Table of Contents

.... 1
3-2-a 1
3.2.b 4
3.2.c 6
3.2.d 8
3.2.f-g 12


```
close all;
clear;
clc;
```

H1:

```
b1 = [0.969531, -1.923772, 0.969531];
a1 = [1, -1.923772, 0.939063];

% H2:
b2 = [0.996088, -1.976468, 0.996088];
a2 = [1, -1.976468, 0.992177];

fs = 400;
```

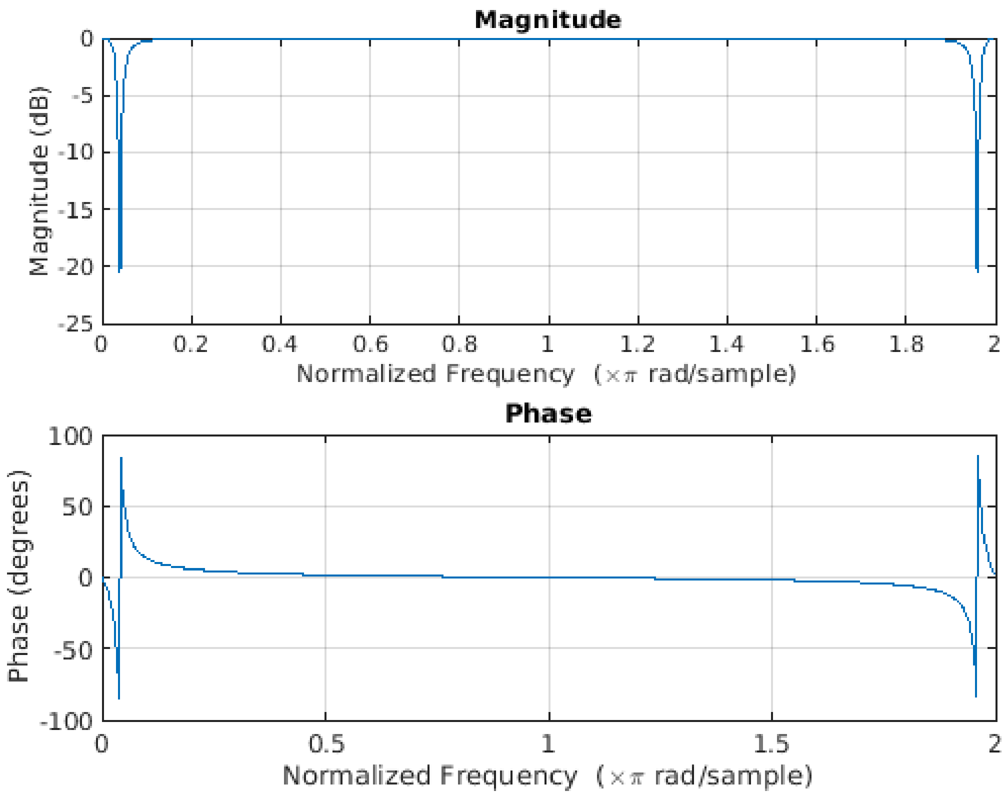
3-2-a

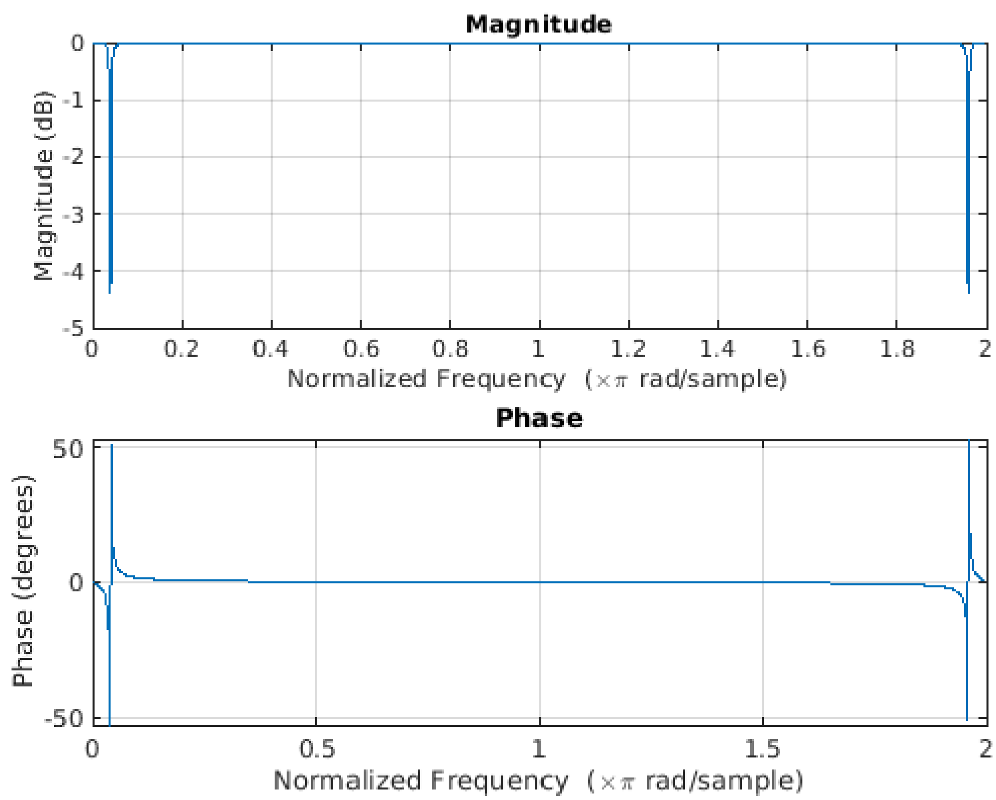
H1:

```
figure('name', 'H1 response')
freqz(b1,a1,1024,'whole');
% H2:
figure('name', 'H2 response')
freqz(b2,a2,1024,'whole');

% Canonical Presentation
csys1 = canon(filt(b1 , a1),'companion')
csys2 = canon(filt(b2 , a2),'companion')
```

```
csys1 =

  A =
             x1        x2
   x1         0   -0.9391
   x2         1     1.924

  B =
        u1
   x1    1
   x2    0

  C =
```

```
             x1         x2
   y1  -0.05862  -0.05368

  D =
           u1
   y1  0.9695

Sample time: unspecified
Discrete-time state-space model.

csys2 =

  A =
            x1       x2
   x1        0  -0.9922
   x2        1    1.976

  B =
       u1
   x1   1
   x2   0

  C =
              x1          x2
   y1  -0.007732   -0.00749

  D =
           u1
   y1  0.9961

Sample time: unspecified
Discrete-time state-space model.
```

