ELEC 418 Homework #4

1)

a. To find the local truncation error of a leap frog method, consider the below equations,

$$\frac{dx(t)}{dt} = \mu x(t) \tag{1}$$

$$\frac{x[m+1]-x[m-1]}{2h} = \mu x[m] \tag{2}$$

Where $x[m+1] = x[m-1] + 2h\mu x[m]$

Let $\mu x[m] \approx \mu x(t)$ where $t = mh$ and h is a constant step size

$$x(t_{m+1}) = x(t_{m-1}) + 2hF(x(t_m), t_m) \tag{3}$$

Where $\mu x(t_m) = F(x(t_m), t_m) = x'(t_m)$

For a perfect equation above equation must be consistent, but it is not. Hence an error accumulates. For the Leap-Frog method local truncation error is,

$$LTE = x(t_{m+1}) - x(t_{m-1}) - 2hF(x(t_m), t_m) \tag{4}$$

To find the error I will expand a Taylor series expansion where,

$$x(t_{m+1}) = x[mh + h] \tag{5}$$

$$x(t_{m-1}) = x[mh - h] \tag{6}$$

$$LTE = [\, x(t_m) + x'(t_m).h + x''(t_m).\frac{h^2}{2} + x'''(t_m).\frac{h^3}{3!} + \cdots] \tag{7}$$

$$-[x(t_m) - x'(t_m).h + x''^{(t_m)}.\frac{h^2}{2} - x'''(t_m).\frac{h^3}{3!} + \cdots]$$

$$-2hF(x(t_m), t_m)$$

Equation simplifies to,

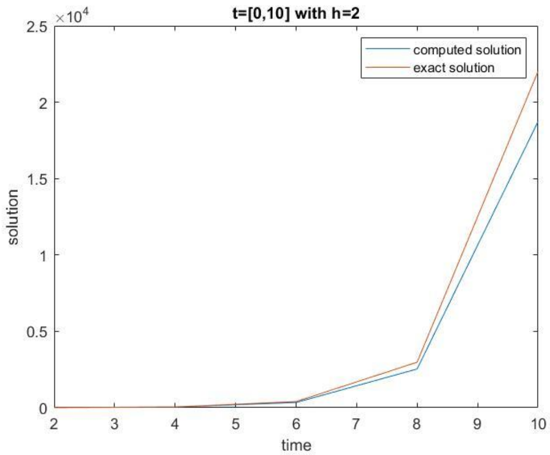
$LTE = x'(t_m).2h - 2hF(x(t_m), t_m) - [x'''(t_m).\frac{h^3}{3!} + \cdots]$ where $x'(t_m) = F(x(t_m), t_m)$,

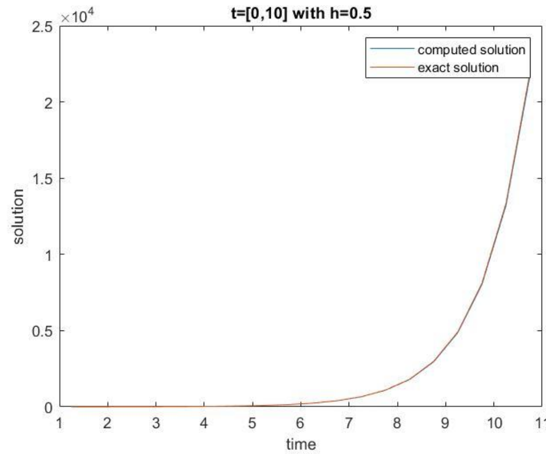
$$LTE = [x'(t_m).\frac{h^3}{3!} + \cdots] = O(h^3) \tag{8}$$

