Ged Miller

University of Oklahoma

ECE 3793-001

Dr. Nathan Goodman

November 9th, 2023

## Introduction:

Filtering signals is key component of signal processing. Filtering allows for background noise to be eliminated, signals to be attenuated, and for data to be layered onto carrier signals and to be extracted. In this experiment, a signals frequency response and impulse response will be extracted from LCCDE poles and zero's in MATLAB.

To calculate these signals given the poles and zero's, the transfer function H(s) was first derived. Then the impulse response was calculated using an inverse Laplace Transform.

Once the frequency response and impulse response have been created, the impulse will be convolved with a sinusoidal input to simulate how the system would respond to the impulse response over a set time. Lastly, the frequency response will be multiplied by the convolved sinusoidal signal to confirm that the steady-state response of a linear time-invariant system to a sinusoidal input can be described by the system's frequency response at the input signal (Figure 1).

$$\lim_{t\to\infty} e^{j2\pi F_0 t} * h(t) = H(F = F_0)e^{j2\pi F_0 t}$$

Figure 1: Steady-state response of a LTI system.

## Methods:

To analyze the relationship between the Steady-state response of a LTI system, the following methods were used. First, properties of LCCDE systems allowed the given poles and zeros to be arranged into a transfer function H(S) as seen in Figure 2.

$$H(s) = A_0 \frac{P(s)}{Q(s)} = \frac{(s-z_1)(s-z_2)\cdots(s-z_M)}{(s-p_1)(s-p_2)\cdots(s-p_N)}.$$

Figure 2: Frequency Response of an LTI System.

After deriving the frequency response, the inverse Laplace transform is used to produce the system's response to an impulse input h(t) using "ilaplace()". Next, MATLAB's plotting functions "plot(), title(), xlabel(), ylabel(), grid, hold, set()" are used to graph relevant functions. The convolution function "conv()" is used to convolve the sinusoidal input to simulate the system's response to a signal over time. Next, partial fraction decomposition functions "zp2tf()" and "residue()"  used to turn the poles into and zeros into polynomials to find the coefficients of the partial fraction expansion. Lastly, these tools are used to simulate six plots that allow the systems response to be compared to sinusoidal inputs when different frequencies "F0" are used.

Results:

After computing the frequency response from the given zero's and poles, a frequency axis with mins, maxes, and spacing is created. The frequency response is evaluated at every value of the frequency axis, graphically creating the frequency response. Then the response is converted to dB.

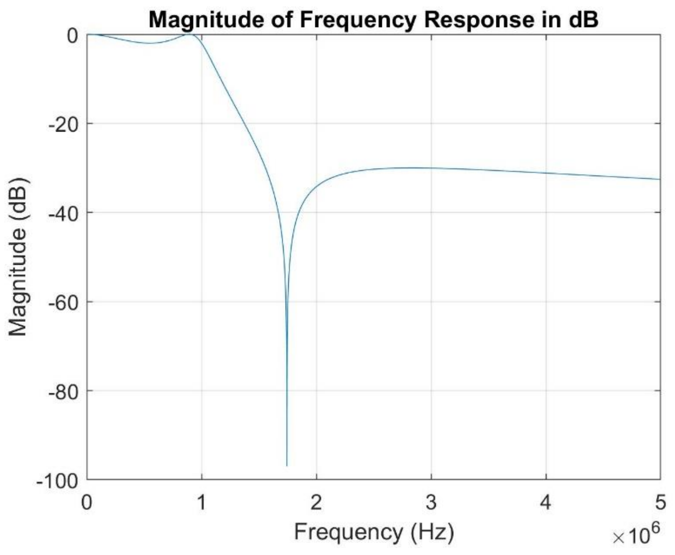

Figure 3: Magnitude of Frequency Response in dB.

Figure 3 demonstrates that the system is greatly attenuated between 1 to 2 MHz. This implies that the filter may be a low-pass or band-stop filter with a cut-off below 1 MHz. Next, the impulse response of the system was created over time.