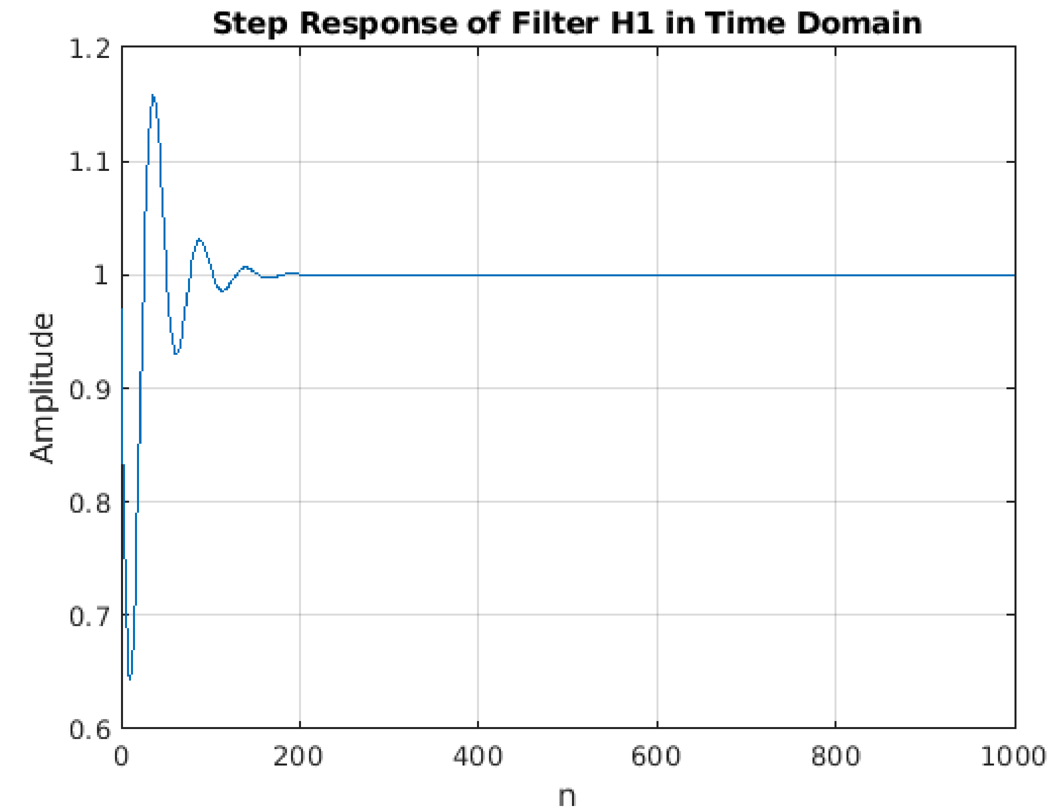

Magnitude
Magnitude (dB)
0
-5
-10
-15
-20
-25
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Normalized Frequency (×π rad/sample)
Phase
Phase (degrees)
100
50
0
-50
-100
0
0.5
1
1.5
2
Normalized Frequency (×π rad/sample)

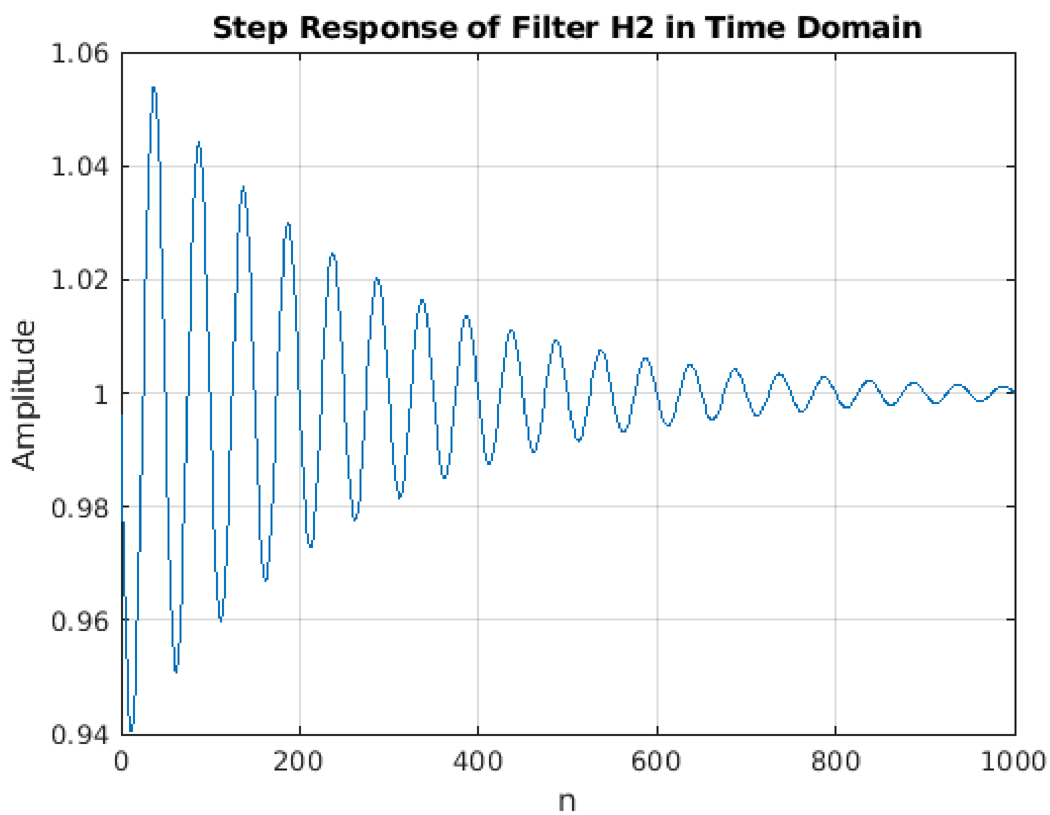

Magnitude
Magnitude (dB)
0
-1
-2
-3
-4
-5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Normalized Frequency (×π rad/sample)
Phase
Phase (degrees)
50
0
-50
0
0.5
1
1.5
2
Normalized Frequency (×π rad/sample)

