EECE-5666 (DSP) : Homework-6 Solutions

Table of Contents

Default Plot Parameters

```
set(0,'defaultfigurepaperunits','inches','defaultfigureunits','inches');
set(0,'defaultaxesfontsize',10); set(0,'defaultaxeslinewidth',1.5);
set(0,'defaultaxestitlefontsize',1.4,'defaultaxeslabelfontsize',1.2);
```

Problem 6.1: Conversion between Filter Specifications

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

(a) Design the following MATLAB function that implements conversions between various filter specifications, as described in the comments below in the code example.

MATLAB function:

```
function [A,B] = speConversion(C,D,typein,typeout)
% typein:  'abs' or 'rel' or 'ana'
% typeout: 'abs' or 'rel' or 'ana'
% C, D:     input specifications
% A, B:     output specifications
%
typein = lower(typein); typeout = lower(typeout);
% Error check
if ~(strcmp(typein,'abs') || strcmp(typein,'rel') || strcmp(typein,'ana'))
    error('typein not recognized')
end
if ~(strcmp(typeout,'abs') || strcmp(typeout,'rel') || strcmp(typeout,'ana'))
    error('typeout not recognized')
end
% When "typein" = "typeout", no conversion
if strcmp(typein,typeout)
    A = C; B = D;
```

```
end
% When "typein' is 'abs'
if strcmp(typein,'abs') && strcmp(typeout,'rel')
    A = 20*log10((1+C)/(1-C)); B = 20*log10((1+C)/D);
elseif strcmp(typein,'abs') && strcmp(typeout,'ana')
    A = 20*log10((1+C)/(1-C)); B = 20*log10((1+C)/D);
    A = sqrt(10^(A/10)-1); B = 10^(B/20);
end
% When 'typein" is 'rel'
if strcmp(typein,'rel') && strcmp(typeout,'abs')
    A = (10^(C/20)-1)/(10^(C/20)+1); B = (1+A)/(10^(D/20));
elseif strcmp(typein,'rel') && strcmp(typeout,'ana')
    A = sqrt(10^(C/10)-1); B = 10^(D/20);
end
% When 'typein" is 'ana'
if strcmp(typein,'ana') && strcmp(typeout,'rel')
    A = 20*log10(sqrt(1+C^2)); B = 20*log10(D);
elseif strcmp(typein,'ana') && strcmp(typeout,'abs')
    A = 20*log10(sqrt(1+C^2)); B = 20*log10(D);
    A = (10^(A/20)-1)/(10^(A/20)+1); B = (1+A)/(10^(B/20));
end
end
```

(b) Convert the following absolute (dB) specifications into (i) relative and (ii) analog specifications:

$$\delta_p = 0.001 \text{ and } \delta_s = 0.005$$

(i) Relative specs:

```
clc; close all; clear;
delp = 0.001; dels = 0.005;
[Ap,As] = speConversion(delp,dels,'abs','rel'); display(Ap); display(As);
```

```
Ap = 0.0174
As = 46.0293
```

(ii) Analog specs:

```
[epsi,A] = speConversion(delp,dels,'abs','ana'); display(epsi); display(A);
```

```
epsi = 0.0633
A = 200.2000
```

(c) Convert the following relative (dB) specifications into (i) absolute and (ii) analog specifications:

$$A_p = 0.025 \text{ dB and } A_s = 60 \text{ dB}$$

(i) Absolute specs:

```
Ap = 0.025; As = 60;
[delp,dels] = speConversion(Ap,As,'rel','abs'); display(delp); display(dels);
```

```
delp = 0.0014
dels = 0.0010
```

(ii) Analog specs:

```
[epsi,A] = speConversion(Ap,As,'rel','ana'); display(epsi); display(A);
```

```
epsi = 0.0760
A = 1000
```

Problem 6.2: Text Problem 10.4 (Page 744)

