Control System Lab (EEP310)

DATE : 31-07-2019

Experiment- 1(A)

Title of the Experiment: To study about ODEs(ordinary differential equations) in Controls System and their solving techniques in Matlab.

Software: Matlab

Theory:

All linear systems can be described by a linear ordinary differential equation. So, by analysing the ODEs obtained, behavior and response of the system can be interpreted. As in controls and system usually we deal with system equations which are in differential form so it require better understanding of ODEs.

Ordinary Differential Equations: An ordinary differential equations (ODE) contains one or more derivatives of a dependent variable, y, w.r.t a single independent variable t, usually refered to as a time. Commonly used ODE Solving Functions are:

1. Ode45
2. Ode23
3. Ode113
4. Ode15s
5. Ode23s
6. Ode23t
7. Ode15i

Systems of ODEs: You can specify any number of coupled ODE equations to solve, and in principle the number of equations is only limited by available computer memory. If the system of equations has n equations,

$$\begin{pmatrix} y'_1 \\ y'_2 \\ \vdots \\ y'_n \end{pmatrix} = \begin{pmatrix} f_1(t, y_1, y_2, \ldots, y_n) \\ f_2(t, y_1, y_2, \ldots, y_n) \\ \vdots \\ f_n(t, y_1, y_2, \ldots, y_n) \end{pmatrix},$$

Using Ode45: Solve non-stiff differential equations — medium order method.

Syntax: $[t, y] = \mathrm{ode45}(\mathrm{odefun}, \mathrm{tspan}, y_0)$

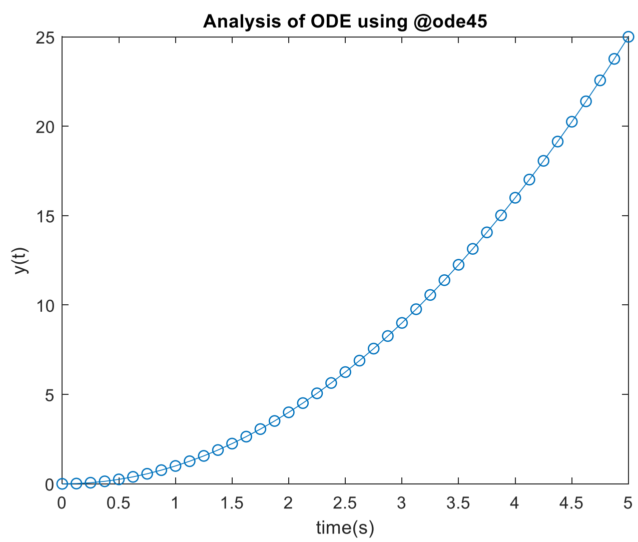
Description: $[t, y] = \mathrm{ode45}(\mathrm{odefun}, \mathrm{tspan}, y_0)$, where $\mathrm{tspan} = [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE y' = 2t. use a time interval of [0,5] and initial conditions $y(0) = 0$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 0;
[t,y] = ode45(@(t,y) 2*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode45');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode23: ode23 can be more efficient than ode45 at problems with crude tolerances, or in the presence of moderate stiffness.

Syntax: $[t, y] = \text{ode23}(\text{odefun}, \text{tspan}, y_0)$

Description: $[t, y] = \text{ode23}(\text{odefun}, \text{tspan}, y_0)$, where tspan $= [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

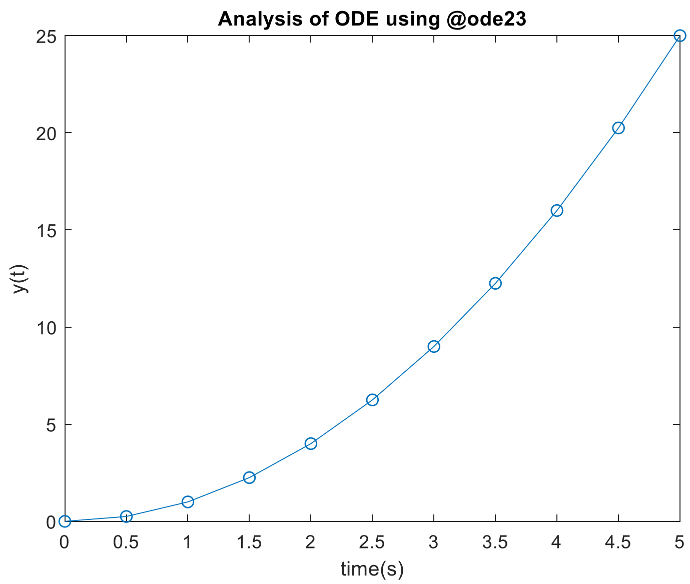
Example: Solve the ODE y' $=$ 2t. use a time interval of [0,5] and initial conditions $y(0) = 0$.

