EEL 3040

Control System

Lab - 1

By - B23EE1035
Anand Kharane

Exp-1 (Introduction to matlab)

Content

1) Objective :-

A) Aim :-

To learn the fundamentals of MATLAB and Simulink and apply them to solve Ordinary Differential Equations (ODEs).

B) Software :- Matlab-Simulink.

C) Discussion :-

This experiment demonstrated the use of MATLAB and Simulink for solving mathematical problems and modeling dynamic systems. Plotting the quadratic polynomial highlighted MATLAB's ability to visualize equations and study system behavior. The ODE was solved numerically in MATLAB using the ODE45 solver and graphically in Simulink through block diagrams consisting of integrators, gains, and summations. Both approaches produced consistent results, showing how MATLAB is effective for computation while Simulink provides an intuitive simulation environment. Together, they form powerful tools for analysis and design in engineering applications.

Exercises :-

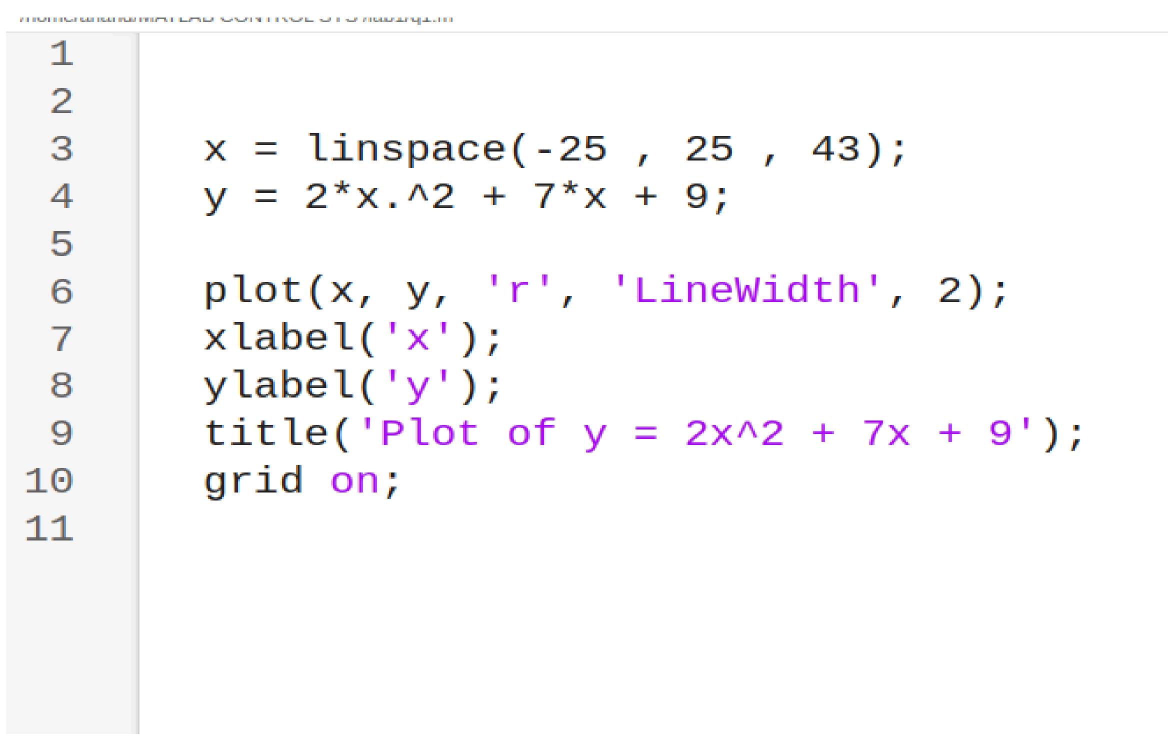
Q .6) Plot the graph of a polynomial equation in MATLAB

For the following equation, plot the values of y in MATLAB for a range of values in x,

y = 2x^2+7x+9

Include the code, plot and the corresponding description.

CODE :-

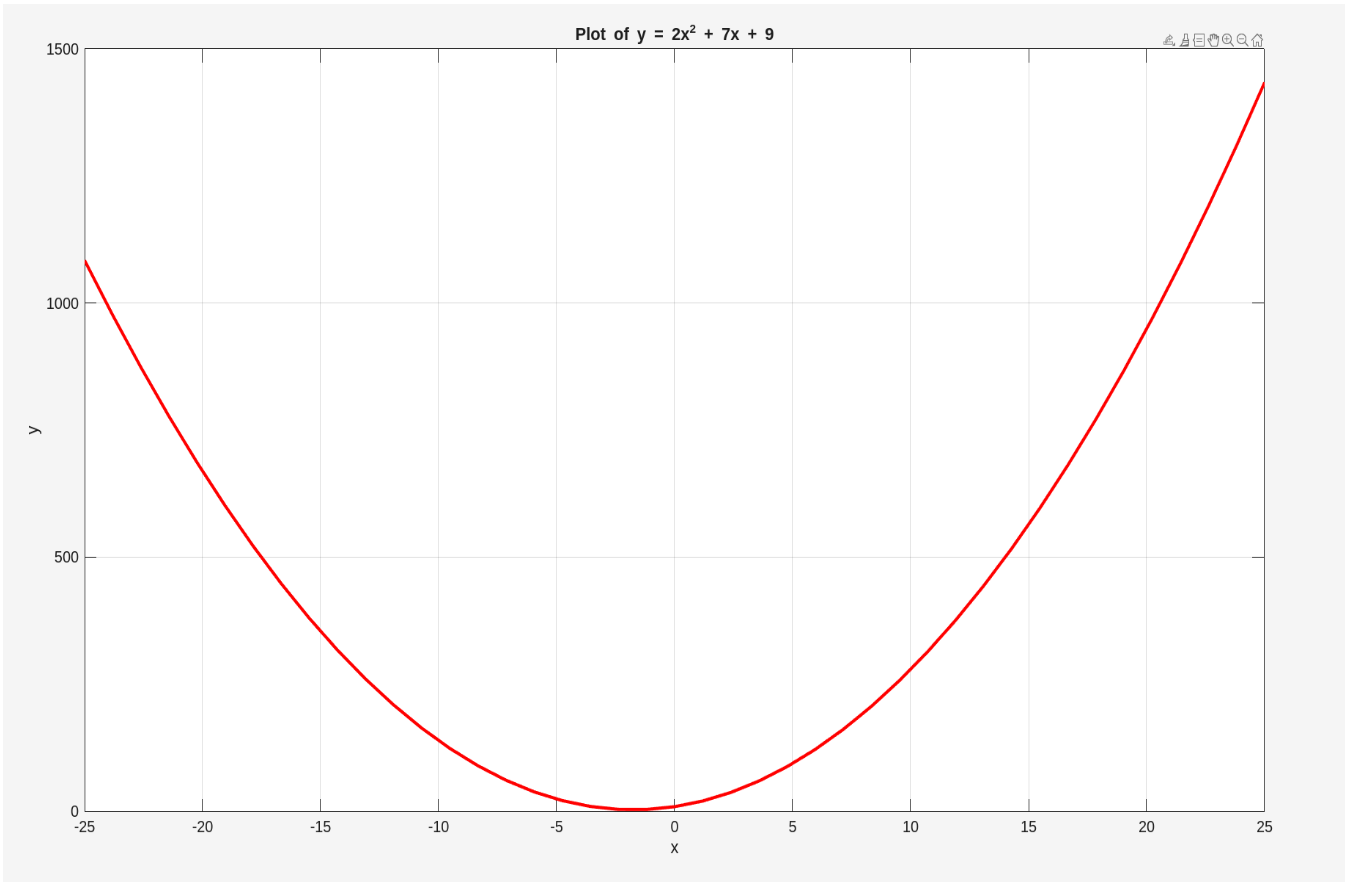


```
x = linspace(-25 , 25 , 43);
y = 2*x.^2 + 7*x + 9;

plot(x, y, 'r', 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Plot of y = 2x^2 + 7x + 9');
grid on;
```

Explanation :-

1) `x = linspace(-25 , 25 , 43);` : - Generates a vector **x** with 43 evenly spaced points between −25 and 25.
2) `y = 2*x.^2 + 7*x + 9;` :- Calculates the values of the quadratic polynomial y = 2x^2 + 7x + 9 for each value in the **x** vector (`.^2` ensures element-wise squaring).
3) `plot(x, y, 'r', 'LineWidth', 2);` : Plots the quadratic polynomial with **x** on the x-axis and **y** on the y-axis, using a **red curve** (`'r'`) with a line thickness of **2.**
4) `xlabel('x');` : Labels the x-axis as **“x”**.
5) `ylabel('y');` : Labels the y-axis as **“y”**.
6) `title('Plot of y = 2x^2 + 7x + 9');` :- Adds the title “*Plot of y = $2x^2 + 7x + 9$* to the plot.
7) `grid on;` : Adds a grid to the plot for better readability and clarity of data points.