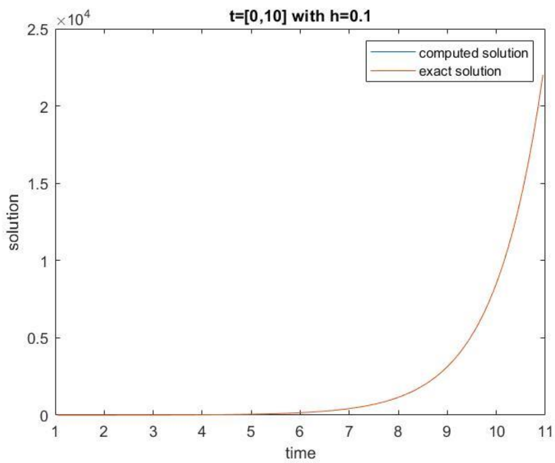
b. From the theorem we know that, if $e_m(h) = O(h^{p+1})$ than $E_m(h) = O(h^p)$,

Hence Global error is; $O(h^2)$. Midpoint method is unstable.

c. Below are the MatLab plots in the interval $t \in [0,10]$ for h=2, 0.5, 0.1. Code has been provided as P1.m







At t=10	**Computed solution**	**Exact solution**
h=2	≈18742	≈22026
h=0.5	≈21798	
h=0.1	≈22017	

d. From this solution it can be seen that with the increasing number of iteration points, approximation becomes closer to exact solution. This is logical since what we are doing in here is taking partial integrals to achieve a solution between given intervals. Remembering the derivation of integral, it is easy to see that with the increasing number of intervals solution becomes closer to exact solution at infinity.

2) -

3)

a. To solve the Van der pol oscillator, we need to reduce it to a first order ordinary differential equation.

$$\frac{d^2x(t)}{dt^2} = -x(t) + \mu(1 - x^2(t))\frac{dx(t)}{dt}$$

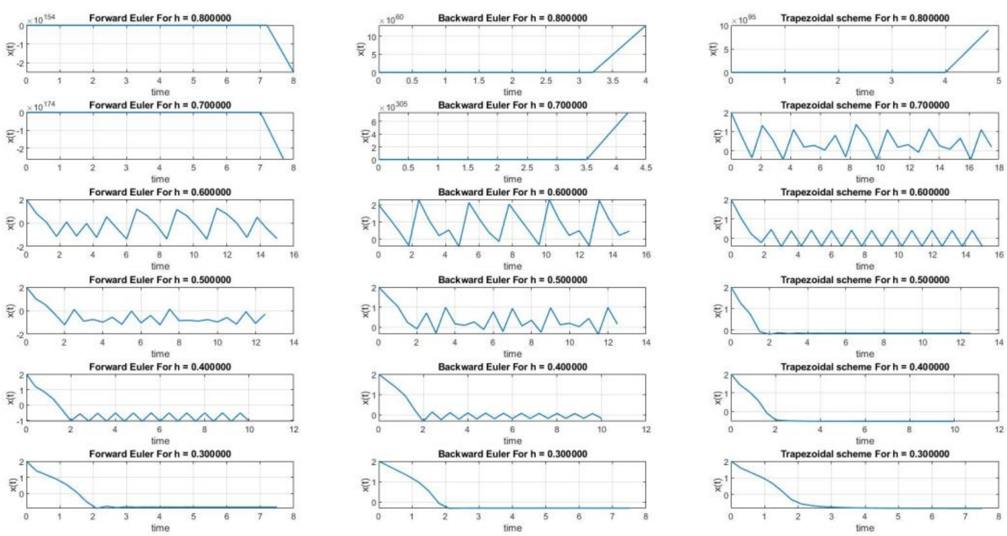
Making the substitution, $\frac{dx(t)}{dt} = x'(t) = x_2(t)$

