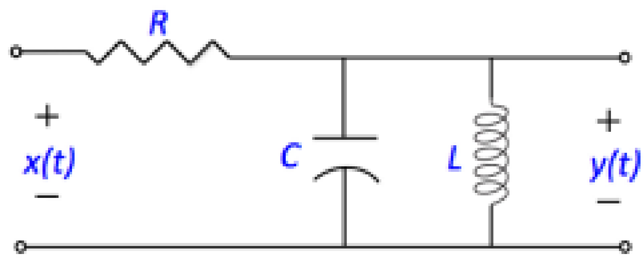
Jason Waseq
ECE103L
Assignment 3

1. Following RLC circuit is described by the differential equation (1). Use Matlab built-in differential equation solver dsolve()to derive the impulse response h(t) for this circuit when R=2Ω, C=1F, L=0.5H. Plot the impulse response h(t) from a range −10≤ t≤ 30.

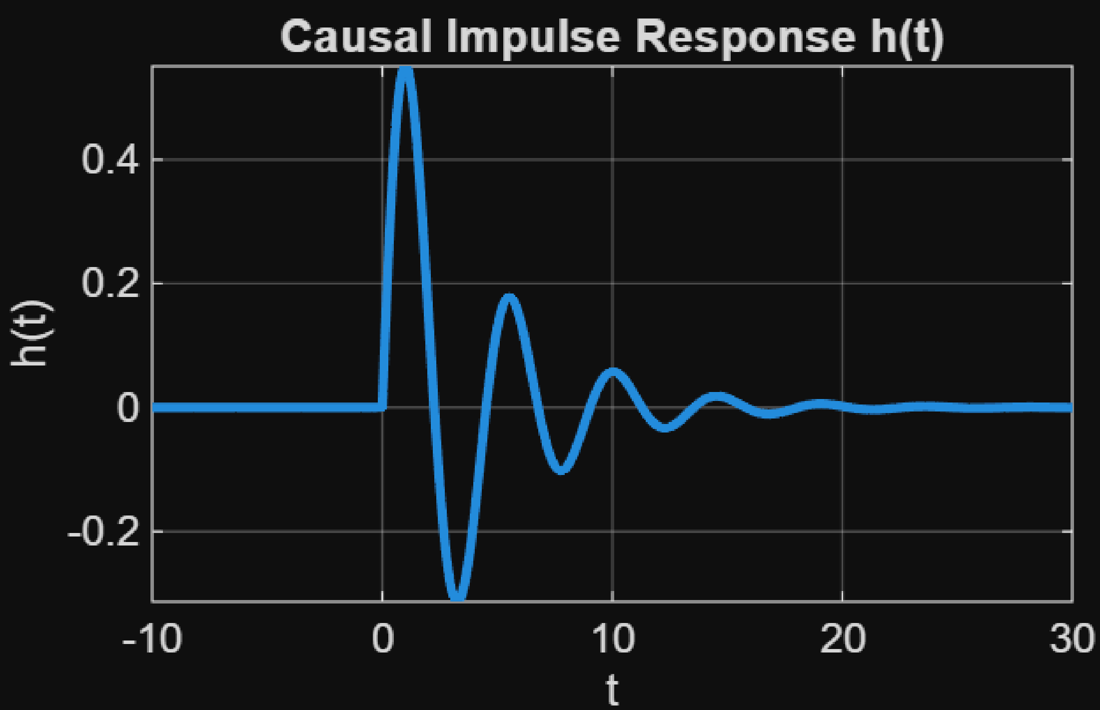


(1) RC (d^2y(t))/(dt^2) + (dy(t))/(dt) + (Ry(t))/L = dx(t)/dt

Final Answer:

```
Causal Impulse Response h(t):
```

$$
\begin{cases} 0 & \text{if } t < 0 \\ \dfrac{4\sqrt{31}\, e^{-\frac{t}{4}} \sin\left(\dfrac{\sqrt{31}\, t}{4}\right)}{31} & \text{otherwise} \end{cases}
$$



The impulse response h(t) is derived using the dsolve() function, and the plot visualizes how the system responds to an impulse input over the specified time range.

Detailed Solution:

1. Define the parameters and the differential equation.
2. Use dsolve() to find the solution.
3. Plot the impulse response over the specified range.