Output :-



Q . 7) Solve an ODE in MATLAB and Simulink, and plot the results obtained .

a)

$$\frac{d^2}{dx^2}(y) = Cos(2x) - y$$

Plot the results obtained and attach the code. Also describe the Toolboxes required for the code to work, if any.

Mathematical Description :-

Let $y_1 = y(1)$, $y_2 = y(2)$.

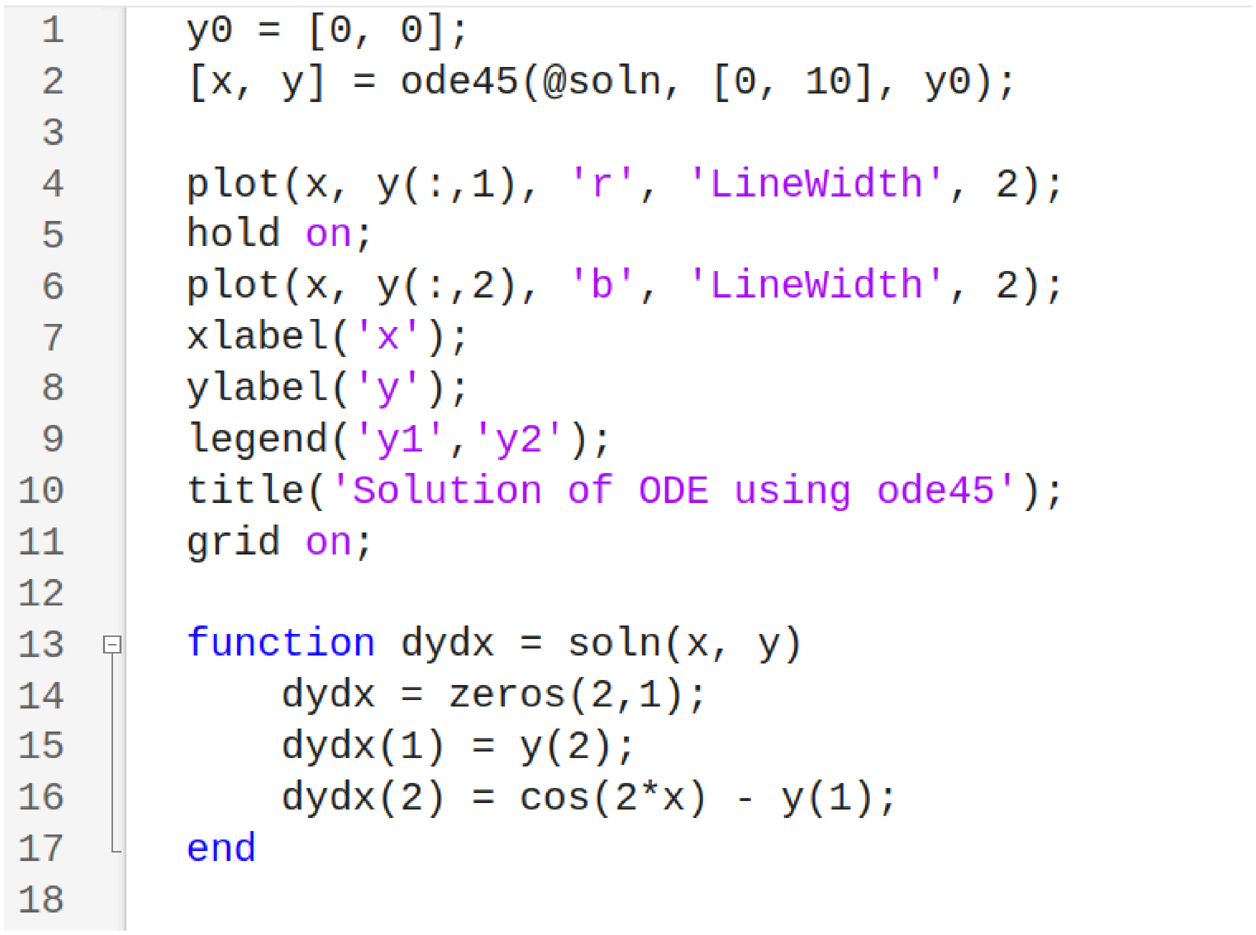
Then the equations are:

$$\frac{dy_1}{dx} = y_2$$

$$\frac{dy_2}{dx} = \cos(2x) - y_1$$

This is a **second-order ODE** written as a **system of two first-order ODEs**.

Code :-



```matlab
y0 = [0, 0];
[x, y] = ode45(@soln, [0, 10], y0);

plot(x, y(:,1), 'r', 'LineWidth', 2);
hold on;
plot(x, y(:,2), 'b', 'LineWidth', 2);
xlabel('x');
ylabel('y');
legend('y1','y2');
title('Solution of ODE using ode45');
grid on;

function dydx = soln(x, y)
    dydx = zeros(2,1);
    dydx(1) = y(2);
    dydx(2) = cos(2*x) - y(1);
end
```

6)

Explanation :-

1) The MATLAB code solves a system of two first-order ODEs using the ode45 solver.
2) ode45 is based on the Runge-Kutta method for numerical integration. The initial conditions of the system are set as [0, 0].
3) The solution is computed over the interval x = 0 to x = 10.
4) The system of equations is defined inside the function
5)

   Soln:-

   The first equation represents dy1/dx=y2

   The second equation represents dy2/dx=cos(2x)−y1

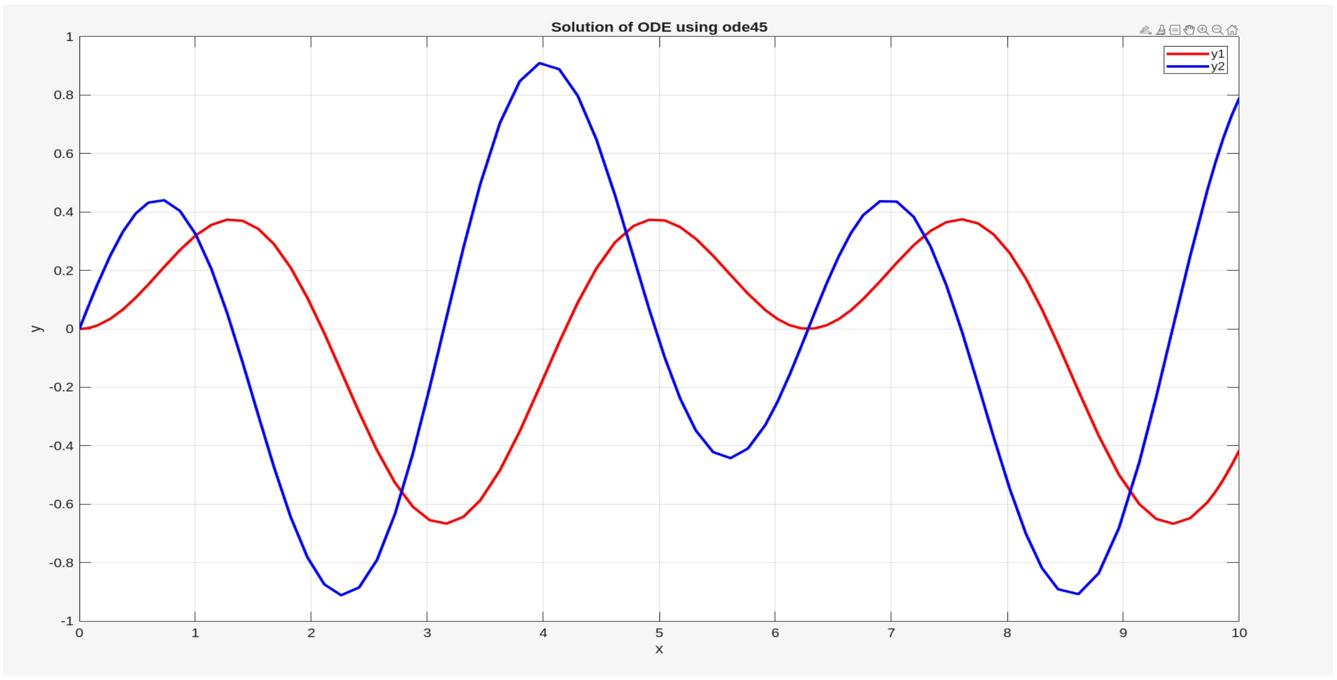
The ode45 function computes the solution and stores it in vectors:

- x → values of the independent variable.
- y → values of the dependent variables y1 and y2.

The results are plotted:

- y1 is shown as a red curve.
- y2 is shown as a blue curve.
- Axis labels, title, legend, and grid improve clarity.

Output :-



Q.7)

b. Model the following using Simulink

$$\frac{d^2}{dt^2}(y) = -2\,\frac{d(y)}{dt} - 5y + 1$$

Describe the blocks used and how each one of them contributes in obtaining the solution of the above equation. Use Scope to observe the output and attach the corresponding plot.

Mathematical Description :-

$$\frac{d^2y}{dx^2} = -2\frac{dy}{dx} - 5y + 1$$

with initial conditions

$$y(0) = 0, \quad \frac{dy}{dx}(0) = 0$$

Let

$$y_1 = y, \quad y_2 = \frac{dy}{dx}$$

Then:

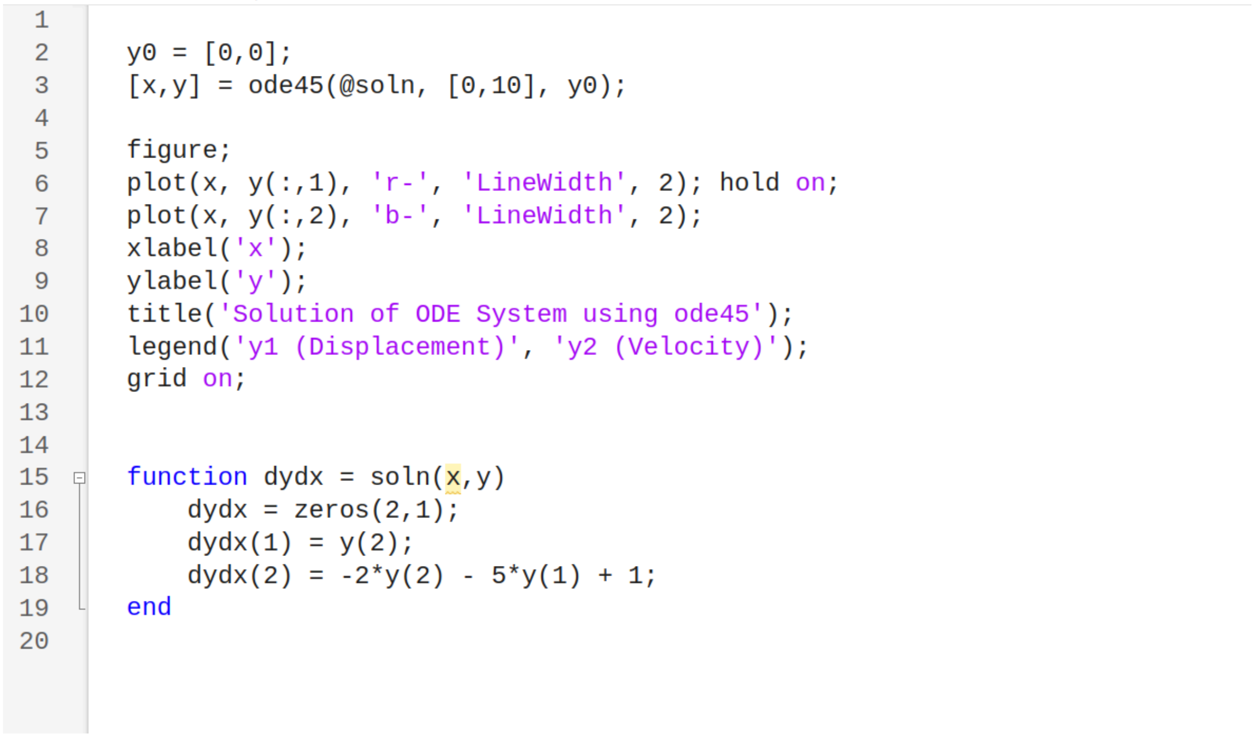
$$\frac{dy_1}{dx} = y_2$$

$$\frac{dy_2}{dx} = -2y_2 - 5y_1 + 1$$

So, the system is:

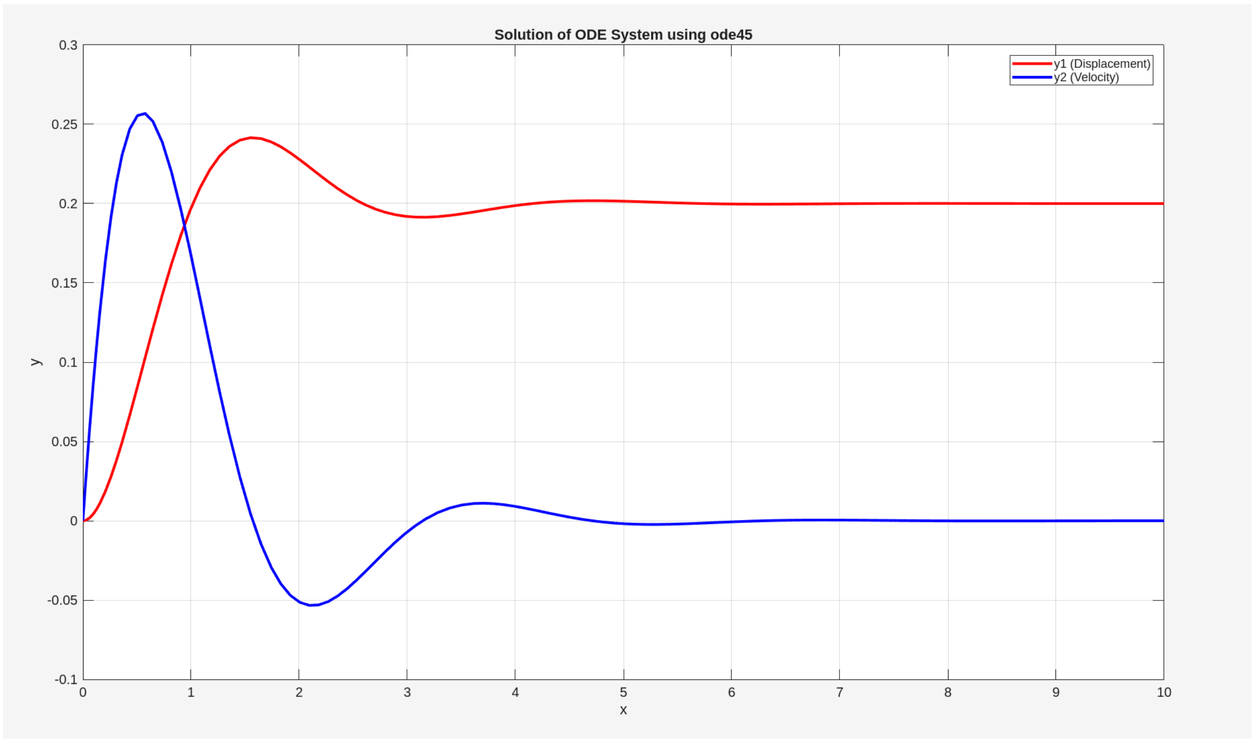
$$\begin{cases} y_1' = y_2 \\ y_2' = -2 \qquad - 5y_1 + 1 \end{cases}$$

