ASSIGNMENT 1

EET305- Signals and Systems

B.Tech Fifth Semester

Department of Electrical and Electronics Engineering Government Engineering College, Barton Hill, Thiruvananthapuram

Q1: A trapezoidal signal is described as follows

$$x(t) = \begin{cases} t, & 0 \le t < 2 \\ 2, & 2 \le t \le 6 \\ 8 - t, & 6 < t \le 8 \\ 0, & \text{otherwise} \end{cases}$$

Simulate the following and generate the corresponding plot.

(a) $x(t)$

(b) $x(t - 3)$

(c) $x(2t)$

(d) $x(\frac{1}{2}t)$

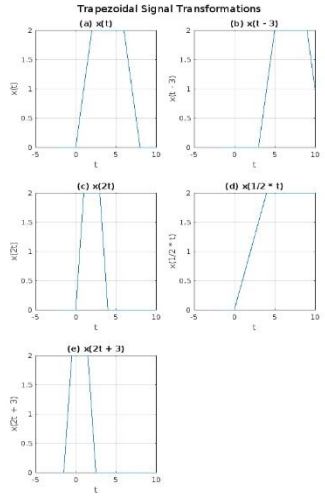
(e) $x(2t + 3)$

Ans:

CODE:

```
% Define time range for plotting
t = linspace(-2, 10, 1000);

% Define the trapezoidal signal function using element-wise operations
x = @(t) (t >= 0 & t < 2) .* t + ...
(t >= 2 & t <= 6) .* 2 + ...
(t > 6 & t <= 8) .* (8 - t);

% Calculate each transformation
x_t = x(t);              % (a) x(t)
x_t_minus_3 = x(t - 3);  % (b) x(t - 3)
x_2t = x(2 * t);         % (c) x(2t)
```

```
x_half_t = x(0.5 * t);     % (d) x(1/2 * t)
x_2t_plus_3 = x(2 * t + 3); % (e) x(2t + 3)

% Plotting each transformation
figure;

% (a) Plot x(t)
subplot(3, 2, 1);
plot(t, x_t);
title('(a) x(t)');
xlabel('t');
ylabel('x(t)');
grid on;

% (b) Plot x(t - 3)
subplot(3, 2, 2);
plot(t, x_t_minus_3);
title('(b) x(t - 3)');
xlabel('t');
ylabel('x(t - 3)');
grid on;

% (c) Plot x(2t)
subplot(3, 2, 3);
plot(t, x_2t);
title('(c) x(2t)');
xlabel('t');
ylabel('x(2t)');
grid on;

% (d) Plot x(1/2 * t)
subplot(3, 2, 4);
plot(t, x_half_t);
title('(d) x(1/2 * t)');
xlabel('t');
ylabel('x(1/2 * t)');
grid on;

% (e) Plot x(2t + 3)
subplot(3, 2, 5);
plot(t, x_2t_plus_3);
title('(e) x(2t + 3)');
xlabel('t');
ylabel('x(2t + 3)');
grid on;

% Adjust layout
sgtitle('Trapezoidal Signal Transformations');
```



Q2: Given the following input signal $x(t)$ and impulse response $h(t)$:

$$x(t) = \begin{cases} 1 & 0 \le t < 3 \\ 0 & \text{otherwise} \end{cases}$$

$$h(t) = e^{-t}, \qquad t \ge 0$$

(a) Plot the input signal $x(t)$ and the impulse response $h(t)$.

(b) Perform the convolution of $x(t)$ and $h(t)$ using MATLAB.

(c) Plot the output signal $y(t)$ obtained after convolution.

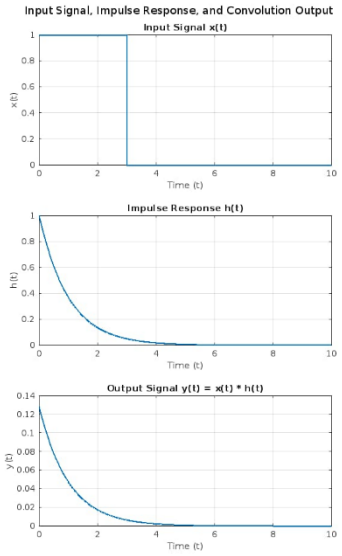
(d) Analyze the system's behavior based on the convolution result.

Ans:

CODE:

```
% Define the time vector for both signals
t = 0:0.01:10; % Adjust the range of time to cover the signals
% (a) Define the input signal x(t) and the impulse response h(t)
x_t = @(t) (t >= 0 & t < 3); % Rectangular pulse from 0 to 3
h_t = @(t) exp(-t) .* (t >= 0); % Exponential decay for t >= 0

% Plot the input signal x(t)
figure;
subplot(3,1,1);
plot(t, x_t(t), 'LineWidth', 1.5);
title('Input Signal x(t)');
xlabel('Time (t)');
ylabel('x(t)');
grid on;

% Plot the impulse response h(t)
subplot(3,1,2);
plot(t, h_t(t), 'LineWidth', 1.5);
title('Impulse Response h(t)');
xlabel('Time (t)');
ylabel('h(t)');
grid on;

% (b) Perform the convolution of x(t) and h(t)
dt = t(2) - t(1);
% Time step conv_result = conv(x_t(t), h_t(t), 'same') * dt; % Convolution

% (c) Plot the output signal y(t) after convolution
subplot(3,1,3);
plot(t, conv_result, 'LineWidth', 1.5);
title('Output Signal y(t) = x(t) * h(t)');
xlabel('Time (t)');
```

```
ylabel('y(t)');
grid on;

% Adjust layout
sgtitle('Input Signal, Impulse Response, and Convolution Output');
```



(d)System's Response to a Pulse: The convolution of a rectangular pulse x(t) and the exponential decay h(t) yields a response that starts at t=0, grows initially as the effect of the pulse accumulates, and then decays exponentially after t=3 (when x(t) becomes zero). This represents how the system "accumulates" the input and then gradually returns to zero as the impulse response decays.

Smoothing Effect: The convolution with an exponential decay implies a smoothing effect on the input pulse. This results in a smoother, delayed version of the input signal.

Memory of the System: The response shows that the system has a form of "memory," as the exponential decay causes the system to retain influence from past values of x(t).

Q3: A system with an impulse response $h(t) = e^{-2t}$ for $t \geq 0$ is excited by a square wave input:

$$x(t) = 1 \quad \text{for} \quad 0 \leq t < 5, \qquad x(t) = 0 \quad \text{otherwise}$$

(a) Define the input square wave in MATLAB.

(b) Perform the convolution of $x(t)$ with $h(t)$ to find the output $y(t)$.

(c) Plot the input, impulse response, and output signals.