MATLAB Code:

```
% Define symbolic impulse response h(t)
syms hz(t)
Dh = diff(hz, t);
D2h = diff(hz, t, 2);

% Differential equation: 2*d²h/dt² + dh/dt + 4*h = 0
```

```
ode = 2*D2h + Dh + 4*hz == 0;

% Initial conditions: h(0) = 0, h'(0) = 1
conds = [hz(0) == 0, Dh(0) == 1];

% Solve the differential equation
hSol(t) = dsolve(ode, conds);
hSol(t) = simplify(hSol);

% Make the solution causal: h(t) = hSol(t) * u(t)
% (i.e., h(t) = 0 for t < 0)
hCausal(t) = piecewise(t < 0, 0, hSol(t));

% Display the result
disp('Causal Impulse Response h(t):');
disp(hCausal);

% Plot h(t) over desired range
fplot(hCausal, [-10, 30], 'LineWidth', 2);
title('Causal Impulse Response h(t)');
xlabel('t');
ylabel('h(t)');
grid on;
```

2. Consider the following input signal

$x_1(t) = 5,\ 0 \leq t < 10$

$\quad\quad\quad 0$, elsewhere

$x_2(t) = 2x_1(t - 10)$

$x_{linear\_comb}(t) = x_1(t) + 2x_1(t - 10)$

Using the example Matlab file simplified_convolution_runtime.m, plot the output signals in three separate figure windows:

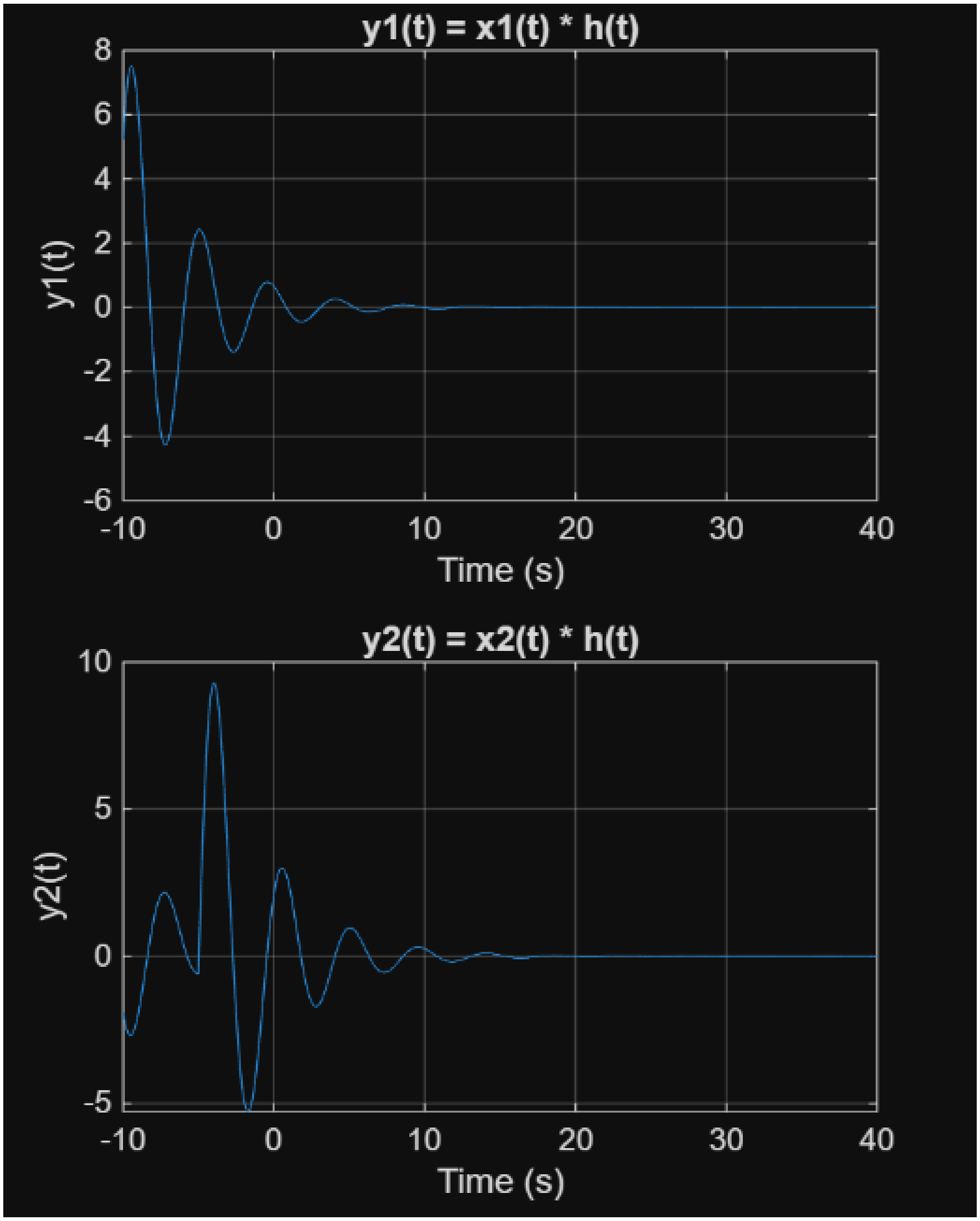
(a) $y_1(t) = x_1(t) * h(t)$

(b) $y_2(t) = x_2(t) * h(t)$

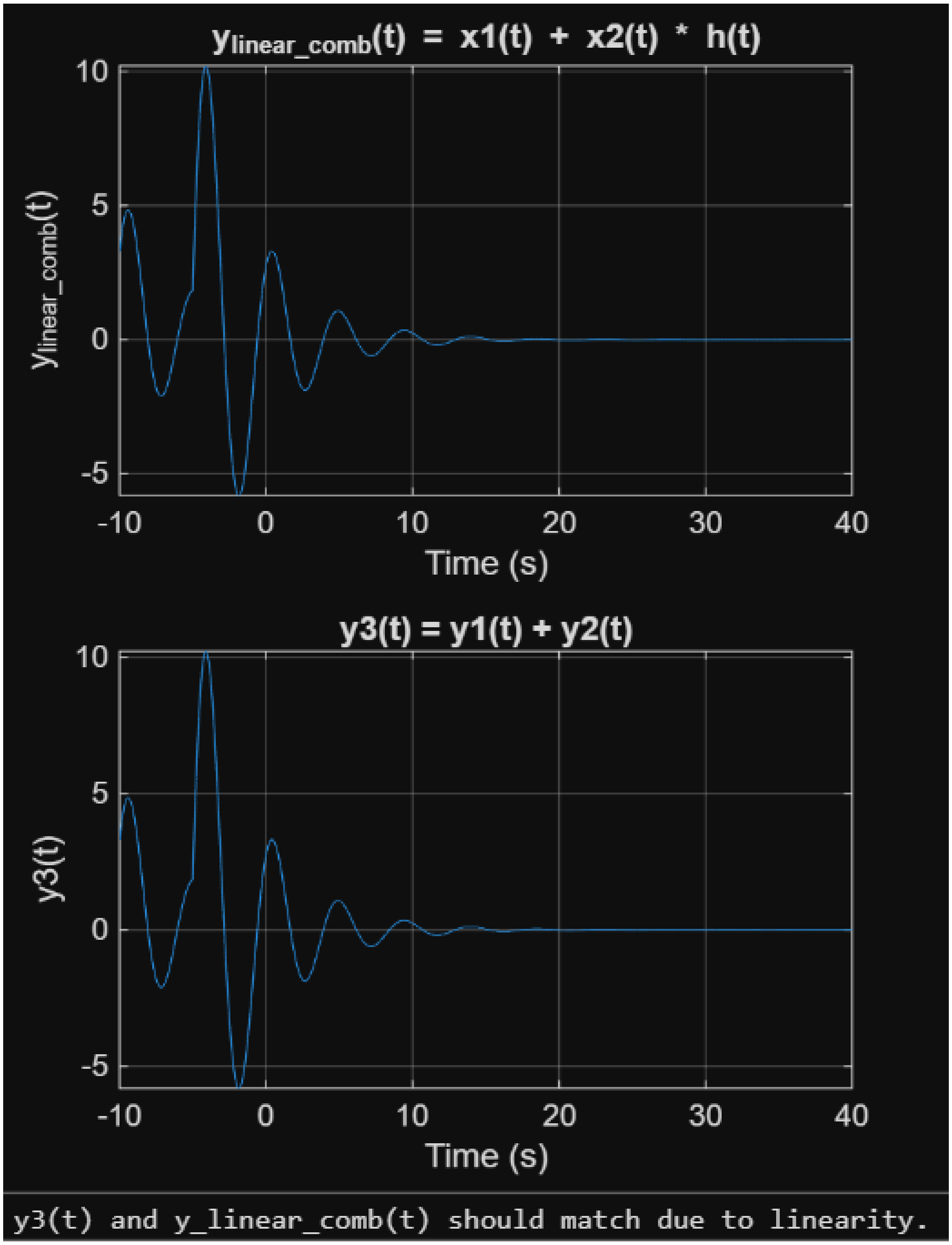
(c) $y_{linear\_comb} = x_{linear\_comb}(t) * h(t)$.

Use the ranges of 'τ' and 't' as $-10 \leq \tau \leq 40$ and $-10 \leq t \leq 40$. Also plot $y_3(t)=y_1(t)+y_2(t)$ and comment on similarity of $y_3(t)$ and $y_{linear\_comb}(t)$.

