|  |  |
| --- | --- |
| **Experiment Number** |  |
| **Date of Experiment** |  |
| **Date of Submission** |  |
| **Name** |  |
| **Roll Number** |  |
| **Section** | ECS-01 |

**Aim of The Experiment :-**

Processing and analysis of continuous time signal using Laplace Transform in

MATLAB.

(Design Problem: Stability analysis of a boiler control system in a power plant using

transfer function as an industrial application)

.

**Equipment and Software Required:-**

The Equipment and Software required are as follows:

 MATLAB R2017 / 2019 / LabVIEW

**Code :-**

% Define time range for signal

t = 0:0.01:10; % Time vector from 0 to 10 with increment of 0.01

% Generate and plot a continuous time signal

f = 1; % Frequency of the signal

signal = sin(2\*pi\*f\*t); % Sinusoidal signal

figure;

plot(t, signal);

title('Continuous Time Signal');

xlabel('Time','FontSize',12,'Fontweight','bold','Color','r');

ylabel('Amplitude','FontSize',12,'Fontweight','bold','Color','b');

% Define Parameters

a = 0.5; % Boiler heat exchange rate

s = tf('s'); % Define the Laplace variable

% Boiler Control System

G = a/(s + a); % Transfer function

% Impulse Response

figure;

impulse(G);

title('Impulse Response of the Boiler Control System');

ylabel('Temperature Change','FontSize',12,'Fontweight','bold','Color','b');

xlabel('Time','FontSize',12,'Fontweight','bold','Color','r');

% Step Response

figure;

step(G);

title('Step Response of the Boiler Control System');

ylabel('Temperature Change','FontSize',12,'Fontweight','bold','Color','b');

xlabel('Time','FontSize',12,'Fontweight','bold','Color','r');

% Poles and Stability Analysispoles = pole(G); % Poles of the system

stable = all(real(poles) < 0); % Check stability

disp(['System is stable: ', num2str(stable)]);

% Performance Analysis

S = stepinfo(G); % Get step info

disp(['Rise Time: ', num2str(S.RiseTime)]);

disp(['Settling Time: ', num2str(S.SettlingTime)]);

disp(['Overshoot: ', num2str(S.Overshoot)]);

% Interpretation and Conclusion

if stable

disp('The system is stable, which is necessary for safe operation of the boiler system.')

if S.Overshoot <= 5

disp('The overshoot is within acceptable range. It indicates the system will not exceed

the desired temperature by a large margin, thus avoiding any potential harm to the

system or any risk of inefficiency.')

else

disp('The overshoot is high. It indicates that the system may exceed the desired

temperature by asignificant margin, leading to potential harm to the system or

inefficiencies.')

end

if S.RiseTime <= 1

disp('The rise time is short, indicating that the boiler can quickly respond to changes

in heatinput.')

else

disp('The rise time is long, indicating that the boiler may take some time to respond to

changes inheat input.')

end

if S.SettlingTime <= 2

disp('The settling time is short, which indicates that the boiler can reach a steady state

quicklyafter a change in control input.')

else

disp('The settling time is long, which indicates that the boiler may take some time to

reach asteady state after a change in control input.')

end

else

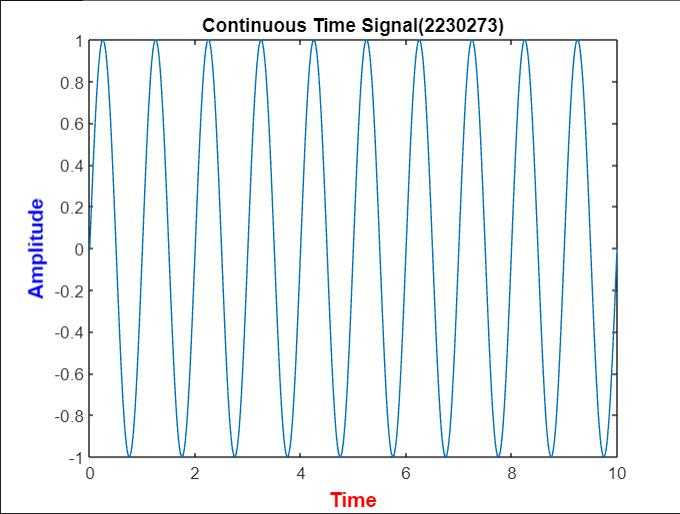
disp('The system is unstable, which is not safe for operating the boiler system. The

parameters of the system need to be adjusted to ensure stability.')