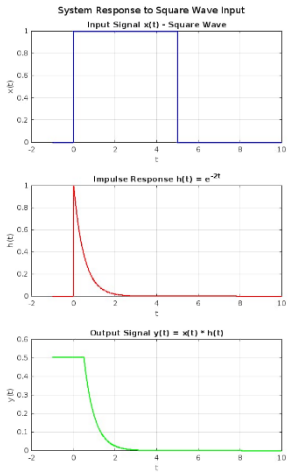
(d) Discuss the system's response to the square wave.

Ans:

CODE:

```
% Define the time range
t = -1:0.01:10;  % Extend beyond the square wave to see the full system response

% (a) Define the square wave input x(t)
x = @(t) (t >= 0 & t < 5);  % Square wave from t = 0 to t = 5

% Define the impulse response h(t)
h = @(t) (t >= 0) .* exp(-2 * t);  % h(t) = e^(-2t) for t >= 0

% Evaluate x(t) and h(t) over the time range
x_t = x(t);
h_t = h(t);

% (b) Perform the convolution of x(t) with h(t)
y_t = conv(x_t, h_t, 'same') * 0.01;  % Scaling by time step (0.01) for accuracy

% Adjust the time axis for y(t) to match t
t_conv = t;

% (c) Plot the input signal x(t), impulse response h(t), and output signal y(t)
figure;

% Plot x(t)
subplot(3, 1, 1);
plot(t, x_t, 'b', 'LineWidth', 1.5);
title('Input Signal x(t) - Square Wave');
xlabel('t');
ylabel('x(t)');
grid on;

% Plot h(t)
subplot(3, 1, 2);
```

```
plot(t, h_t, 'r', 'LineWidth', 1.5);
title('Impulse Response h(t) = e^{-2t}');
xlabel('t');
ylabel('h(t)');
grid on;

% Plot y(t)
subplot(3, 1, 3);
plot(t_conv, y_t, 'g', 'LineWidth', 1.5);
title('Output Signal y(t) = x(t) * h(t)');
xlabel('t');
ylabel('y(t)');
grid on;

% Overall title
sgtitle('System Response to Square Wave Input');
```



(d)The system's response to the square wave input reflects the cumulative effect of the decaying exponential impulse response, h(t), over the duration of the input. Initially, as x(t) is constant (from0≤t<5), the convolution causes y(t) to increase, indicating a gradual build-up in response. After t=5, as the input x(t) returns to zero, the response begins to decay due to the nature of h(t). This behavior demonstrates the system's memory effect, where it retains the influence of past inputs but gradually diminishes it over time due to the exponential decay in h(t).

Q4: Consider a system with an impulse response $h(t) = e^{-t}$ representing a low-pass filter. The input signal is a sum of two sinusoids:

$$x(t) = \sin(2\pi f_1 t) + \sin(2\pi f_2 t), \qquad f_1 = 1 \text{ Hz}, \qquad f_2 = 10 \text{ Hz}$$

(a) Plot the input signal $x(t)$ for $0 \leq t \leq 10$ seconds.

(b) Perform the convolution of $x(t)$ with $h(t)$ in MATLAB.

(c) Plot the output signal $y(t)$.

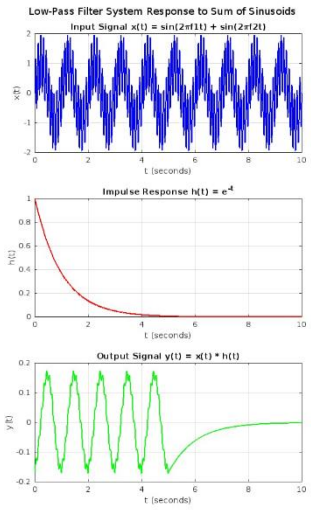
(d) Discuss the effect of the system on the two sinusoidal components (low-pass filtering behavior).

Ans:

CODE:

```
% Define the parameters
f1 = 1;  % Frequency of first sinusoid in Hz
f2 = 10; % Frequency of second sinusoid in Hz
t = 0:0.01:10;  % Time vector from 0 to 10 seconds with a sampling step of 0.01 seconds

% (a) Define and plot the input signal x(t)
x_t = sin(2 * pi * f1 * t) + sin(2 * pi * f2 * t);  % Input signal x(t)

% Plot x(t)
figure;
subplot(3, 1, 1);
plot(t, x_t, 'b', 'LineWidth', 1.5);
title('Input Signal x(t) = sin(2\pif1t) + sin(2\pif2t)');
xlabel('t (seconds)');
ylabel('x(t)');
grid on;

% (b) Define the impulse response h(t)
h_t = exp(-t);  % Impulse response h(t) = e^(-t)

% (c) Perform the convolution of x(t) and h(t) to obtain y(t)
y_t = conv(x_t, h_t, 'same') * 0.01;  % Scale by time step (0.01) for accurate result

% Plot h(t)
subplot(3, 1, 2);
plot(t, h_t, 'r', 'LineWidth', 1.5);
title('Impulse Response h(t) = e^{-t}');
```

```
xlabel('t (seconds)');
ylabel('h(t)');
grid on;

% (d) Plot the output signal y(t)
subplot(3, 1, 3);
plot(t, y_t, 'g', 'LineWidth', 1.5);
title('Output Signal y(t) = x(t) * h(t)');
xlabel('t (seconds)');
ylabel('y(t)');
grid on;

% Overall title
sgtitle('Low-Pass Filter System Response to Sum of Sinusoids');
```



(d) The impulse response h(t)=e^{-t} acts as a low-pass filter, meaning it attenuates high-frequency components more than low-frequency ones. When x(t) (containing frequencies f_1=1 Hz and f_2 =10 Hz) is convolved with h(t), the system passes the lower frequency component (1 Hz) with minimal attenuation, while the higher frequency component (10 Hz) is significantly dampened. As a result, the output y(t) mainly retains the 1 Hz component, with the 10 Hz component reduced, demonstrating the filtering effect. This response illustrates how the low-pass filter removes high-frequency signals, allowing low-frequency ones to pass through.

Q5: Given two arbitrary continuous-time signals:

$$x(t) = \sin(2\pi t), \quad 0 \le t \le 2, \qquad h(t) = t, \quad 0 \le t \le 1$$

(a) Write MATLAB code to define $x(t)$ and $h(t)$ as functions.

(b) Compute the convolution $y(t) = x(t) * h(t)$ using MATLAB's conv function.

(c) Plot the original signals $x(t)$ and $h(t)$, as well as the output $y(t)$.