$$x'(t) = x_2(t) \quad (1)$$

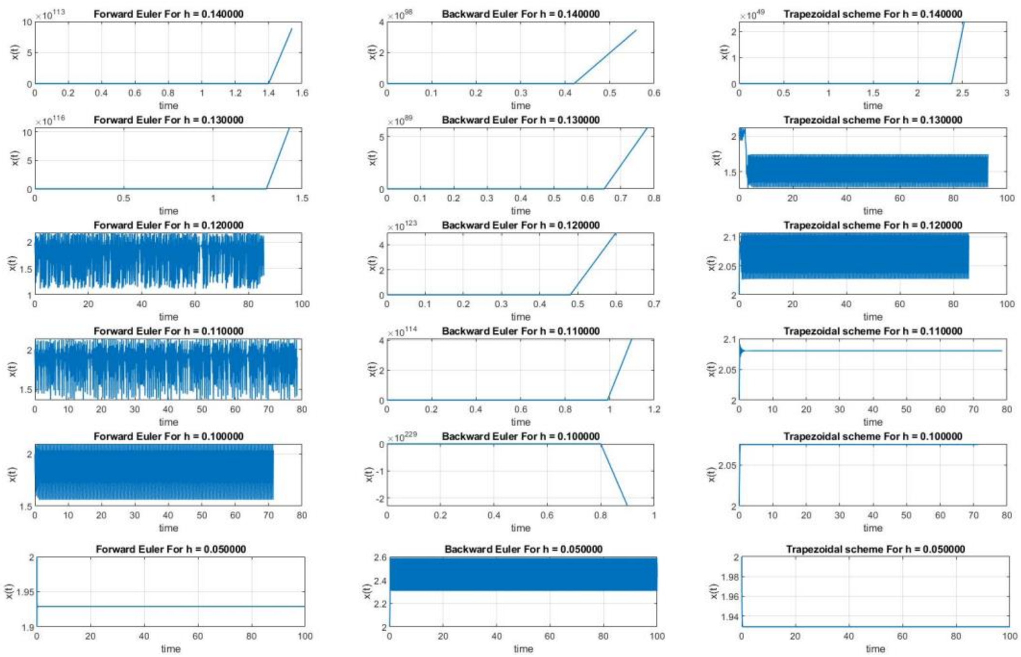
$$x_2'(t) = \mu\left(1 - x^2(t)\right)x_2(t) - x(t) \quad (2)$$

b. Below are the different schemes implement for the different values of h. First set of graphs are for the μ=1 and the second set of plots are for the μ=10 where $t \in [0, \max(20,10\mu)]$. MatLab code has been provided as *P3.m*.

For μ=1,



For μ=10,



➔Because of the immense number oscillations in the plots where oscillation happens MatLab plot shows the plots fully filled in. this happens since too many oscillations are stacked in.

c. As it can be clearly seen that when the oscillation starts for the Forward Euler, oscillation happens below a certain point. On the other hand for the Backward Euler oscillation happens above that same point. For the increasing value of μ, that point gets close to the initial point of x(0)=2. Also increasing number of h greatly increases the number of oscillations. Also for the increasing number of mu h gets smaller to have an oscillation.

4) Complete code for this part has been provided as *P4.m*.

a. Computed coefficients for explicit LMM are below in the graph. For explicit, $\beta_0 = 0$.

We are given initial values for $\alpha_0 = 1$, $\alpha_1 = -1$ and $\alpha_3 = \ldots = \alpha_k = 0$. Also $\alpha_2 = 0$ since $\sum_{j=0}^{k} \alpha_k = 0$.

Coefficients	k=1	k=2	k=3	k=4
β_1	1	3	2.8447	4.6146
β_2	-	-2	-13.7664	-21.9171
B_3	-	-	11.8917	112.8947
B_4	-	-	-	-91.5922

Δt is LTE proportional to k+1.

b. Computed coefficients for implicit LMM are below in the graph.

We are given initial values for $\alpha_0 = 1$, $\alpha_1 = -1$ and $\alpha_3 = \ldots = \alpha_k = 0$. Also $\alpha_2 = 0$ since $\sum_{j=0}^{k} \alpha_k = 0$.

Coefficients	k=1	k=2	k=3	k=4
β_0	0.5	0.3662	0.3208	0.2915
β_1	0.5	0.8283	1.0440	1.2729
β_2	-	-0.1944	-0.3637	-1.2487
β_3	-	-	0.9989	4.5884
β_4	-	-	-	0.9040

Δt is LTE proportional to k.

c. Computed coefficients for implicit LMM are below in the graph.

We are given initial values for $\alpha_0 = 1$, and $\beta_1 = \ldots = \beta_k = 0$.

Coefficients	k=1	k=2	k=3	k=4
α_1	-1	-2.0847	-2.9689	-3.6640
α_2	-	1.0847	5.1445	10.1553
α_3	-	-	-3.1756	-23.6955
α_4	-	-	-	16.2042
β_0	0.6	0.4421	0.3735	0.3363

Δt is LTE proportional to k.