Design an FIR linear-phase, bandstop digital filter approximating the ideal frequency response

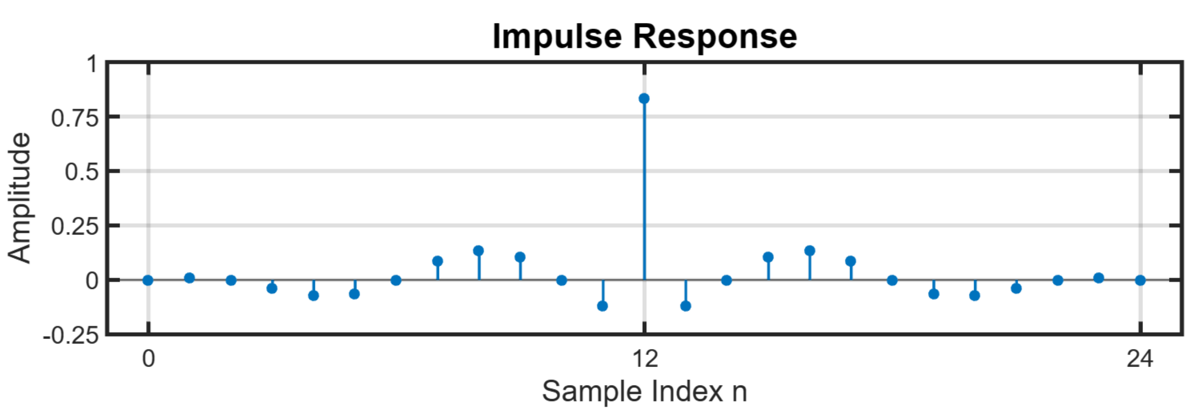
$$H_d(\omega) = \begin{cases} 1, & \text{for } |\omega| \le \dfrac{\pi}{6} \\ 0, & \text{for } \dfrac{\pi}{6} < |\omega| \le \dfrac{\pi}{3} \\ 1, & \text{for } \dfrac{\pi}{3} < |\omega| \le \pi \end{cases}$$

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

(a) Determine the coefficients of a 25-tap filter based on the window method with a rectangular window. Provide a **`stem`** plot of the impulse response $h(n)$.

MATLAB script for computation and plot:

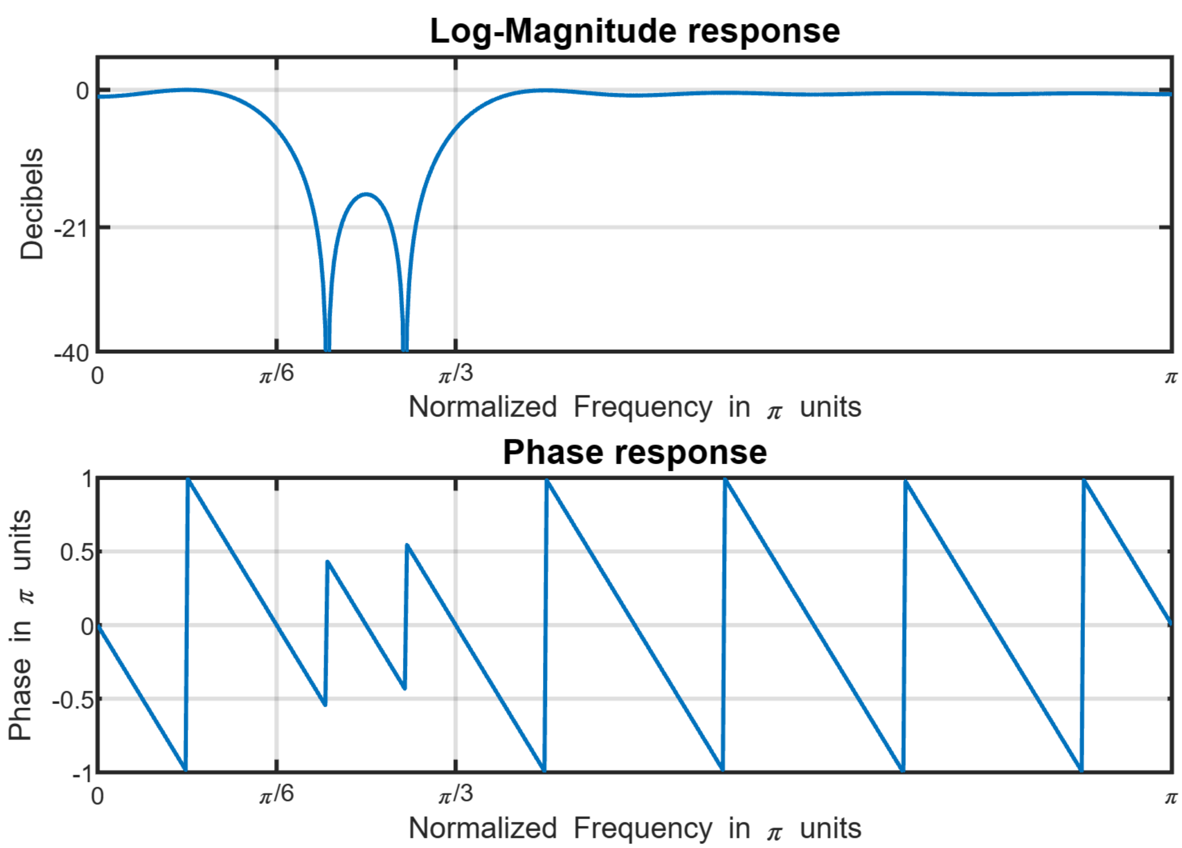
```
M = 25; wc1 = pi/6; wc2 = pi/3; w_rec = boxcar(M)';
hd = ideal_lp(wc1,M)+ideal_lp(pi,M)-ideal_lp(wc2,M);
h = hd.*w_rec;
figure('units','inches','position',[0,0,7,2]);
stem(0:M-1,h,'filled','linewidth',1,'MarkerSize',3);
xlabel('Sample Index n'); ylabel('Amplitude');
title('Impulse Response'); grid; axis([-1,M,-0.25,1])
set(gca,'xtick',[0,(M-1)/2,M-1],'ytick',-0.25:0.25:1);
```



(b) Determine and plot the log-magnitude (dB) and phase response of the filter using two rows and one column subplots over $0 \le \omega \le \pi$.

MATLAB script for computation and plot:

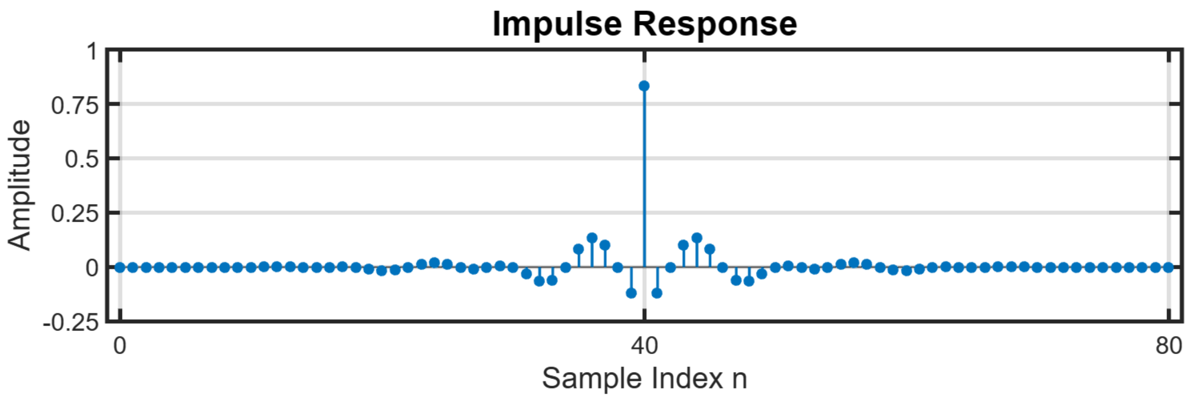
```
% [db, mag, pha, grd, w] = freqz_m(h, 1);
% om = linspace(0,1,501);
om = linspace(0,1,501)*pi; H = freqz(h,1,om);
figure('Units','inches','position',[0,0,7,4.5]);
subplot(2,1,1); % Log-magnitude plot
Hmag = abs(H); Hdb = 20*log10(Hmag/max(Hmag));
plot(om/pi,Hdb,'linewidth',1.5); axis([0,1,-40,5]);
xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude response');
set(gca,'xtick',[0,1/6,1/3,1],'ytick',[-40,-21,0]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
subplot(2,1,2); % Phase plot
plot(om/pi,angle(H)/pi,'LineWidth',1.5); axis([0,1,-1,1])
xlabel('Normalized Frequency in \pi units');
ylabel('Phase in \pi units'); title('Phase response');
set(gca,'xtick',[0,1/6,1/3,1]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
```