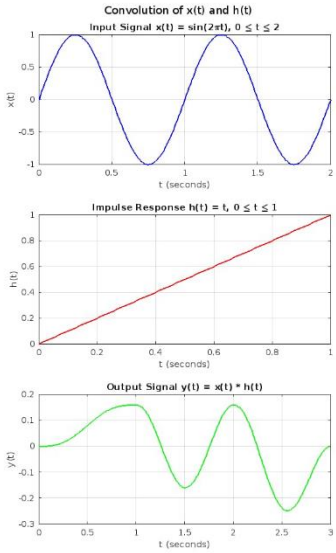
(d) Interpret the physical meaning of the convolution in this case.

Ans:

CODE:

```
% Define the time range
t_x = 0:0.01:2;  % Time range for x(t) from 0 to 2
t_h = 0:0.01:1;  % Time range for h(t) from 0 to 1
t_y = 0:0.01:(2 + 1); % Extended time range for y(t) from 0 to 3 (since convolution extends the signal)

% (a) Define x(t) and h(t) as functions
x_t = sin(2 * pi * t_x);  % x(t) = sin(2 * pi * t) for 0 <= t <= 2
h_t = t_h;              % h(t) = t for 0 <= t <= 1

% (b) Compute the convolution y(t) = x(t) * h(t)
y_t = conv(x_t, h_t, 'full') * 0.01;  % Scale by time step (0.01) for accurate result

% (c) Plot x(t), h(t), and y(t)
figure;

% Plot x(t)
subplot(3, 1, 1);
plot(t_x, x_t, 'b', 'LineWidth', 1.5);
title('Input Signal x(t) = sin(2\pit), 0 \leq t \leq 2');
xlabel('t (seconds)');
ylabel('x(t)');
grid on;

% Plot h(t)
subplot(3, 1, 2);
plot(t_h, h_t, 'r', 'LineWidth', 1.5);
title('Impulse Response h(t) = t, 0 \leq t \leq 1');
xlabel('t (seconds)');
ylabel('h(t)');
```

```
grid on;

% Plot y(t)
subplot(3, 1, 3);
plot(t_y, y_t, 'g', 'LineWidth', 1.5);
title('Output Signal y(t) = x(t) * h(t)');
xlabel('t (seconds)');
ylabel('y(t)');
grid on;

% Overall title
sgtitle('Convolution of x(t) and h(t)');
```



(d)The convolution y(t)=x(t)∗h(t) represents the output of a system with impulse response h(t)=t when excited by the input x(t)=sin(2πt). In physical terms, convolution accumulates the weighted effect of the signal x(t) over time. Here, h(t)=t grows linearly with t, implying that the system scales the input signal proportionally to time, gradually emphasizing contributions from later parts of the input signal as time progresses. As a result, the output y(t) shows a delayed and modified version of the sinusoidal input, where the initial response is low, increasing over time due to the linear nature of h(t). This behavior illustrates how convolution acts as a "smoothing" operation, distributing the influence of x(t) over a broader time period based on the characteristics of h(t).

Q6: Consider a system with the following impulse response $h(t)$:

$$h_1(t) = e^{-t} \quad \text{for} \quad t \geq 0$$
$$h_2(t) = e^{-2t} \quad \text{for} \quad t \geq 0$$

The input signal is $x(t) = \sin(2\pi t)$ for $0 \leq t \leq 5$.

(a) Compute the convolution of $x(t)$ with both impulse responses $h_1(t)$ and $h_2(t)$ in MATLAB.

(b) Plot the output signals for both cases

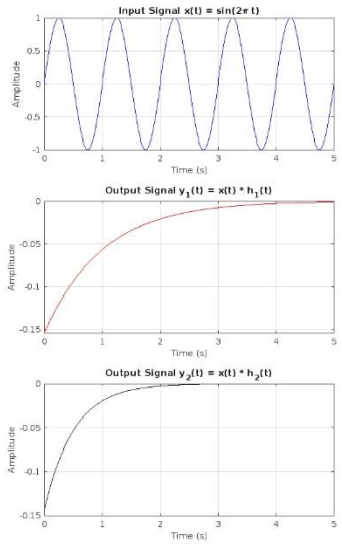
(c) Compare and contrast the outputs based on the different impulse responses. Discuss how the change in the impulse response affects the output.

Ans:

CODE:

```
% (a) Define the input signal x(t) = sin(2*pi*t) for 0 <= t <= 5
t = 0:0.01:5;  % Time vector with 0.01s step size
x_t = sin(2*pi*t);  % Input signal

% Define the two impulse responses
t_h = 0:0.01:10;  % Time vector for h(t), large enough to capture convolution effects
h1_t = exp(-t_h);  % h1(t) = e^(-t)
h2_t = exp(-2*t_h);  % h2(t) = e^(-2t)

% (b) Compute the convolutions y1(t) = x(t) * h1(t) and y2(t) = x(t) * h2(t)
y1_t = conv(x_t, h1_t, 'same') * 0.01;  % Convolution with h1(t)
y2_t = conv(x_t, h2_t, 'same') * 0.01;  % Convolution with h2(t)

% (b) Plot the output signals y1(t) and y2(t)
figure;

% Plot for y1(t)
subplot(3,1,1);
plot(t, x_t, 'b');
title('Input Signal x(t) = sin(2\pi t)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

% Plot for the output with h1(t)
subplot(3,1,2);
plot(t, y1_t, 'r');
title('Output Signal y_1(t) = x(t) * h_1(t)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

% Plot for the output with h2(t)
subplot(3,1,3);
```

```
plot(t, y2_t, 'k');
title('Output Signal y_2(t) = x(t) * h_2(t)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

% (c) Comparison and interpretation
disp('Interpretation:');
disp('h_1(t) = e^(-t) decays slowly, meaning it retains more of the oscillations in the input signal.');
disp('h_2(t) = e^(-2t) decays faster, meaning it suppresses oscillations more quickly.');
disp('Thus, y_2(t) will have a quicker attenuation of the sinusoidal component compared to y_1(t).');
```



(c) Output Shape and Damping: Since $h_1(t)$ and $h_2(t)$ represent different decaying exponentials, the outputs $y_1(t)$ and $y_2(t)$ will show different damping characteristics. The impulse response $h_1(t)$ with a slower decay rate (e^{-t}) will retain more of the sinusoidal oscillations compared to $h_2(t)$, which decays faster (e^{-2t}).

Amplitude Decay: The output with $h_2(t)$ will have a more rapid decay in amplitude over time due to the faster exponential decay, while the output with $h_1(t)$ will exhibit a slower decay, sustaining the oscillations for a longer period.

In summary, the difference in impulse response directly affects the decay rate of the output signal, impacting how long the oscillations in x(t) persist in the response

Q7: A periodic square wave $x(t)$ with period $T = 2\pi$ is defined as:

$$x(t) = \begin{cases} 1, & 0 \le t < \pi \\ -1, & \pi \le t < 2\pi \end{cases}$$

(a) Compute the Fourier series coefficients a_n, b_n, and a_0 for the square wave.

