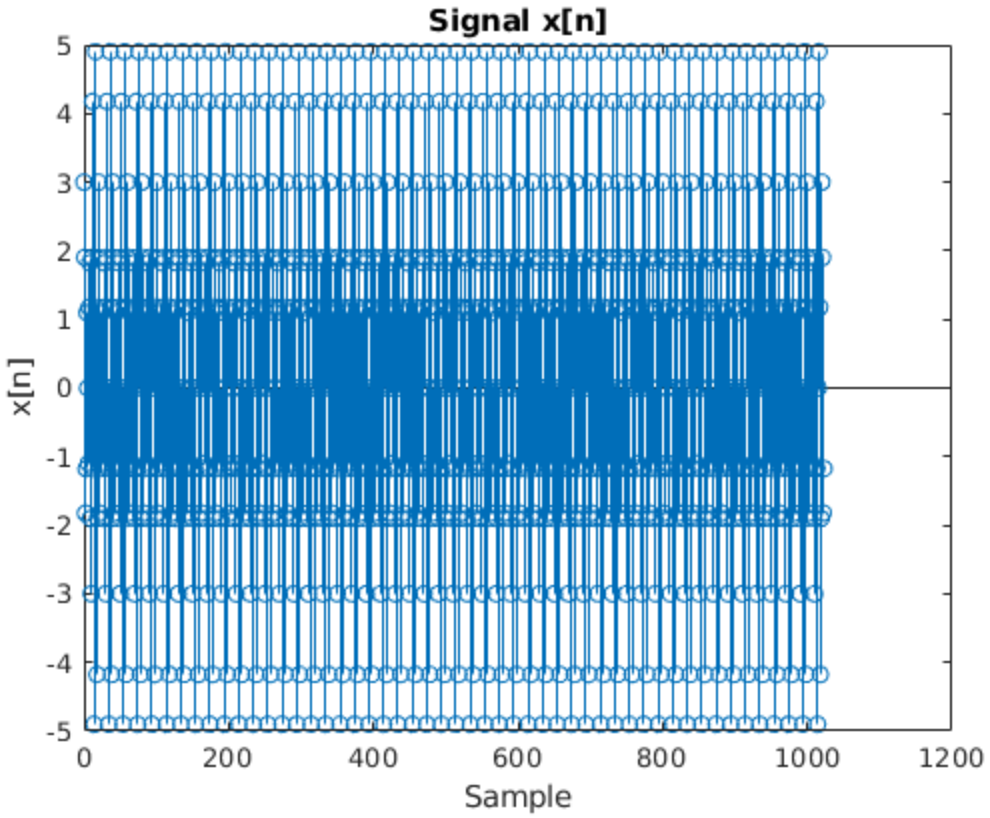
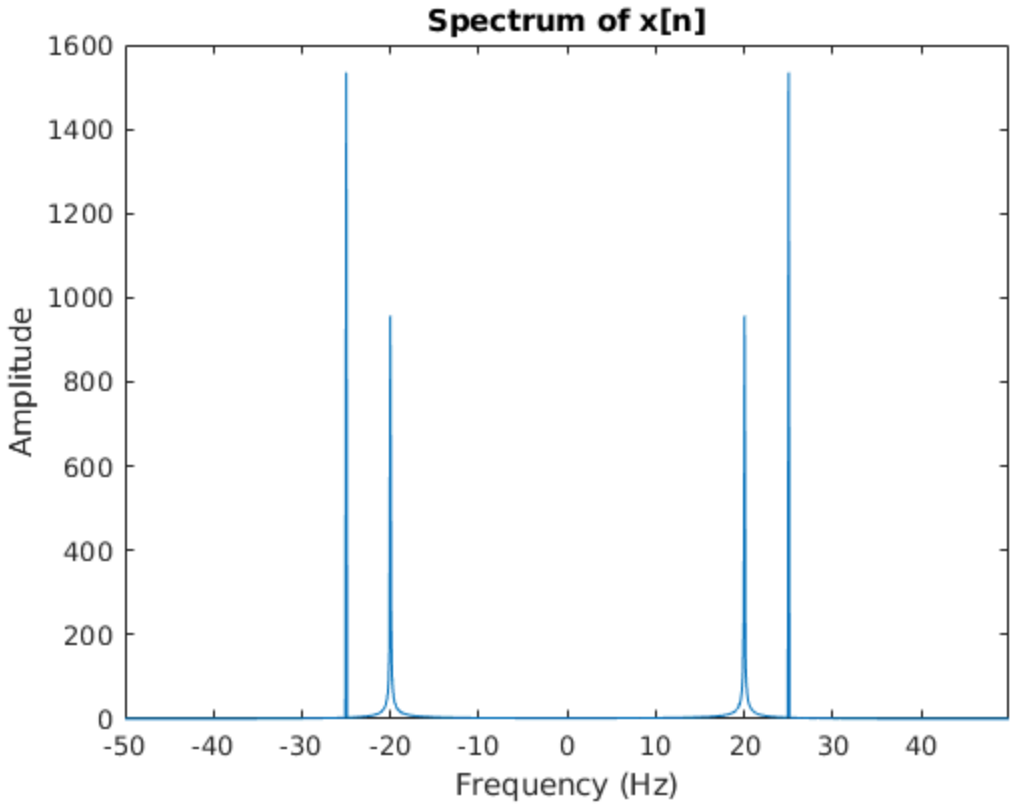
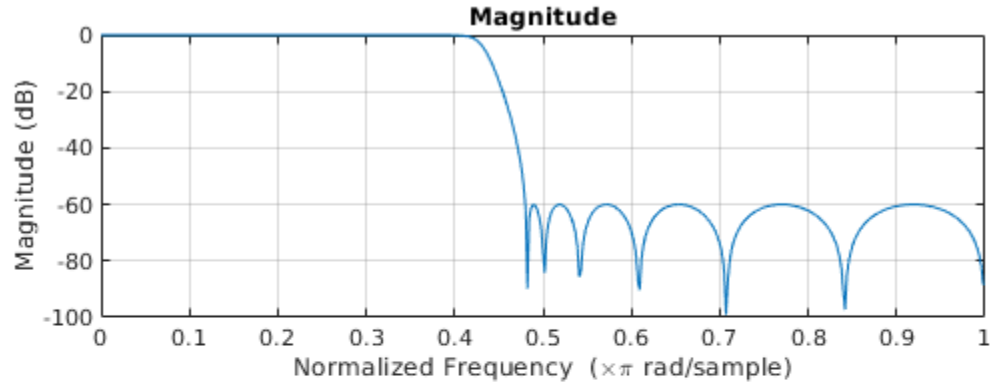
```
%!-------------------------------------------------------------------------
%! DSP HW9 #2
%!  - Create analog signal for x(t) = 2sin(pi20t) + 3cos(pi50t)
%!  - Design Filter with pa < 1dB, ps > 50dB. Filter out higher frequency
%!-------------------------------------------------------------------------

%! Enviorment
clc; clear;
N = 1024;
Fs = 100;
Ts = 1/Fs;
n = (0:N-1)*Ts;
f = ((0:N-1)-N/2)*Fs/N;
filt = load('hw9_2_filtercoef');

%! Signal
xn = 2*sin(40*pi*n) + 3*cos(50*pi*n);

%! Filter
yn = filter(filt.Num, filt.Den, xn);

% Plot
figure()
stem((0:N-1), xn)
title('Signal x[n]')
xlabel('Sample')
ylabel('x[n]')

figure()
plot(f, abs(fftshift(fft(xn))))
xlim([f(1), f(N)])
title('Spectrum of x[n]')
xlabel('Frequency (Hz)')
ylabel('Amplitude')

figure()
freqz(filt.Num, filt.Den)

figure()
stem((0:N/2-1), yn(1:N/2))
title('Signal y[n]')
xlabel('Samples')
ylabel('y[n]')

figure()
plot(f, abs(fftshift(fft(yn))))
xlim([f(1), f(N)])
title('Spectrum of y[n]')
xlabel('Frequency (Hz)')
ylabel('Amplitude')

disp(['The filter was able to reduce the second frequency almost' ...
```