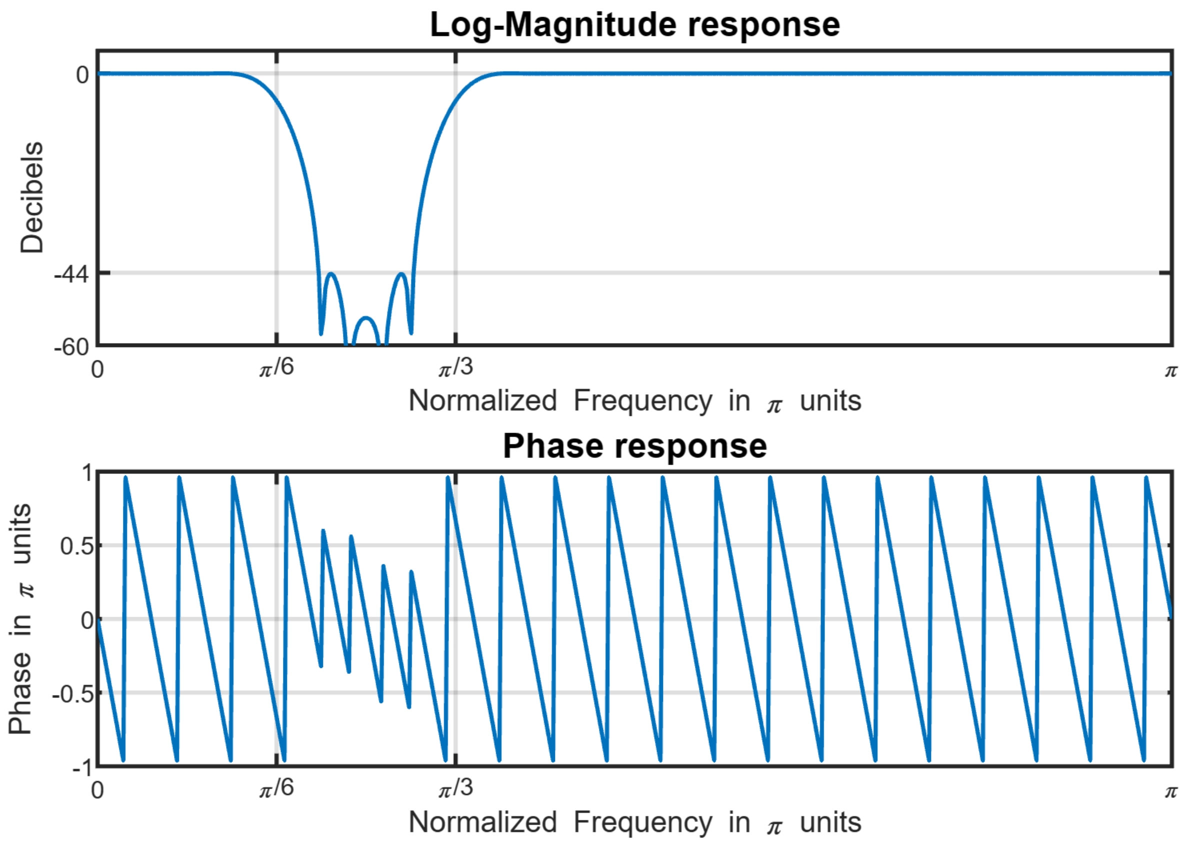
(c) Repeat parts (a) and (b) using the Hann window but using a 81-tap filter.

