MODULE 06 HOMEWORK

3/12/18

EN.525.718.81.SP18 MULTIRATE SIGNAL PROCESSING

SPRING 2018

By: Zachary H. Sweep

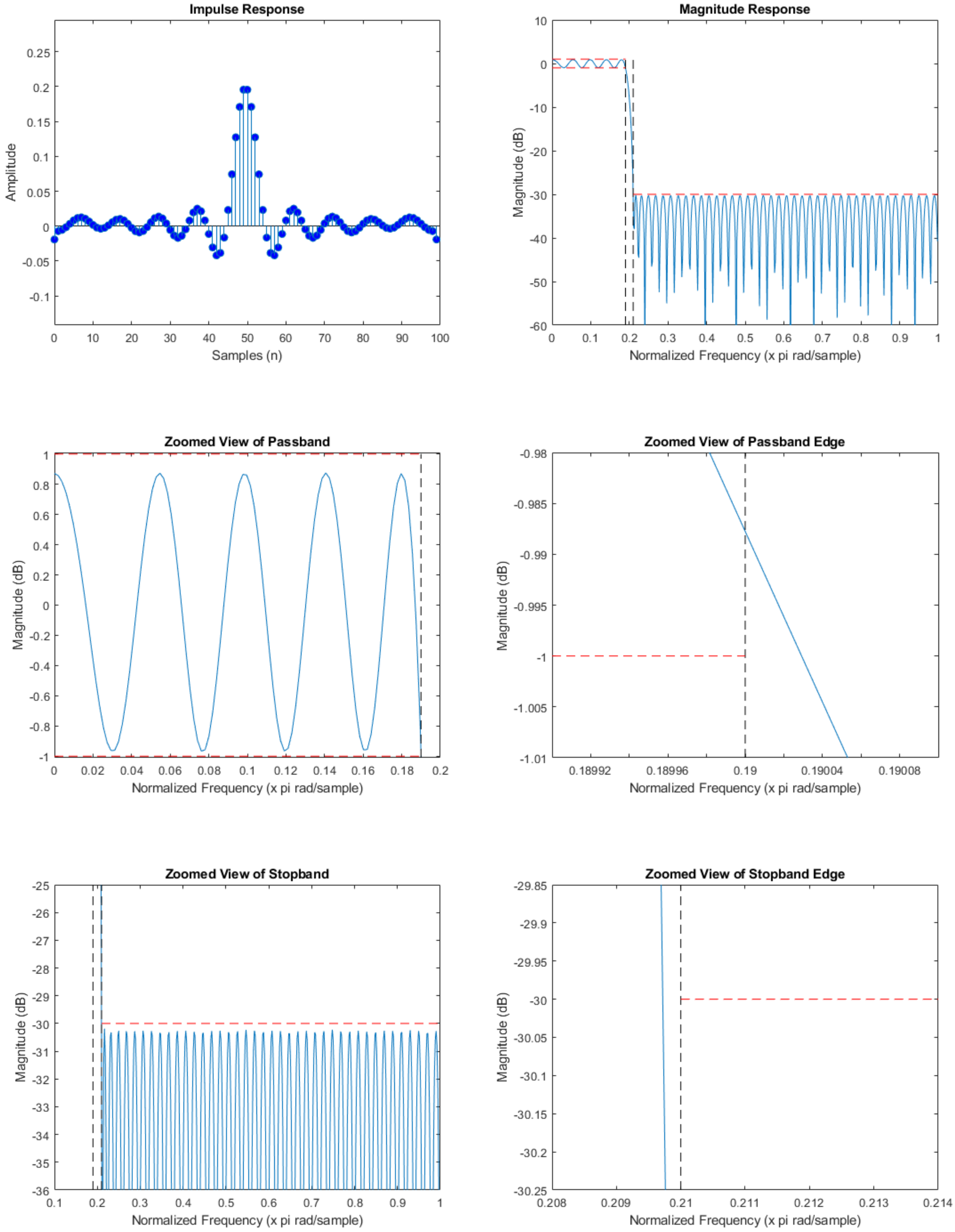
Contents

Problem 1

Using the Parks-McClellan algorithm, design a lowpass decimation filter for M=5 with the following specifications: ωp=π/M-Δω, ωs=π/M+Δω, Rp=1dB, Rs=30dB with Δω= π/100. Plot the impulse response and the frequency response magnitude. Show clearly that your design meets all specifications by plotting the specification template on the frequency response graph.

```
% Decimation Parameter
M = 5;

% Transition band
dw = pi/100;

% Passband/Stopband parameters
wp = pi/M - dw;
ws = pi/M + dw;

% Passband/Stopband ripple parameters
Rp =  1;    %dB
Rs = 30;    %dB

Rp_linear = min([(1-10^(-Rp/20)) (10^(Rp/20)-1)]);

% Compute estimated filter order and other design params using firpmord()
[N,fo,mo,w] = firpmord([wp/pi ws/pi], [1 0], [Rp_linear 10^(-Rs/20)]);

% Increase filter order by 3 since original order was under estimated.
% Compute FIR filter impulse response
N = N + 3;
b = firpm(N,fo,mo,w);

% Compute Frequency Response
[H,w] = freqz(b);

% Plot Impulse Response
figure(1);
stem((0:N),b,'MarkerFaceColor','b');
ylim([min(b)-0.1 max(b)+0.1]);
title('Impulse Response');
xlabel('Samples (n)'); ylabel('Amplitude');

% Plot Magnitude Frequency Response
figure(2);
plot(w/pi,20*log10(abs(H)));
axis([0 1 -60 10]);
title('Magnitude Response');
xlabel('Normalized Frequency (x pi rad/sample)'); ylabel('Magnitude (dB)');
line([wp/pi wp/pi],[-60 Rp],'color','black','LineStyle','--');
line([ws/pi ws/pi],[-60 Rp],'color','black','LineStyle','--');
line([0 wp/pi],[-Rp -Rp],'color','red','LineStyle','--');
line([0 wp/pi],[Rp Rp],'color','red','LineStyle','--');
line([ws/pi 1],[-Rs -Rs],'color','red','LineStyle','--');

% Zoomed view of passband
figure(3);
plot(w/pi,20*log10(abs(H)));
axis([0 0.2 -1.01 1.01]);
title('Zoomed View of Passband');