(b) Plot the square wave and its Fourier series approximation using the first 5, 10,

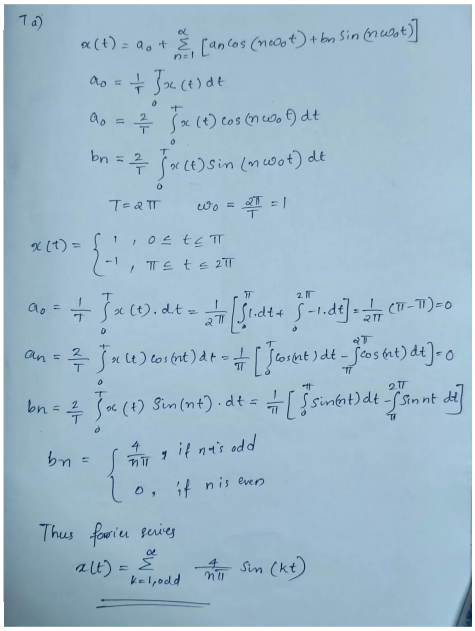
and 20 terms of the series.

(c) Use MATLAB to compute the Fourier series and plot the approximations for the given number of terms.

(d) Discuss the convergence of the Fourier series to the square wave as the number of terms increases.

Ans:

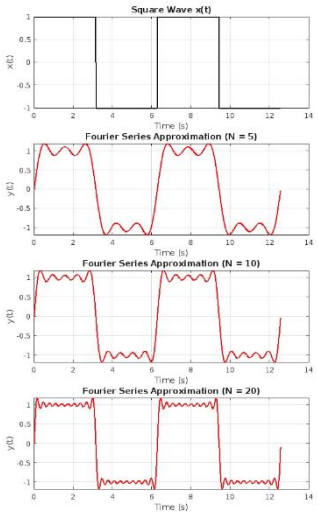
(a)



CODE:

```
% Parameters
T = 2 * pi;         % Period
t = 0:0.01:2*T;     % Time vector
x = zeros(size(t)); % Initialize square wave

% Define the square wave
for i = 1:length(t)
   if mod(t(i), T) < pi
      x(i) = 1;
   else
      x(i) = -1;
   end
end

% Number of terms for approximations
```

```
N_values = [5, 10, 20];

% Prepare the figure
figure;

% Plot the square wave
subplot(length(N_values) + 1, 1, 1);
plot(t, x, 'k', 'LineWidth', 2);
title('Square Wave x(t)');
xlabel('Time (s)');
ylabel('x(t)');
grid on;

% Fourier series approximation
for j = 1:length(N_values)
    N = N_values(j);
    y_approx = zeros(size(t)); % Initialize the approximation

    % Sum the first N terms of the Fourier series
    for n = 1:N
        bn = (2 * (1 - (-1)^n)) / (n * pi); % Calculate b_n
        y_approx = y_approx + bn * sin(n * t); % Add the n-th term
    end

    % Plot the approximation
    subplot(length(N_values) + 1, 1, j + 1);
    plot(t, y_approx, 'r', 'LineWidth', 2);
    title(['Fourier Series Approximation (N = ', num2str(N), ')']);
    xlabel('Time (s)');
    ylabel('y(t)');
    grid on;
end
```



(d) As the number of terms in the Fourier series increases, the approximation converges more closely to the square wave. However, the convergence exhibits the Gibbs phenomenon: overshoots near the discontinuities at t=0 and t=π which do not fully diminish even as more terms are added. As we include more terms, these oscillations get narrower, but the amplitude of the overshoot remains around 9% of the jump (or 0.09 × 2 = 0.18) in the square wave.

Q8: A periodic sawtooth wave is defined by:

$$x(t) = \frac{t}{\pi}, \quad -\pi \leq t < \pi$$

This signal repeats with a period $T = 2\pi$.

(a) Derive the Fourier series coefficients for the sawtooth wave.

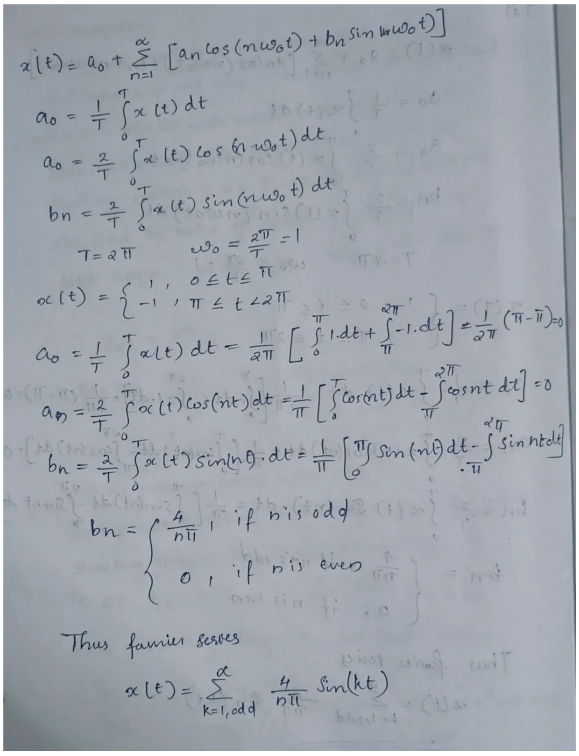
(b) Using MATLAB, plot the original sawtooth wave and its Fourier series approximations using 5, 10, and 20 terms.

(c) Comment on the accuracy of the approximation and explain why the Gibbs phenomenon occurs at discontinuities.

(d) Analyze the impact of the number of harmonics on the quality of the approximation.