MATLAB script for computation and plot:

```
M = 81; wc1 = pi/6; wc2 = pi/3; w_hann = hann(M)';
hd = ideal_lp(wc1,M) + ideal_lp(pi,M) - ideal_lp(wc2,M);
h = hd.*w_hann; H = freqz(h,1,om);
figure('units','inches','position',[0,0,7,2]);
stem(0:M-1,h,'filled','linewidth',1,'MarkerSize',3);
xlabel('Sample Index n'); ylabel('Amplitude');
title('Impulse Response'); grid; axis([-1,M,-0.25,1])
set(gca,'xtick',[0,(M-1)/2,M-1],'ytick',-0.25:0.25:1);
```



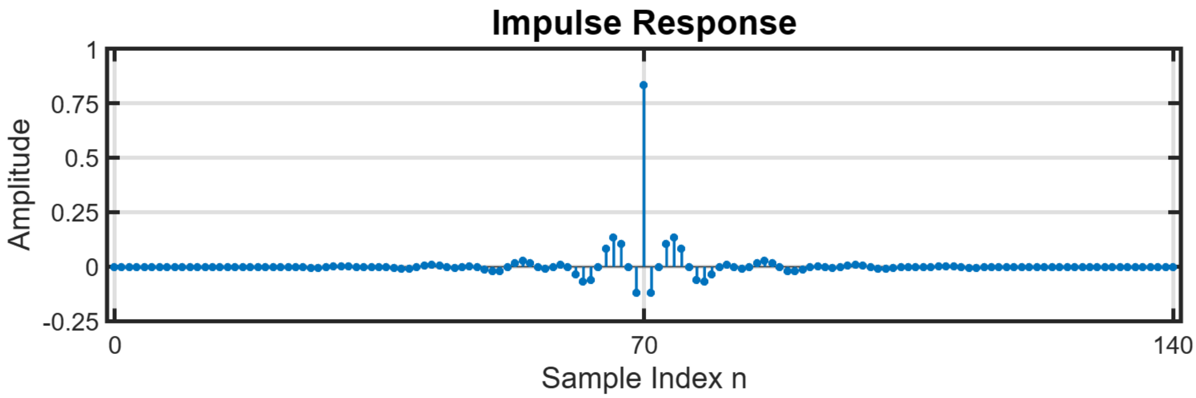

```
figure('Units','inches','position',[0,0,7,4.5]);
subplot(2,1,1); % Log-magnitude plot
Hmag = abs(H); Hdb = 20*log10(Hmag/max(Hmag));
plot(om/pi,Hdb,'linewidth',1.5); axis([0,1,-60,5]);
xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude response');
set(gca,'xtick',[0,1/6,1/3,1],'ytick',[-60,-44,0]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
subplot(2,1,2); % Phase plot
plot(om/pi,angle(H)/pi,'LineWidth',1.5); axis([0,1,-1,1])
xlabel('Normalized Frequency in \pi units');
ylabel('Phase in \pi units'); title('Phase response');
set(gca,'xtick',[0,1/6,1/3,1]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
```