3.2.b

```
N = 10e4;
n = 0:N;
x_step = ones(1, N + 1);

% H1
y_step = filter(b1, a1, x_step);

figure('Name', 'Step Response');
plot(n, y_step, "LineWidth", 1.5);
title("Step Response of Filter H1 in Time Domain");
xlabel("n");
ylabel("Amplitude");
xlim([0 1000]);
grid on;

settling_time_value = find(abs(y_step - y_step(end)) >= 0.01, 1, 'last') + 1;
fprintf("Setteling sample (H1): %d\n", settling_time_value);

% H2
y_step = filter(b2, a2, x_step);

figure('Name', 'Step Response');
plot(n, y_step, "LineWidth", 1.5);
title("Step Response of Filter H2 in Time Domain");
xlabel("n");
ylabel("Amplitude");
xlim([0 1000]);
grid on;

settling_time_value = find(abs(y_step - y_step(end)) >= 0.01, 1, 'last') + 1;
fprintf("Setteling sample (H2): %d\n", settling_time_value);
```

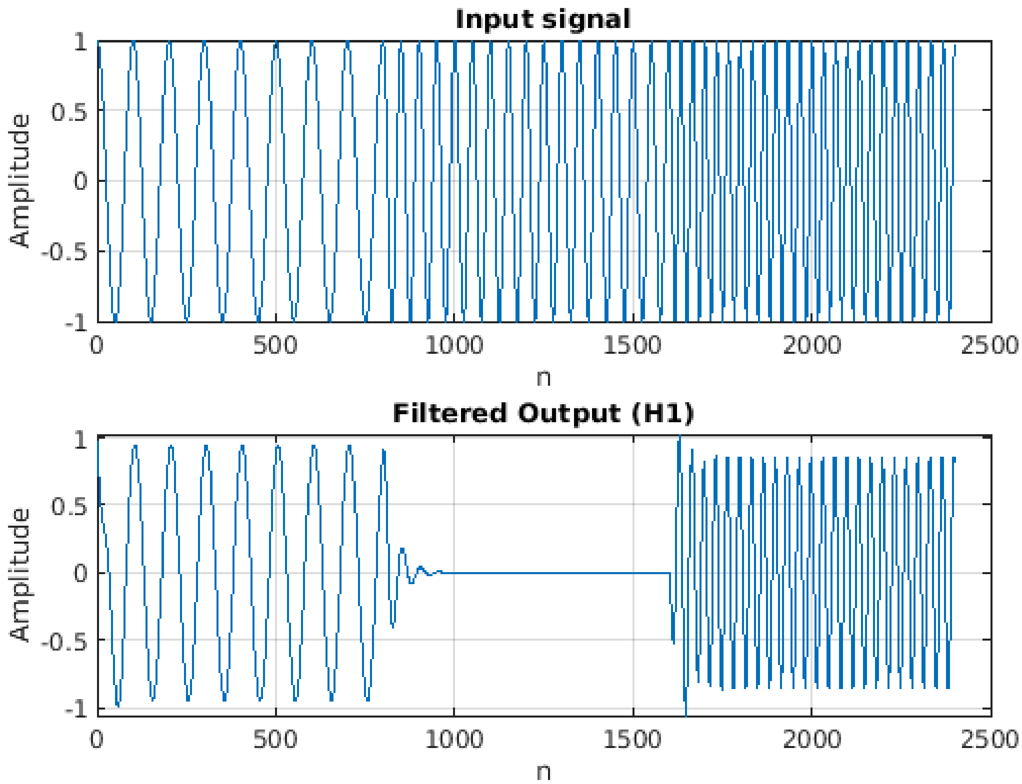
```
Setteling sample (H1): 122
Setteling sample (H2): 465
```

