

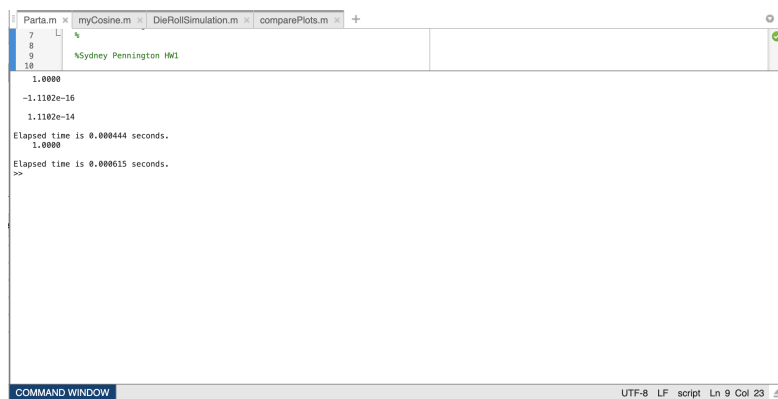
Sydney Pennington
Bayesian Data Analysis
Fall 2020
HW1p: Programming Homework #1
Instructor: Prof. Kevin H. Knuth
Date : 3 Sep 2020

Part A:

I first created a for loop to add 0.1 to 10 times and this yielded the result -1.1102e-16, which is not the same answer of $0.1 * 10 = 1$. It therefore proves that computers do make mistakes. When calculating the percent error, I stored the value of x minus 1 in variable y and then got the absolute value of y and divided that value by 100, which then displayed a percent error of 1.1102e-14.

Part B:

I listed the values within the vector named vector one by one and then used the sum function to calculate the sum of the vector to then display the result. In this case there is no error and the result is 1. I placed tic at the beginning of then for loop for part A and then toc twice after the for loop and after the percent error calculations. This resulted in 0.000444 seconds and 0.000615 elapsed time calculations respectively.



```
PartA.m x myCosine.m x DieRollSimulation.m x comparePlots.m x
7 %
8 %
9 %Sydney Pennington HW1
10
1.0000
-1.1102e-16
1.1102e-14
Elapsed time is 0.000444 seconds.
1.0000
Elapsed time is 0.000615 seconds.
>>
```

Part C:

I essentially was guided by the email Professor Knuth sent then class about the myCosine function. In terms of figuring out the terms needed for this function, I realized going up to six terms is ideal for this cosine in addition to dividing each number in the Taylor series using the factorial function in Matlab. I also adjusted the printed decimal value in formatting the print statement and printing to three decimal places. As a result, the user inputs the x value in the console and then calls the myCosine function and prints out the corresponding y value on the cosine curve.

```
>> x=2*pi
x =
    6.2832
>> myCosine(x)
Result: 1.000.
ans =
    1
>>
```



Part D:

In plotting both functions, I didn't notice any difference in graphing the functions within the domain 0 to 2π .

Part E:

I used the random function for each die simulation within this section of the homework. I stored the results for random roll die for the six sided die in the variable results and then altered those columns to then show ones forever a 6 was rolled. I then used the sum functions to get the sums of all the ones and then used that number to divide it over the number of times the die was rolled, in this case, 1000 times. I repeated this framework for the 12 sided and 18 sided die. My probability for each die was close to 6, which contrasts the probability when calculating $2/12$, or $3/18$, but that be could a computer error as we saw in Part A.