Second Order Digital Filter Design Using Bilinear Transform

SHOHRUKH DADAKHON KHASAN – 19067905
EMİN EMİRHAN ŞENER – 20067040
RECEP YUSUF GÜLTAŞ – 1806A046
ENES EMEKSİZ – 19067015

MKT 4820 | Digital Signal Processing | Group - 1 | 21.05.2024

Table of Contents

LIST OF SYMBOLS

Ω_s = Stopband frequency in rad/sec

Ω_p= Passband frequency in rad/sec

Ω_c = Cutoff frequency in rad/sec

α_s = Stopband attenuation in dB

α_p = Passband attenuation in dB

f_c = Cutoff frequency in Hertz

ABBREVIATIONS

dB = Decibel

FIGURE LIST

INTRODUCTION

PROJECT OVERVIEW

Title: Design and Implementation of a Second Order Digital Filter Using Bilinear Transform

Objective: To design and implement a second-order digital filter using MATLAB and the bilinear transform method. This project will involve the design, simulation, and analysis of the filter's performance in the time and frequency domains, including manual calculations and drawings.

BACKGROUND

Digital signal processing (DSP) is a critical area in modern engineering, with applications ranging from audio and speech processing to telecommunications and control systems. Digital filters are fundamental components in DSP systems, used to manipulate or enhance signals in various ways. This project focuses on the design of a second-order digital filter, providing practical insights into theoretical concepts taught in DSP courses.

I. FILTER DESIGN

I.I FILTER SPECIFICATIONS

Desired filter specifications are given below.

	Filter Specifications
Type	Low Pass Filter
Cutoff frequency (wc)	500 Hz
Passband Ripple	1 DB
Transition width	100 Hz
Stopband Attenuation	40 db
Method	Butterworth
Order	2nd

I.II ANALOG FILTER TRANSFER FUNCTION

In here we derived the analog filter transfer function.

Firstly to visualize the filter specifications better we plotted the Bode plot by hand.

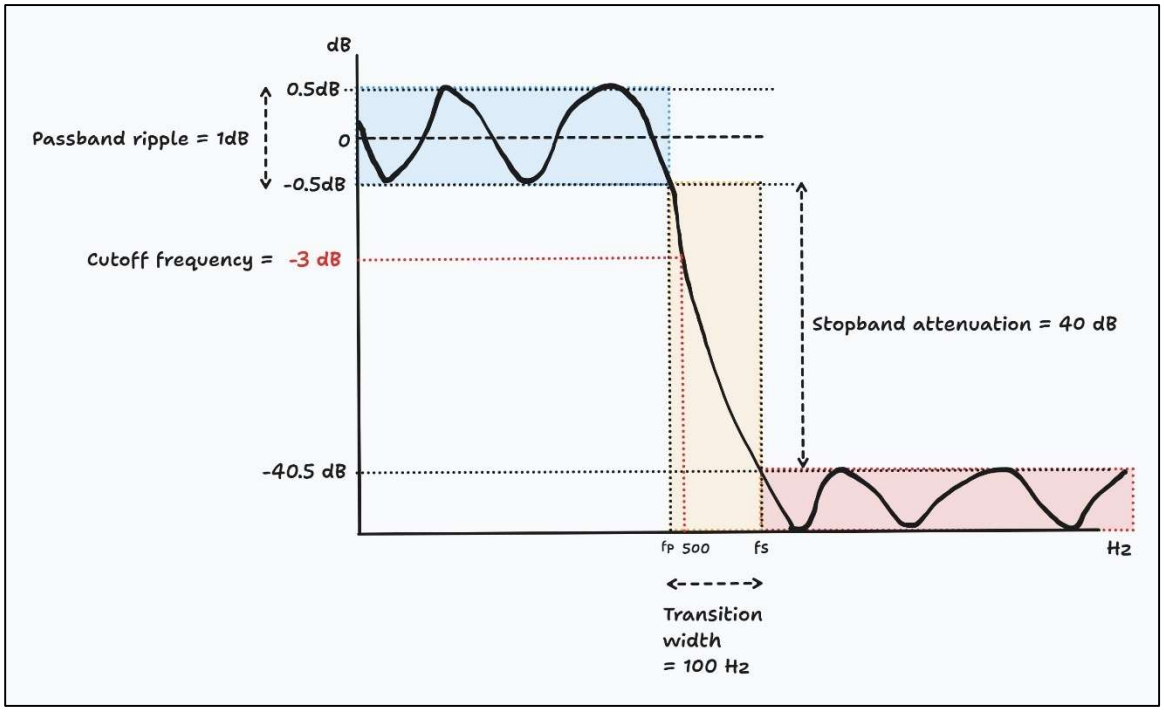


Figure 1: Hand drawn Bode plot

$$\Omega_s = Stopband\ frequency\ in\ \frac{rad}{sec} = 2 * \pi * f_s$$

$$\Omega_p = Passband\ frequency\ in\ \frac{rad}{sec} = 2 * \pi * f_p$$

$$\alpha_s = Stopband\ attenuation\ in\ dB = 40\ dB$$

$$\alpha_p = Passband\ attenuation\ in\ dB = Passband\ ripple = 1\ dB$$

We can derive the second order transfer function as below.

$$H(s) = \frac{{\omega_c}^2}{s^2 + 2 * \zeta * \omega_c * s + {\omega_c}^2}$$

$$\omega_c = 2 * \pi * f_c$$

$$H(s) = \frac{(2 * \pi * f_c)^2}{s^2 + 2 * \zeta * (2 * \pi * f_c) * s + (2 * \pi * f_c)^2}$$

We found the continuous transfer function. But we need to get the digital transfer function. For this we will use Bilinear transform method.