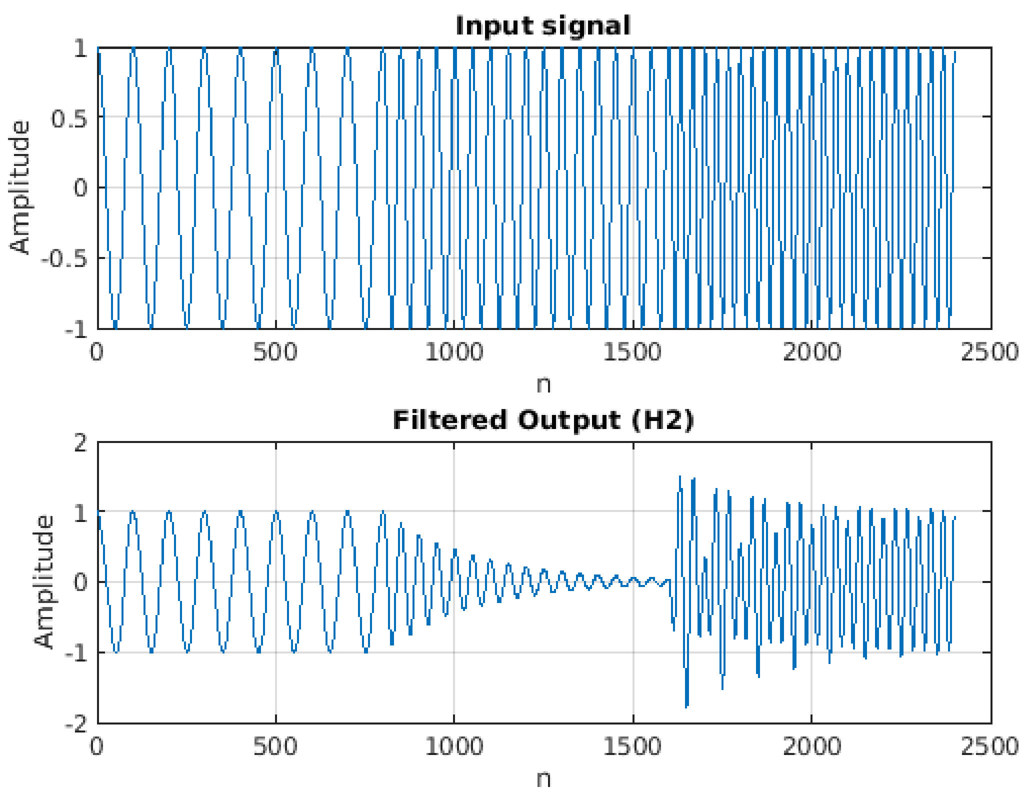

Step Response of Filter H1 in Time Domain
Amplitude
1.2
1.1
1
0.9
0.8
0.7
0.6
0
200
400
600
800
1000
n


Step Response of Filter H2 in Time Domain
Amplitude
1.06
1.04
1.02
1
0.98
0.96
0.94
0
200
400
600
800
1000
n

3.2.c

```
f = [4, 8 ,12];
fs = 400;

t1 = 0:(1/fs):(2-1/fs);
t2 = 2:(1/fs):(4-1/fs);
t3 = 4:(1/fs):(6-1/fs);
x = [cos(2*pi*f(1)*t1),cos(2*pi*f(2)*t2),cos(2*pi*f(3)*t3)];

% H1:
figure('Name', 'H1');
subplot(2,1,1);
plot(x, "LineWidth", 1.5);
title("Input signal");
xlabel("n");
ylabel("Amplitude");
grid on;

y1 = filter(b1, a1, x);
subplot(2,1,2);
plot(y1, "LineWidth", 1.5);
title("Filtered Output (H1)");
xlabel("n");
ylabel("Amplitude");
grid on;

% H2:
figure('Name', 'H2');
subplot(2,1,1);
plot(x, "LineWidth", 1.5);
title("Input signal");
xlabel("n");
ylabel("Amplitude");
grid on;

y2 = filter(b2, a2, x);
subplot(2,1,2);
plot(y2, "LineWidth", 1.5);
title("Filtered Output (H2)");
xlabel("n");
ylabel("Amplitude");
grid on;
```