Code :-

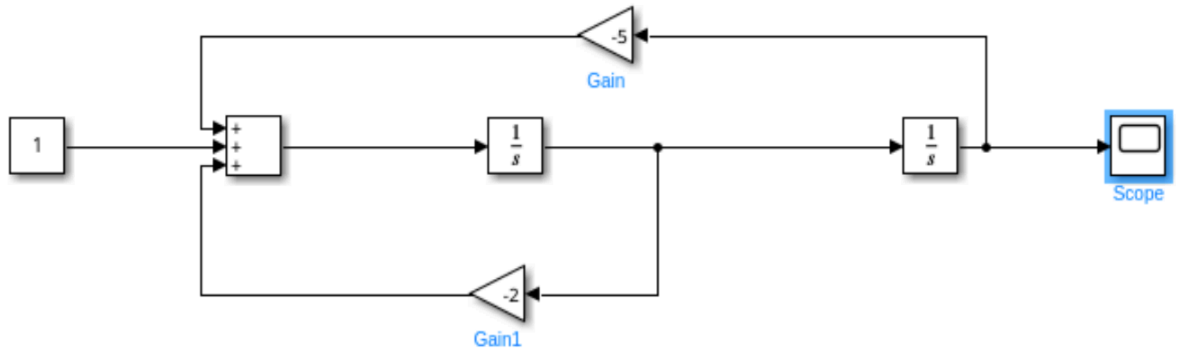


```

y0 = [0,0];
[x,y] = ode45(@soln, [0,10], y0);

figure;
plot(x, y(:,1), 'r-', 'LineWidth', 2); hold on;
plot(x, y(:,2), 'b-', 'LineWidth', 2);
xlabel('x');
ylabel('y');
title('Solution of ODE System using ode45');
legend('y1 (Displacement)', 'y2 (Velocity)');
grid on;


function dydx = soln(x,y)
    dydx = zeros(2,1);
    dydx(1) = y(2);
    dydx(2) = -2*y(2) - 5*y(1) + 1;
end

```

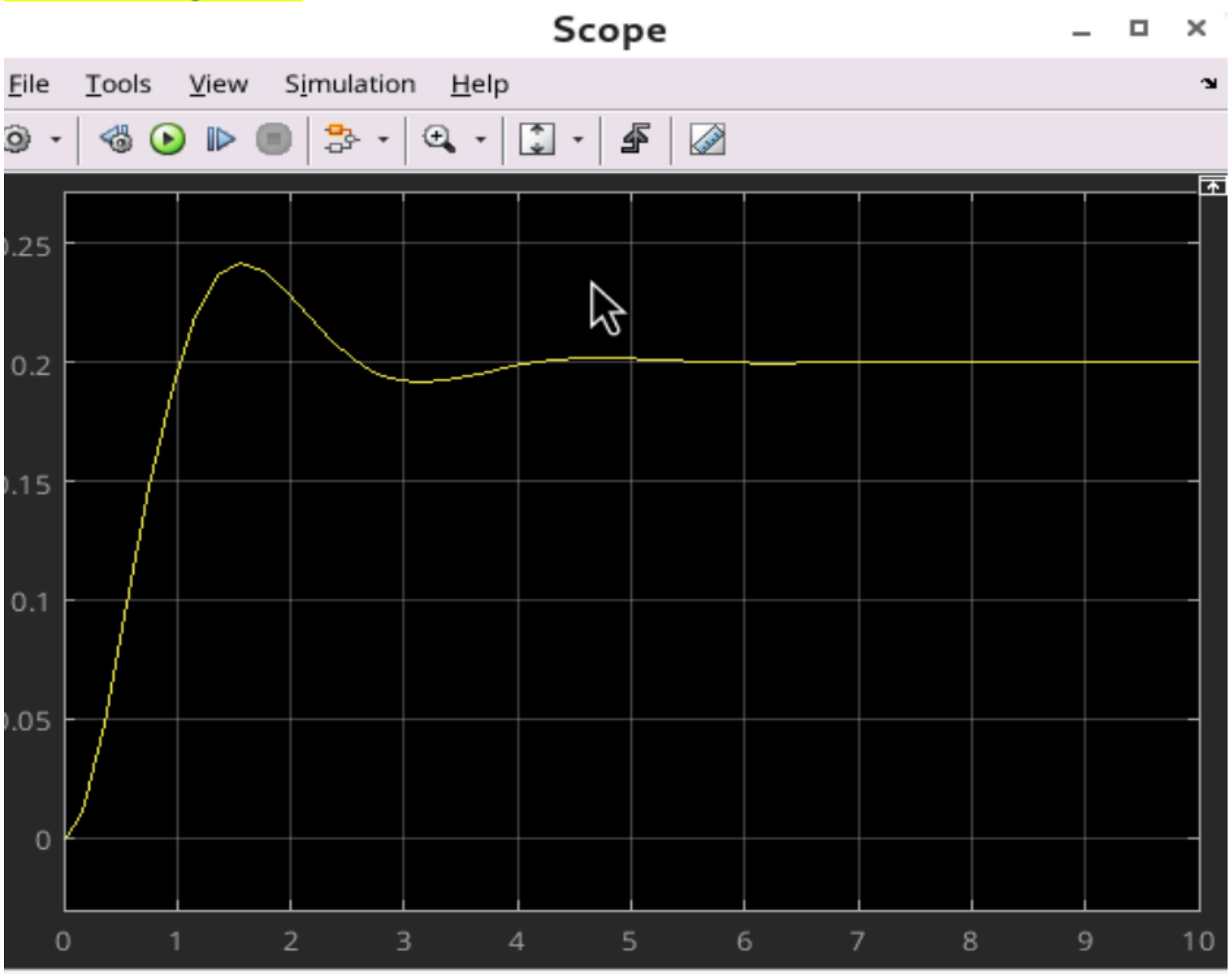
OutPut :-



Simulink Diagram :-



Scope Diagram :-



Exp-2 (Dc Motor Control)

Content

1) Objective : -

a) Aim :- Modeling of P-controller based d.c. motor in Simulink using state space model (using differential equations) and transfer functions based model.

b) Software :- Matlab-Simulink.

c) OverView :- This experiment will illustrate the modeling and design of separately excited d.c. motor using state space model and transfer function block diagram. Thereafter, a P-controller will be designed to control the speed of the motor.

