COMPUTER SCIENCE 349A, FALL 2022
ASSIGNMENT #4 - 20 MARKS

DUE TUESDAY NOVEMBER 8, 2022 (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
(a) Write a MATLAB or Python function

```
Bisect ( xl , xu , eps , imax, f )
```

that returns an approximation to a root of function f, in interval $[x_l, x_u]$, corresponding to the pseudocode given in Handout #8 for the Bisection method (in `xl` it is an "ell" not a "one").

The only differences from that given algorithm are the following:

- print a caption for your computed approximations by inserting the following statement just before the while statement:

  MATLAB

  ```
  fprintf( ' iteration     approximation \n')
  ```

  Python

  ```
  print( ' iteration     approximation \n')
  ```

- print each successive computed approximation by inserting the following statement after the computation of x_r at the beginning of the while loop:

  MATLAB

  ```
  fprintf( ' %6.0f %18.8f \n', i, xr )
  ```

  Python

  ```
  print('{:6.0f} {:18.8f}\n'.format(i,xr))
  ```

- print a message to indicate that the algorithm has failed to converge in `imax` steps by replacing the last statement in the pseudocode by the following:

  MATLAB

  ```
  fprintf( ' failed to converge in %g iterations\n', imax )
  ```

  Python

  ```
  print( ' failed to converge in ',imax,' iterations\n')
  ```

Use the function Bisect() to solve the following two problems (parts (b) and (c)). In each case, you will need to write another function file with header

```
function  y = f(x)
```

corresponding to the function of which you are computing a zero.
(b) Use Bisect to solve the following problem, you are designing a spherical tank to hold water for a small village in a developing country. The volume that it holds can be computed as

$$V = \pi h^2 \frac{3R - h}{3}$$

where V is the volume in m^3, h is the depth of the water in the tank in meters, and R is the tank radius in meters. If $R = 4.1$, determine the depth h that the tank must be filled to in order that the tank holds 45 m^3 of water. In Bisect, use an initial interval of $[0, 4.1]$, eps $= 10^{-4}$ and imax $= 20$ (as was done in Handout 8).
(c) Use Bisect to solve the following problem, The velocity v of a falling parachutist of mass m kg is given by

$$v = \frac{gm}{c}(1 - e^{-ct/m})$$

, where $g = 9.81$ meters per second squared. For a parachutist with drag coefficient $c = 13.5$ kg/s, compute the mass m so that the velocity is $v = 40$ meters per second at time $t = 10$ seconds. In Bisect, use an initial interval of $[1, 100]$, eps $= 10^{-4}$ and imax $= 20$.

Question #2 - 8 Marks

For this question your are going to create a function 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 for Newton's method corresponding to the following pseudocode:

function root = Newton(x_0, ε, `imax`)
$i \leftarrow 1$
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) print statements for output (same as question 1).

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

Pass the additional parameters `f` and `fp` for f and f' and use the function handle @functionname to pass the functions if using MATLAB. (similar to the way f was a parameter in the bisection function in my code).

(b) **(2 points)**

An oscillating current in an electric circuit is described by $i = 9e^{-t}\cos(2\pi t)$, where t is in seconds. Determine all positive values of t such that $i = 3.5$. The solution to this problem can be posed as determining the positive zeros of a function $f(t)$. Use the MATLAB function `ezplot` to draw a graph of the function $f(t)$ on the interval $[-3, 2]$. The following MATLAB statement illustrates the syntax of `ezplot`:

```
ezplot('exp(-x)+cos(pi*x)', [-1  5])
```

will cause a graphics window to open and will display the graph of the function $e^{-x} + \cos(\pi x)$ on the interval $[-1, 5]$.

DELIVARABLES: Include the call to `ezplot` and the resulting plot.

(c) **(2 points)** Careful analysis of the graph of the function $f(t)$ in part (b) shows that $f(t)$ has only one positive zero, and it is in the interval $[0, 1]$.

Use the function `Newton()` you wrote in (a) with

$$x_0 = 0.1, \varepsilon = 10^{-6}, \text{ and } imax = 20$$

to solve the above problem.

Note: You will need to write function files with headers something like

```
function y = f(t)
```

and

```
function y = fp(t)
```

corresponding to the function $f(t)$ that you are computing a zero of, and its derivative $f'(t)$.

(d) **(2 points)** Although the above function $f(t)$ has only one positive zero, it has many negative zeros. The use of Newton's method to compute the positive zero is very sensitive to the choice of an initial condition. Show this by using Newton as follows: with

$$x_0 = 0.5, \varepsilon = 10^{-6}, \text{ and } imax = 20$$

and

$$x_0 = 0.9, \varepsilon = 10^{-6}, \text{ and } imax = 20.$$

Question #3 - 6 Marks.
Consider the polynomial, $f(x) = x^5 + 2x^4 - 5x^2 - 7x - 3$, and solve the following questions by hand.

(a) **(3 points)** Use Horner's algorithm to compute $f(1)$ and $f'(1)$.

(b) **(1 point)** Suppose we use the values in (a) to obtain x_1 using Newton's method, as an approximation to a true root of $f(x)$. What is the approximate deflated polynomial?

(c) **(2 points)** If instead we let Newton continue after calculating x_1, until it converges to the actual root $x_t = -1$, what is the order of convergence? Justify your answer.