Input signal
Amplitude
n
Filtered Output (H1)
Amplitude
n

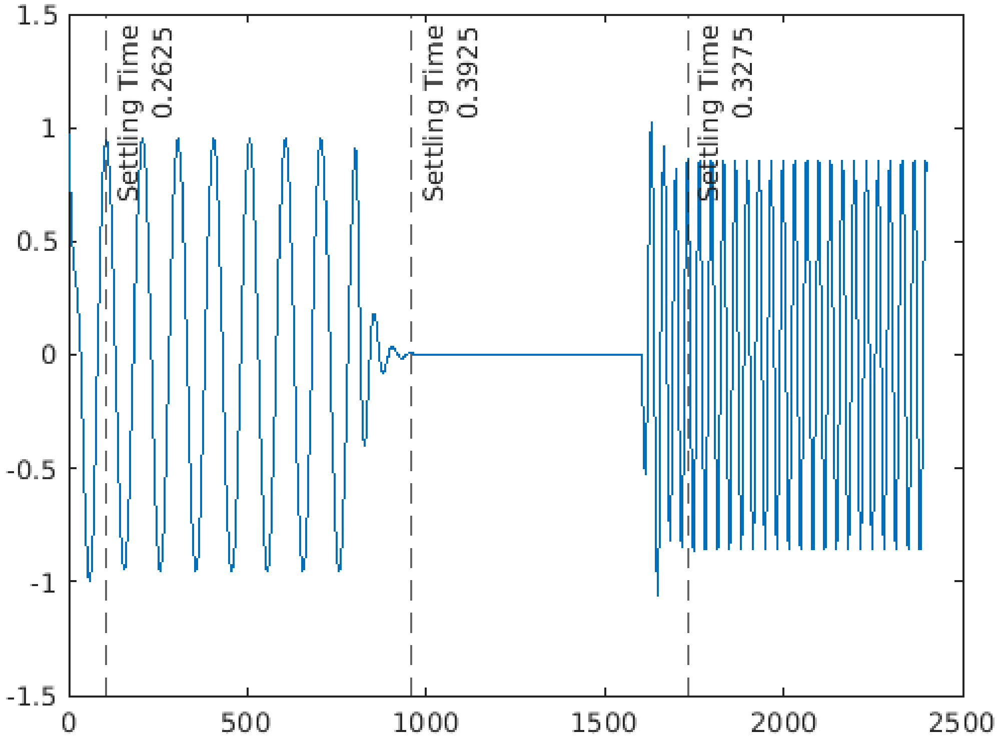

Input signal
Amplitude
n
Filtered Output (H2)
Amplitude
n

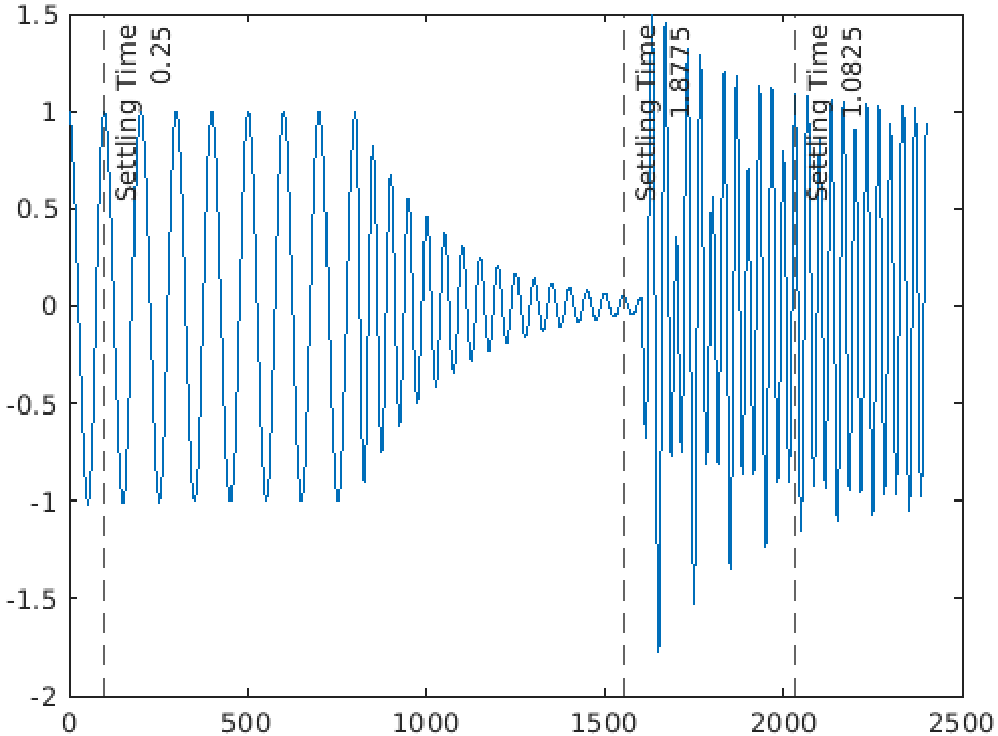
3.2.d

```
fs  = 400;
Nf = 4096;
[h1, w1] = freqz(b1, a1, Nf);
[h2, w2] = freqz(b2, a2, Nf);

w1 = w1./pi.*fs./2;
w2 = w2./pi.*fs./2;

% H1
i4 = find(abs(w1-4) < 0.02);
h1g4 = abs(h1(i4(1))) % 4Hz gain of H1

i8 = find(abs(w1-8) < 0.02);
h1g8 = abs(h1(i8(1))) % 8Hz gain of H1

i12 = find(abs(w1-12) < 0.02);
h1g12 = abs(h1(i12(1))) % 12Hz gain of H1

% H2
i4 = find(abs(w2-4) < 0.02);
h2g4 = abs(h2(i4(1))) % 4Hz gain of H2

i8 = find(abs(w2-8) < 0.02);
h2g8 = abs(h2(i8(1))) % 8Hz gain of H2

i12 = find(abs(w2-12) < 0.02);
h2g12 = abs(h2(i12(1))) % 12Hz gain of H2


% Plotting Settling Times
[peaksH1_value, peaksH1_index] = findpeaks(y1);
[peaksH2_value, peaksH2_index] = findpeaks(y2);

% H1 Settling Times
figure('name', 'Settling Times for H1');
plot(y1);
i=1;
sections = [0,2,4].*fs; % Start of signal sections
freqs = [4,8,12];
current_section = 1;
for delta_g=peaksH1_value(2:end) - peaksH1_value(1:end-1) % Calculating
 changes
       if (peaksH1_index(i) < sections(current_section)) % Skip other samples
 if settled one is already found
           i = i + 1;
           if (i > length(peaksH1_index))
               break
           end
           continue;
       end
```

```
        if (abs(delta_g) < 0.01) % Assumming that when the peaks value dont
 fluctuate as much, the output is settled
            xline(peaksH1_index(i),'--',{'Settling Time', peaksH1_index(i) *
 (1/fs) - sections(current_section)/fs});
            fprintf("H1, Settling amplitude for f=%.0f: %.2f\n",
 freqs(current_section), y1(peaksH1_index(i)));
            current_section = current_section + 1; % Move on to next section
 when first settling point is found
            if (current_section > length(sections))