```

```
xlabel('Normalized Frequency (x pi rad/sample)'); ylabel('Magnitude (dB)');
line([wp/pi wp/pi],[-60 Rp],'color','black','LineStyle','--');
line([ws/pi ws/pi],[-60 Rp],'color','black','LineStyle','--');
line([0 wp/pi],[-Rp -Rp],'color','red','LineStyle','--');
line([0 wp/pi],[Rp Rp],'color','red','LineStyle','--');
line([ws/pi 1],[-Rs -Rs],'color','red','LineStyle','--');

% Zoomed view of passband edge
figure(4);
plot(w/pi,20*log10(abs(H)));
axis([0.1899 0.1901 -1.01 -0.980]);
title('Zoomed View of Passband Edge');
xlabel('Normalized Frequency (x pi rad/sample)'); ylabel('Magnitude (dB)');
line([wp/pi wp/pi],[-60 Rp],'color','black','LineStyle','--');
line([ws/pi ws/pi],[-60 Rp],'color','black','LineStyle','--');
line([0 wp/pi],[-Rp -Rp],'color','red','LineStyle','--');
line([0 wp/pi],[Rp Rp],'color','red','LineStyle','--');
line([ws/pi 1],[-Rs -Rs],'color','red','LineStyle','--');

% Zoomed view of stopband
figure(5);
plot(w/pi,20*log10(abs(H)));
axis([0.1 1 -36 -25]);
title('Zoomed View of Stopband');
xlabel('Normalized Frequency (x pi rad/sample)'); ylabel('Magnitude (dB)');
line([wp/pi wp/pi],[-60 Rp],'color','black','LineStyle','--');
line([ws/pi ws/pi],[-60 Rp],'color','black','LineStyle','--');
line([0 wp/pi],[-Rp -Rp],'color','red','LineStyle','--');
line([0 wp/pi],[Rp Rp],'color','red','LineStyle','--');
line([ws/pi 1],[-Rs -Rs],'color','red','LineStyle','--');

% Zoomed view of stopband edge
figure(6);
plot(w/pi,20*log10(abs(H)));
axis([0.208 0.214 -30.25 -29.85]);
title('Zoomed View of Stopband Edge');
xlabel('Normalized Frequency (x pi rad/sample)'); ylabel('Magnitude (dB)');
line([wp/pi wp/pi],[-60 Rp],'color','black','LineStyle','--');
line([ws/pi ws/pi],[-60 Rp],'color','black','LineStyle','--');
line([0 wp/pi],[-Rp -Rp],'color','red','LineStyle','--');
line([0 wp/pi],[Rp Rp],'color','red','LineStyle','--');
line([ws/pi 1],[-Rs -Rs],'color','red','LineStyle','--');
```