➔Looking for the every part for the every value of k, $\sum_{j=0}^{k} \alpha_k = 0$ holds

5)

a. MatLab code has been provided as P5.m user should choose uncomment fa, fb or fc to solve.

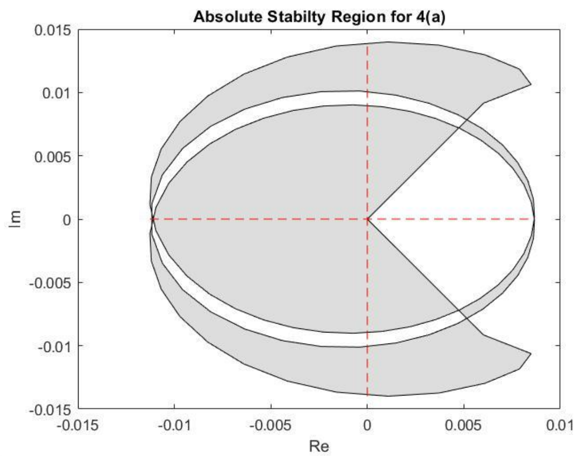
$\sum_{j=0}^{k} \alpha_k y^{l-j} = \Delta t \sum_{j=0}^{k} \mathrm{b}_k F(y^{l-j}, \mathrm{t})$ now describing the right and left side as,

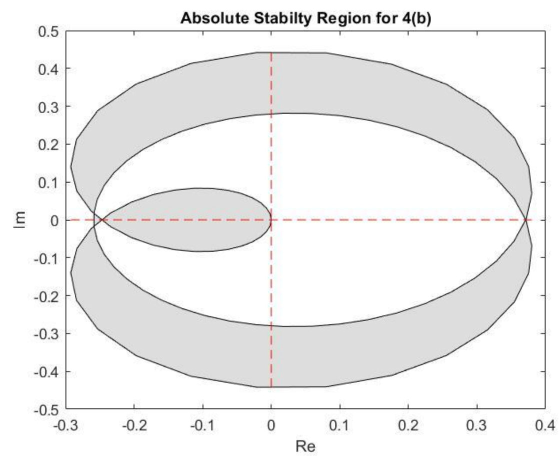
$$\sigma(E) = \beta_0 E^k + \beta_1 E^{k-1} + \beta_2 E^{k-2} + \cdots + \beta_k E^0 \tag{1}$$

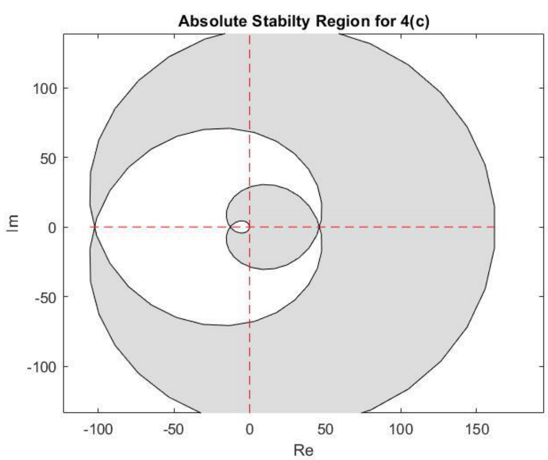
$$\rho(E) = \alpha_0 E^k + \alpha_1 E^{k-1} + \alpha_2 E^{k-2} + \cdots + \alpha_k E^0 \tag{2}$$

$$r(\theta) = \frac{\rho(e^{j\theta})}{\sigma(e^{j\theta})} \qquad 0 \le \theta \le 2\pi \tag{3}$$

b. Below are the regions of absolute stability for k=1, 2, 3, 4.







➔ For the equations at 4(a) and 4(b) formulas are stable since region is inside z=1.For the equation 4(c) we cannot say such thing since looking at the biggest magnitude of its root 0.1427 + 2.6062i, z=2.6101 and it's clear that the region is not in the magnitude of root, hence system is not stable.