                break
            end
        end
        i=i+1;
end

% H2 Settling Times
figure('name', 'Settling Times for H2');
plot(y2);
i=1;
sections = [0,2,4].*fs; % Start of signal sections
current_section = 1;
for delta_g=peaksH2_value(2:end) - peaksH2_value(1:end-1) % Calculating
 changes
        if (peaksH2_index(i) < sections(current_section)) % Skip other samples
 if settled one is already found
            i = i + 1;
            if (i > length(peaksH2_index))
                break
            end
            continue;
        end
        if (abs(delta_g) < 0.01) % Assumming that when the peak values don't
 fluctuate as much, the output has settled
            xline(peaksH2_index(i),'--',{'Settling Time', peaksH2_index(i) *
 (1/fs) - sections(current_section)/fs});
            fprintf("H2, Settling amplitude for f=%.0f: %.2f\n",
 freqs(current_section), y2(peaksH2_index(i)));
            current_section = current_section + 1; % Move on to next section
 when first settling point is found
            if (current_section > length(sections))
                break
            end
        end
        i=i+1;
end


% Comparing Settlign Time with stepinfo nswer
sih1 = stepinfo(filt(b1,a1),'SettlingTimeThreshold',0.01); %Step Info H1
sih2 = stepinfo(filt(b2,a2),'SettlingTimeThreshold',0.01); %Step Info H2

fprintf("Settling time from stepinfo for H1:%d\n", sih1.SettlingTime);
fprintf("Settling time from stepinfo for H2:%d\n", sih2.SettlingTime);
```

```
h1g4 =

    0.9483


h1g8 =

    0.0039


h1g12 =

    0.8584


h2g4 =

    0.9990


h2g8 =

    0.0321


h2g12 =

    0.9973

H1, Settling amplitude for f=4: 0.95
H1, Settling amplitude for f=8: 0.01
H1, Settling amplitude for f=12: 0.87
H2, Settling amplitude for f=4: 1.00
H2, Settling amplitude for f=8: 0.05
H2, Settling amplitude for f=12: 1.08
Settling time from stepinfo for H1:121
Settling time from stepinfo for H2:464
```