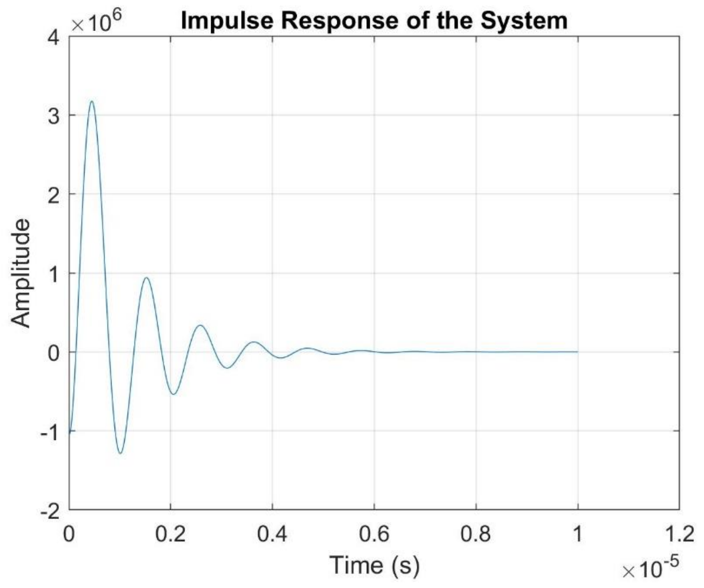

Figure 4: Impulse Response of the System over time.

Figure 4 shows that the impulse response of the system decreases in amplitude over time indicating that the system is stable, time-invariant, and demonstrates the systems tendency to the steady state value.

Convolving the sinusoidal signal x(t) with the impulse response demonstrates the system response at a specific input as seen in Figure 5. From the data below, the input signal is greatly amplified by the impulse response and then reaches steady-state. This demonstrates the transient behavior of the filter.

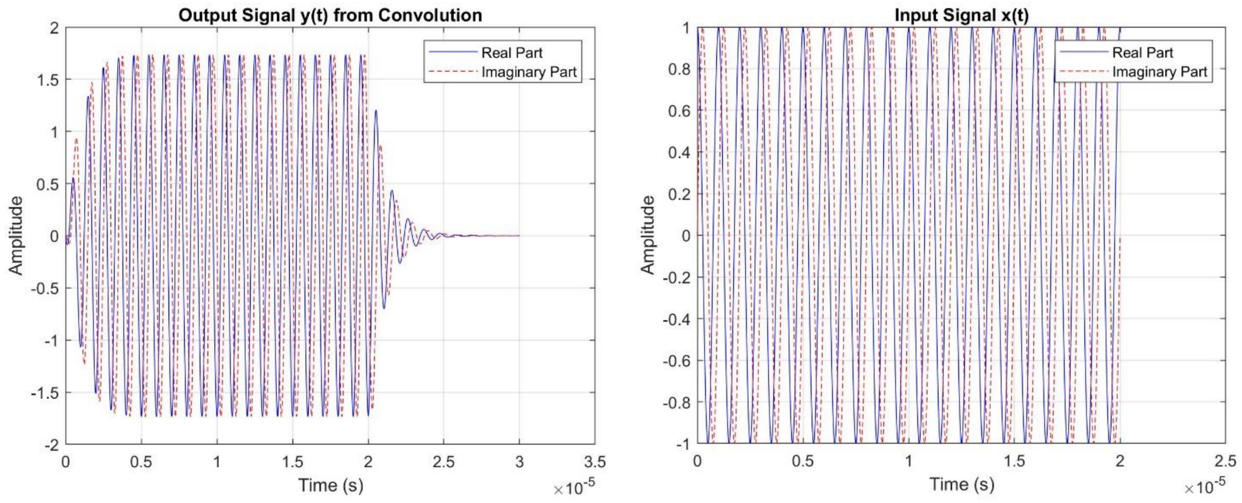

Figure 5: Input Sinusoidal signal and output after convolution with he impulse response.