Code:

```
clc;
clear all;
close all;
```

```
tspan = [0 5];
y0 = 0;
[t,y] = ode23(@(t,y) 2*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode23');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode113: ode113 can be more efficient than ode45 at problems with stringent error tolerances, or when the ODE function is expensive to evaluate.

Syntax: $[t, y] = \text{ode113}(\text{odefun}, \text{tspan}, y_0)$

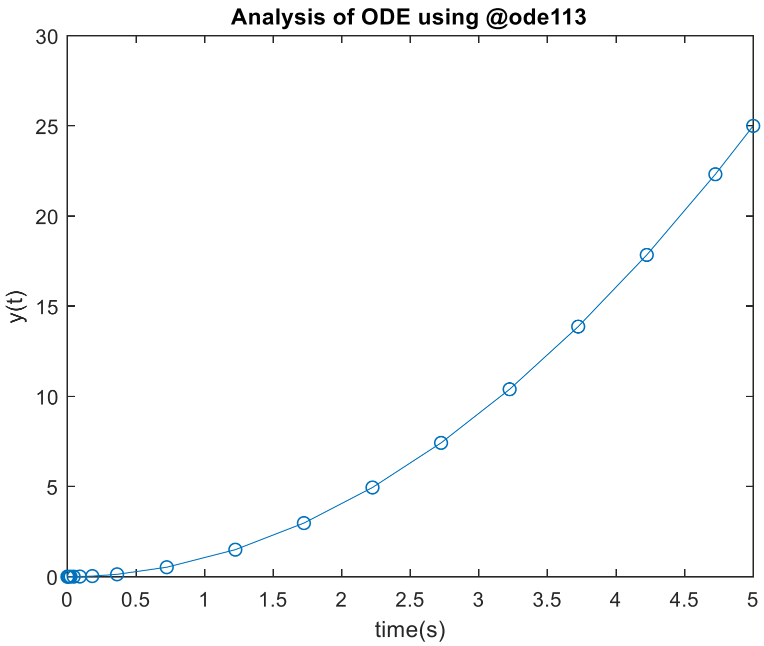
Description: $[t, y] = \text{ode113}(\text{odefun}, \text{tspan}, y_0)$, where $\text{tspan} = [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE $y' = 2t$. use a time interval of [0,5] and initial conditions $y(0) = 0$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 0;
[t,y] = ode113(@(t,y) 2*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode113');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode15s: Try ode15 and ode45 fails or is inefficient and you suspect that the problem is stiff. Also use ode15s when solving differential algebraic equations(DAEs).

Syntax: $[\mathrm{t}, \mathrm{y}] = $ ode15s(odefun, tspan, y_0)

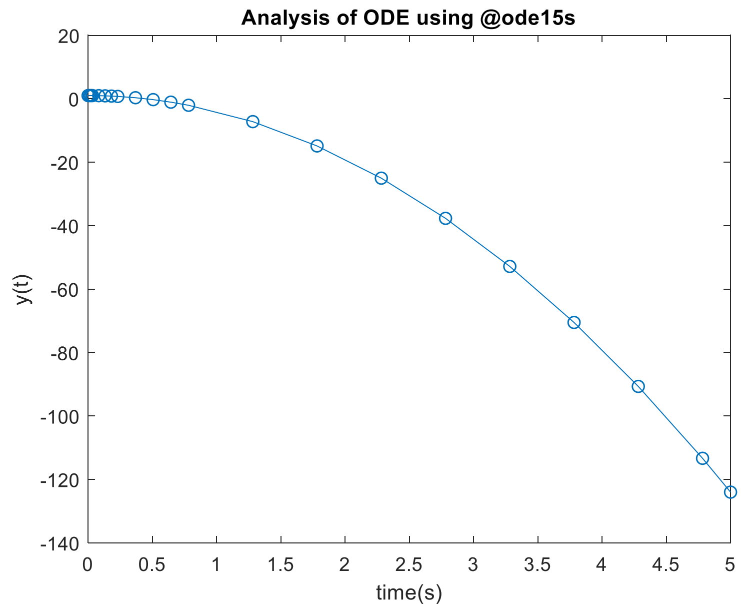
Description: $[\mathrm{t}, \mathrm{y}] = $ ode15s(odefun, tspan, y_0), where tspan $= [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE $y' = -10t$. use a time interval of [0,5] and initial conditions $y(0) = 1$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 1;