I.III CONVERSION OF ANALOG FILTER TO DIGITAL FILTER

We applied the bilinear transform formula to convert the analog filter to a digital filter.

Suppose that we have,

$$y(t) = \int_{t_0}^{t} x(\tau) * d\tau = y(t - T) + \int_{t-T}^{t} x(\tau) * d\tau$$

Using Trapezoidal rule, this integral can be written as,

$$y(t) \approx y(t - T) + \frac{T}{2} * \big(x(t) + x(t - T)\big)$$

Or in discrete domain,

$$y[n] = y[n - 1] + \frac{T}{2} * (x[n] + x[n - 1])$$

$$\mathrm{H(z)} = \frac{Y(z)}{X(z)} = \frac{T}{2} * \frac{z+1}{z-1}$$

$$\frac{1}{s} = \frac{T}{2} * \frac{z + 1}{z - 1}$$

$$s = \frac{2}{T} * \frac{z - 1}{z + 1}$$

Applying bilinear transformation we get our filter representation in discrete domain.

$$s = \frac{2}{T} * \frac{1 - z^{-1}}{1 + z^{-1}}$$

$$H_d(z) = \frac{{\omega_c}^2}{\left(\frac{2}{T} * \frac{1 - z^{-1}}{1 + z^{-1}}\right)^2 + 2 * \zeta * \omega_c * \left(\frac{2}{T} * \frac{1 - z^{-1}}{1 + z^{-1}}\right) + (\omega_c)^2}$$

Prewarping cutoff can be defined to satisfy cutoff (non-linear to linear) as below.

$$\acute{\omega_c} = \frac{2}{T} * \tan\left(\frac{\omega_c * T}{2}\right)$$

$$H_d(z) = \frac{(\acute{\omega_c})^2}{\left(\frac{2}{T} * \frac{1 - z^{-1}}{1 + z^{-1}}\right)^2 + 2 * \zeta * \acute{\omega_c} * \left(\frac{2}{T} * \frac{1 - z^{-1}}{1 + z^{-1}}\right) + (\acute{\omega_c})^2}$$

$$\frac{Y(z)}{X(z)} =$$

$$\frac{(\acute{\omega_c})^2 * (1 + z^{-1})^2}{\left(\frac{2}{T} * (1 - z^{-1})\right)^2 + 2 * \zeta * \acute{\omega_c} * \left(\frac{2}{T} * (1 - z^{-1}) * (1 + z^{-1})\right) + (\acute{\omega_c})^2 * (1 + z^{-1})^2}$$

$$D = \frac{4}{T^2} * (1 - 2 * z^{-1} + z^{-2}) + \frac{4}{T} * \zeta * \acute{\omega_c} - \frac{4}{T} * \zeta * \acute{\omega_c} * z^{-2} + {\acute{\omega_c}}^2 (1 + 2 * z^{-1} + z^{-2})$$

$$D = \left[\frac{4}{T^2} + \frac{4}{T} * \zeta * \acute{\omega_c} + {\acute{\omega_c}}^2\right] + \left[2 * {\acute{\omega_c}}^2 - \frac{8}{T^2}\right] * z^{-1} + \left[{\acute{\omega_c}}^2 + \frac{4}{T^2} - \frac{4}{T} * \zeta * \acute{\omega_c}\right] * z^{-2}$$

$$N = \acute{\omega_c}^2 + 2 * \acute{\omega_c}^2 * z^{-1} + \acute{\omega_c}^2 * z^{-2}$$

$$\frac{N}{D} = \frac{Y(z)}{X(z)} = a_0 * y[n] + a_1 * y[n-1] + a_2 * y[n-2] =$$

$$b_0 * x[n] + b_1 * x[n-1] + b_2 * x[n-2]$$

$$y[n] = b_0 * x[n] + b_1 * x[n-1] + b_2 * x[n-2] - a_1 * y[n-1] - a_2 * y[n-2]$$

This is the respresentation of the second order low pass filter respresentation of our design in discrete domain.