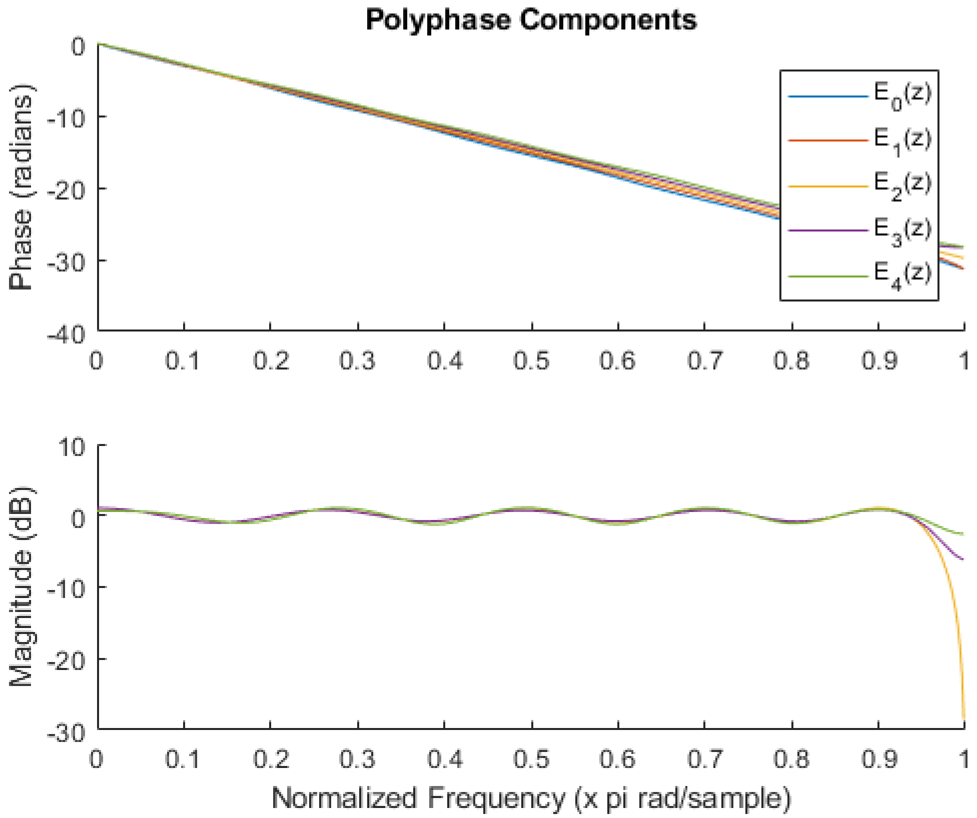

Impulse Response
Amplitude
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
0
10
20
30
40
50
60
70
80
90
100
Samples (n)
Magnitude Response
10
0
-10
-20
-30
-40
-50
-60
Magnitude (dB)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (x pi rad/sample)
Zoomed View of Passband
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
Magnitude (dB)
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Normalized Frequency (x pi rad/sample)
Zoomed View of Passband Edge
-0.98
-0.985
-0.99
-0.995
-1
-1.005
-1.01
Magnitude (dB)
0.18992
0.18996
0.19
0.19004
0.19008
Normalized Frequency (x pi rad/sample)
Zoomed View of Stopband
-25
-26
-27
-28
-29
-30
-31
-32
-33
-34
-35
-36
Magnitude (dB)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (x pi rad/sample)
Zoomed View of Stopband Edge
-29.85
-29.9
-29.95
-30
-30.05
-30.1
-30.15
-30.2
-30.25
Magnitude (dB)
0.208
0.209
0.21
0.211
0.212
0.213
0.214
Normalized Frequency (x pi rad/sample)