[t,y] = ode15s(@(t,y) -10*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode15s');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode23s: ode23s can be more efficient than ode15s at problems with crude error tolerances. It can solve some stiff problems for which od15s is not effective. Ode23s computes the Jacobian so it is beneficial to provide the jacobian via odeset to maximize efficiency and accuracy.

Syntax: [t, y] = ode23s(odefun, tspan, y_0)

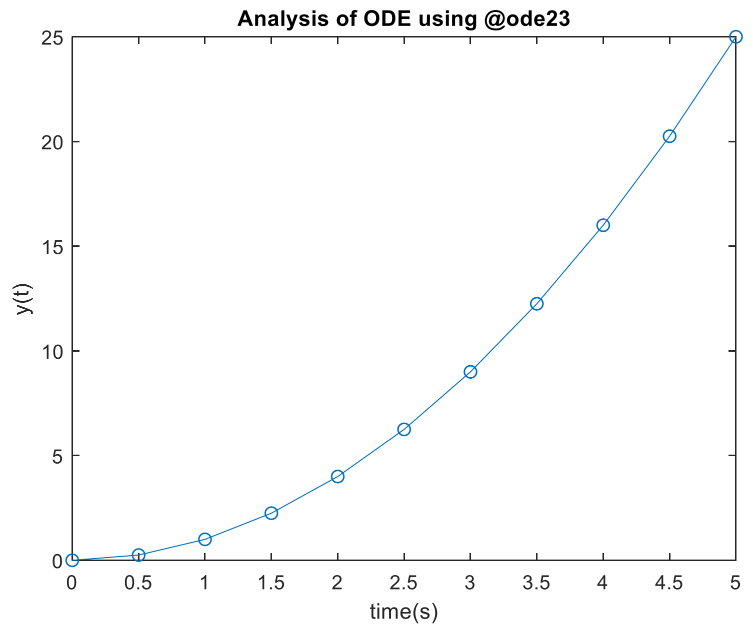
Description: [t, y] = ode23s(odefun, tspan, y_0), where tspan = [t_0 t_f], integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE y' $= -10t$. use a time interval of [0,5] and initial conditions $y(0) = 1$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 1;
[t,y] = ode23s(@(t,y) -10*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode23s');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode23t: use ode23t if the problem is only moderately stiff and you need a solution without numerical damping.

Syntax: $[t, y] = \text{ode23t}(\text{odefun}, \text{tspan}, y_0)$

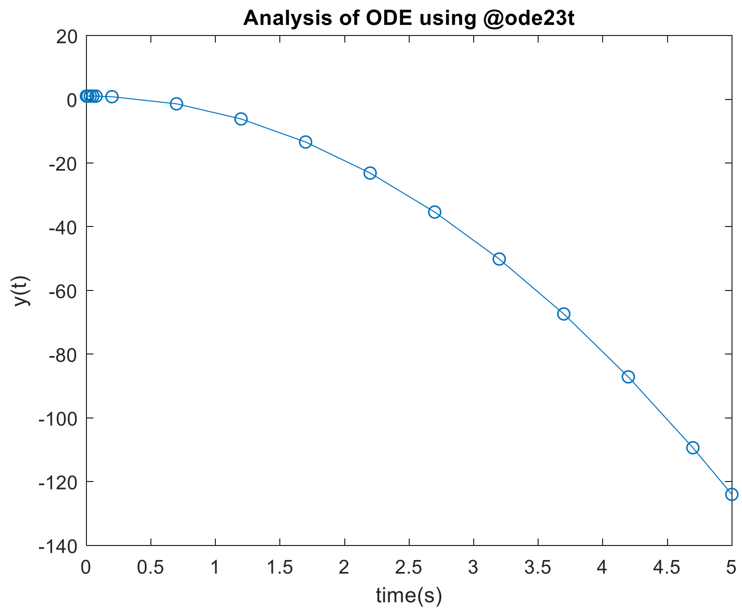
Description: $[t, y] = \text{ode23t}(\text{odefun}, \text{tspan}, y_0)$, where tspan $= [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE $y' = -10t$. use a time interval of [0,5] and initial conditions $y(0) = 1$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 1;
[t,y] = ode23t(@(t,y) -10*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode23t');
xlabel('time(s)'),ylabel('y(t)');
```