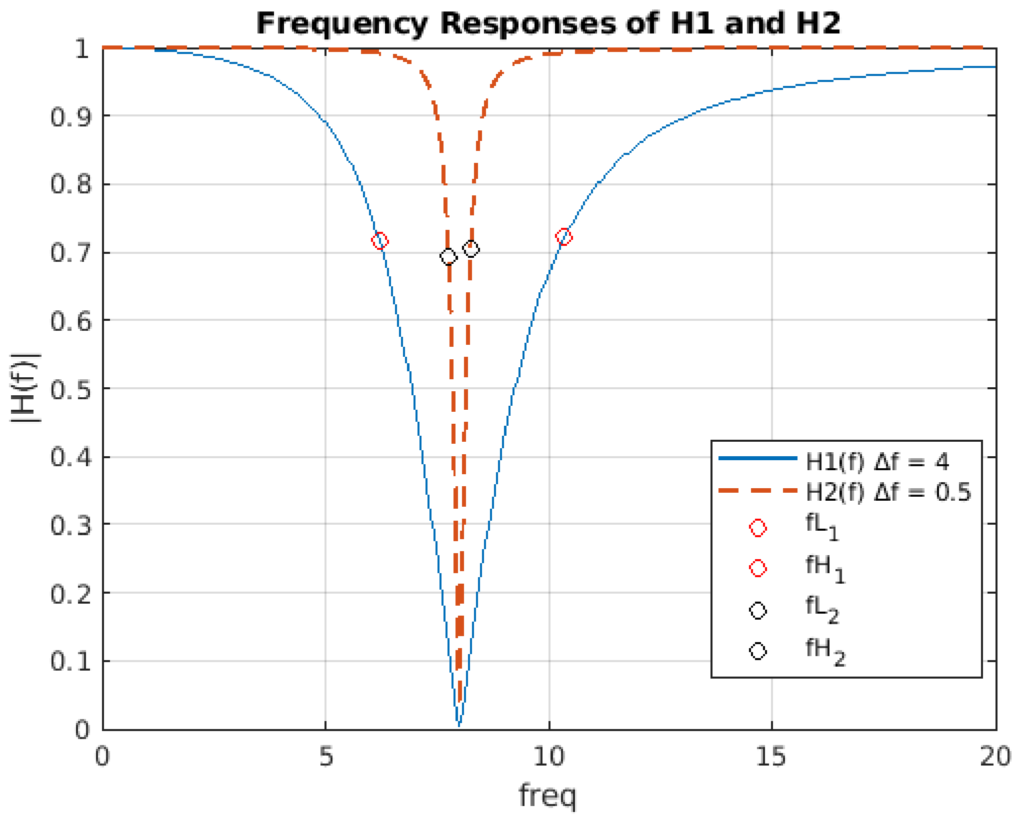

Settling Time
0.2625
Settling Time
0.3925
Settling Time
0.3275


Settling Time
0.25
Settling Time
1.8775
Settling Time
1.0825

3.2.f-g

```
fs  = 400;
Nf = 4096;
[h1, w1] = freqz(b1, a1, Nf);
[h2, w2] = freqz(b2, a2, Nf);

% Plotting frequency response of H1 and H2
figure('name', "Notch Filter Responses")
plot(w1/pi*fs/2, abs(h1), 'LineWidth', 1.5);
xlim([0, 20]);
grid on;
xlabel('freq');
ylabel('|H(f)|');
hold on;
title('Frequency Responses of H1 and H2');
plot(w2/pi*fs/2, abs(h2), '--', 'LineWidth', 1.5);

% Finding section of array which resides in the bandwidth of response
bw1 = find(abs(abs(h1) - .5^.5*max(abs(h1))) < 0.02);
bw2 = find(abs(abs(h2) - .5^.5*max(abs(h2))) < 0.02);

fli1 = bw1(1); % F_l_1 index
fhi1 = bw1(end); % F_h_1 index

fli2 = bw2(1); % F_l_2 index
fhi2 = bw2(end); % F_h_2 index

plot(w1(fli1)/pi*fs/2, abs(h1(fli1)), 'ro', 'LineWidth', 1.5);
plot(w1(fhi1)/pi*fs/2, abs(h1(fhi1)), 'ro', 'LineWidth', 1.5);

plot(w2(fli2)/pi*fs/2, abs(h2(fli2)), 'ko', 'LineWidth', 1.5);
plot(w2(fhi2)/pi*fs/2, abs(h2(fhi2)), 'ko', 'LineWidth', 1.5);


legend('H1(f) #f = 4', 'H2(f) #f =
 0.5','fL_1','fH_1','fL_2','fH_2', 'Location','best');

fprintf("H1:\nFh-Fl=%f\n\nH2:\nFh-Fl=%f\n", (fhi1-fli1)*(fs/Nf/2), (fhi2-
fli2)*(fs/Nf/2))
```

```
H1:
Fh-Fl=4.150391

H2:
Fh-Fl=0.488281
```



Published with MATLAB® R2023a