Problem 2

Determine the Type I polyphase representation for the filter designed in Part 1 and plot the magnitude and phase response for each polyphase section.

```
h = b;

% Create each polyphase impulse response by resampling h(n) from problem 1.
e0 = M*h(1:M:end);
e1 = M*h(2:M:end);
e2 = M*h(3:M:end);
e3 = M*h(4:M:end);
e4 = M*h(5:M:end);

% Interleaving polyphase components so that I could plot the impulse
% response and make sure it matched the direct form impulse response.
eL = zeros(1, length(e0)+length(e1)+length(e2)+length(e3)+length(e4));
eL(1:M:end) = e0;
eL(2:M:end) = e1;
eL(3:M:end) = e2;
eL(4:M:end) = e3;
eL(5:M:end) = e4;

% Compute magnitude response for each polyphase component
[E0,w0] = freqz(e0);
[E1,w1] = freqz(e1);
[E2,w2] = freqz(e2);
[E3,w3] = freqz(e3);
[E4,w4] = freqz(e4);

% Compute phase response for each polyphase component
[P0,wp0] = phasez(e0);
[P1,wp1] = phasez(e1);
[P2,wp2] = phasez(e2);
[P3,wp3] = phasez(e3);
[P4,wp4] = phasez(e4);

% Plot phase response for e0-e4
figure(7);
subplot(211)
hold on
plot(wp0/pi, P0);
plot(wp1/pi, P1);
plot(wp2/pi, P2);
plot(wp3/pi, P3);
plot(wp4/pi, P4);
title('Polyphase Components');
ylabel('Phase (radians)');
legend('E_0(z)','E_1(z)','E_2(z)','E_3(z)','E_4(z)');
hold off

% Plot magnitude response for e0-e4
subplot(212)
hold on
plot(w0/pi, 20*log10(abs(E0)));
plot(w1/pi, 20*log10(abs(E1)));
plot(w2/pi, 20*log10(abs(E2)));
plot(w3/pi, 20*log10(abs(E3)));
plot(w4/pi, 20*log10(abs(E4)));
```

```
xlabel('Normalized Frequency (x pi rad/sample)');
ylabel('Magnitude (dB)');
hold off
```



Problem 3

Evaluate the response of a system consisting of the decimation filter designed in Part 1 followed by a M-sample compressor for the following inputs: x(n) = cos(ω0n), ω0= π/50, ω0= π/50+ 2π/5, ω0= π/50+ 4π/5. Determine the output frequency and approximate peak amplitude (steady state) for each input frequency. Make a rough comparison of the relative amplitudes and compare with predictions based on the passband/stopband specifications for the decimation filter.