Output :



Using Ode15i:

Syntax: $[t, y] = od15i(odefun, tspan, y_0)$

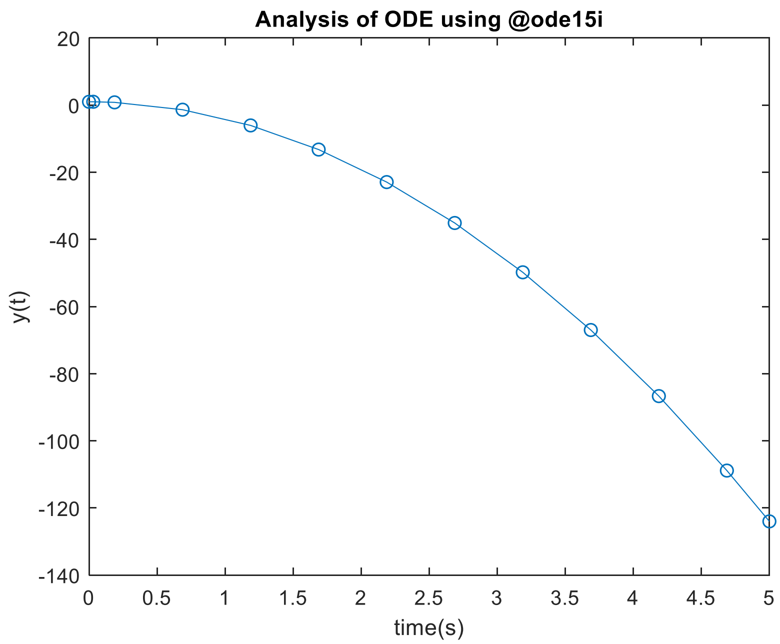
Description: $[t, y] = ode15i(odefun, tspan, y_0)$, where tspan $= [t_0\ t_f]$, integrates the system of differential equations $y' = f(t, y)$ from t_0 to t_f with initial conditions y_0. Each row in the solution array y corresponds to a value returned in column vector t.

Example: Solve the ODE $y' = -10t$. use a time interval of [0,5] and initial conditions $y(0) = 1$.

Code:

```
clc;
clear all;
close all;
tspan = [0 5];
y0 = 1;
[t,y] = ode15i(@(t,y) -10*t,tspan,y0);
plot(t,y);
title('Analysis of ODE using @ode15i');
xlabel('time(s)'),ylabel('y(t)');
```

Output:



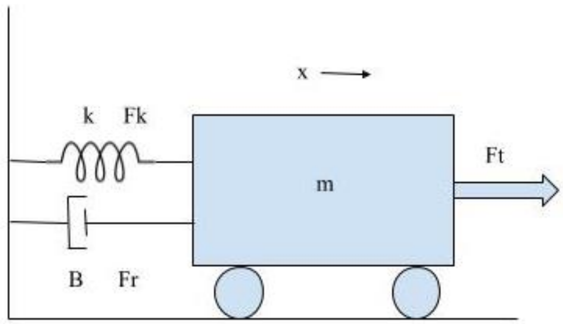
Conclusions: Various MATLAB techniques to solve an ODE were studied and analyzed. ode45, ode23 and ode113 solve nonstiff differential equations. ode15s, ode23s, ode23t and ode23tb solve nonstiff differential equations. An ordinary differential equation problem is stiff if the solution being sought is varying slowly, but there are nearby solutions that vary rapidly, so the numerical method must take small steps to obtain satisfactory results. Nonstiff methods can solve stiff problems; they just take a long time to do it. ode45 is medium ordered method. ode23 and ode23s are low ordered methods while ode15s ande113 are variable ordered methods. ode23t and ode23tb are based on the trapezoidal rule.