2) Modeling and Design of D.C Motor :-

a) State Space Variables :-

$$\frac{d\omega}{dt} = \frac{1}{J}(K_t i - \omega) \qquad (1)$$

$$\frac{di}{dt} = \frac{1}{L}(V - Ri - K_e \omega) \qquad (2)$$

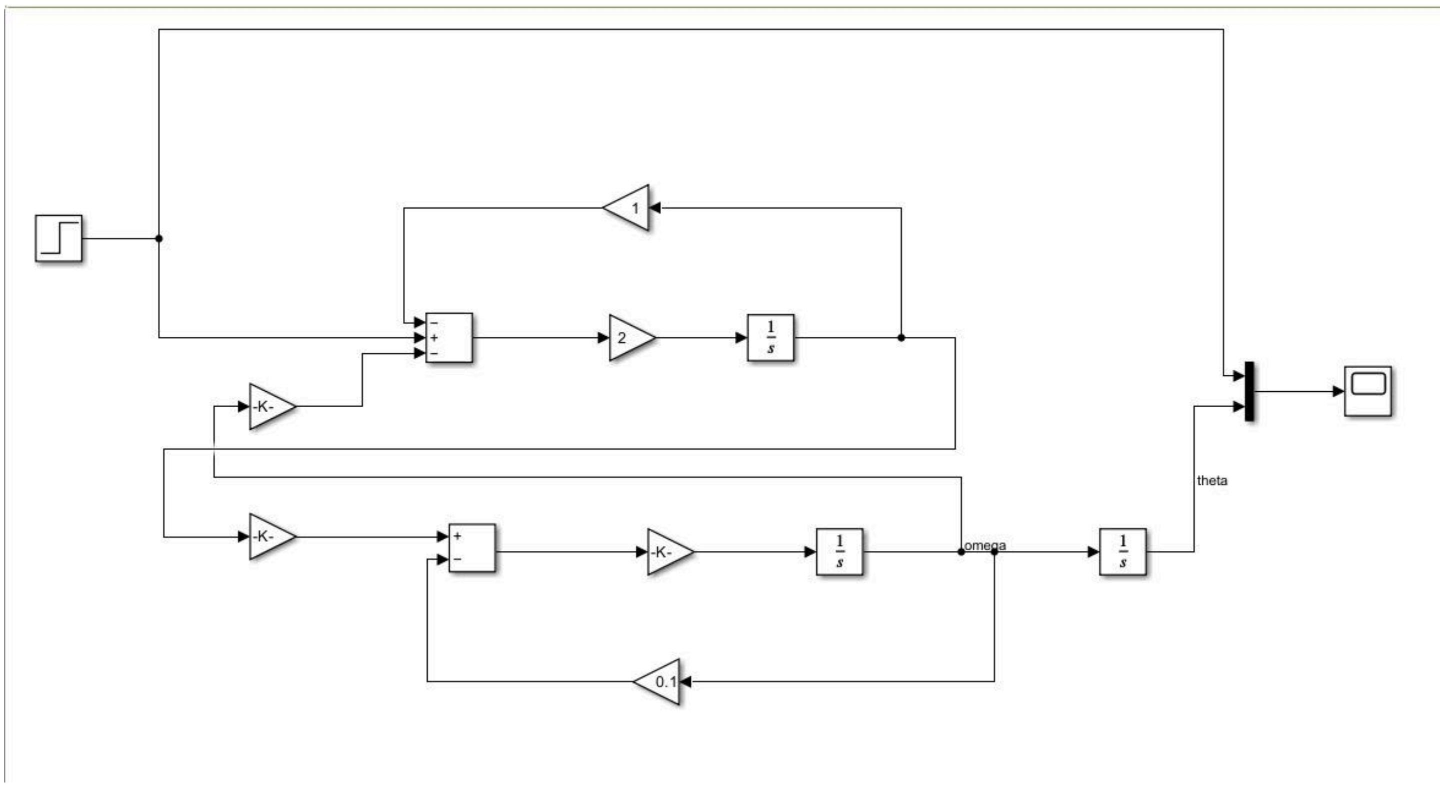
$$\frac{d\theta}{dt} = \omega \qquad (3)$$

b) Designing the Mathematical model of D.C Motor :-

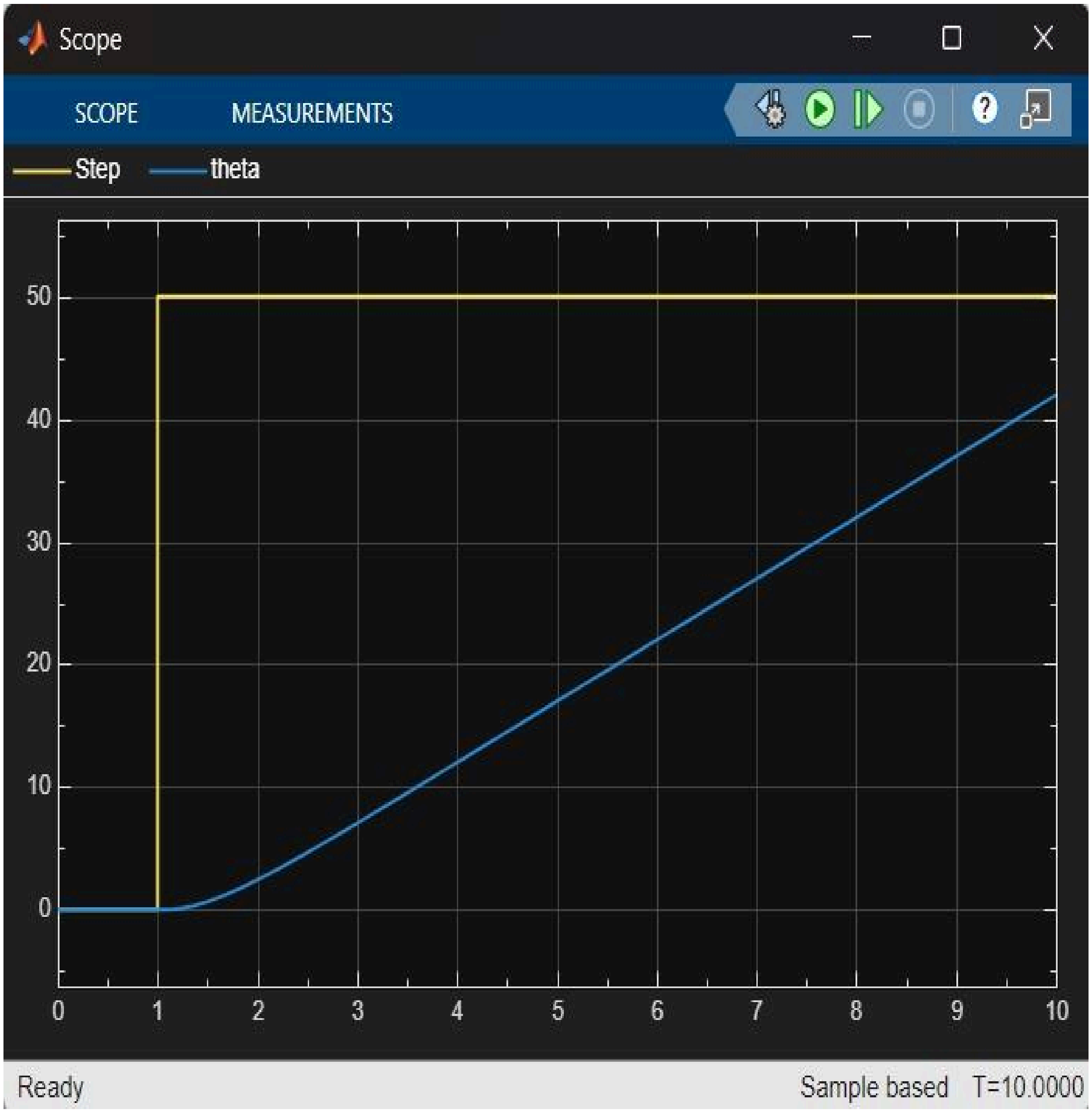
1) ω = speed
2) θ = rotation angle
3) i = armature current
4) R = armature resistor
5) L = inductance of armature coil

6) Ke = Back-emf proportion constant
7) J = Moment of inertia
8) b = Friction constant
9) Kt = Torque constant
10) V = Input voltage

In the diagram given below I have used 2 integrators and some adders for our purpose. In MUX we are observing the input voltage, Revolutions, and current via MUX for combined input. The Gain blocks used are constant values which we need to multiply to variables to get our desired results.