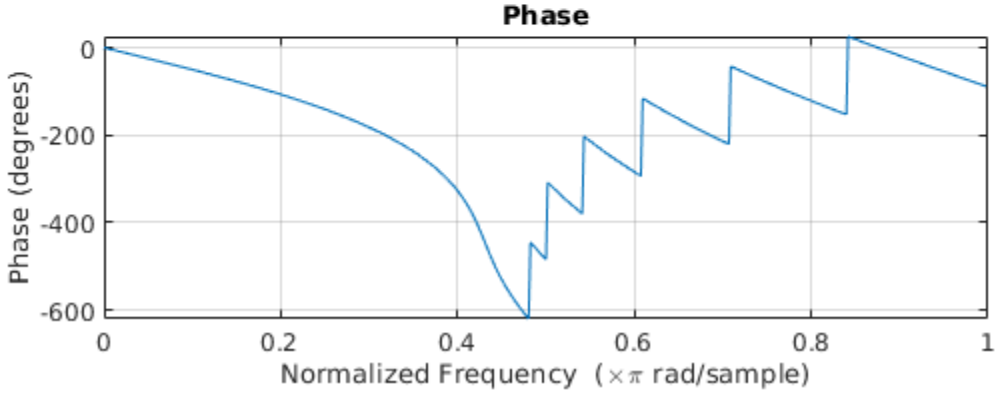
```
    ' completely but the first part of the signal was not perfect while'...
    ' the filter taps filled up'])
```