I.IV FILTER COEFFICIENTS

We have calculated the filter coefficients using MATLAB functions.

```
%% calculating coefficients

w_c = 2*pi*f_c;
w_c_prime = (2/T)*tan(w_c*(T/2));

a0 = (4/(T^2)) + ((4/T)*zeta*w_c_prime) + (w_c_prime^2);
a1 = 2*(w_c_prime^2) - (8/(T^2));
a1 = a1/a0;
a2 = w_c_prime^2 + (4/(T^2)) - ((4/T)*zeta*w_c_prime);
a2 = a2/a0;

b0 = (w_c_prime^2);
b0 = b0/a0;
b1 = (2*(w_c_prime^2));
b1 = b1/a0;
b2 = w_c_prime^2;
b2 = b2/a0;

a0 = a0/a0;
```

Calculated coefficients using MATLAB functions are as below.

$$a_0 = 1.0000, \quad a_1 = -0.3710, \quad a_2 = 0.2007$$

$$b_0 = 0.2074, \quad b_1 = 0.4148, \quad b_2 = 0.2074$$

II. MATLAB SCRIPT DEVELOPMENT

II.I ANALOG FILTER DESIGN WITH MATLAB

In this section we wrote a MATLAB script to design the analog filter in continuous domain.

```
f_c = 500;
zeta = 0.7;
omega_c = 2*pi*f_c;

f = logspace(2,4,400); % 100 to 10 kHz
omega = 2*pi*f;

s = 1i*omega;


H = (omega_c.^2)./(s.^2 + 2.*zeta*omega_c*s + omega_c.^2);

% calculate magnitude and phase
magnitude = 20*log10(abs(H));
phase = angle(H) * (180/pi);

% plot the magnitude
subplot(2,1,1);
semilogx(f, magnitude)
grid on;
title('Bode plot of H(s) = 1/(s^2 + 2*zeta*w_c*s + w_c^2)');
xlabel('Frequency (Hz)')
ylabel('Magnitude (dB)')

subplot(2,1,2);
semilogx(f, phase)
grid on;
xlabel('Frequency (Hz)')
ylabel('Phase (degree)')
```

II.II CONVERSION OF ANALOG FILTER TO DIGITAL FILTER WITH MATLAB

We used the bilinear function to transform the analog filter to a digital filter using MATLAB.

```
f_c = 500; % cutoff frequency
f_s = 2500; % sampling frequency
zeta = 0.7;

frequencies = [1, 450, 800]; % frequency of applied sinusoidal signal

A = 1;
phi = 0;
T = 1/f_s;
N = f_s*1;

%% calculating coefficients
w_c = 2*pi*f_c;
w_c_prime = (2/T)*tan(w_c*(T/2));

a0 = (4/(T^2)) + ((4/T)*zeta*w_c_prime) + (w_c_prime^2);
a1 = 2*(w_c_prime^2) - (8/(T^2));
a1 = a1/a0;
a2 = w_c_prime^2 + (4/(T^2)) - ((4/T)*zeta*w_c_prime);
a2 = a2/a0;

b0 = (w_c_prime^2);
b0 = b0/a0;
b1 = (2*(w_c_prime^2));
b1 = b1/a0;
b2 = w_c_prime^2;
b2 = b2/a0;

a0 = a0/a0;

%% input signal
n = 0:1:N-1;

figure

for i=1:length(frequencies)
    f = frequencies(i);
```

```
    x = A*sin(2*pi*f*n*T + phi);

    y = zeros(N, 1);

    for j=3:N
        y(j) = b0*x(j) + b1*x(j-1) + b2*x(j-2) - a1*y(j-1) - a2*y(j-2);
    end

    subplot(length(frequencies),1,i);
    plot(n*T, x, 'b', n*T, y, 'r')
    title(sprintf('%d Hz input and Filtered Output Signals', f));
    xlabel('Time');
    ylabel('Amplitude');
    legend('Input Signal', 'Filtered Signal', 'Location','best');
    grid on;
end

a = [a0 a1 a2];
b = [b0 b1 b2];
[H,f] = freqz(b, a, [], f_s);

figure
plot(f, 20*log10(abs(H)));
title('Freq-Mag Response of Discrete Filter Design');
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
grid on;
```

II.III APPLICATION OF DIGITAL FILTER TO A GIVEN SIGNAL

We created a function to apply the digital filter to a given signal using MATLAB.

```
% We defined the Butterworth filter application function as below
function y = applyButterworthFilter(x, fs, fc, zeta)
    % Calculating the normalized cutoff frequency
    T = 1/fs;
    omega_c = 2 * pi * fc;
    omega_c_prime = (2 / T) * tan(omega_c * (T / 2));

    % Calculating filter coefficients
    a0 = (4 / (T^2)) + (4 * zeta * omega_c_prime / T) + omega_c_prime^2;
```

```
    a1 = (2 * omega_c_prime^2) - (8 / (T^2));
    a2 = (4 / (T^2)) - (4 * zeta * omega_c_prime / T) + omega_c_prime^2;

    b0 = omega_c_prime^2;
    b1 = 2 * omega_c_prime^2;
    b2 = omega_c_prime^2;

    a1 = a1 / a0;
    a2 = a2 / a0;
    b0 = b0 / a0;
    b1 = b1 / a0;
    b2 = b2 / a0;
    a0 = a0 / a0;  % normalize a0 to 1

    % Applying filter to the input signal
    N = length(x);
    y = zeros(N, 1);
    for j = 3:N
        y(j) = b0*x(j) + b1*x(j-1) + b2*x(j-2) - a1*y(j-1) - a2*y(j-2);
    end
end
```

III. SIMULATION AND ANALYSIS

III.I TEST SIGNAL GENERATION

We generated test signals on 3 different frequencies. The frequencies are 1 Hz, 450 Hz and 800 Hz.