Repeating the experiment at different frequencies reveals that the system is linear and time-invariant since the shape of the output is consistent across frequencies from 100kHz to 10MHz. The systems also reach steady state across frequencies. Lastly, the Output Signal Comparison in Figure 6 demonstrates that the outputs of the frequency response and the convolution have consistent phase and amplitude to the steady-state, supporting the theory that the steady-state response of a linear time-invariant system to a sinusoidal input can be described by the system's frequency response at the input signal.

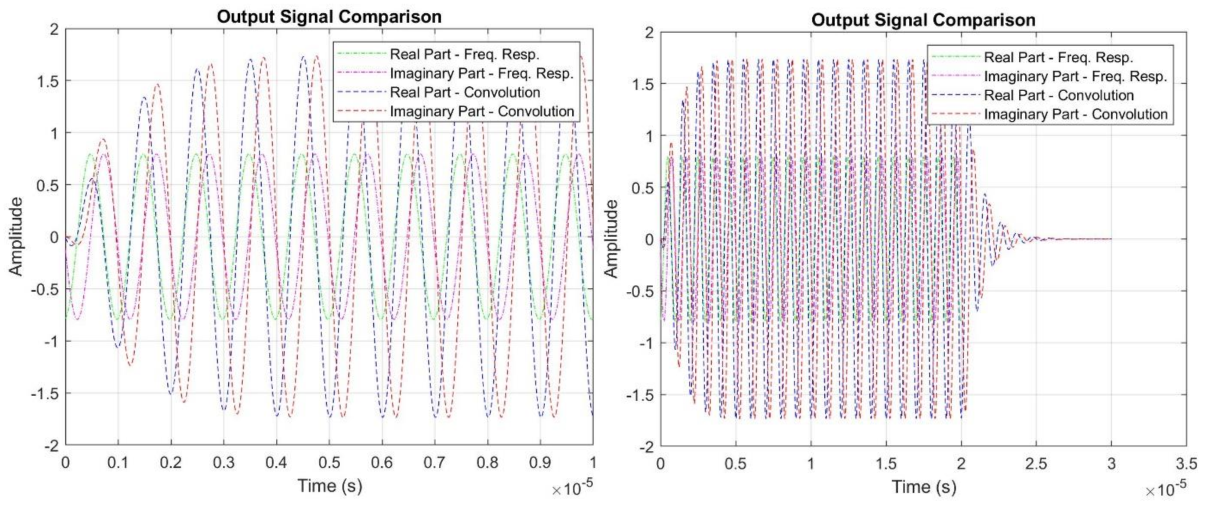

Figure 6: Comparison of Frequency Response and Convolution.

Conclusion:

The experiment successfully provides evidence that the steady-state response of a linear time-invariant system to a sinusoidal input can be described by the system's frequency response at the input signal. This was shown experimentally by working backwards, extracting the transfer function from poles and zero's the impulse response was calculated. With the impulse response, the system's response was simulated to various sinusoidal inputs. Convolving the sinusoidal inputs to the impulse response allowed for the attributes of the system to be categorized. These being that the system is linear and time-invariant.

Lastly, the comparison of the outputs from the frequency response multiplication and the convolution provided evidence supporting the theoretical prediction (Figure 1). The observed consistency in phase and amplitude in the steady-state across different frequencies reinforces the understanding of LTI systems and their behavior under sinusoidal inputs. Thus, the data supports the theory.

The signal processing knowledge explored in this experiment can be expanded upon to other areas of signal processing such as advanced filtering, real world applications, and noise reduction. The next logical step would be to test the systems with a non-sinusoidal signal and record how the system is effected.