6)

a. To determine the exact solution,

$$\frac{dx(t)}{dt} = -100[x(t) - \cos(t)] - \sin(t) \tag{1}$$

$$\frac{dx(t)}{dt} - 100x(t) = 100\cos(t) - \sin(t) \tag{2}$$

First, finding the homogeneous solution,

$$\frac{dx(t)}{dt} - 100x(t) = 0$$

$$X_h = Ae^{-100t} \tag{3}$$

Second finding the particular solution,

$$X_p = Acos(t) + Bsin(t) \tag{4}$$

X_P satisfies the equation 2, looking at X_h,

$$\frac{dX_h}{dt} - 100X_h = 100\cos(t) - \sin(t)$$

$$-Asin(t) + Bsin(t) + 100Acos(t) + 100Bsin(t) = 100\cos(t) - \sin(t) \tag{5}$$

Solving for A and B yields to, A=1, B=0. Hence X_p becomes,

$$X_p = cos(t) \tag{6}$$

Now finding the solution,

$$x(t) = X_h + X_p \tag{7}$$

$$x(t) = Ae^{-100t} + \cos(t) \tag{8}$$

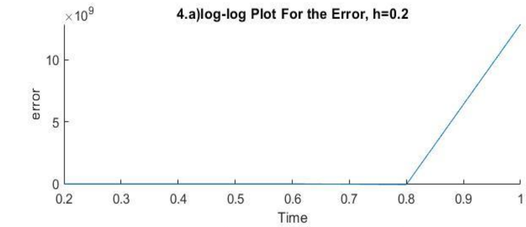
Applying initial condition$x(0) = 0$, $x(0) = A + 1$. Than A becomes 0. Hence,

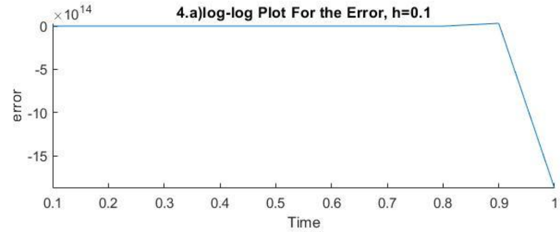
$$x(t) = \cos(t) \tag{9}$$

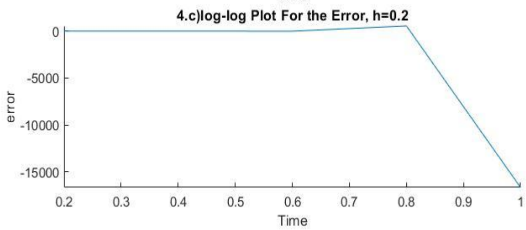
b. Implemented solution for part a with IVP & LMM provided as P6.m and commented in the code file.

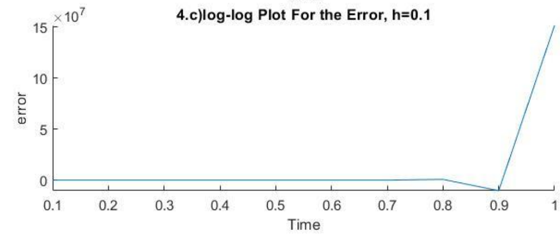
c. Implemented solution for part c with IVP & LMM provided as P6.m and commented in the code file.

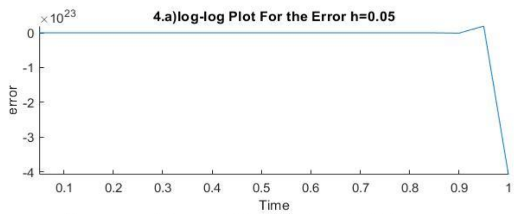
d. Below are the graphs at h= 0.2, 0.1, 0.05, 0.02, 0.01, 0.005, 0.001 implemented with the formulas derived at 4.a and 4.b. Code has been provided as *P6.m*.