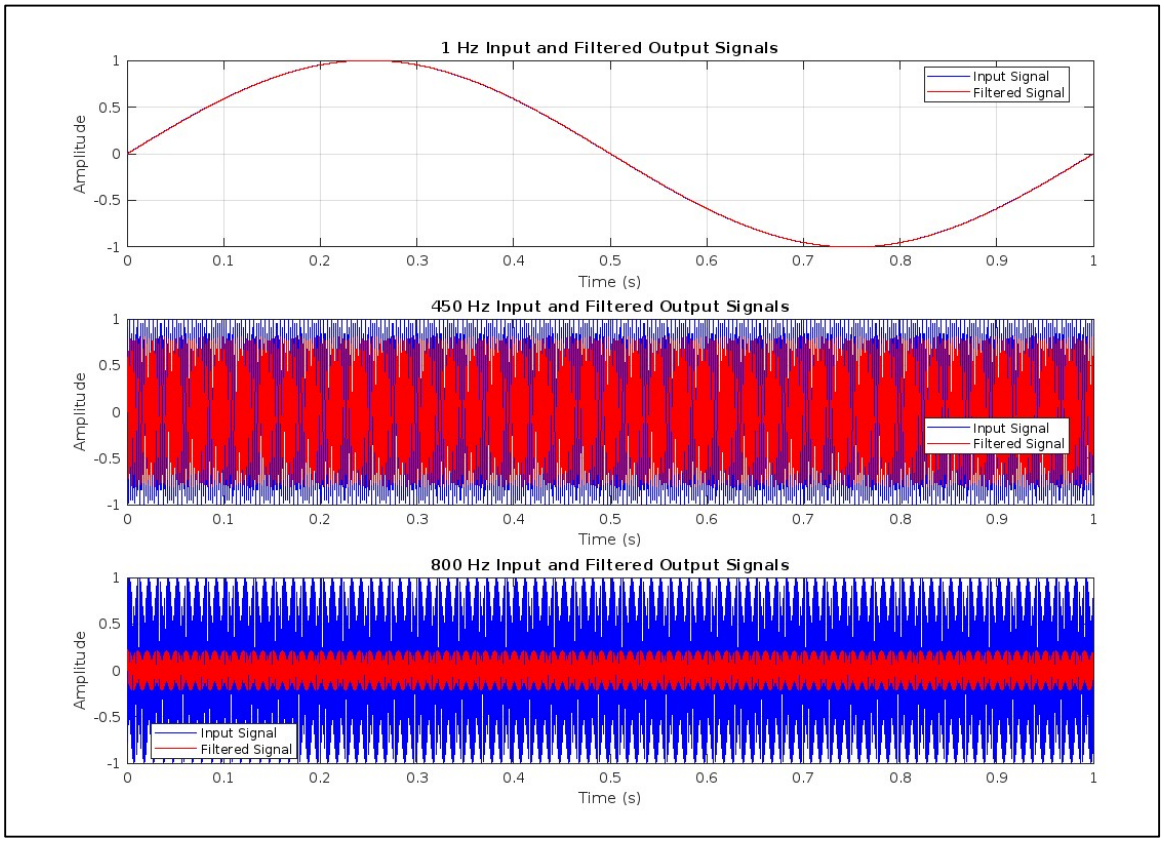


Figure 2: Generated test signals

III.II FILTER'S TIME-DOMAIN RESPONSE

Instruction: Calculate the filter's time-domain response to the test signal. (BY HAND)

Transfer function,

$$H(s) = \frac{{\omega_c}^2}{s^2 + 2 * \zeta * \omega_c * s + {\omega_c}^2}$$

Sinusoidal input,

$$I = \sin(\omega * T)$$

$$I(s) = \frac{\omega}{(s^2 + \omega^2)}$$

Taking the inverse Laplace transformation to obtain the time domain response,

$$\mathcal{L}^{-1}\left\{\frac{{\omega_c}^2}{s^2 + 2 * \zeta * \omega_c * s + {\omega_c}^2} * \frac{\omega}{(s^2 + \omega^2)}\right\}$$

III.III FREQUENCY RESPONSE OF THE FILTER

Instruction: Derive the frequency response of the filter (BY HAND)

$$H(s) = \frac{{\omega_c}^2}{s^2 + 2 * \zeta * \omega_c * s + {\omega_c}^2}$$

Substituting $s = j\omega$

$$H(j\omega) = \frac{(2 * \pi * 500)^2}{(j\omega)^2 + 2 * 0.7 * (2 * \pi * 500) * (j\omega) + (2 * \pi * 500)^2}$$

Magnitude response;

$$|H(j\omega)| = \sqrt{Re\big(H(j\omega)\big)^2 + Im\big(H(j\omega)\big)^2}$$

Phase response;

$$\angle H(j\omega) = \tan^{-1}\left(\frac{Im\big(H(j\omega)\big)}{Re\big(H(j\omega)\big)}\right)$$

III.IV APPLICATION OF DIGITAL FILTER TO THE TEST SIGNAL

We applied the digital filter to the test signals we previously created. The function we created previously named “applyButterworthFilter” is used.