# Code Appendix:

```
close all;

clc;

clear;


%https://www.mathworks.com/help/matlab/ref/colon.html?searchHighlight=%3A&s_tid=srchtitle_support_results_1_%253A


%Create symbolic scalar variables and functions, and matrix variables and

%functions:

%https://www.mathworks.com/help/symbolic/syms.html

syms s t

A0 = 816542.09;

%Zeros

z1 = 10^6 * (0 + 1i*10.9555);


%complex conjugate of z1

z2 = 10^6 * (0 - 1i*10.9555);


%Define the poles

p1 = 10^6 * (-0.93452 + 1i*5.96819);


%complex conjugate of p1

p2 = 10^6 * (-0.93452 - 1i*5.96819);

p3 = 10^6 * (-2.685583 + 1i*0);


%Computing the transfer function using inverse Laplace:


%Transfer function of H(s):

Hs = A0 * ((s - z1)*(s - z2)) / ((s - p1)*(s - p2)*(s - p3));


%Inverse Laplace transform to find h(t):

%https://www.mathworks.com/help/symbolic/sym.ilaplace.html

ht = ilaplace(Hs, s, t);


%Simplifying the result:

%https://www.mathworks.com/help/symbolic/simplify.html

htsimplified = simplify(ht);
```

```
%Displaying the result:

disp(htsimplified);


%Setting the simplified transfer function to a variable:

ht= htsimplified;


%Part One:




%One a): Frequency axis parameters spanning from min to max frequency:

fmin = 0; %Min frequency in Hz

fmax = 5e6; %Max frequency in Hz

freqSpace = 500; %Frequency spacing in Hz


%The frequency axis:

f = fmin:freqSpace:fmax; %Frequency values from fmin to fmax with spacing freqSpace


% Given poles and zeros:

p1 = 1e6 * (-0.93452 + 1i*5.96819);

%Since p2 is given as the complex conjugate of p1:

p2 = conj(p1);

p3 = 1e6 * (-2.685583 + 1i*0);

z1 = 1e6 * (0 + 1i*10.9555);

%z2 is the complex conjugate of z1:

z2 = conj(z1);




%One b): Evaluate the frequency response:

s = 1i*2*pi*f;

HF = A0 * ((s - z1).*(s - z2)) ./ ((s - p1).*(s - p2).*(s - p3));


%One c): Frequency response in dB:

HFdB = 20 * log10(abs(HF));




%One d): Plotting

figure;

plot(f, HFdB);
```

```
title('Magnitude of Frequency Response in dB');

xlabel('Frequency (Hz)');

ylabel('Magnitude (dB)');

grid on;


%Set the font size:

set(gca, 'fontsize', 12);


%One e): Save the plot to a jpeg:

print -djpeg95 'magnitudeoffrequencyresponse.jpg';




%Part Two:


%Time axis parameters

tmin = 0; %Min time in seconds

tmax = 10e-6; %Max time in seconds

timeSpace = 10e-9; %Time spacing in seconds


%Create the time axis:

t = tmin:timeSpace:tmax;


%Impluse Response from Inverse Laplace Transform:

h = (8924341383397411074527017375553*exp(-2685583*t))/3323060306750653050388848 - exp(t*(- 934520 -
5968190i))*(155273069383179795805434260535/166153015337532652519424 - 29021145762525979693258113795304065i/198326552921461800287980224512)
- exp(t*(- 934520 + 5968190i))*(155273069383179795805434260535/166153015337532652519424 +
29021145762525979693258113795304065i/198326552921461800287980224512);


%Sinusoidal input:

F0 = 10e6; %Frequency


%Sinusoidal input:

xt = sin(2*pi*F0*t);


%Convolution:

%https://www.mathworks.com/help/matlab/ref/conv.html?searchHighlight=conv&s_tid=srchtitle_support_results_1_conv

yt = conv(xt, h, 'same');


%Plot:

figure;
```

```
plot(t, yt);

title('Output Signal after Convolution');

xlabel('Time (s)');

ylabel('Amplitude');

grid on;


%Saving the plot:

print -dpng 'convolutionoutput.png';