Ans:

(a)



$$x(t) = a_0 + \sum_{n=1}^{\infty} [a_n \cos(n\omega_0 t) + b_n \sin(n\omega_0 t)]$$

$$a_0 = \frac{1}{T}\int_0^T x(t)\,dt$$

$$a_0 = \frac{2}{T}\int_0^T x(t)\cos(n\omega_0 t)\,dt$$

$$b_n = \frac{2}{T}\int_0^T x(t)\sin(n\omega_0 t)\,dt$$

$T = 2\pi \qquad \omega_0 = \frac{2\pi}{T} = 1$

$$x(t) = \begin{cases} 1, & 0 \le t \le \pi \\ -1, & \pi \le t < 2\pi \end{cases}$$

$$a_0 = \frac{1}{T}\int_0^T x(t)\,dt = \frac{1}{2\pi}\left[\int_0^{\pi} 1\,dt + \int_{\pi}^{2\pi} -1\,dt\right] = \frac{1}{2\pi}(\pi - \pi) = 0$$

$$a_n = \frac{2}{T}\int_0^T x(t)\cos(nt)\,dt = \frac{1}{\pi}\left[\int_0^{\pi}\cos(nt)\,dt - \int_{\pi}^{2\pi}\cos nt\,dt\right] = 0$$

$$b_n = \frac{2}{T}\int_0^T x(t)\sin(nt)\,dt = \frac{1}{\pi}\left[\int_0^{\pi}\sin(nt)\,dt - \int_{\pi}^{2\pi}\sin nt\,dt\right]$$

$$b_n = \begin{cases} \frac{4}{n\pi}, & \text{if } n \text{ is odd} \\ 0, & \text{if } n \text{ is even} \end{cases}$$

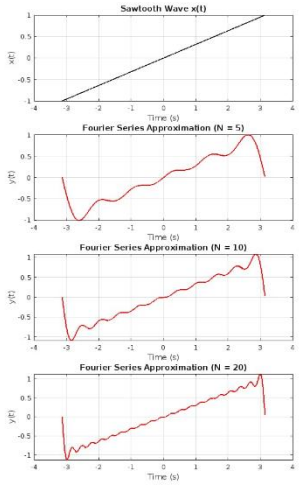
Thus fourier series

$$x(t) = \sum_{k=1,\text{odd}}^{\infty} \frac{4}{n\pi}\sin(kt)$$

CODE:

```
% Parameters
T = 2 * pi;           % Period
t = -pi:0.01:pi;      % Time vector
x = t / pi;           % Sawtooth wave

% Number of terms for approximations
N_values = [5, 10, 20];

% Prepare the figure
figure;

% Plot the original sawtooth wave
subplot(length(N_values) + 1, 1, 1);
plot(t, x, 'k', 'LineWidth', 2);
title('Sawtooth Wave x(t)');
xlabel('Time (s)');
ylabel('x(t)');
grid on;

% Fourier series approximation
for j = 1:length(N_values)
    N = N_values(j);
```

```
    y_approx = zeros(size(t)); % Initialize the approximation

% Sum the first N terms of the Fourier series
    for n = 1:N
        bn = -2 * ((-1)^n) / (n * pi); % Calculate b_n
        y_approx = y_approx + bn * sin(n * t); % Add the n-th term
    end

    % Plot the approximation
    subplot(length(N_values) + 1, 1, j + 1);
    plot(t, y_approx, 'r', 'LineWidth', 2);
    title(['Fourier Series Approximation (N = ', num2str(N), ')']);
    xlabel('Time (s)');
    ylabel('y(t)');
    grid on;
end
```



(c)As the number of terms in the Fourier series approximation increases, the approximation of the sawtooth wave improves. However, near the discontinuity at t = ±π, there are persistent oscillations known as the Gibbs phenomenon. This phenomenon occurs because the Fourier series approximation of a signal with a discontinuity cannot converge exactly at the discontinuity. Instead, it exhibits an overshoot of about 9% of the jump discontinuity. As more terms are added, the oscillations become narrower but do not disappear completely.

(d) The quality of the Fourier series approximation improves as the number of harmonics (terms) increases. With more terms, the approximation more closely follows the shape of the original sawtooth wave, especially between the discontinuities. However, due to the Gibbs phenomenon, there will always be a small overshoot at the points of discontinuity, although the overshoots are localized and narrow with an increasing number of harmonics.

Q9: A triangular wave $x(t)$ has period $T = 2\pi$ and is defined as:

$$x(t) = \begin{cases} \frac{t}{\pi}, & 0 \le t \le \pi \\ -\frac{t}{\pi} + 2, & \pi \le t \le 2\pi \end{cases}$$

(a) Compute the Fourier series coefficients for the triangular wave.

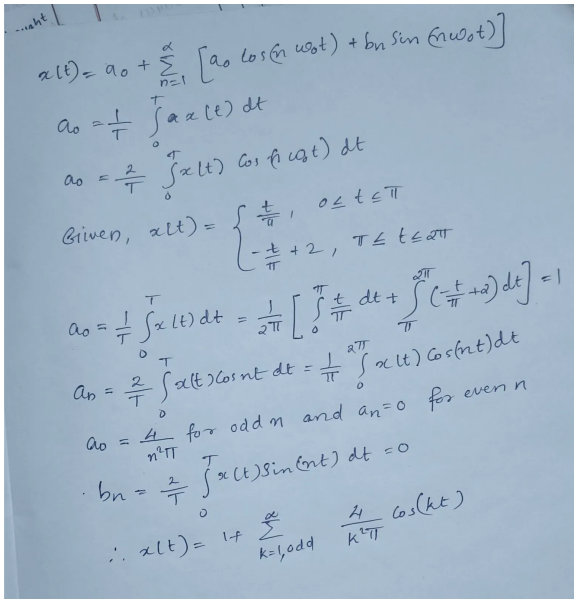
(b) Plot the triangular wave and its Fourier series approximations using MATLAB with 5, 10, and 20 terms.

(c) Discuss the symmetry properties of the triangular wave and their impact on the Fourier coefficients.

(d) Compare the rate of convergence of the Fourier series for the triangular wave with that of the square wave.

Ans:

(a)