Here is the MATLAB code where we called the applyButterworthFilter function:

```
% Example signal parameters
fs = 2500; % Sampling frequency
fc = 500;  % Cutoff frequency
zeta = 0.7; % Damping factor
frequencies = [1, 450, 800]; % Frequencies of applied sinusoidal signals
A = 1;
phi = 0;
T = 1/fs;
N = fs * 1; % 1 second of data

% Time vector
n = 0:1:N-1;

figure;
for i = 1:length(frequencies)
    f = frequencies(i);
    x = A * sin(2 * pi * f * n * T + phi);
    y = applyButterworthFilter(x, fs, fc, zeta);

    subplot(length(frequencies), 1, i);
    plot(n*T, x, 'b', n*T, y, 'r');
    title(sprintf('%d Hz Input and Filtered Output Signals', f));
    xlabel('Time (s)');
    ylabel('Amplitude');
    legend('Input Signal', 'Filtered Signal', 'Location', 'best');
    grid on;
end
```

III.V TIME DOMAIN RESPONSE OF THE FILTERED SIGNAL

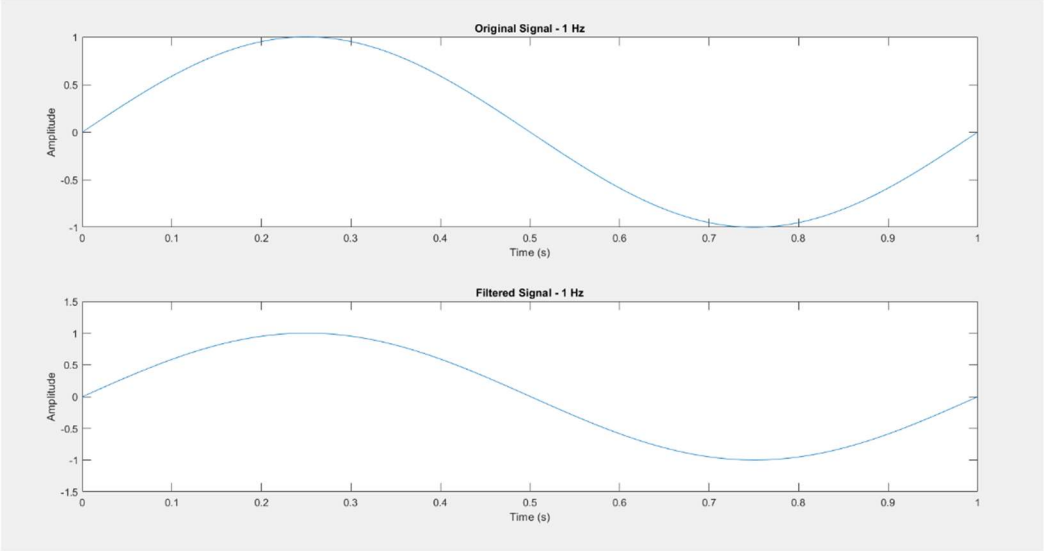