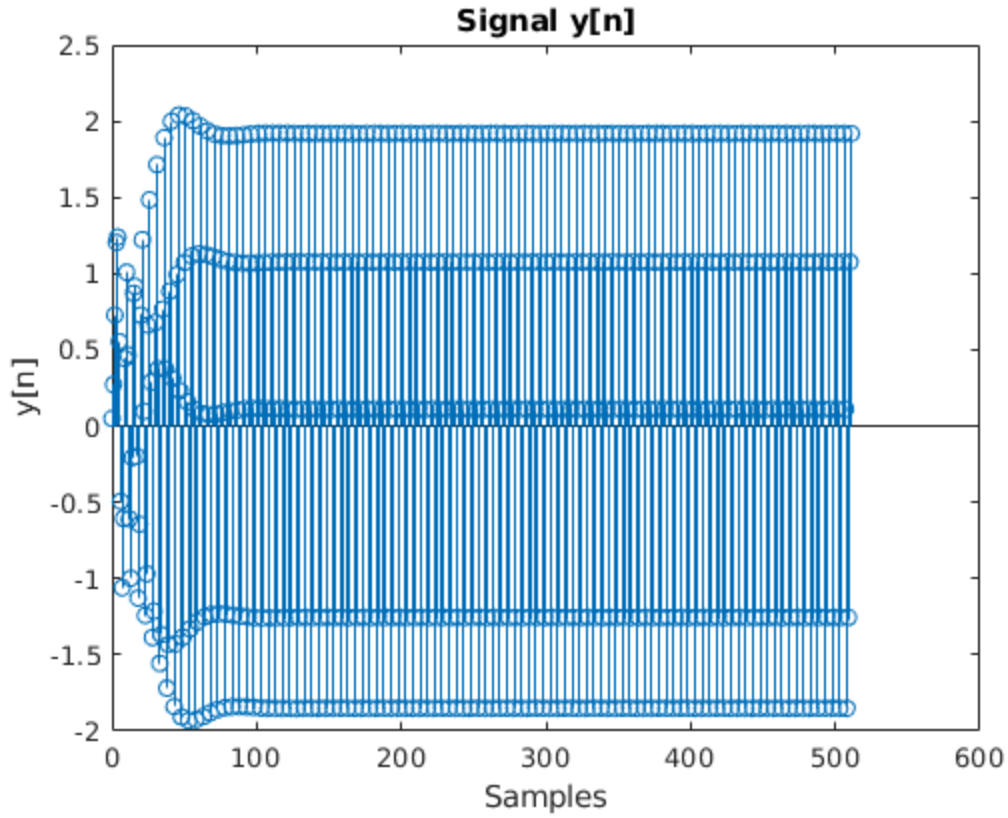
The filter was able to reduce the second frequency almost completely but the first part of the signal was not perfect while the filter taps filled up