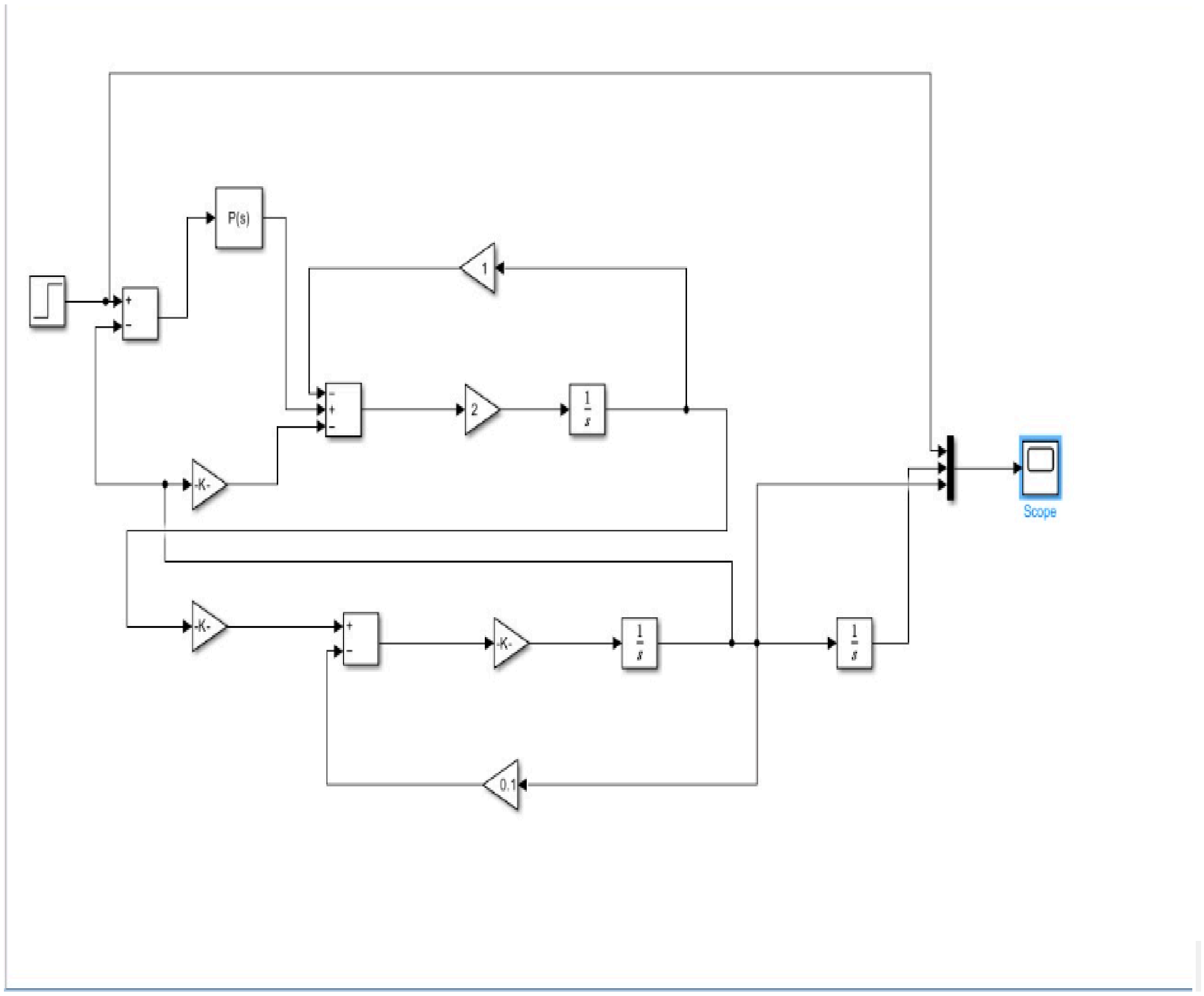
c) Scope Diagram :-



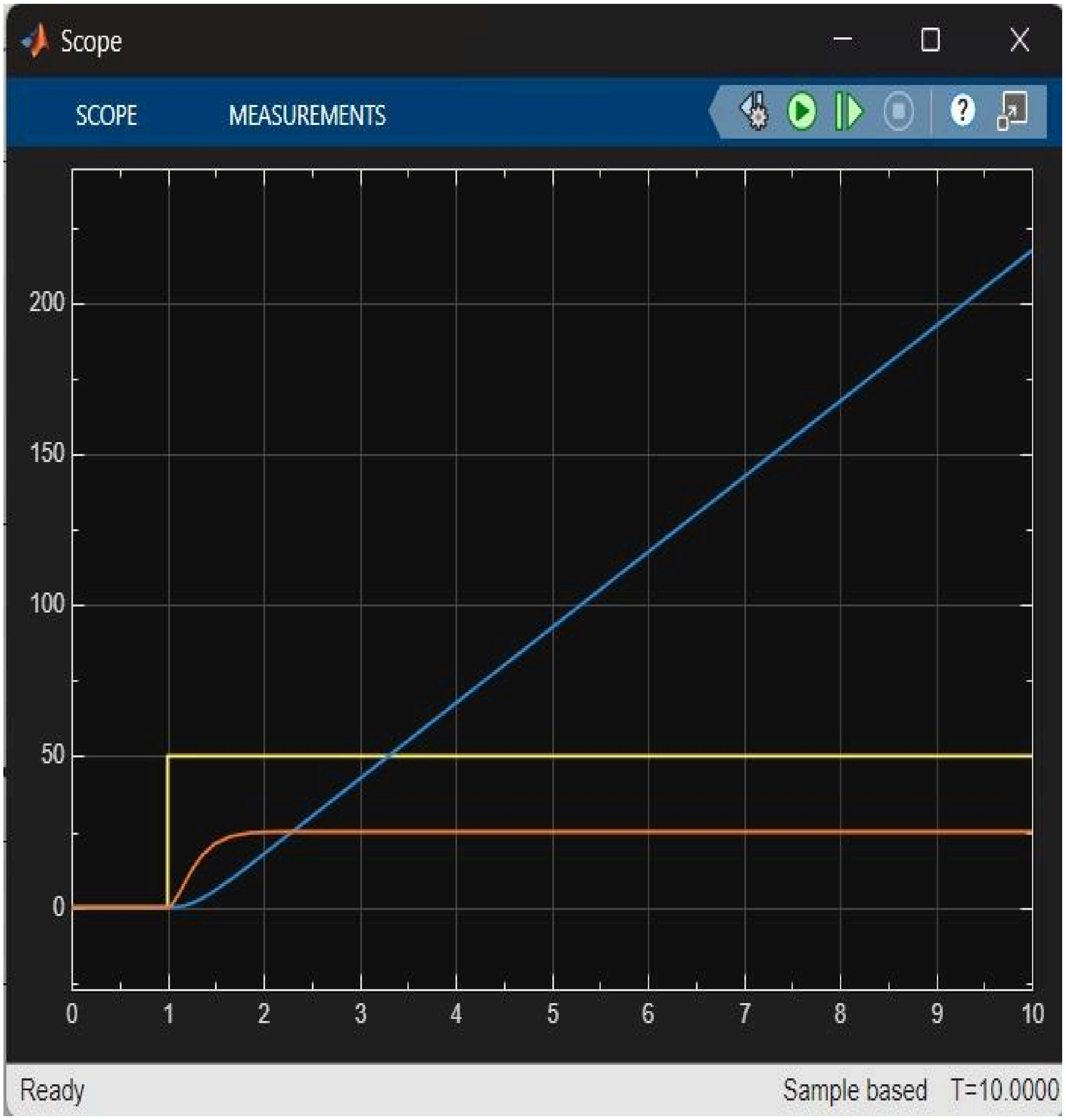
3. Feedback Control System :-

a) Mathematical Model :-

Now, our motive is to control the speed of the motor using a P-controller. For controlling speed (ω) of the motor, we will use a proportional controller. P controller regulates a constant gain say Kp in a feedback control system. This gain regulates the speed error, eω = ωdesired – ω. The speed error passes from the P-controller and becomes the input voltage, v. The final Simulink model of the DC motor with the proportional controller is :-