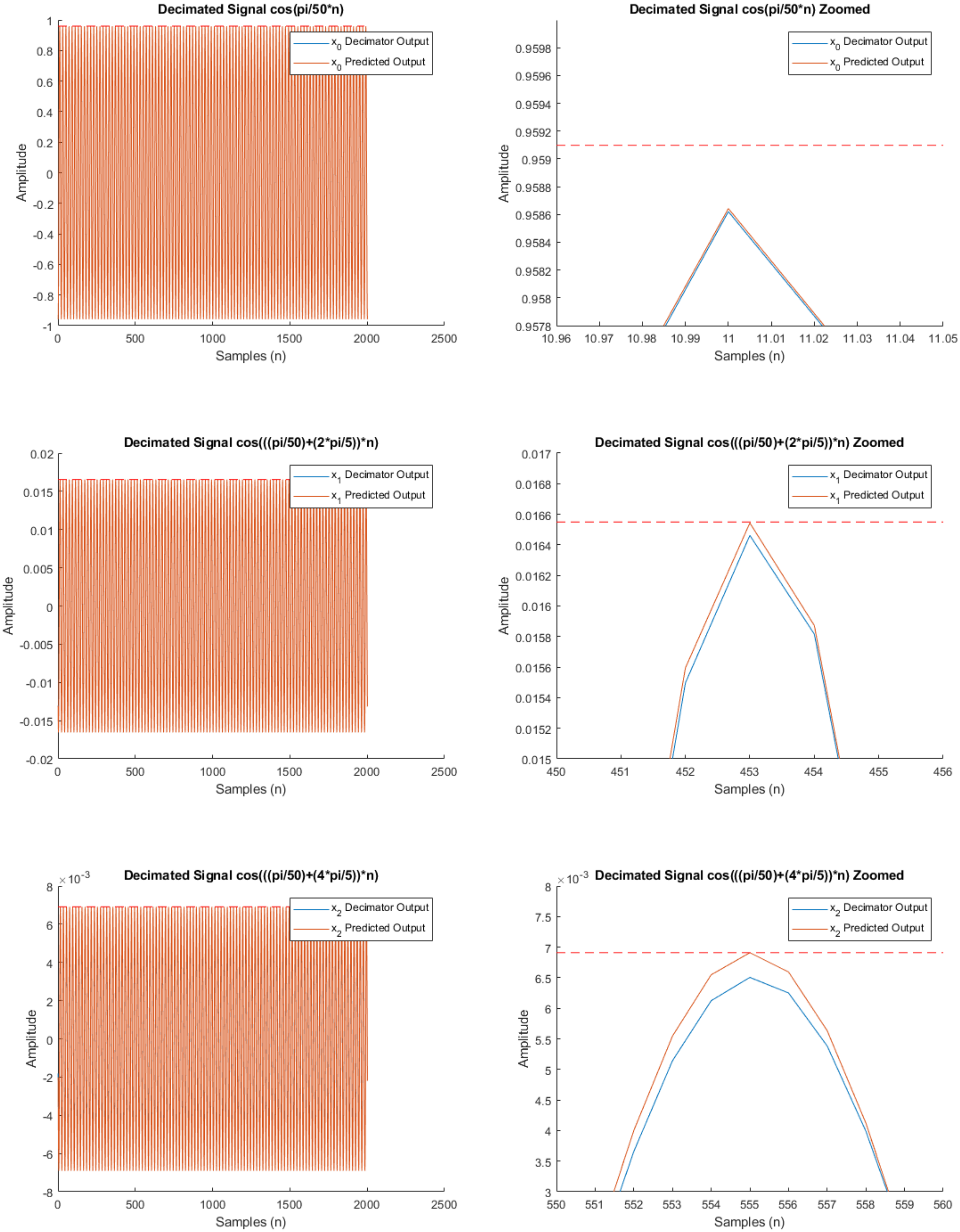
This problem clearly exemplifies the effects of aliasing that can occur during decimation. The lowpass filter from problem 1 did not have enough stop band attenuation to prevent the signals of frequency w0=pi/50+2pi/5 and w0=pi/50+4pi/5 from having a noticeable presence in the output of the system. As a result of the aliasing these two signals appeared on the output of the system as having a frequency of w0 = pi/50.

```
% Re-Compute Frequency response of decimation filter from problem 1
[H,w] = freqz(b);

% Input signal frequencies
w_0 = pi/50;
w_1 = w_0 + (2*pi)/5;
w_2 = w_0 + (4*pi)/5;

% Create input signals x0, x1, and x2 according to the above frequencies
```

```
n = (0:10000);
x0 = cos(w_0*n);
x1 = cos(w_1*n);
x2 = cos(w_2*n);


%******************** Output Signal Predictions ***************************
% This small section is where I predict what the output will look like for
% each input signal.

% Interpolate H and w of Decimation filter in problem 1 and find the
% frequencies closest to w_0, w_1, and w_1 for more accurate predictions.
H_interp = interp(H,1000);
w_interp = interp(w,1000);
[m,i0] = min(abs(w_interp-w_0));
[m,i1] = min(abs(w_interp-w_1));
[m,i2] = min(abs(w_interp-w_2));

% Predict output amplitude for input signals x0, x1, and x2
A0    = abs(H_interp(i0));
A1    = abs(H_interp(i1));
A2    = abs(H_interp(i2));

% Predict phase offset for input signals x0, x1, and x2
ph0 = angle(H_interp(i0));
ph1 = angle(H_interp(i1));
ph2 = angle(H_interp(i2));

% Predict normalized frequency for input signals x0,x1, and x2
f = min(abs((-(M-1):M-1)*(2*pi/M) + [-w_0, w_0]'));
f0_est = min(f)/pi;
f = min(abs((-(M-1):M-1)*(2*pi/M) + [-w_1, w_1]'));
f1_est = min(f)/pi;
f = min(abs((-(M-1):M-1)*(2*pi/M) + [-w_2, w_2]'));
f2_est = min(f)/pi;

% Apply amplitude, phase, and frequency predictions to signal template.
x0_est = A0*cos(w_interp(i0)*n - ph0);
x1_est = A1*cos(w_interp(i1)*n - ph1);
x2_est = A2*cos(w_interp(i2)*n - ph2);

% Downsample the prediction siganls by factor M
x0_est = downsample(x0_est,M);
x1_est = downsample(x1_est,M);
x2_est = downsample(x2_est,M);

%******************** End of Output Signal Predictions ********************


%******************** Compute Actual Sytem Output *************************

% Apply convolve filter impulse response with each input signal.
x0_filt = conv(x0,h);
x1_filt = conv(x1,h);
x2_filt = conv(x2,h);

% Keep only the steady state portions of the filtered signals. Steady
%state begins at N+1 and ends at length(filtered signal) - N+1 where
% N = filter order
```

```
ind_start = N+1;
ind_end = length(x0_filt)-(N+1);
x0_filt = x0_filt(ind_start:ind_end);
x1_filt = x1_filt(ind_start:ind_end);
x2_filt = x2_filt(ind_start:ind_end);

% Downsample the filtered signal by factor M by retrieving every Mth
% sample from each filterd signal
x0_ds = downsample(x0_filt,M);
x1_ds = downsample(x1_filt,M);
x2_ds = downsample(x2_filt,M);

% Plot filtered/downsampled signals x0_ds, x1_ds, and x2_ds as well as
% predicted outputs
figure(11);
hold on
plot(x0_ds);
plot(x0_est);
line([0 length(x0_ds)],[A0 A0],'color','red','LineStyle','--');
title('Decimated Signal cos(pi/50*n)'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_0 Decimator Output','x_0 Predicted Output'); hold off

figure(12);
hold on
plot(x0_ds);
plot(x0_est);
line([0 length(x0_ds)],[A0 A0],'color','red','LineStyle','--');
