COMPUTER SCIENCE 349A, SPRING 2025, ASSIGNMENT #1

DUE SUNDAY JANUARY 19, 2025 (11:30 p.m. PST)

To facilitate the grading of a large number of submissions, please ensure that your assignments conform to the following requirements - any deviation will result in getting a zero for the particular question or assignment.

- All assignments should be submitted through the Brightspace course website using the Crowdmark tool. Each question will be submitted separately in Crowdmark. You may be asked to provide code for any functions (.m or .py). No languages other than MATLAB or Python will be accepted.
- Crowdmark only accepts PDF (or image files) for upload. Thus you will need to submit PDF images of your .m (or .py) files, rather than the actual code file. However, you should retain copy of all the files you create for later reference.
- **PLEASE DO NOT COPY THE ASSIGNMENT DESCRIPTION IN YOUR SUBMISSION**
- The answers 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.
- You may use Python instead of MATLAB anywhere the instructions say MATLAB. In that case replace M-FILE or .m file with .py file. Using the Spyder IDE for this is recommended.

Question #1 - 8 marks

A MATLAB function M-file for Euler's method (as described in class and on pages 17-19 of the textbook) for solving the differential equation

$$\frac{dv}{dt} = g - \frac{c}{m}v$$

(this is (1.9) on page 14) is given below. As discussed in class, this numerical method is obtained by approximating $\frac{dv}{dt}$ at time t_i by $\frac{v(t_{i+1})-v(t_i)}{t_{i+1}-t_i}$, which results in the computed approximation

$$v(t_{t+1}) = v(t_i) + \left[g - \frac{c}{m}v(t_i)\right](t_{i+1} - t_i).$$

To create a MATLAB function M-file, either type **edit** in the Command Window or select **HOME** → **New** → **Script** or select **HOME** → **New** → **Function** (the latter gives you a template for creating a function).

Each of these options will open a new window (an Editor window) in which to type in the MATLAB statements for Euler's method. Enter the following, (Note - don't copy and paste as the quotes might not work). Statements starting with % are comments, documenting the MATLAB code.

```
function Euler(m,c,g,t0,v0,tn,n)
%  print headings and initial conditions
fprintf('values of t    approximations v(t)\n')
fprintf('%8.3f',t0),fprintf('%19.4f\n',v0)
%  compute step size h
h=(tn-t0)/n;
%  set t,v to the initial values
t=t0;
v=v0;
%  compute v(t) over n time steps using Euler's method
for i=1:n
     v=v+(g-c/m*v)*h;
     t=t+h;
     fprintf('%8.3f',t),fprintf('%19.4f\n',v)
end
```

To save your M-file, select

EDITOR → **Save** → **Save As...**

At the top of this window, it should say

File name: Euler.m

Save as type: MATLAB files (*.m)

Select

Save

to save your file, and close the Editor window.

In order to use the above function, you must specify values for the 7 local parameters in the function Euler:

`m` is the mass of the falling object
`c` is the drag coefficient
`g` is the gravity constant
`t0` is the initial time, `v0` is the initial velocity
`tn` is the final time at which the velocity is to be computed
`n` is the number of time steps into which $[t_0, t_n]$ is divided

Thus, in the function `Euler`, the step size $h = (t_n - t_0)/n$ is computed, and Euler's method is used to compute an approximation to the solution $v(t)$ of the differential equation at the n points (values of time)

$$t_0 + h,\ t_0 + 2h,\ t_0 + 3h,\ t_0 + 4h,\ \ldots,\ t_0 + nh \ = \ t_n.$$

For example, in order to use Euler to solve the problem given in Example 1.2 on page 17 and to save your results in a file for printing, you could enter the following (in the MATLAB Command Window):

diary filename
Euler (68.1 , 12.5 , 9.81, 0 , 0 , 12 , 6)

{*the desired results should appear here*}

diary off

(a) Create a working copy of the Euler function using MATLAB or Python. Submit a copy of your function definition (as a PDF file).

(b) Use Euler to solve the differential equation using $m = 75.3\ kg$, $c = 13.1\ kg/s$ and initial conditions $v(0) = 0$ on the time interval $[0, 16]$ using 40 time steps and for a free-falling body on Mars where the gravitational constant is $g = 3.72\ m/s^2$. Submit the call to Euler and the resulting output.