b) Scope Diagram :-



4. Linear Dynamic Model of DC-DC Boost [Post Lab Work]

(a) Mathematical Model

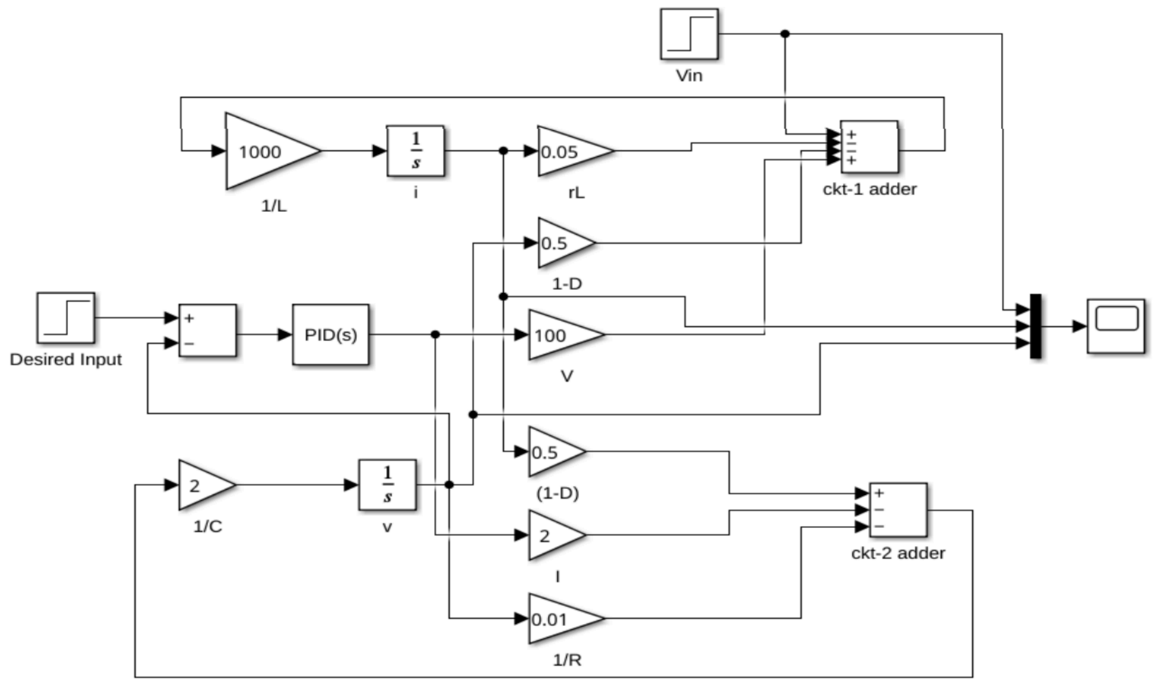
The mathematical model of the DC-DC boost converter can be described by the following differential equations:

$$\frac{di}{dt} = \frac{1}{L}\left(V_{in} - i \cdot r_L - (1-D)\cdot V + D \cdot V\right) \tag{4}$$

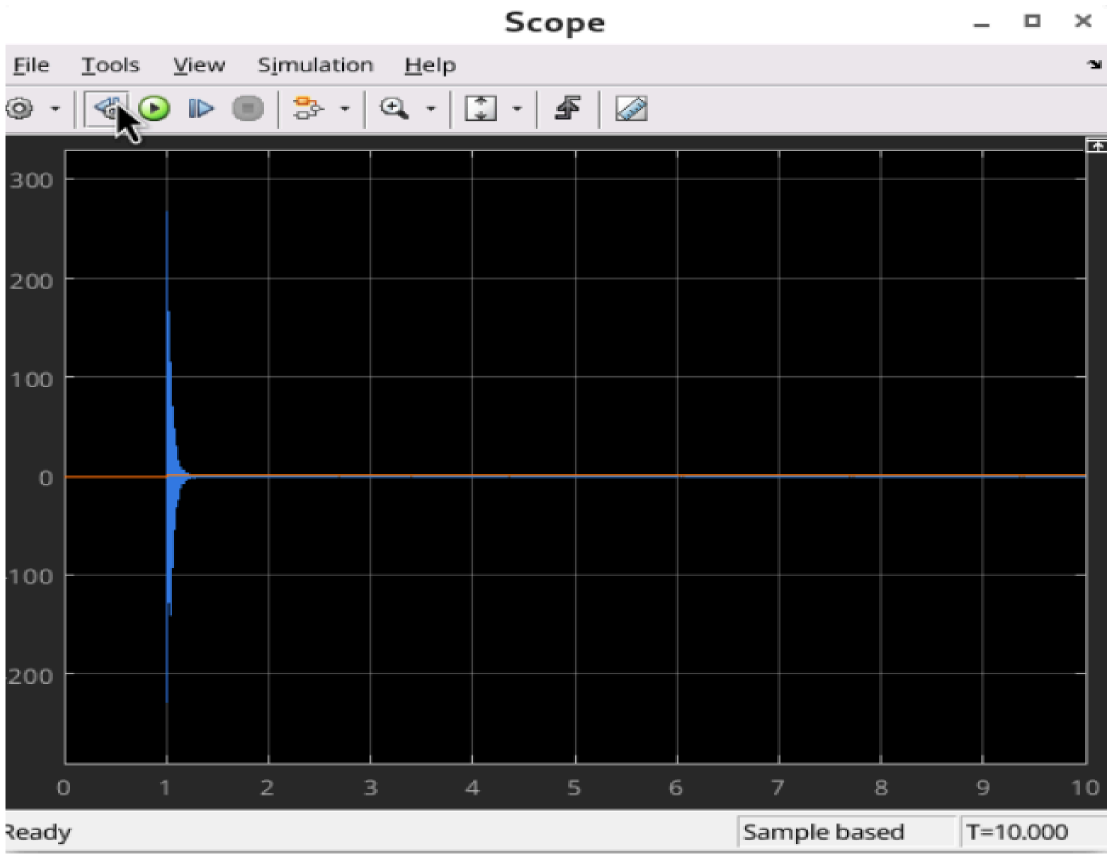
$$\frac{dv}{dt} = \frac{1}{C}\left((1-D)\cdot i - D \cdot i - \frac{V}{R}\right) \tag{5}$$

Parameters and Values

Parameter	Value
V_{in}	50 V
L	1 mH
C	500 mF
R	100 Ohms
r_L	0.05 Ohms
V	100 V
D (Duty Cycle)	0.5
I	$\frac{V}{R(1-D)} = 2$