title('Decimated Signal cos(pi/50*n) Zoomed'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_0 Decimator Output','x_0 Predicted Output');
axis([10.96 11.05 0.9578 0.960]);h old off

figure(13);
hold on
plot(x1_ds);
plot(x1_est);
line([0 length(x1_ds)],[A1 A1],'color','red','LineStyle','--');
title('Decimated Signal cos(((pi/50)+(2*pi/5))*n)'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_1 Decimator Output','x_1 Predicted Output'); hold off

figure(14);
hold on
plot(x1_ds);
plot(x1_est);
line([0 length(x1_ds)],[A1 A1],'color','red','LineStyle','--');
title('Decimated Signal cos(((pi/50)+(2*pi/5))*n) Zoomed'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_1 Decimator Output','x_1 Predicted Output');
axis([450 456 0.015 0.017]); hold off

figure(15);
hold on
plot(x2_ds);
plot(x2_est);
line([0 length(x2_ds)],[A2 A2],'color','red','LineStyle','--');
title('Decimated Signal cos(((pi/50)+(4*pi/5))*n)'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_2 Decimator Output','x_2 Predicted Output'); hold off

figure(16);
hold on
plot(x2_ds);
plot(x2_est);
line([0 length(x2_ds)],[A2 A2],'color','red','LineStyle','--');
title('Decimated Signal cos(((pi/50)+(4*pi/5))*n) Zoomed'); xlabel('Samples (n)'); ylabel('Amplitude');
legend('x_2 Decimator Output','x_2 Predicted Output');
axis([550 560 3e-3 8e-3]); hold off

% Compute FFT of output signal x0_ds, x1_ds, and x2_ds
H0_ds = fft(x0_ds);
H1_ds = fft(x1_ds);
H2_ds = fft(x2_ds);
L = length(H0_ds);
w = linspace(0,1,L/2)/M;

% Find the peak amplitude, frequency, and phase offset of output signals
% by finding the index corresponding to the highest absolute magnitude
% of the FFT.
[m0,i0] = max(abs(H0_ds/L)*2);
[m1,i1] = max(abs(H1_ds/L)*2);
[m2,i2] = max(abs(H2_ds/L)*2);

H0_actual = abs(H0_ds(i0)/L)*2;
H1_actual = abs(H1_ds(i1)/L)*2;
H2_actual = abs(H2_ds(i2)/L)*2;

ph0_actual = angle(H0_ds(i0));
ph1_actual = angle(H1_ds(i1));
ph2_actual = angle(H2_ds(i2));

w0_actual = w(i0);
w1_actual = w(i1);
w2_actual = w(i2);

% Compare to Predictions made previously
fprintf('\nPredicted Output Signal for Input x0\n');
fprintf('Predicted:   Amplitude = %f, Frequency = %f, Phase Offset = %f\n',A0,f0_est,ph0);
fprintf('Actual:      Amplitude = %f, Frequency = %f, Phase Offset = %f\n',H0_actual,w0_actual,ph0_actual);
fprintf('\nPredicted Vs Actual Output Signal for Input x1\n');
fprintf('Predicted:   Amplitude = %f, Frequency = %f, Phase Offset = %f\n',A1,f1_est,ph1);
fprintf('Actual:      Amplitude = %f, Frequency = %f, Phase Offset = %f\n',H1_actual,w1_actual,ph1_actual);
fprintf('\nPredicted Vs Actual Output Signal for Input x2\n');
fprintf('Predicted:   Amplitude = %f, Frequency = %f, Phase Offset = %f\n',A2,f1_est,ph2);
fprintf('Actual:      Amplitude = %f, Frequency = %f, Phase Offset = %f\n',H2_actual,w2_actual,ph2_actual);
```

```
Predicted Vs Actual Output Signal for Input x0
Predicted:   Amplitude = 0.959098, Frequency = 0.020000, Phase Offset = -3.110810
Actual:      Amplitude = 0.955399, Frequency = 0.020020, Phase Offset = -3.015992

Predicted Vs Actual Output Signal for Input x1
Predicted:   Amplitude = 0.016549, Frequency = 0.020000, Phase Offset = -2.484833
Actual:      Amplitude = 0.016403, Frequency = 0.020020, Phase Offset = 2.639107

Predicted Vs Actual Output Signal for Input x2
Predicted:   Amplitude = 0.006909, Frequency = 0.020000, Phase Offset = -1.866467
Actual:      Amplitude = 0.006480, Frequency = 0.020020, Phase Offset = 2.010678
```