DATE : 31-07-2019

Experiment 1(B)

Title of the experiment:

1. To solve the ode equation: $y(x) = x^2 + x + 4$.
2. To model the physical system using 2nd order ODE.



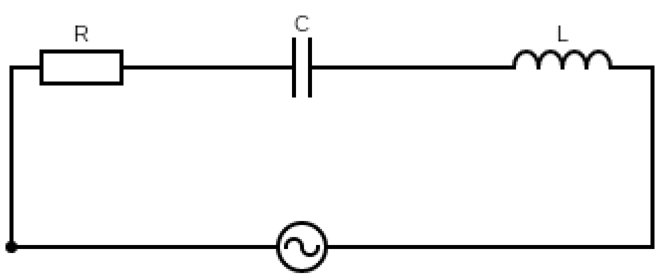
a. Assume: $k = 0, m = 750\text{ kg}, B = 30\text{ Ns/m}, Fr = 300N$ form a 1st order ODE.

b. Using Mupad solve the 1st order ODE with initial condition $v(0) = 0$.

c. Plot ‘v’ vs ‘t’ in MATLAB for t: 1 to 200s

d. Plot using @ode45 Function.

3.



$V_m \cos(wt)$

$V_m = 10V, w = 50\text{ rad/sec}, L = 2H, C = 1F, R = 1\Omega, i(0) = 0, i'(0) = 0.$

a. To model the given RLC circuit in series model in Matlab to a 2nd order ODE.

b. Solve using Mupad for given values.

c. Plot ‘i’ vs ‘t’ for t: 0 to 200 sec.

Software Used: Matlab 2018

Theory:

1. **Mupad**: Mupad is a GUI driven MATLAB package that helps you do algebra, calculus, as well as to graph and visualize functions. As you know, MATLAB is good for writing simple programs and working with numbers, but is cumbersome for doing symbolic calculations. Mupad is useful here.

2. **ode45:** Syntax » $[t, y] = \text{ode45}(\text{odefun}, \text{tspan}, y_0)$
   Where tspan = [t0 tf], integrates the system of differential equations y′=f (t,y) from t_0 to t_f with initial conditions y0. Each row in the solution array y corresponds to a value returned in column vector t.
   $[t, y, te, ye, ie] = \text{ode45}(\text{odefun}, \text{tspan}, y_0, \text{options})$ additionally finds where functions of (t, y), called event functions, are zero. In the output, t_e is the time of the event, y_e is the solution at the time of the event, and i_e is the index of the triggered event.

Code 1:

```
eq:= ode({y(x) = x^2 + x + 4},y(x))
solve(eq)
```

Output:

$\{x^2 + x + 4\}$

Code 2:

a) Balancing forces on the free body diagram we get:

$$F - kx - B\frac{dx}{dt} = m\frac{d^2x}{dt^2}$$

and Substituting the given k, B, m and $v(t) = \frac{dx}{dt}$, we get:

$$25v'(t) + v(t) = 10.$$

b) **Using Mupad:**

```
eq:= ode({25*v'(t) + v(t) = 10,v(0)=0},v(t))
solve(eq)
```

c) **Using Editor:**

```
clc;
close all;
clear all;
syms t;
v(t) = 10 - 10*exp(-t/25);
t = 0:200;
```

```
plot(t,v(t));
title('Analysis of Physical System');
xlabel('time(s)'),ylabel('v(t)');
```