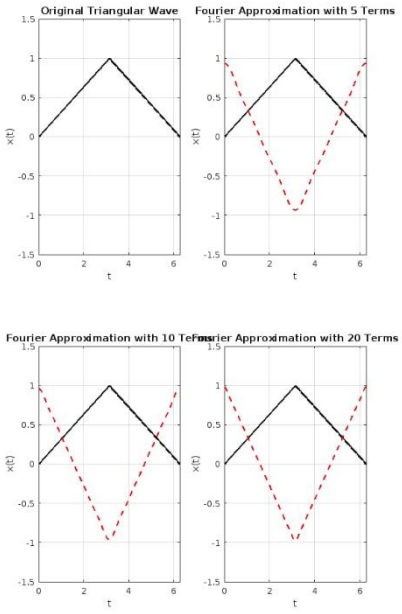


CODE:

```
% MATLAB script to compute Fourier series of a periodic triangular wave
% Period T = 2*pi
T = 2*pi;
omega0 = 2*pi/T;  % Fundamental frequency
N = 20;  % Maximum number of terms to compute in the Fourier series
t = linspace(0, 2*pi, 1000);  % Time vector for plotting % Triangular wave definition
x_t = @(t) (t/pi).(t>=0 & t<=pi) + (-t/pi + 2).(t>pi & t<=2*pi);  % Triangular wave from 0 to 2*pi
```

```
% Initialize Fourier coefficients
a0 = 0;
an = zeros(1, N);
bn = zeros(1, N);

% Compute the Fourier coefficients
for n = 1:N
   if mod(n,2) == 1  % Only odd harmonics contribute
      an(n) = 8 / (pi^2 * n^2);  % an coefficient for triangular wave
   else
      an(n) = 0;  % Even terms are zero
   end
   bn(n) = 0;  % bn terms are zero for triangular wave (even function)
end

 % Plot the original triangular wave
figure;
subplot(2, 2, 1);
plot(t, x_t(t), 'k', 'LineWidth', 1.5);
title('Original Triangular Wave');
xlabel('t');
ylabel('x(t)');
axis([0 2*pi -1.5 1.5]);
grid on;

% Function to compute Fourier series approximation
fourier_approx = @(t, N) a0/2 + sum(an(1:N)' .* cos((1:N)' * omega0 .* t), 1);

% Plot Fourier approximations with 5, 10, and 20 terms
terms_to_plot = [5, 10, 20];
for i = 1:length(terms_to_plot)
   N_terms = terms_to_plot(i);
   subplot(2, 2, i+1);
   plot(t, x_t(t), 'k', 'LineWidth', 1.5);  % Plot original triangular wave
   hold on;
   plot(t, fourier_approx(t, N_terms), 'r--', 'LineWidth', 1.5);  % Plot Fourier series approximation
   title(['Fourier Approximation with ', num2str(N_terms), ' Terms']);
   xlabel('t');
```

```
    ylabel('x(t)');
    axis([0 2*pi -1.5 1.5]);
    grid on;
end

% Display the convergence discussion
disp('As the number of terms increases, the Fourier series approximation becomes closer to the original triangular wave.');
disp('Since the triangular wave is continuous and has a finite slope, the Fourier series converges faster compared to the square wave.');
```



(c) The triangular wave is an even function and has half-wave symmetry. This symmetry implies:

Only cosine terms (even harmonics) are present in the Fourier series, meaning all b_n coefficients are zero.
Only odd harmonic terms contribute to the Fourier series because the waveform has odd symmetry in each half-period.
Due to the smooth nature of the triangular wave (continuous with a continuous first derivative), the Fourier coefficients a_n decay faster (proportionally to $1/n^2$) than for a square wave (where the coefficients decay as 1/n).

(d) The Fourier series for the triangular wave converges faster than for the square wave. This is because the triangular wave is continuous, while the square wave has discontinuities. For the square wave, the Fourier coefficients decay as 1/n, resulting in slower convergence and more pronounced Gibbs phenomenon at the discontinuities.
For the triangular wave, the coefficients decay as $1/n^2$, leading to faster convergence and reduced Gibbs phenomenon. The smoother waveform of the triangular wave allows for a more accurate approximation with fewer terms in the Fourier series.

Q10: A half-wave rectified sine wave is defined by:

$$x(t) = \begin{cases} \sin(t), & 0 \le t \le \pi \\ 0, & \pi < t \le 2\pi \end{cases}$$

(a) Derive the Fourier series coefficients for the half-wave rectified sine wave.

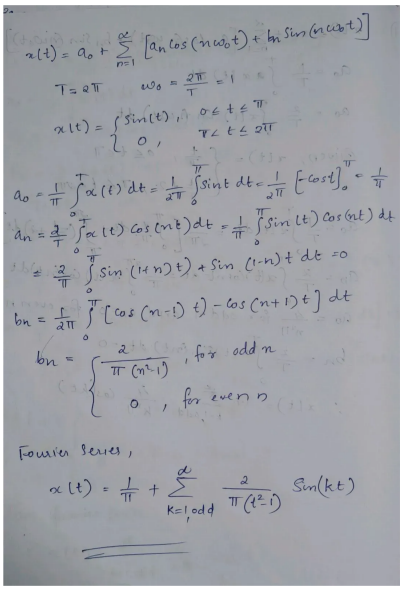
(b) Using MATLAB, plot the original signal and its Fourier series approximations for 5, 10, and 20 terms.

(c) Analyze the frequency spectrum of the signal and explain the presence of both sine and cosine terms in the Fourier series.

(d) Comment on the physical interpretation of the Fourier coefficients.

Ans:

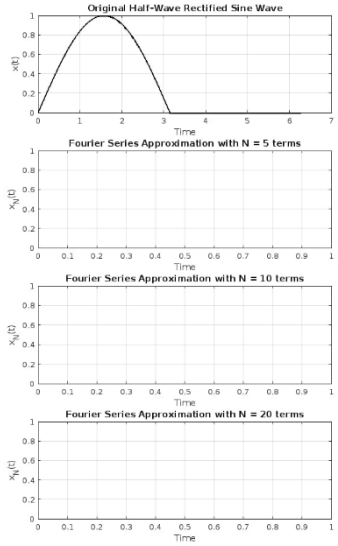
(a)



(b) CODE:

```
% Define time vector and period
T = 2 * pi;
t = linspace(0, T, 1000);

% Define the original half-wave rectified sine wave
x_t = (t <= pi) .* sin(t);

% Plot original signal
figure;
subplot(4,1,1);
plot(t, x_t, 'k', 'LineWidth', 1.5);
title('Original Half-Wave Rectified Sine Wave');
xlabel('Time');
```

```
ylabel('x(t)');
grid on;

% Fourier series approximation
terms = [5, 10, 20];
for i = 1:length(terms)
   N = terms(i);
   y_t = ones(size(t)) / pi; % Start with a0 term

   for n = 1:N
      b_n = (2 / (pi * (1 - n^2))) * (1 - (-1)^n); % Fourier coefficient b_n
      y_t = y_t + b_n * sin(n * t); % Sum up to N terms
   end

   % Plot Fourier approximation
   subplot(4,1,i+1);
   plot(t, y_t, 'r', 'LineWidth', 1.5);
   title(['Fourier Series Approximation with N = ', num2str(N), ' terms']);
   xlabel('Time');
   ylabel(['x_N(t)']);
   grid on;
end
```



(c)The frequency spectrum of the signal shows that only sine terms are present due to the shape of the half-wave rectified sine wave. Since the signal is asymmetric and non-zero only over part of its period, it does not exhibit even symmetry (which would produce cosine terms). Only sine terms contribute to the Fourier series, with the primary frequencies corresponding to odd harmonics due to the structure of the half-wave rectification.

(d)The Fourier coefficients b_n represent the amplitude of each sine component in the approximation of $x(t)$. The decay in b_n values with increasing n implies that higher frequencies have lower amplitudes,

contributing less to the overall shape of the signal. This aligns with the concept that the half-wave rectified sine wave has sharp transitions, and these transitions require higher frequency components for accurate approximation.