Decimated Signal cos(pi/50*n)
Amplitude
Samples (n)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
500
1000
1500
2000
2500
x_0 Decimator Output
x_0 Predicted Output
Decimated Signal cos(pi/50*n) Zoomed
Amplitude
Samples (n)
0.9598
0.9596
0.9594
0.9592
0.959
0.9588
0.9586
0.9584
0.9582
0.958
0.9578
10.96
10.97
10.98
10.99
11
11.01
11.02
11.03
11.04
11.05
x_0 Decimator Output
x_0 Predicted Output
Decimated Signal cos(((pi/50)+(2*pi/5))*n)
Amplitude
Samples (n)
0.02
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
-0.02
0
500
1000
1500
2000
2500
x_1 Decimator Output
x_1 Predicted Output
Decimated Signal cos(((pi/50)+(2*pi/5))*n) Zoomed
Amplitude
Samples (n)
0.017
0.0168
0.0166
0.0164
0.0162
0.016
0.0158
0.0156
0.0154
0.0152
0.015
450
451
452
453
454
455
456
x_1 Decimator Output
x_1 Predicted Output
Decimated Signal cos(((pi/50)+(4*pi/5))*n)
$\times 10^{-3}$
Amplitude
Samples (n)
8
6
4
2
0
-2
-4
-6
-8
0
500
1000
1500
2000
2500
x_2 Decimator Output
x_2 Predicted Output
Decimated Signal cos(((pi/50)+(4*pi/5))*n) Zoomed
$\times 10^{-3}$
Amplitude
Samples (n)
8
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
550
551
552
553
554
555
556
557
558
559
560
x_2 Decimator Output
x_2 Predicted Output

Problem 4

Describe, in general terms, how your observations in Part 3 would be changed by the use of a decimation filter with a stopband decay rate of 1/f (as opposed to the minimax stopband characteristic of the Parks-McClellan design).

```
% If the stop band of the filter in part 1 decayed at a rate of 1/f the
% amplitude of signals x1 and x2 would be less significant at the output
% of the filter. Since the stopband of the filter in part 1 has a mini-max
% characteristic all frequencies in the stopband are generally attenuated
% the same. On the other hand, with a stopband decay rate of 1/f, higher
% frequency input signals are attenuated more. This would reduce the
% effects of aliasing in the output of the system.
```

Published with MATLAB® R2017b