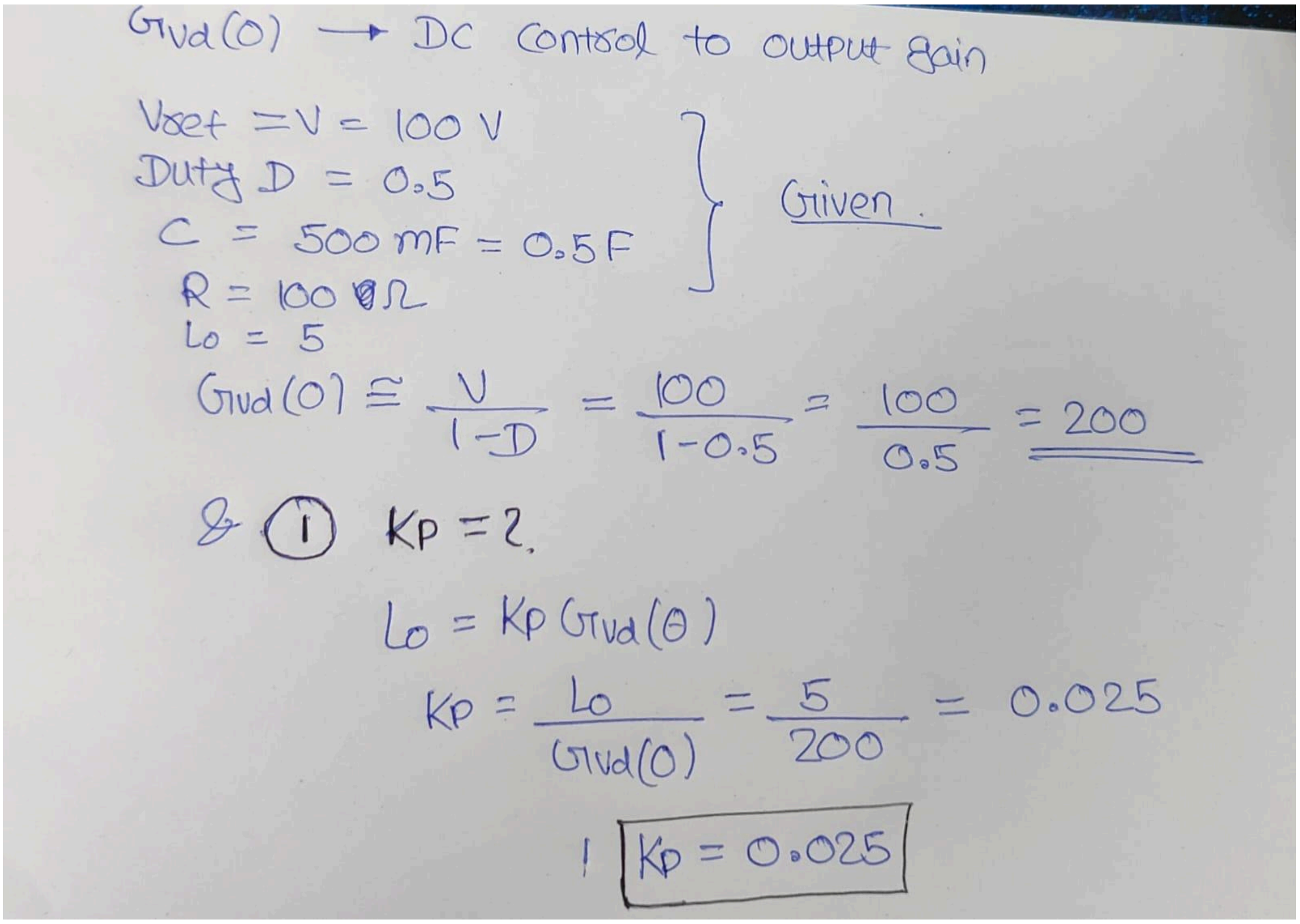
Scope Diagram :-



Observation Table :-

Parameters	Values
Value of the Kp	0.025
Settling time of voltage	200 s
Error between desired and actual voltage	16.7 v
Is there any speed overshoot or undershoot?	No

Calculation :-



$G_{vd}(0) \longrightarrow$ DC control to output gain

$V_{ref} = V = 100\ V$
Duty $D = 0.5$
$C = 500\ mF = 0.5 F$
$R = 100\ \Omega$
$L_o = 5$

} Given.

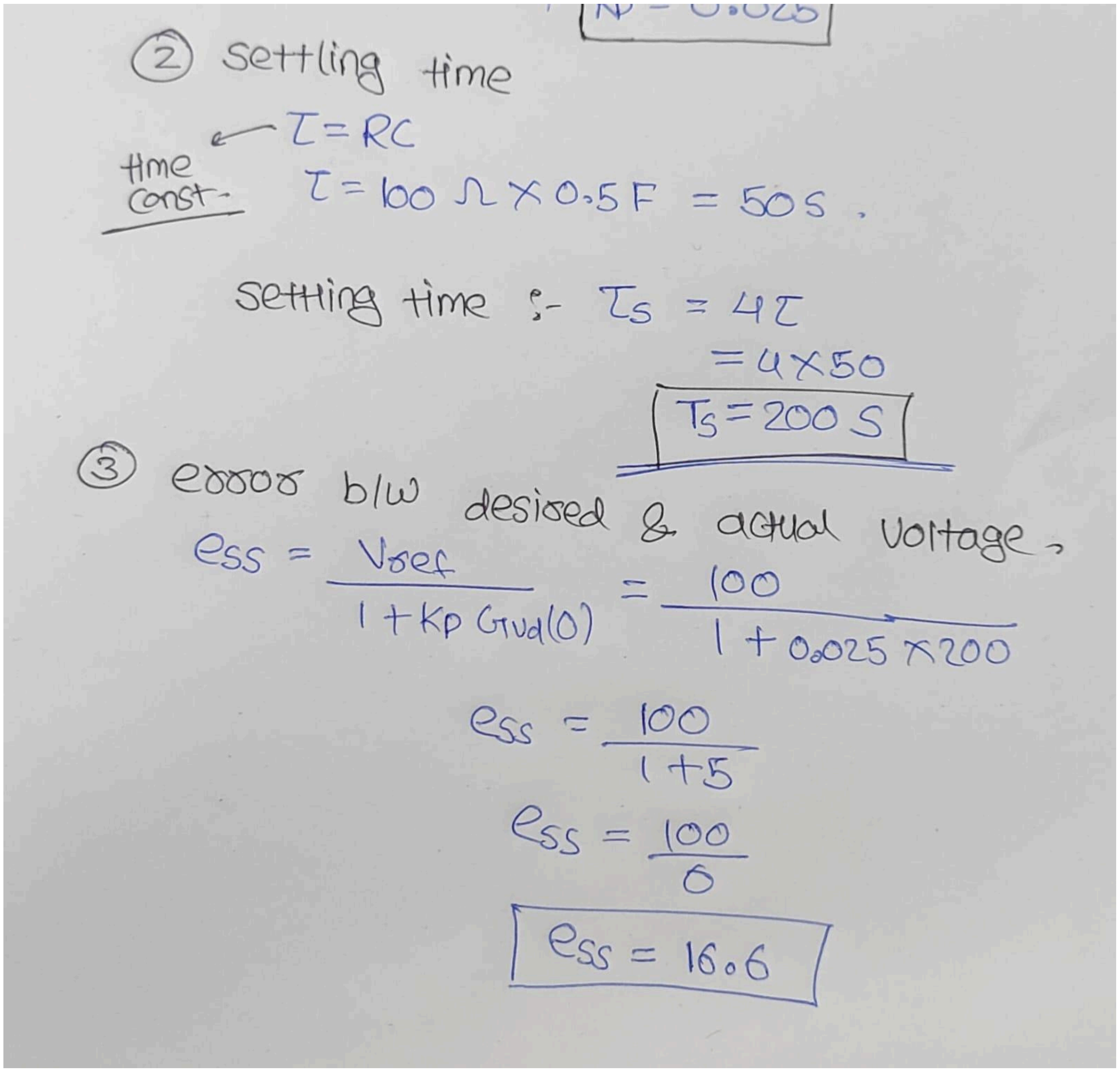
$$G_{vd}(0) \cong \frac{V}{1-D} = \frac{100}{1-0.5} = \frac{100}{0.5} = \underline{\underline{200}}$$

& ① $K_p = 2$,

$$L_o = K_p G_{vd}(0)$$

$$K_p = \frac{L_o}{G_{vd}(0)} = \frac{5}{200} = 0.025$$

$$\boxed{K_p = 0.025}$$



$\boxed{K_P = 0.025}$

② Settling time

$\leftarrow \tau = RC$

time const.

$\tau = 100\,\Omega \times 0.5\,F = 50\,s$.

Settling time :- $T_s = 4\tau$

$= 4 \times 50$

$\boxed{T_s = 200\,S}$

③ error b/w desired & actual voltage.

$$e_{ss} = \frac{V_{ref}}{1 + K_P G_{vd}(0)} = \frac{100}{1 + 0.025 \times 200}$$

$$e_{ss} = \frac{100}{1+5}$$

$$e_{ss} = \frac{100}{6}$$

$$\boxed{e_{ss} = 16.6}$$

Q4) Is there any speed overshoot or undershoot?

Ans :- No , there is no significant speed overshoot or undershoot.

With the chosen parameters , the closed-loop has low bandwidth and is strongly dominated by the slow output pole . A small controller gain and a very large time constant produce a slow, well-damped response, so the speed (and voltage) rises smoothly toward the steady value without appreciable overshoot or undershoot.