(c) Using the same parameters from Q1(b) calculate the corresponding velocities using the analytic solution. That is, create a function for the following formula, using the numerical values from Q1(b) for the constants,

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

and run it for $t = [t0 : h : tn]$. Submit the MATLAB definition of the function and the results of running it.

(d) What is the terminal velocity of this free-falling body of mass 75.3 kg on Mars? (You may approximate this roughly with experimental results from your function or solve it analytically.) Submit the details of your solution.

Question #2 - 6 Marks.

Newton's law of cooling says that the temperature of a body changes at a rate proportional to the difference between its temperature and that of the surrrounding medium (the ambient temperature),

$$\frac{dT}{dt} = -k(T - T_a)$$

where T is the temperature of the body ($^\circ C$), t is time (minutes), k is the proportionality constant (per minute), and T_a is the ambient temperature ($^\circ C$).

(a) Modify your function Euler in Question 1 so that it will use Euler's method to solve this differential equation. Use the function header (Note that the last argmument is h and not n here.)

    Euler2(k , Ta, t0 , T0 , tn , h)

where Ta = T_a, the initial condition T0 = $T(t_0)$, tn is the final value of t in the numerical solution, and h is the step size.

Submit a copy of your function definition (as a PDF file).

(b) Use Euler2 to compute a numerical approximation to the above differential equation using $k = 0.019$/min, $T_a = 20^\circ C$ and initial condition $T(0) = 68^\circ C$ on the time interval $[0, 12]$ using a step size of 0.125 minutes.

Submit the call to Euler2 and the resulting output

(c) Use the fact the exact analytic solution of this problem is

$$T(t) = 20 + 48e^{-0.019t}$$

to compute (either in MATLAB, Python or using your calculator) the relative error in the computed solution at $t = 12$.

Submit your calculations, and the computed relative error.

Question #3 - 6 Marks

The constant π, which is the ratio between the circumference of a circle and its diameter, whose value to 50 decimal digits is

$$3.14159265358979323846264338327950288419716939937510$$

(a) (1 mark) Let $p = 3.141592$ represent the "true" value of π (technically, just a 7 digit approximation of π). Use p to evaluate the relative error of p_i for each $i = 1, 2, 3, 4$, where i is the number of significant digts of p we use in our approximation. That is, let $p_1 = 3$ and calculate the relative error, then let $p_2 = 3.1$ and calculate the relative error, etc.

Submit your calculations and the resulting relative error value (as a decimal and a percentage) for each i

(b) (3 marks) There are many inifinite series that may be used to compute the value of π. For example, $\pi = 4 \cdot (F(2) + F(3))$, where

$$F(x) = \frac{1}{x} - \frac{1}{3x^3} + \frac{1}{5x^5} - \frac{1}{7x^7} + \cdots = \sum_{j=0}^{\infty} \frac{(-1)^j}{(2j+1)x^{2j+1}}$$

Now use p to evaluate the relative error of q_j for each $j = 0, 1, 2, ..., n$, where j corresponds to the jth partial sum from the formula above and where n is the smallest value of j such that q_j has 4 signifcant digits of p. That is, let $q_0 = 4(\frac{1}{2} + \frac{1}{3})$, $q_1 = 4(\frac{1}{2} - \frac{1}{24} + \frac{1}{3} - \frac{1}{81})$, $q_2 = 4(\frac{1}{2} - \frac{1}{24} + \frac{1}{160} + \frac{1}{3} - \frac{1}{81} + \frac{1}{1215})$ etc. Use the relative error between each q_j and p to determine the number of significant digits of each approximation.

Submit your calculations for each j, showing the value of q_j, the relative error of q_j with respect to p, and the resulting number of significant figures.

(c) (2 marks) The following infinite series, discovered independently by Madhava, Leibniz and Gregory, is another way to compute π:

$$\pi = 4(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots) = 4\sum_{j=0}^{\infty} \frac{(-1)^j}{2j+1}$$

Using the value of n from Part (b), compute the relative error obtained by estimated p using $4\sum_{j=0}^{n} \frac{(-1)^j}{2j+1}$

Submit your calculation showing the estimated value and its relative error.