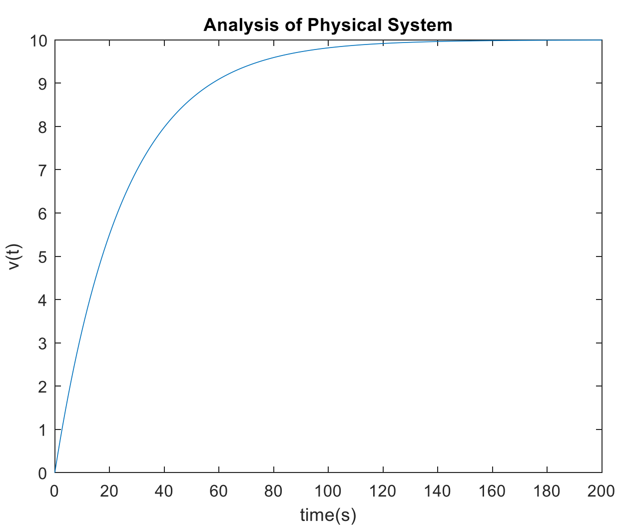
d) **Using @ode45:**

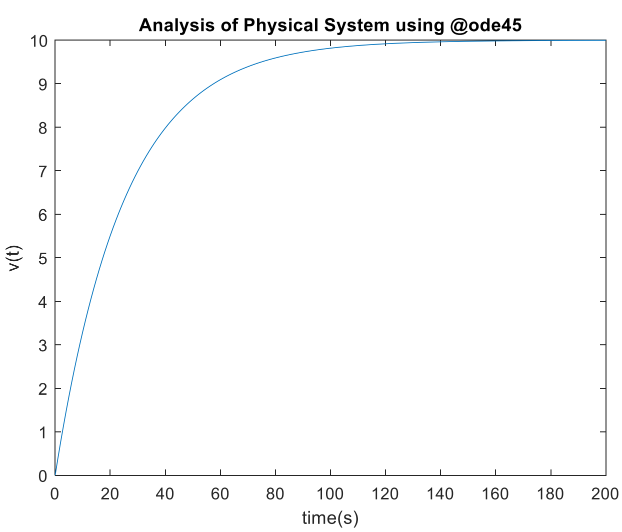
```
function dv = asdf(t,v)
dv = (10-v)/25;
end
clc;
close all;
clear all;
[t,v] = ode45(@asdf,0:200,0);
plot(t,v);
title('Analysis of Physical System using @ode45');
xlabel('time(s)'),ylabel('v(t)');
```

Output:

$\mathrm{ode}\big(25v'(x) + v(x) - 10, v(t)\big)$

$\{10 - 10e^{\frac{-x}{15}}\}$





Conclusions: 2nd order derivative equation were derived for given Physical System, 1st ODE solved using mupad editor and graphs were plot for the same using ode45 function and matlab text editor.

Code 3:

a) Adding the voltages to $V_m \sin(wt)$ with same current i we get:

$$V_m \cos(wt) \ = Ri(t) + L\frac{di(t)}{dt} + C\int i(t)dt$$

or

$$V_m\, w\sin(wt) + \ R\frac{di(t)}{dt} + L\frac{d^2 i(t)}{dt^2} + Ci(t) = 0$$

Using initial conditions we get: $500\cos(50t) = i'(t) + 2i''(t) + i(t)$.

b) **<u>Using Mupad:</u>**

```
eq:= ode({500*sin(50*t) + i'(t) + 2*i''(t) + i(t) =
0,i(0)=0,i'(0) = 0},i(t))
solve(eq)
```

c) **<u>Using Editor:</u>**

```
clc
clear all
close all
syms t;
i(t) = cos((7^(1/2)*t)/4)*((12500*cos(50*t -
(7^(1/2)*t)/4))/24992501 … - (499925000*7^(1/2)*exp(-
t/4)*sin((7^(1/2)*t)/4))/174947507;
t = 0:1:200;
plot(t,i(t));
title('Analysis of Electrical System');
xlabel('time(s)'),ylabel('i(t)');
```

Output:

$$\mathrm{ode}(\{i'(0) = 0,\ i(0) = 0,\ 2\,i''(t) + i'(t) + i(t) + 500\sin(50\,t)\},\ i(t))$$

$$\Big\{\cos(\sigma_5)\left(\frac{12500\,\sigma_3}{24992501} + \frac{12500\,\sigma_4}{24992501} + \frac{1249750\,\sigma_1}{24992501} + \frac{1249750\,\sigma_2}{24992501} + \frac{1250250\sqrt{7}\,\sigma_3}{174947507}\right.$$

$$\left. - \frac{1250250\sqrt{7}\,\sigma_4}{174947507} + \frac{249962500\sqrt{7}\,\sigma_1}{174947507} - \frac{249962500\sqrt{7}\,\sigma_2}{174947507}\right) - \frac{25000\,e^{-\frac{t}{4}}\cos(\sigma_5)}{24992501}$$

$$+\sin(\sigma_5)\left(\frac{1249750\,\sigma_3}{24992501} - \frac{1249750\,\sigma_4}{24992501} - \frac{12500\,\sigma_1}{24992501} + \frac{12500\,\sigma_2}{24992501} + \frac{249962500\sqrt{7}\,\sigma_3}{174947507}\right.$$