Final Answer:



y1(t) = x1(t) * h(t)
y1(t)
Time (s)
y2(t) = x2(t) * h(t)
y2(t)
Time (s)



The code provided will generate the required plots for y1(t), y2(t), ylinearcomb(t), and y3(t). The output signals y3(t) and ylinearcomb(t) should exhibit similarity, as y3(t) represents the combined response of the individual components of the linear combination.

Detailed Solution:
To plot the output signals for the given input signals x1(t), x2(t), and xlinearcomb(t) using MATLAB, we will follow these steps:

1. Define the input signals.
2. Define the impulse response h(t).

3. Perform convolution for each output signal.
4. Plot the results in separate figure windows.
5. Comment on the similarity between y3(t) and ylinearcomb(t).

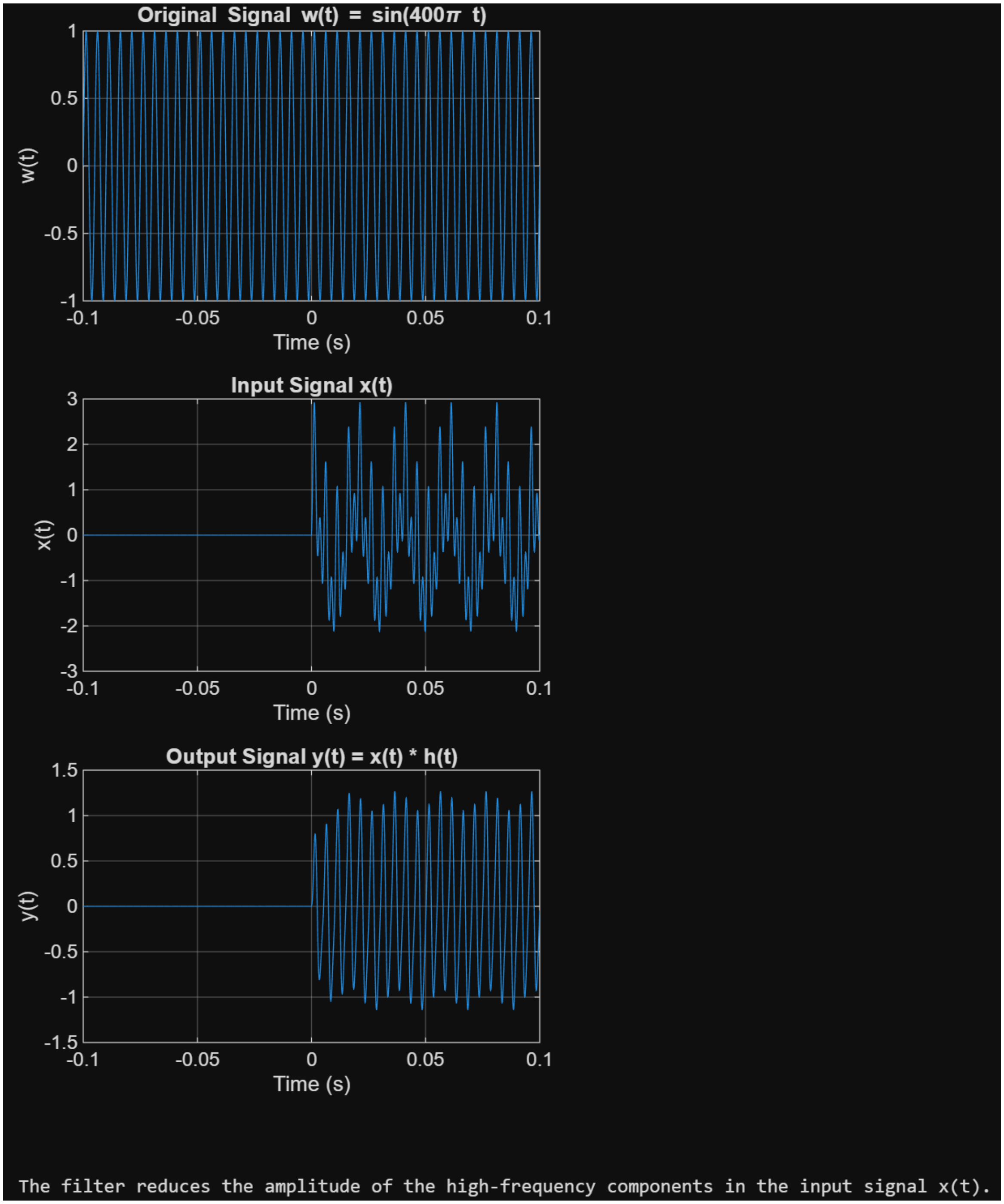
MATLAB Code:

```
% Define time vector
t = -10:0.01:40;

% Convert symbolic h(t) to a numeric function handle
h_func = matlabFunction(hSol);

% Evaluate h(t) numerically
h_vals = h_func(t);

% Define x1(t): 5 for 0 <= t < 10
x1_vals = 5 * (t >= 0 & t < 10);

% Define x2(t): time-shifted version of x1(t)
x2_vals = 2 * (t >= 10 & t < 20);  % since x2(t) = 2 * x1(t - 10)

% Linear combination of inputs
x_comb_vals = x1_vals + x2_vals;

% Convolution step size
dt = 0.01;

% Compute convolutions
y1 = conv(x1_vals, h_vals, 'same') * dt;
y2 = conv(x2_vals, h_vals, 'same') * dt;
y_comb = conv(x_comb_vals, h_vals, 'same') * dt;

% Plot y1(t)
figure;
plot(t, y1);
title('y1(t) = x1(t) * h(t)');
xlabel('Time (s)');
ylabel('y1(t)');
grid on;

% Plot y2(t)
figure;
plot(t, y2);
title('y2(t) = x2(t) * h(t)');
xlabel('Time (s)');
```

```
ylabel('y2(t)');
grid on;

% Plot y_comb(t)
figure;
plot(t, y_comb);
title('y_{linear\_comb}(t) = x1(t) + x2(t) * h(t)');
xlabel('Time (s)');
ylabel('y_{linear\_comb}(t)');
grid on;

% Plot y3 = y1 + y2
y3 = y1 + y2;

figure;
plot(t, y3);
title('y3(t) = y1(t) + y2(t)');
xlabel('Time (s)');
ylabel('y3(t)');
grid on;

% Comparison
disp('y3(t) and y_linear_comb(t) should match due to linearity.');
```