Figure 3: Time-domain response of the filtered signal with frequency of 1 Hz

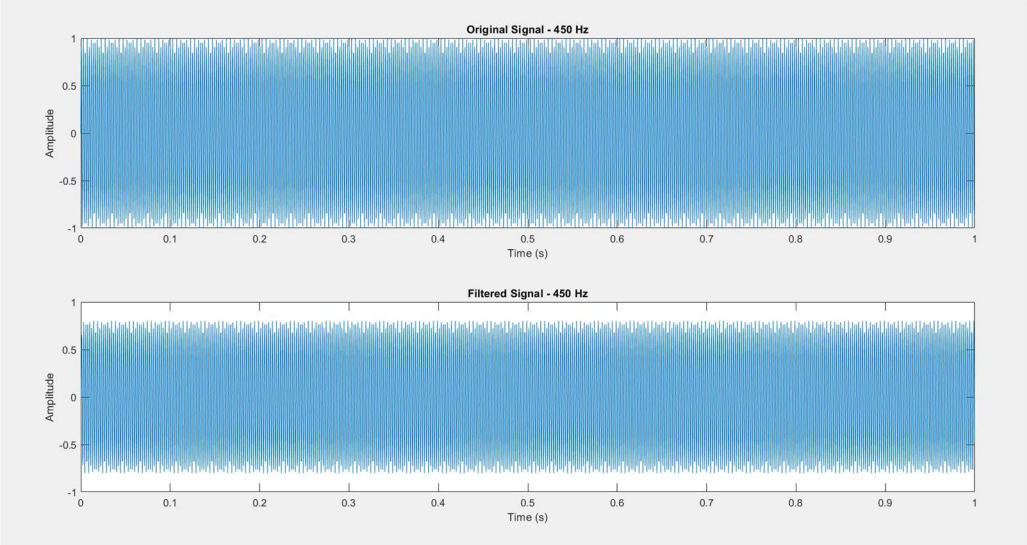


Figure 4: Time-domain response of the filtered signal with frequency of 450 Hz

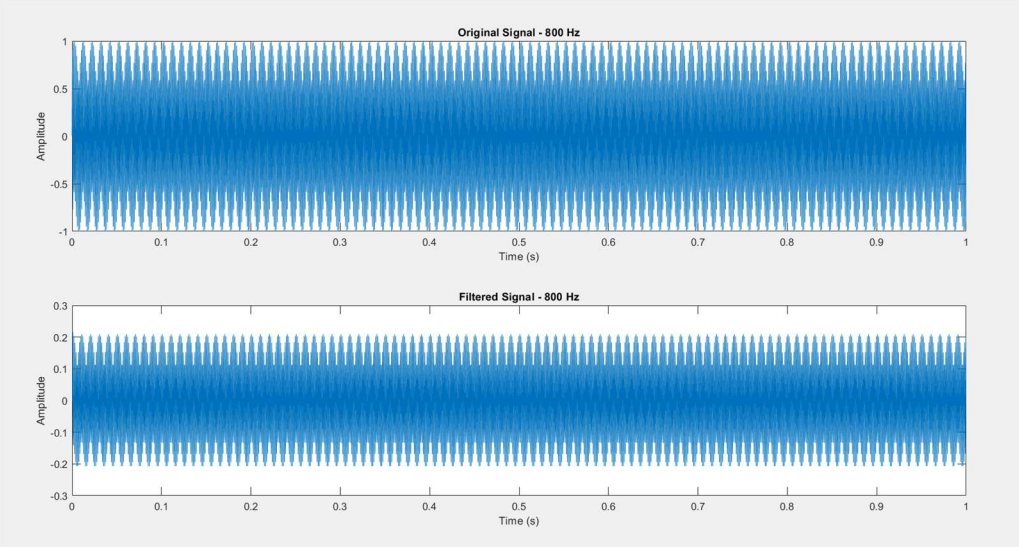


Figure 5: Time-domain response of the filtered signal with frequency of 800 Hz

III.VI FREQUENCY RESPONSE OF THE FILTER

We plotted the frequency response of the signal using the freqz function of MATLAB.

```
% Plotting the frequency response of the filter
a = [a0 a1 a2];
b = [b0 b1 b2];
[H, f] = freqz(b, a, [], f_s);

figure
plot(f, 20 * log10(abs(H)));
title('Frequency-Magnitude Response of Discrete Filter Design');
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
grid on;
```

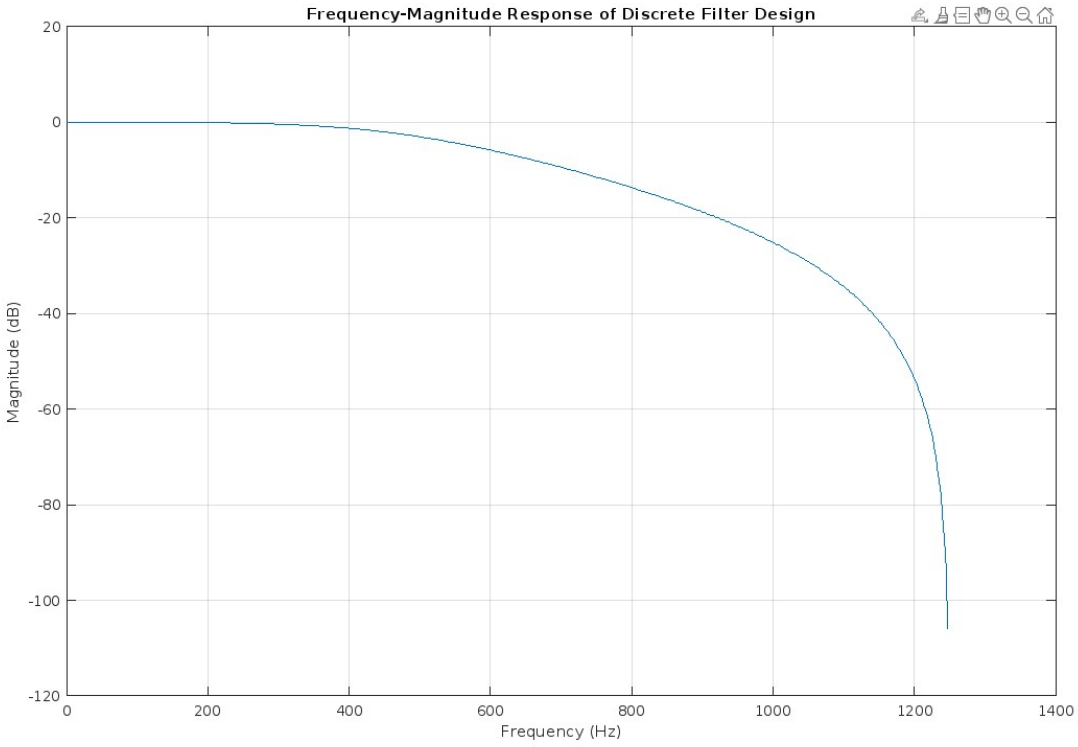


Figure 6: Frequency response of the filter

IV. PERFORMANCE EVALUATION

IV.I HOW DOES THE FILTER AFFECT THE SIGNAL?