$$\left. + \frac{249962500\sqrt{7}\,\sigma_4}{174947507} - \frac{1250250\sqrt{7}\,\sigma_1}{174947507} - \frac{1250250\sqrt{7}\,\sigma_2}{174947507}\right) - \frac{499925000\sqrt{7}\,e^{-\frac{t}{4}}\sin(\sigma_5)}{174947507}\Big\}$$

where

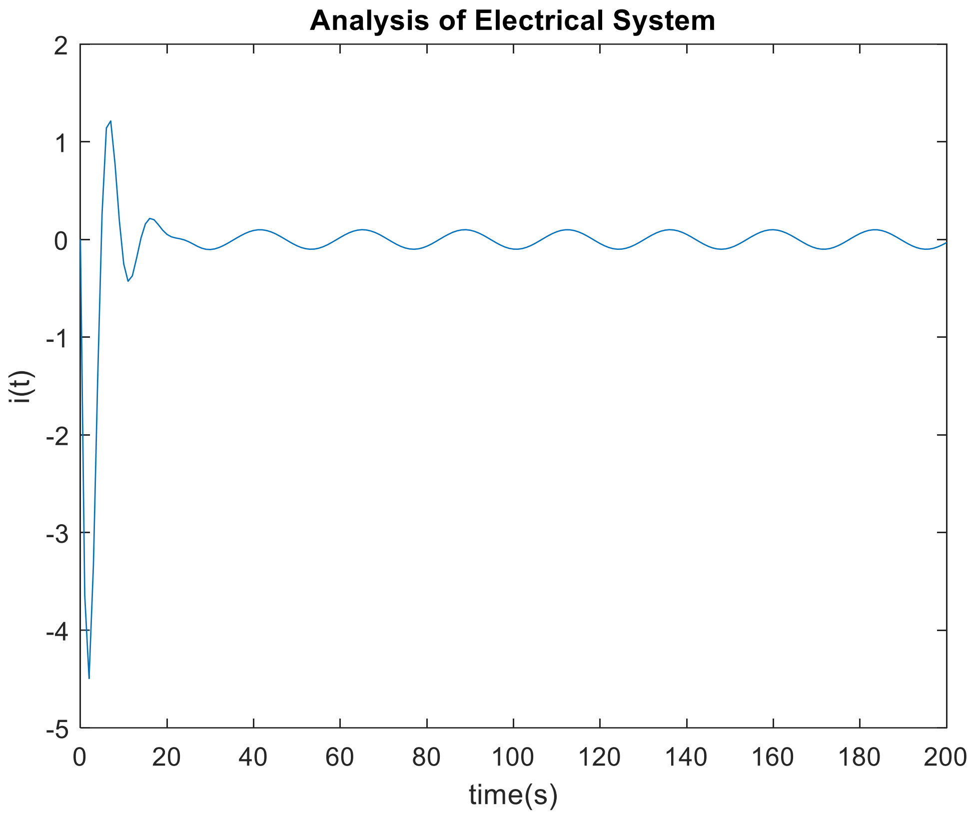
$\sigma_1 = \sin(50\,t - \sigma_5)$

$\sigma_2 = \sin(50\,t + \sigma_5)$

$\sigma_3 = \cos(50\,t - \sigma_5)$

$\sigma_4 = \cos(50\,t + \sigma_5)$

$\sigma_5 = \frac{\sqrt{7}\,t}{4}$



Conclusions: 2^{nd} order derivative equation were derived for given System, solved using mupad editor and graph was plot for the same. The characteristics of simple RLC circuit were analyze using Mupad Editor and Matlab.