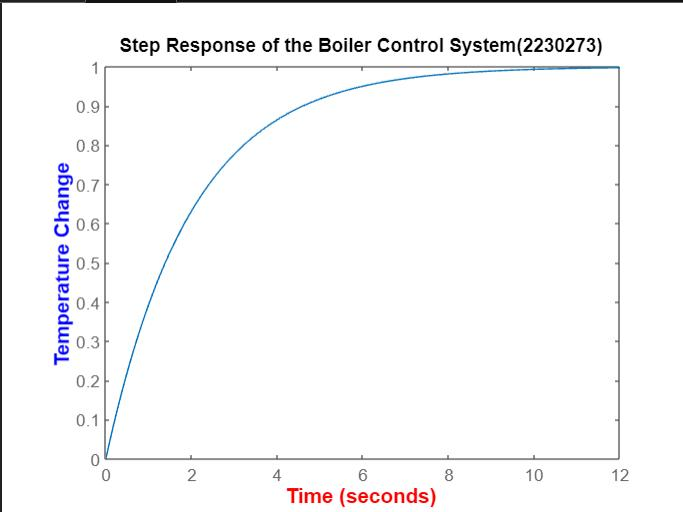
end

**Block diagram:**

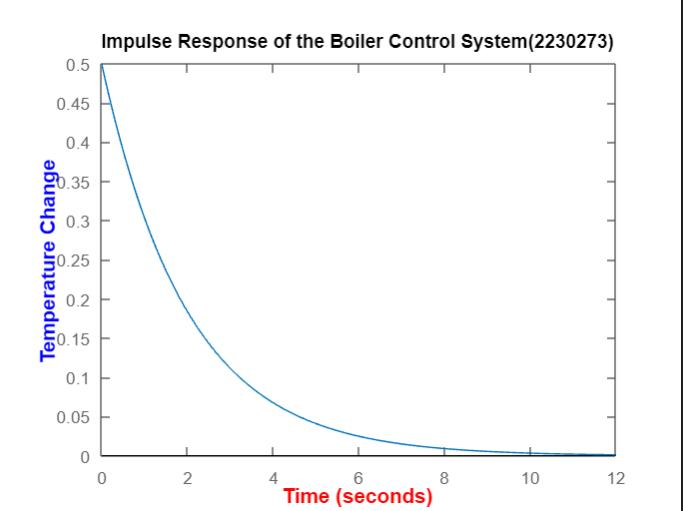
**Continuous time signal**



**step response of boiler control system**



**Impulse response of the boiler control system**

’

**Inference of the experiment:**

Using transfer functions in the design of a boiler control system for a power plant can aid in the analysis and prediction of the system's stability. Transfer functions are mathematical expressions that, while taking the dynamics of the system into account, relate an input to an output. You can ascertain the stability of the system by examining the transfer function's poles and zeros. When all of the poles have negative real portions, the system is stable. Ensuring the stability of the boiler control system is vital in order to avert unstable behavior and any mishaps within the power plant.

**Conslusion:**

Finally, by creating a variety of signals including unit steps, impulse signals, sine/cosine functions, exponential functions, and ramp functions, this experiment taught us the fundamentals of MATLAB. The Laplace transform can be used to process and analyze continuous-time signals with great power in MATLAB. You can efficiently convert your signals into the Laplace domain by following the previously described steps. This will allow you to perform a variety of analysis tasks, including transfer function determination, pole-zero analysis, and system response analysis. A deeper comprehension of signal behavior is facilitated by the ease with which results can be plotted.

Additionally, MATLAB's built-in functions for Laplace and inverse Laplace transforms streamline the process, making it accessible to both beginners and experienced users. Overall, MATLAB's capabilities in conjunction with the Laplace transform offer a comprehensive toolkit for engineers and researchers working with continuous-time signal analysis