%Poles and Zeros into column vectors:

p = [1e6*(-0.93452 + 1i*5.96819), 1e6*(-0.93452 - 1i*5.96819), 1e6*(-2.685583)];

z = [1e6*(1i*10.9555), 1e6*(-1i*10.9555)];

p = p(:);

z = z(:);


%Using given command to turn poles and zeros into a transfer function form:

%https://www.mathworks.com/help/matlab/ref/residue.html

[b, a] = zp2tf(z, p, A0);


%Partial fraction expansion:

[r, p, k] = residue(b, a);


%Time axis parameters

tmin = 0; %Mini time in seconds

tmax = 10e-6; %Max time in seconds

timeSpace = 10e-9; %Time spacing in seconds


%Time axis:

t = tmin:timeSpace:tmax;


%Input the impulse response into the system:

%https://www.mathworks.com/help/matlab/ref/zeros.html

ht = zeros(size(t));


%Calculate each term in the partial fraction expansion

for i = 1:length(r)

    if imag(p(i)) == 0

        ht = ht + r(i) * exp(p(i) * t);
    else

        ht = ht + 2 * real(r(i) * exp(p(i) * t));

    end

end


%Include only real parts:

%https://www.mathworks.com/help/matlab/ref/real.html?searchHighlight=real&s_tid=srchtitle_support_results_1_real

ht = real(ht);


%Plotting

figure;

plot(t, ht);

title('Impulse Response of the System');

xlabel('Time (s)');

ylabel('Amplitude');

grid on;


%Set the font size for readability

set(gca, 'fontsize', 12);


%Save the plot to a file

print -djpeg95 'impulse_response.jpg';


%Part Three:


%Frequency of the sinusoidal input

F0 = 10e6; % 1 MHz


%Time array expansion:

tth = tmax; %End time of the impulse response time array

textended = 0:timeSpace:2*tth; %Extended time array


%Input signal Calculation:

xt = exp(1i*2*pi*F0*textended);


%Plot of the real and imaginary parts of the input signal:

figure;

plot(textended, real(xt), 'b', textended, imag(xt), 'r--');

title('Input Signal x(t)');
```

```
xlabel('Time (s)');

ylabel('Amplitude');

legend('Real Part', 'Imaginary Part');

grid on;


hold off;


%Save the plot:

print -djpeg95 'input_signal.jpg';


%Convolution of the input signal with the impulse response:

yt = conv(xt, ht) * timeSpace;


%Adjusting time array for convolution size increase:

tconv = 0:timeSpace:(length(yt)-1)*timeSpace;


%Plot Real and Imaginary parts:

figure;

plot(tconv, real(yt), 'b', tconv, imag(yt), 'r--');

title('Output Signal y(t) from Convolution');

xlabel('Time (s)');

ylabel('Amplitude');

legend('Real Part', 'Imaginary Part');

grid on;


hold off;


%Save the plot:

print -djpeg95 'output_convolution.jpg';


%Frequency response at F0:

sF0 = 1i*2*pi*F0;

HF0 = A0 * ((sF0 - z1)*(sF0 - z2)) / ((sF0 - p1)*(sF0 - p2)*(sF0 - p3));


%Output signal from scalar multiplication:

ytfreqresp = xt * HF0;


%Plot the outputs:

figure;
```

```
plot(textended, real(ytfreqresp(1:length(textended))), 'g-.', ...
    textended, imag(ytfreqresp(1:length(textended))), 'm-.', ...
    tconv, real(yt), 'b--', ... %Add this line for the real part of convolution
    tconv, imag(yt), 'r--');    %Add this line for the imaginary part of convolution
title('Output Signal Comparison');
xlabel('Time (s)');
ylabel('Amplitude');
legend('Real Part - Freq. Resp.', 'Imaginary Part - Freq. Resp.', ...
      'Real Part - Convolution', 'Imaginary Part - Convolution');
  xlim([0,1e-5]);
grid on;

hold off;

%Save the plot:
print -djpeg95 'output_comparison.jpg';
```