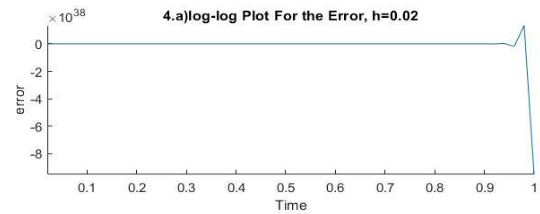


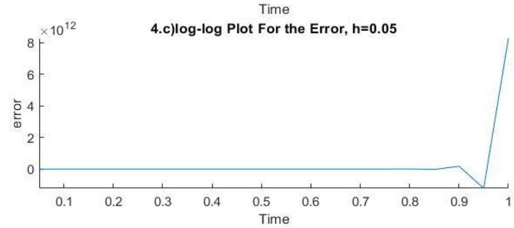


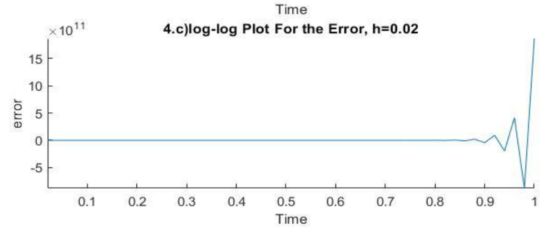


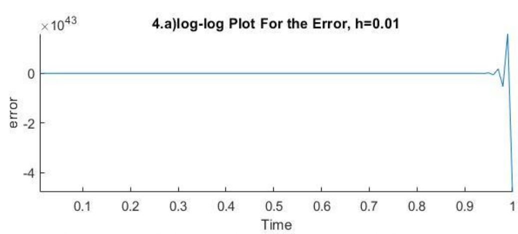


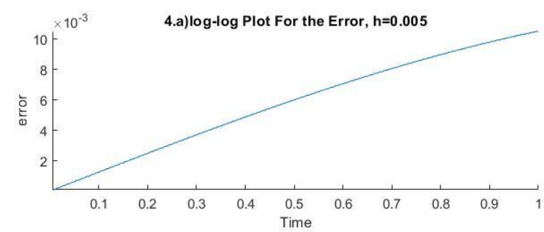


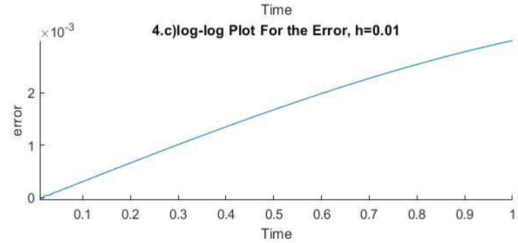


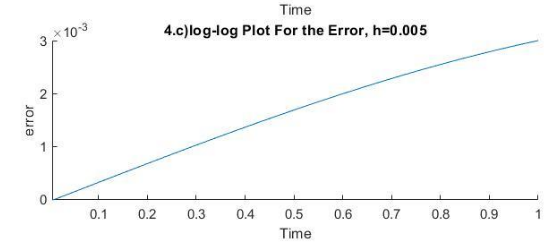


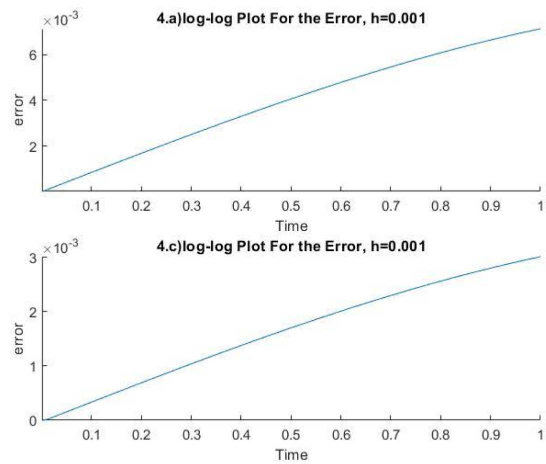












e. It is clearly seen that with the increasing number of step size formulas becomes more reliable and gets closer to the exact solution. On the other hand formula derived at part c clearly gives less error, hence it is more reliable. Not only staying with that, formula derived at part c needs less step size to become relatively more reliable. Then again methods fail catastrophically at some step sizes. For the formula of part a this threshold is h=0.005 while for the formula of part c it is h=0.01