corner of your console pane.

2) Copy the code above and paste it just below it. Modify the code so that it will do the same thing for the probability of getting 6 when you throw a die.

Hint: you only need to modify the vector `c(0,1)` inside `sample()` command. We had `c(0,1)` with 0 standing for Tail and 1 for Head. And by summing the sequence of 0's and 1's we have the number of times we get Heads. For this problem you want to count the number of events where the outcome is 6, ignoring all other numbers we get. [So, you should have `c(?,?,?,?,?,?)` instead of `c(0,1)`. Each "?" is of course 1 or 0.] Your result should converge to 1/6 = 0.166...

Multiplication Rule for Independent Events

Let us toss a coin, and throw a die at the same time. According to the multiplication rule, the probability of getting a head and 6 on the die is

$$P = \frac{1}{2} \times \frac{1}{6} = \frac{1}{12} \approx 0.083$$

Verify the multiplication rule by combining the previous 2 cases. Again, the only place you need to modify is Line 5 of the original code. In this case we want each element of x to be 1 if you get a head and 6, and 0 otherwise. For the coin-toss experiment we may have something like

x = 0 1 1 0 1 0 0 …

where 1's are heads, and for the die experiment

x = 1 0 0 0 1 0 0 …

where 1 represent 6 on the die. In order to find events where a head and 6 occurred simultaneously, we only need to multiply the two vectors element by element to get

x = 0 0 0 0 1 0 0 …

Element-by-element multiplication is done in R as

```
Vector1 * Vector2
```

Thus, you just need to combine the `sample(…)` commands (which are vectors) from the previous code blocks.