COMPUTER SCIENCE 349A, FALL 2021
ASSIGNMENT #3 - 20 MARKS

DUE FRIDAY OCTOBER 26, 2021 (11:30 p.m. PST)

This is a really large class and the logistics of grading assignments are challenging. Me and the markers require your help in making this process go smoothly. Please ensure that your assignments conform to the following requirements - any violation will result in getting a zero for the particular assignment.

- All assignments should be submitted electronically through the Brightspace course website and shoud be a **SINGLE PDF FILE** with all the material plus files for any functions (.m or .py). No other formats will be accepted.. Handwritten answers are ok but they will need to be scanned and merged into a single pdf file together with any code examples and associated plots.
- The assignment number, student name and student number should be clearly visible on the top of every page of your assignment submission.
- **PLEASE DO NOT COPY THE ASSIGNMENT DESCRIPTION IN YOUR SUBMISSION**
- The asnwers to the questions should be in the same order as in the assignment specification.
- Some of the questions of the assignments are recycled from previous years but typically with small changes in either the description or the numbers. Any submission that contains numbers from previous years in any questions will be immediately graded with zero.
- Any general assignment related questions can be posted in the Discusscion Forum in Brightspace for the assignment.
- Any specific assignment related questions (details about your solution) should be e-mailed (rlittle@uvic.ca) to me and have a subject line of the form CSC349A Assignment X, where X is the number of the corresponding assignment.
- You may use Python instead of MATLAB anywhere the instructions say MATLAB. In that case replace M-FILE or .m file with .py file. Again, I recommend the Spyder IDE for this.

Question #1 - 6 marks.
Consider the function

$$g(x) = \frac{\ln x}{x}$$

where $x > 0$.

(a) **(2 points)** Show $g(x)$ is ill-conditioned for values of x close to 1. Do this by calculating the condition number at $\tilde{x} = 1.0025$.

(b) **(4 points)** Now, calculate $g(x)$ when $x = 1.003$ and compare the relative error between x and $\tilde{x}$ to the relative erorr between $g(x)$ and $g(\tilde{x})$? Briefly explain how the condition number in (a) predicts these values.

Question #2 - 6 Marks

(a) **(2 points)** Compute exactly (using algebra) the four roots of the polynomial equation $P(x) = 0$, where

$$\begin{aligned} P(x) &= x^4 - 6x^3 + 13.5x^2 - 13.5x + 5.06249999 \\ &= (x - 1.5)^4 - 10^{-8} \end{aligned}$$

This can be most easily done by noting that $P(x) = 0$ implies that $(x - 1.5)^4 = 10^{-8}$. This polynomial equation has 2 real roots and 2 complex conjugate roots.

(b) **(2 points)** Consider the perturbed polynomial

$$\begin{aligned} Q(x) &= x^4 - 6x^3 + 13.5x^2 - 13.5x + 5.0625 \\ &= (x - 1.5)^4 \end{aligned}$$

which is obtained by perturbing one coefficient of $P(x)$ by 10^{-8}. What is the relative change in the coefficient 5.06249999, from $P(x)$ to $Q(x)$? In $Q(x) = 0$, all four roots are equal to 1.5, what is the relative change in the largest root from (a) with respect to the given perturbation?

(c) **(2 points)** From the above, what can you conclude about the condition of the problem of computing the roots of the equation $P(x) = 0$?

Question #3 - 8 Marks
For this question your are going to create a function in MATLAB or Python for approximating a root of a function using the Newton-Raphson method. You will then use your Newton function to solve an engineering problem from the textbook.

(a) **(2 points)** Write a function for Newton's method corresponding to the following pseudocode:

function root = Newton(x_0, ε, `imax`, f, fp)
$i \leftarrow 1$
output heading
while $i \leq$ `imax`
  $root \leftarrow x_0 - f(x_0)/f'(x_0)$
  output i, root
  if $|1 - x_0/root| < \varepsilon$
    return
  end if
  $i \leftarrow i + 1$
  $x_0 \leftarrow root$
end while
output "failed to converge"

Use the following (or similar) MATLAB print statements for output (print() statements in Python, similarly formatted).

```
fprintf ( ' iteration     approximation \n')
fprintf ( ' %6.0f %18.8f \n', i, root )
fprintf ( ' failed to converge in %g iterations\n', imax )
```

Note the parameters `f` and `fp` are for f and f' and should use the function handle @functionname to pass the functions.

DELIVERABLES: A copy of your MATLAB .m or Python .py file as part of your PDF and the files themselves.

(b) (**2 points**) Let

$$f(x) = \cos(x + \sqrt{2}) + \frac{x^2}{2} + \sqrt{2}x$$

and use your function in (a) to approximate a zero of $f(x)$ with $x_0 = 1$, $\varepsilon = 10^{-6}$, and imax = 50. You will need to create MATLAB (or Python) functions for $f(x)$ and $f'(x)$, to pass as arguments to Newton.

DELIVERABLES: Copies of your .m files (or .py files) for the function and its derivative and the call and output of your `Newton` function in your PDF.

(c) (**2 points**) Let x_a denote your approximate root found in part (b) and calculate $f(x_a)$, $f'(x_a)$, and the relative error of your approximation where the true root is $x_t = -\sqrt{2}$ in MATLAB (or Python). Note, your approximation should only have 4 significant figures which, given the precision of MATLAB or Python, is not great.

DELIVARABLES: Include your calls and output in the PDF.

(d) (**2 points**) Plot the graph of the function $f(x)$ from (b) on interval [-7, 5].

This can be done by entering the following MATLAB commands:

```
>> x = [-7: 0.01: 5];
>> y = function_name(x);
>> plot(x,y)
```

This will cause a graphics window to open, and you can print the graph. You should be able to tell by looking at this graph why the approximation in (c) is good but not great.

In Python I used the following commands:

```
>> import numpy as np
>> import matplotlib.pyplot as plt
>> x = np.arange(-7,5.01,0.01)
>> y = function_name(x)
>> plt.plot(x,y)
```

DELIVARABLES: Include the call to `plot` and the resulting plot in the PDF.