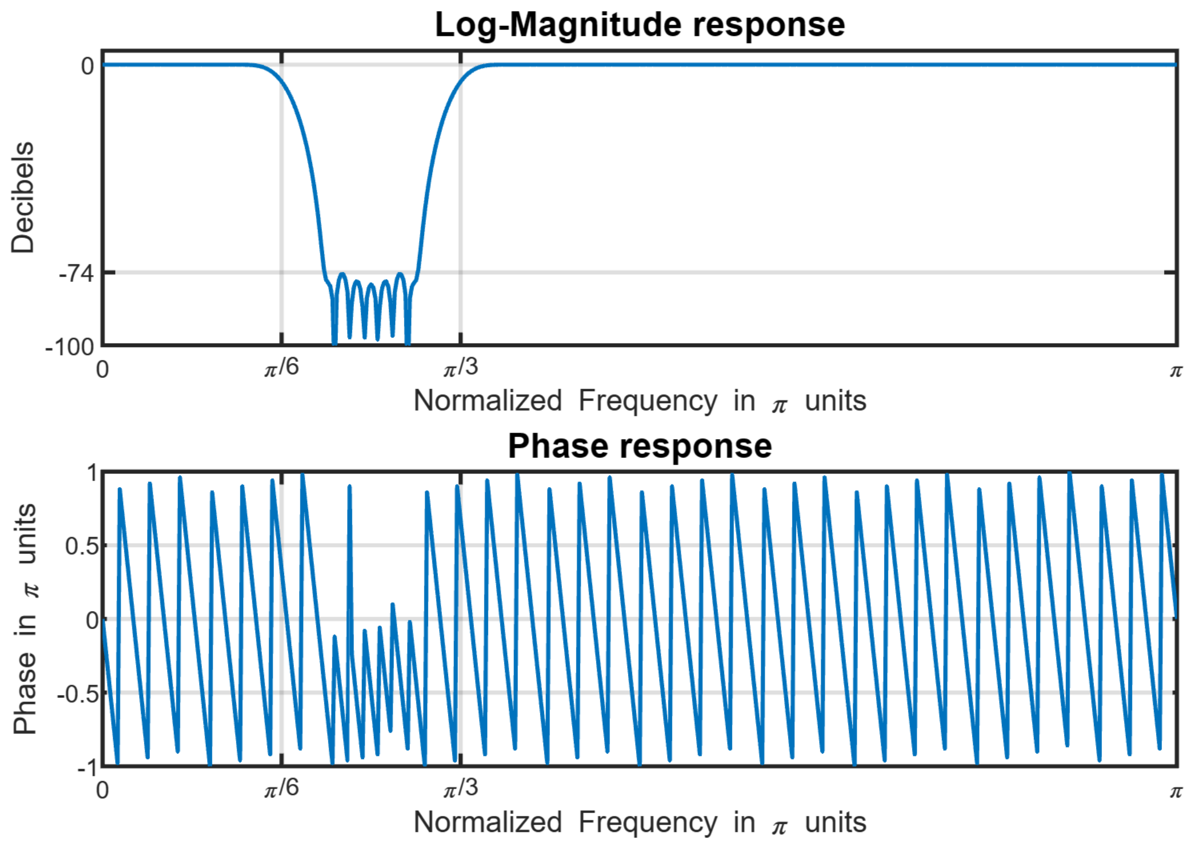
(d) Repeat parts (a) and (b) using the Blackman window but using a 151-tap filter.