Q11: A system is described by the following second-order differential equation:

$$\frac{d^2y(t)}{dt^2} + 4\frac{dy(t)}{dt} + 8y(t) = 5u(t)$$

where $u(t)$ is the input, and $y(t)$ is the output.

(a) Derive the transfer function $H(s) = \frac{Y(s)}{U(s)}$ for the system.

(b) Simulate the step response of the system using MATLAB.

(c) Plot the step response and determine if the system reaches a steady state.

Ans:

(a) Convert the differential equation to laplace domain

$$s^2 Y(s) + 4s\,Y(s) + 8\,Y(s) = 5U(s)$$

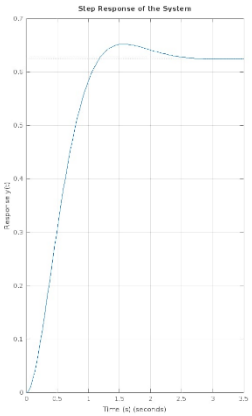
$$(s^2 + 4s + 8)\,Y(s) = 5U(s)$$

$$H(s) = 5 / (s^2 + 4s + 8)$$

(b) CODE:

```
% MATLAB Code to Simulate the Step Response
 % Define the transfer function coefficients
num = [5]; % Numerator coefficients
den = [1 4 8]; % Denominator coefficients

% Create the transfer function
H = tf(num, den);

% Simulate the step response
figure;
step(H);
title('Step Response of the System');
xlabel('Time (s)');
ylabel('Response y(t)');
```

grid on;



Since the poles are in the left half of the s-plane, the system will stabilize, and thus it reaches a steady state.

Q12: A majority of modern trains and local transit vehicles utilize electric traction mo- tors. The electric motor drive for a railway vehicle is shown in block diagram form in Figure 1, incorporating the necessary control of the velocity of the vehicle. Af- ter solving the differential equations and substituting system parameters we get 2. Ignore the disturbance torque $T_d(s)$.

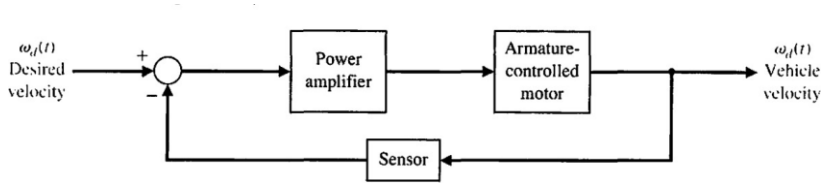


Figure 1: Speed control of an electric motor traction

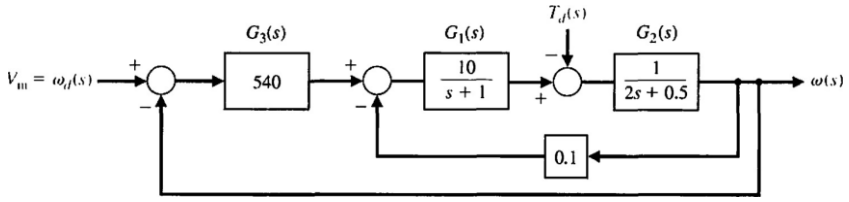


Figure 2: Speed control of an electric motor traction after substituting system parameters

(a) Find the overall transfer function of the system $\frac{\omega(s)}{\dot{\omega}_g(s)}$

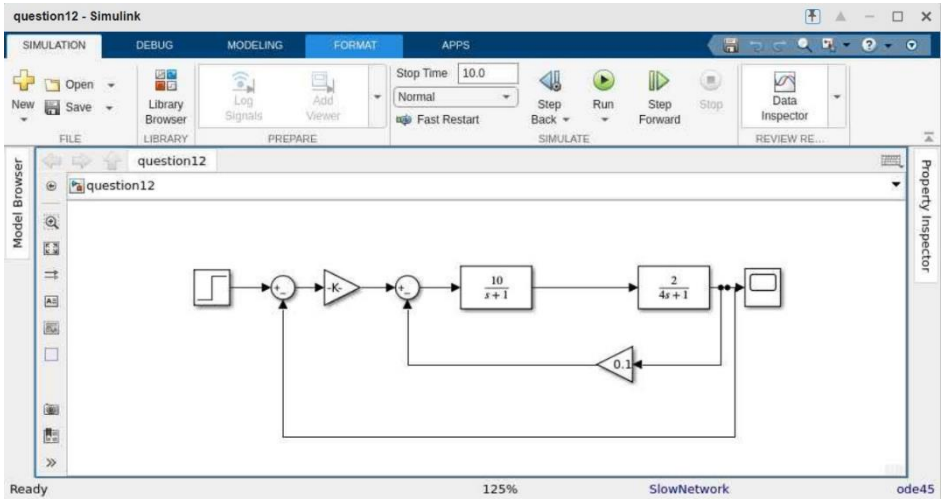
(b) Implement the block diagram representation of the system shown in 2 in SIMULINK and find the overall transfer function.

(c) Simulate the step response and plot the figure.

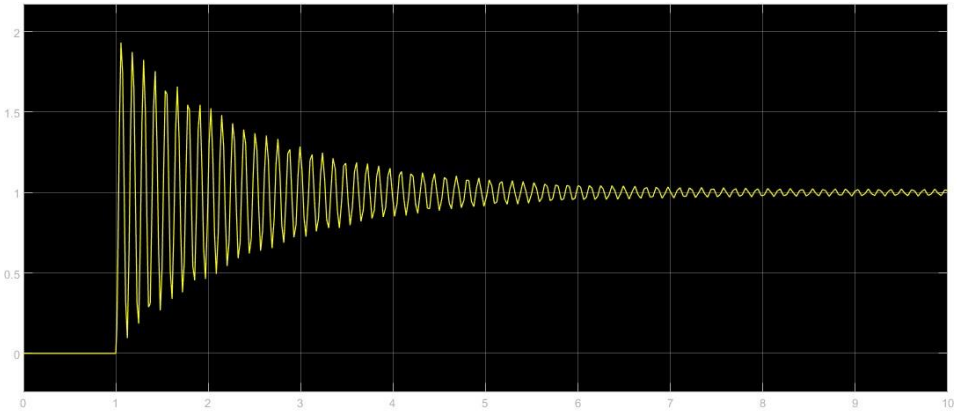
Ans:

(a)

(b)


question12 - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Open
New
Save
FILE
Library Browser
LIBRARY
Log Signals
Add Viewer
PREPARE
Stop Time
10.0
Normal
Fast Restart
Step Back
Run
Step Forward
Stop
SIMULATE
Data Inspector
REVIEW RE...
Model Browser
question12
Property Inspector
-K-
10
s+1
2
4s+1
0.1
Ready
125%
SlowNetwork
ode45

(c)


2
1.5
1
0.5
0
0
1
2
3
4
5
6
7
8
9
10

Q13: Given the transfer function of a system:

$$H(s) = \frac{10(s + 1)}{(s^2 + 6s + 10)}$$

(a) Find the poles and zeros of the system.