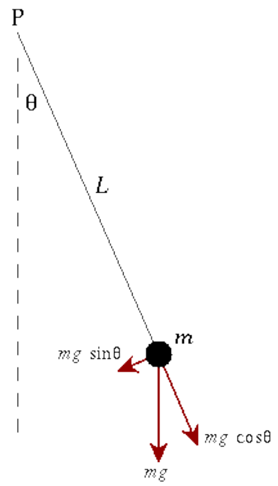
Control System Lab (EEP310)

DATE : 31-07-2019

Assignment 1

Aim:

1. Assume $\Theta = \pi/6, t = 0$ to 2π and $l = 2m$.



a. To model the given physical system to a 2nd order ODE.
b. Solve the 2nd Order Differential Equation.
c. Plot 'i' vs 't' for t: 0-200

Code:

a) Balancing forces on the free body diagram we get:

$$ML^2\frac{d^2\theta}{dt^2} + mg\sin(\theta)L = 0$$

and assuming the values of m, g be 1kg, $9.8m/s^2$ respectively and $\theta'(0) = 0$ and $\theta(0) = \pi/6$, we get:

$$\theta''(t) + 10\sin(\theta(t)) = 0.$$

Also assuming θ tends to small value.

b) **Using Mupad:**

```
eq:= ode({q''(t) = -4.8*q(t),q(0) = PI/6,q'(0) = 0},q(t))
x(t) :=solve(eq)
```

c) **Using Editor:**

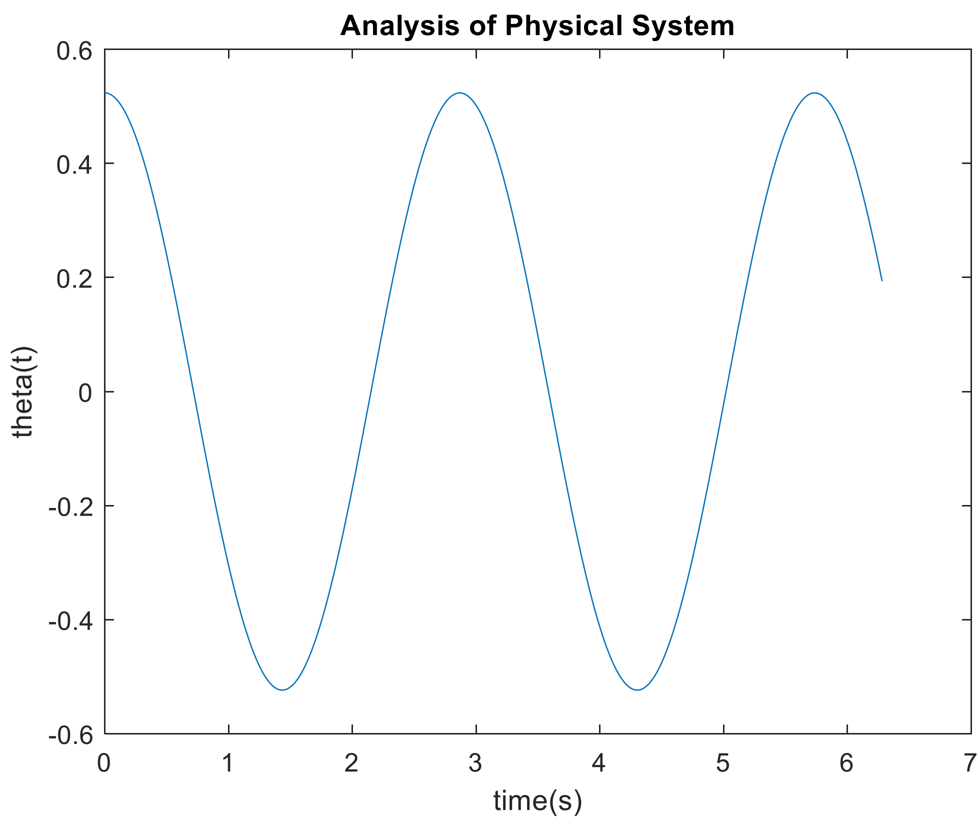
```
Clc;
clear all;
close all;
syms t;
theta(t) = 0.5235987756*cos(2.19089023*t);
t = 0:0.01:2*pi;
plot(t,theta(t));
```

```
title('Analysis of Physical System');
xlabel('time(s)'),ylabel('theta(t)');
```

Output:

$$\text{ode}\left(\left\{q'(0) = 0,\ q''(t) + 4.8\, q(t),\ q(0) = \frac{\pi}{6}\right\},\ q(t)\right)$$

$$\{0.5235987756 \cos(2.19089023\, t)\}$$



Conclusions: 2nd order derivative equation were derived for given free pendulum System, solved using mupad editor and graph was plot for the same. It was observed that the pendulum slow down its oscillation over time due damping force acting on it. Hence, after sometime the pendulum starts to come in its steady state.