AM 147: Computational Methods and Applications: Winter 2021
Homework #2

Instructor: Abhishek Halder

Due: January 19, 2021

NOTE: Please submit your Homework as a single zip file named `YourlastnameYourfirstnameHW2.zip` via CANVAS. For example, `HalderAbhishekHW2.zip`. Please strictly follow the capital and small letters in the filename of the zip file you submit. You may not receive full credit if you do not follow the file-naming conventions. Your zip file should contain all .m files (MATLAB scripts) for the questions below.

Your zip file must be uploaded to CANVAS by 11:59 PM Pacific Time on the due date. The uploads in CANVAS are time-stamped, so please don't wait till last moment. Late homework will not be accepted.

Problem 1

Numerical errors **(20 points)**

For real x, the n-term Taylor series approximation of $\exp(x)$ is $\displaystyle\sum_{j=0}^{n-1} \frac{x^j}{j!}$. Consider computing the 10-term Taylor series approximation of $\exp(-5)$ in two different ways:

(i) $\displaystyle\exp(-5) \approx \sum_{j=0}^{9} \frac{(-5)^j}{j!}$,

(ii) $\displaystyle\exp(-5) = \frac{1}{\exp(5)} \approx \frac{1}{\sum_{j=0}^{9} \frac{(5)^j}{j!}}$.

Typing `exp(-5)` in MATLAB command prompt (with default double precision short format) returns 0.0067. Let us call this as "true value". Submit a MATLAB script (.m file) named `YourlastnameYourfirstnameHW2p1.m` that computes the relative errors in using formula (i) and formula (ii). For your script, you may find MATLAB in-built command `factorial` useful.

Problem 2

Which positive integer generates the longest sequence (20 + 10 = 30 **points**)

Take any positive integer n, and perform the following operation recursively to generate a sequence until you end up with 1.

$$f(n) = \begin{cases} \dfrac{n}{2}, & \text{if } n \text{ is even,} \\ 3n+1, & \text{if } n \text{ is odd.} \end{cases}$$

Here are some examples.

Starting with 1, the sequence is 1.

Starting with 2, the sequence is $2 \to 1$.

Starting with 3, the sequence is $3 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$.

Starting with 4, the sequence is $4 \to 2 \to 1$.

It is known that starting from any positive integer $\leq 2^{68}$, we are guaranteed to end up with 1.

We are interested in the number of steps needed to reach 1. In the examples above, starting with 1, we need 0 steps to reach 1. Starting with 2, we need 1 step to reach 1. Starting with 3, we need 7 steps to reach 1. Starting with 4, we need 2 steps to reach 1.

Write a MATLAB script `YourlastnameYourfirstnameHW2p2.m` that takes a positive integer n as input, and outputs the "starting with positive integer" in the interval $[1, n]$ for which the number of steps is the largest.

For example, if the input to your code is $n = 4$, then the output of your code should be 3. This is because 3 is the positive integer in the interval $[1, 4]$ for which we get the maximum number of steps (7). Starting with any other integer in $[1, 4]$, the number 1 is reached in less than 7 steps.

Use your code to output the "starting with positive integer" in the interval $[1, 500]$ for which the number of steps is the largest. Also compute the corresponding maximum number of steps.

Hint: look up the command `max` in MATLAB documentation.