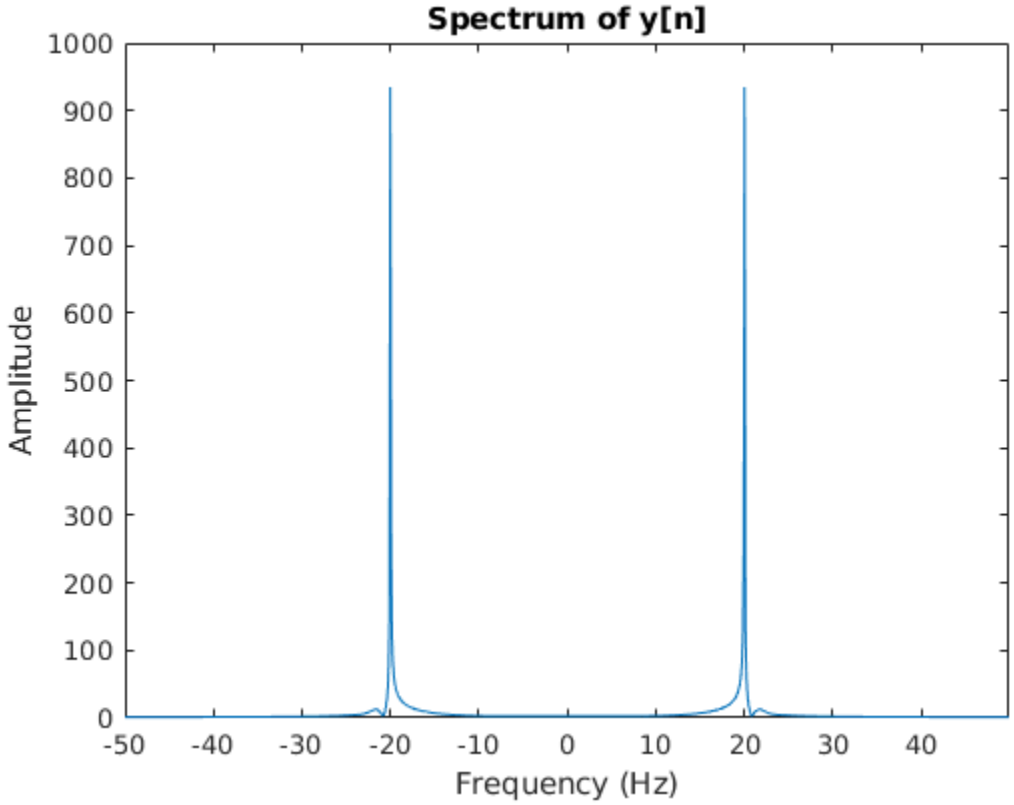



Spectrum of x[n]
Amplitude
Frequency (Hz)


Magnitude
Magnitude (dB)
Normalized Frequency (×π rad/sample)


Phase
Phase (degrees)
Normalized Frequency (×π rad/sample)


Signal y[n]
y[n]
Samples
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
100
200
300
400
500
600


Spectrum of y[n]
Amplitude
Frequency (Hz)
1000
900
800
700
600
500
400
300
200
100
0
-50
-40
-30
-20
-10
0
10
20
30
40

Published with MATLAB® R2023a