On Figure 3, we can see the original test signal with frequency 1 Hz and the filtered version of that signal. As expected, since 1Hz is well below the 500 Hz Cutoff frequency, the change is minimal and insignificant.

On Figure 4, the test signal on frequency 450 Hz is presented with the filtered version of it. We observed a slight attenuation, but most of the signal passes through, indicating proximity to the Cutoff frequency.

Lastly, on Figure 5, we can observe the test signal with frequency 800 Hz along the filtered version of that test signal. A significant amount of attenuation is present, since 800 Hz is well above the 500 Hz Cutoff frequency which means we have implemented an effective filtering.

IV.II EXAMINING THE FREQUENCY RESPONSE

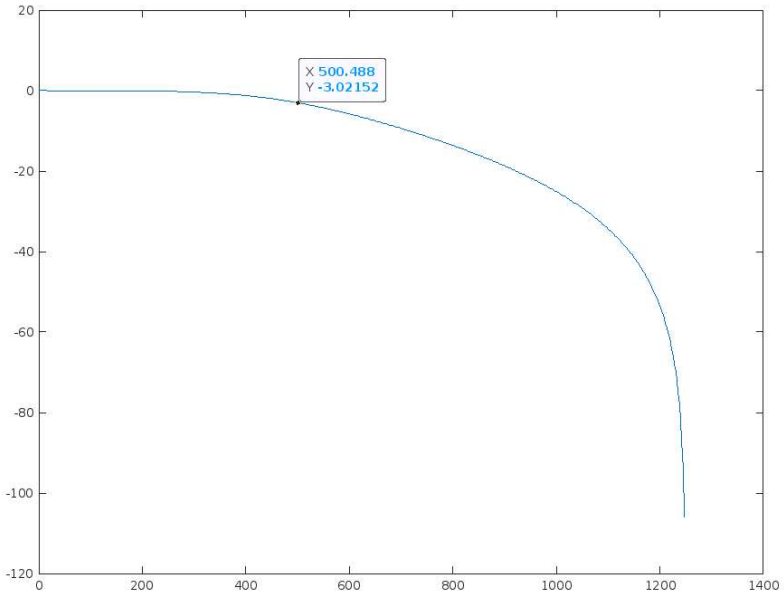


Figure 7: dB at Cutoff frequency

```
figure
y_cutoff = -3;
x_cutoff = interp1(20*log10(abs(H)), f, y_cutoff, "linear");

y_stopband = -40;
x_stopband = interp1(20*log10(abs(H)), f, y_stopband, "linear");

y_passband = -0.5;
x_passband = interp1(20*log10(abs(H)), f, y_passband, "linear");

disp(x_cutoff);
disp(x_stopband);
disp(x_passband);

plot(f, 20*log10(abs(H)));
title('Freq-Mag Response of Discrete Filter Design');
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
grid on;
```

Using the MATLAB code above we examined the transition width, cutoff frequency and stopband frequency values. Here are the results:

IV.III KEY PERFORMANCE METRICS

Cutoff frequency @ -3 dB = 499.5477 Hz

Stopband frequency @ -40 dB attenuation = 1141.1 Hz

Passband frequency @ -0.5 dB = 322.7583 Hz

IV.III BODE PLOT, PASSBAND RIPPLE, TRANSITION WIDTH, AND STOPBAND RIPPLE

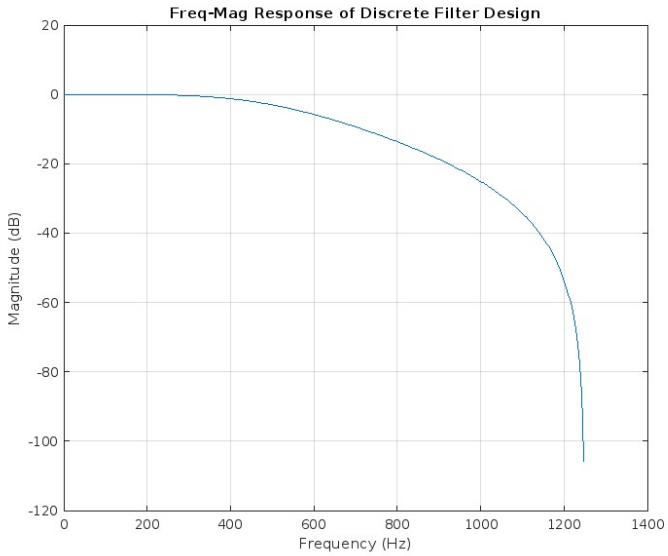


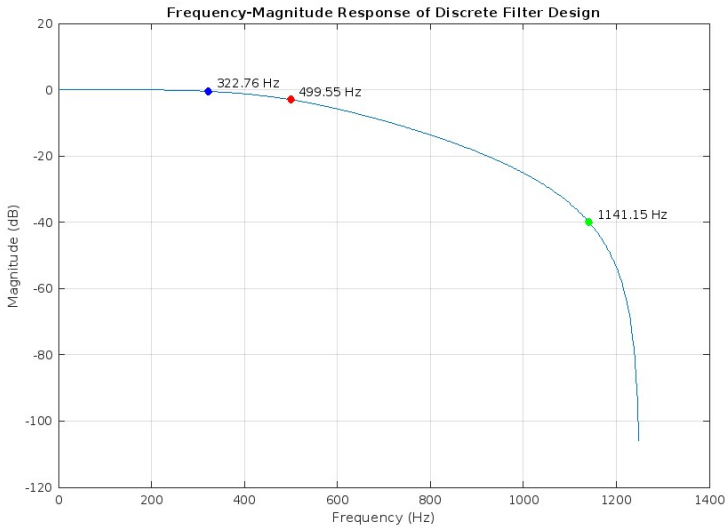
Figure 8: Bode plot

Passband ripple and Stopband ripple cannot be calculated since in our simulation environment Butterworth filter acts as an ideal low pass filter and it does not have a ripple as it would have in a real world filter system.

$$Transition\ width = Stopband\ frequency - Passband\ frequency$$

$$Transition\ width = 1141.1 - 322.758 = 818.342\ \text{Hz}$$

IV.IV KEY PERFORMANCE METRICS WITH MATLAB



Cutoff frequency at -3 dB: ***499.5477 Hz***

Stopband frequency at -40 dB: ***1141.1457 Hz***

Passband frequency at -0.5 dB: ***322.7583 Hz***

IV.V BODE PLOT WITH MATLAB

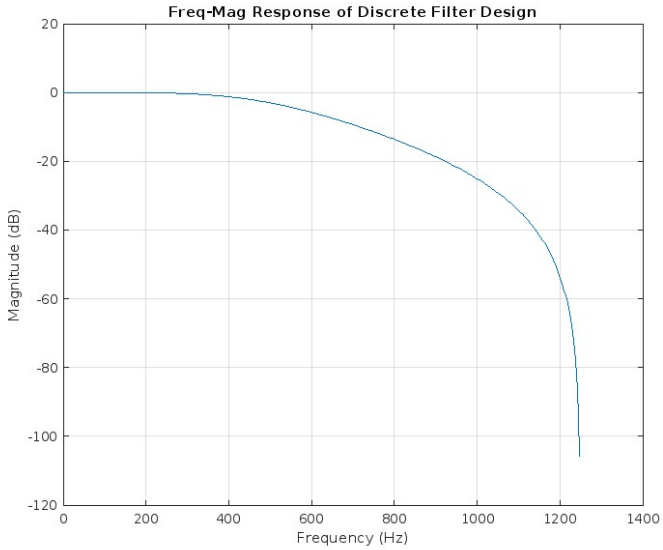
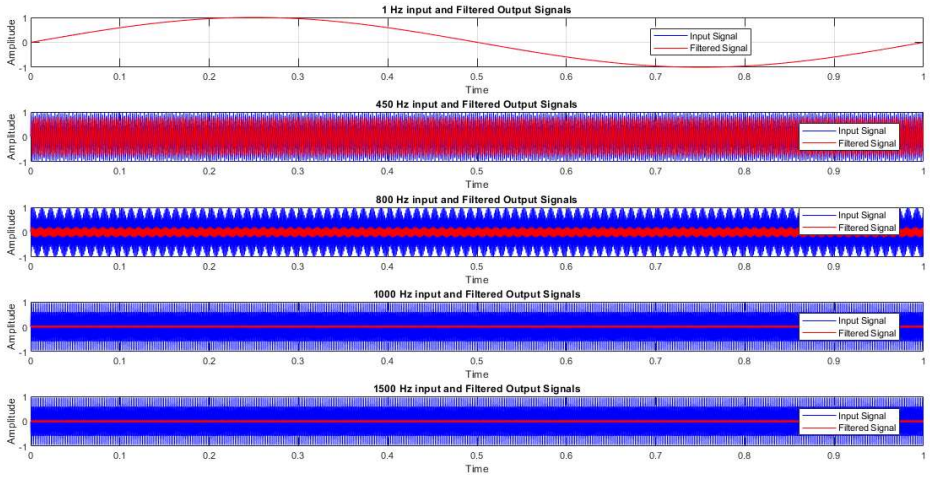


Figure 9: Bode plot

V. RESULT AND DISCUSSION

V.I PLOTS



VI. CONCLUSION

Given transition width of 100 Hz is too small for these specifications of Butterworth Low Pass Filter.

Potential Areas for Improvement:

Order of the Filter: Increasing the order could result in a steeper roll-off, improving attenuation in the stopband at the cost of potential increased complexity and computational requirements.

Passband Ripple: Though minimal in Butterworth filters, if specific applications require even flatter passband characteristics, considering filters like Chebyshev or Elliptic might be beneficial.

Phase Response: If phase linearity is critical, exploring filters with linear phase characteristics, such as Bessel filters, could enhance performance.