(b) Plot the pole-zero map in MATLAB.

(c) Discuss the stability of the system based on the pole locations.

Ans:

(a) Zeros: Zeros of the transfer function occur when the numerator is zero.

$10(s+1) = 0$
$\Rightarrow s = -1$
So, the zero of the system is at $s = -1$

Poles: These are found by setting the denominator equal to zero:

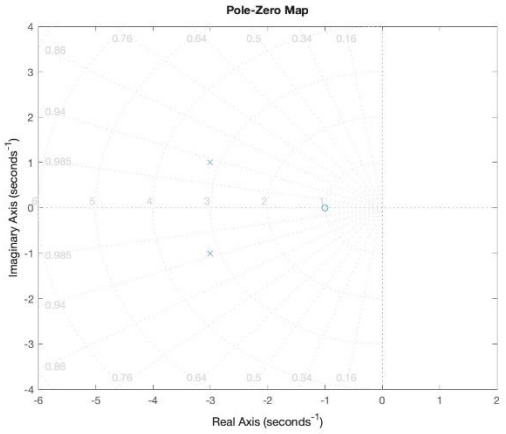
$s^2 + 6s + 10 = 0$

Solving this quadratic equation we get the poles of the system as,

$s = -3 + j$ and $-3 - j$

(b) CODE:

```
% Define the transfer function H(s) = 10 * (s + 1) / (s^2 + 6s + 10)
numerator = [10, 10];    % Coefficients of the numerator 10(s + 1)
denominator = [1, 6, 10]; % Coefficients of the denominator s^2 + 6s + 10

% Create the transfer function
H = tf(numerator, denominator);

% Plot pole-zero map
pzmap(H);
grid on;
title('Pole-Zero Map');
```



For stability in continuous-time systems, all poles must have negative real parts (i.e., they must be in the left half of the complex plane).

The poles are at
$s = -3 + i$ and $s = -3 - i$, both of which have negative real parts.
Since both poles are located in the left half of the complex plane, the system is stable.

Q14: Consider the following transfer function:

$$H(s) = \frac{7}{s^2 + 3s + 2}$$

(a) Find the poles of the system.
(b) Simulate the impulse response of the system.
(c) Plot the impulse response and analyze if the system is stable based on the response.

Ans:

(a) The poles of the system are the values of s that make the denominator zero.

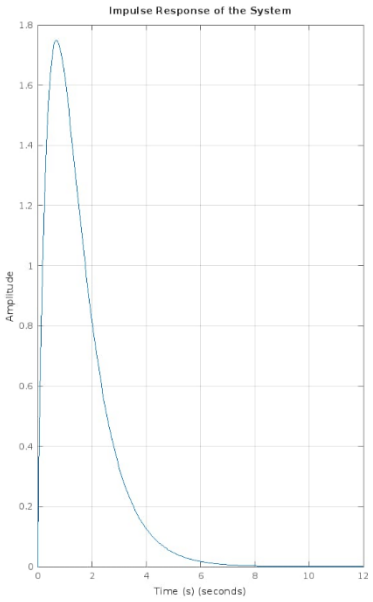
i.e., $s^2 + 3s + 2 = 0$
$(s + 1)(s + 2) = 0$

So the poles are $s = -1$ and $s = -2$.

(b) CODE:

```
% Define the numerator and denominator of the transfer function
numerator = 7;
```

```
denominator = [1, 3, 2];

% Create the transfer function
sys = tf(numerator, denominator);

% Simulate the impulse response
impulse(sys);

% Add title and labels
title('Impulse Response of the System');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
```



Q15: Given a system with the transfer function:

$$H(s) = \frac{(s + 1)}{(s^2 + 4s + 4)}$$

(a) Find the poles and zeros of the system.
(b) Plot the step response of the system.
(c) Generate the pole-zero map and comment on the system's stability.

Ans:

(a) Zeroes of the system are found by setting the numerator equal to zero.

$S + 1 = 0 \Rightarrow s = -1$ is the zero of the system.

Poles of the system are found by equating denominator to zero.

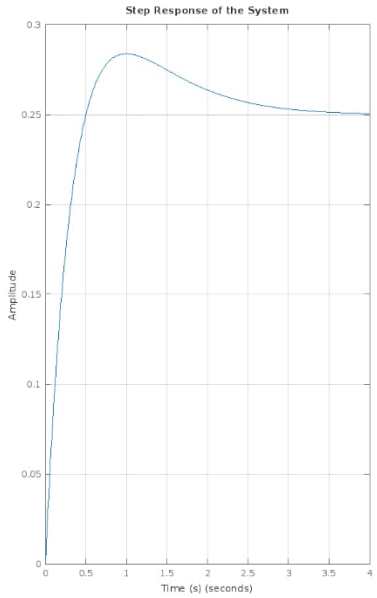
i.e., $s^2 + 4s + 4 = 0$

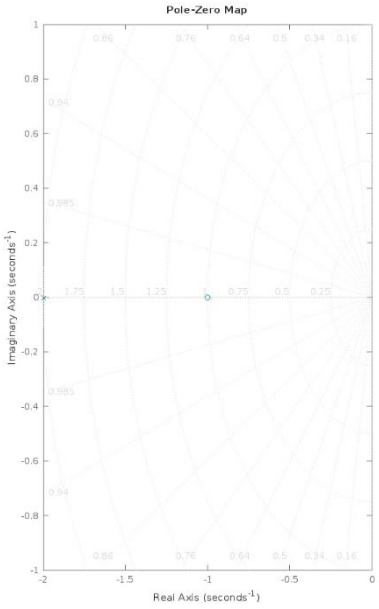
$(s + 2)(s + 2) = 0$

Thus the system has 2 poles at $s = -2$.

(b) CODE:

```
% Define the transfer function: H(s) = (s + 1) / (s^2 + 4s + 4)
% Numerator and denominator of the transfer function
numerator = [1, 1];     % (s + 1)
denominator = [1, 4, 4]; % (s^2 + 4s + 4)

% Create the transfer function
sys = tf(numerator, denominator);

% (b) Plot the step response of the system
figure;
step(sys); % Plot the step response
title('Step Response of the System');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;

% (c) Generate the Pole-Zero Map
figure;
pzmap(sys); % Plot the pole-zero map
title('Pole-Zero Map');
grid on;
```





(c) For this transfer function:

The poles are at $s = -2$

s=−2 (double pole).

Since all poles are in the left half of the s-plane, the system is stable.

PARVATHY S RAJ
S5EEE
58