3. A single-tone signal $w(t) = \sin(400\pi t)$ is transmitted to an audio amplifier and speaker to produce a high-temperature warning for a silicon crystal-growing factory. A filter having impulse response $h(t)= 400e^{-200t}\cos(400\pi t)u(t)$ has been designed to reduce additive interference in the received signal. Using Matlab in-built convolution function: conv(), find the filter output signal y(t), when the received signal is $x(t) = [\cos(100\pi t)+ \sin(400\pi t)- \cos(800\pi t)]u(t)$ (signal w(t) was corrupted by interference and resulted in an input signal x(t)). Plot the output signal, the input signal, and w(t)for the range of $-0.1 \le t \le 0.1$. Comment on the effect of the filter on the signal. While solving this problem, pay attention to the time resolution/step (dT) you need to use.

Final Answer:



The output of the MATLAB code will display three plots: the original signal w(t), the input signal x(t), and the filtered output signal y(t). The filter effectively reduces the amplitude of the high-frequency components in the input signal x(t), allowing the desired frequency components to be emphasized while attenuating the interference.

Detailed Solution:

Step 1: Define the Signals

- The original signal w(t): w(t) = sin(400pit)
- The input signal x(t): x(t) = cos(100pt) + sin(400pit) - cos(800pit)

- The filter impulse response h(t): h(t) = 400e^(-200t)cos(400pi t)u(t)

MATLAB Code:

```
% Define time vector
t = -0.1:0.0001:0.1; % Time range with high resolution

% Define the original signal w(t)
w = sin(400 * pi * t);

% Define the input signal x(t)
x = cos(100 * pi * t) + sin(400 * pi * t) - cos(800 * pi * t);
x(t < 0) = 0; % Apply unit step function u(t)

% Define the filter impulse response h(t)
h = 400 * exp(-200 * t) .* cos(400 * pi * t);
h(t < 0) = 0; % Apply unit step function u(t)

% Perform convolution
y = conv(x, h, 'same') * 0.0001; % Convolution and scale by dT

% Plot the original signal w(t)
figure;
plot(t, w);
title('Original Signal w(t) = sin(400\pi t)');
xlabel('Time (s)');
ylabel('w(t)');
grid on;

% Plot the input signal x(t)
figure;
plot(t, x);
title('Input Signal x(t)');
xlabel('Time (s)');
ylabel('x(t)');
grid on;

% Plot the output signal y(t)
figure;
plot(t, y);
title('Output Signal y(t) = x(t) * h(t)');
xlabel('Time (s)');
ylabel('y(t)');
grid on;
```

```
% Comment on the effect of the filter
disp('The filter reduces the amplitude of the high-frequency components in the input
signal x(t).');
```