COMPUTER SCIENCE 349A, SPRING 2017
ASSIGNMENT #1 - 20 MARKS

DUE FRIDAY JANUARY 20, 2017 (11:55 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 ConneX course website and shoud be **SINGLE PDF FILES**. 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 assignment related email questions should have a subject line of the form CSC349A Assignment X, where X is the number of the corresponding assignment.
- The total number of points for this assignment is 20.

Question #1 - 10 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. 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 copy of the MATLAB function Euler in your installation of MATLAB. Modify the function so that it creates and returns two matrices, one containing the times used in the calculations and one for the resulting velocities.

Note, we index matrices in MATLAB with round brackets and we start indexing at 1 NOT 0. That is, if t is a vector representing the times, then $t(1)$ should contain time t_0 and $t(2)$ should contain time $t_1 = t_0 + h$, etc. The final matrices should be

$t = [t_0\, t_1\, \ldots\, t_n]$ and $v = [v(t_0)\, v(t_1)\, \ldots\, v(t_n)]$.

DELIVERABLES: A copy of the M-FILE in your pdf.

(b) Use Euler to solve the differential equation using $m = 70$ kg, $c = 12$ kg/s and initial conditions $v(0) = 0$ on the time interval $[0, 12]$ using 15 time steps and $g = 9.81$.

DELIVERABLES: The function call to `Euler` and the resulting output.

(c) Using the same parameters from Q1(b) solve the differential equation analytically in MATLAB. That is, create a function for the formula,

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

and run it for $t = [t0 : h : tn]$.

DELIVERABLES: The function, the call and the results.

(d) Plot the results from Q1(b) and Q1(c) together to compare.

DELIVERABLES: The commands and resulting plot.

(e) What is the terminal velocity of this free-falling body of mass 70 kg? (You may approximate this roughly with experimental results from your function or solve it analytically.)

DELIVERABLES: MATLAB output if using MATLAB or your analytic solution.

s_m	s_e	e_1	e_2	t_1	t_2	t_3	t_4

Question #2 - 5 Marks.

Consider a ternary, normalized floating-point number system that is base 3. Analogous to a bit, a ternary digit is a trit. Assume that a hypothetical ternary computer uses the following floating-point representation:

where s_m is the sign of the mantissa and s_e is the sign of the exponent (0 for positive, 1 for negative), t_1, t_2, t_3 and t_4 are the trits of the mantissa, and e_1, e_2 are the trits of the exponent where each trit is 0,1 or 2. Write the following numbers (parts (a) to (d)) using this representation where X_{10} is used to indicate that the number provided is in decimal. Show all your work.

(a) $\left(3 + \frac{2}{9}\right)_{10}$

(b) -55_{10}

(c) The largest number that can be represented in this system.

(d) The smallest positive non-zero number that can be represented in this system.

(e) What is the spacing for numbers between 3_{10} and 9_{10} in this ternary floating-point representation system? Your answer should be in decimal. Don't just provide the answer but also show your reasoning.

Question #3 - 5 Marks

The function e^{-x} can be approximated by its McLaurin series expansion as follows (note the alternating + and −):

$$e^{-x} \approx 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \ldots \pm \frac{x^n}{n!}$$

Alternatively, note that $e^{-x} = \frac{1}{e^x}$. Thus, e^{-x} can also be apporximated by 1 over the McLaurin series expansion of e^x. That is,

$$e^{-x} \approx \frac{1}{1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!}}$$

Approximate e^{-2} using both approaches above for $n = 1, 2, ..., 6$ and compare each approximation to the true value of $e^{-2} = 0.135335...$, using the true relative error. What conclusions can you make about the two approaches?