MATLAB script for computation and plot:

```
M = 141; wc1 = pi/6; wc2 = pi/3; w_blk = blackman(M)';
hd = ideal_lp(wc1,M)+ideal_lp(pi,M)-ideal_lp(wc2,M);
h = hd.*w_blk;H = freqz(h,1,om);
figure('units','inches','position',[0,0,7,2]);
stem(0:M-1,h,'filled','linewidth',1,'MarkerSize',2);
xlabel('Sample Index n'); ylabel('Amplitude');
title('Impulse Response'); grid; axis([-1,M,-0.25,1])
set(gca,'xtick',[0,(M-1)/2,M-1],'ytick',-0.25:0.25:1);
```




```
figure('Units','inches','position',[0,0,7,4.5]);
subplot(2,1,1); % Log-magnitude plot
Hmag = abs(H); Hdb = 20*log10(Hmag/max(Hmag));
plot(om/pi,Hdb,'linewidth',1.5); axis([0,1,-100,5]);
xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude response');
set(gca,'xtick',[0,1/6,1/3,1],'ytick',[-100,-74,0]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
subplot(2,1,2); % Phase plot
plot(om/pi,angle(H)/pi,'LineWidth',1.5); axis([0,1,-1,1])
xlabel('Normalized Frequency in \pi units');
ylabel('Phase in \pi units'); title('Phase response');
set(gca,'xtick',[0,1/6,1/3,1]); grid;
set(gca,'xticklabel',{'0','\pi/6','\pi/3','\pi'});
```



(e) From your plots in partsa (b) through (d), can you verify the achieved minimum stopband attenuations as given in Table on Slide-14 in Powerpoint 10b? Explain clearly to get full credit.

Answer: In part (c) and part (c), the designed filter with Hann and Blackman window achieved the minimum steopband attenuations as given in the Table in Slides-10b, 44dB and 74 dB, respectively. However, the filter designed with rectangular window in part (b) didn't achieve the required one, which is 21dB. This is because of the lower order used in part (b).

Problem 6.3: Text Problem 10.7 (Page 744)

This is a revised version.

A linear-phase FIR filter of length $L = M + 1 = 15$ has a symmetric unit sample response $h(n)$ and an amplitude response $H_r(\omega)$ that satisfies the condition

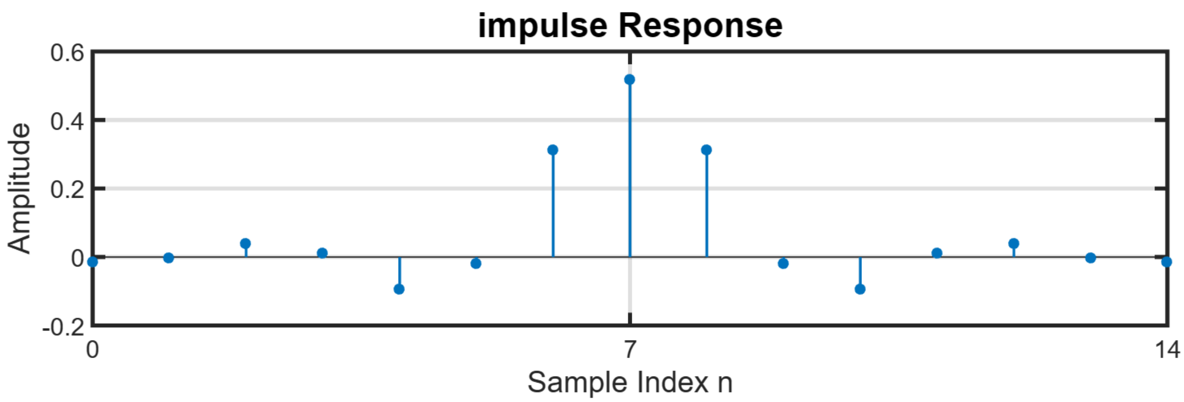
$$H_r\left(\frac{2\pi k}{15}\right) = H_r(\omega_k)\Big|_{\omega_k = \frac{2\pi k}{15}} = \begin{cases} 1, & k = 0, 1, 2, 3 \\ 0.4, & k = 4 \\ 0, & k = 5, 6, 7 \end{cases}$$

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

(a) Determine the coefficients of the sample response $h(n)$ and provide its **`stem`** plot.

MATLAB script for plot:

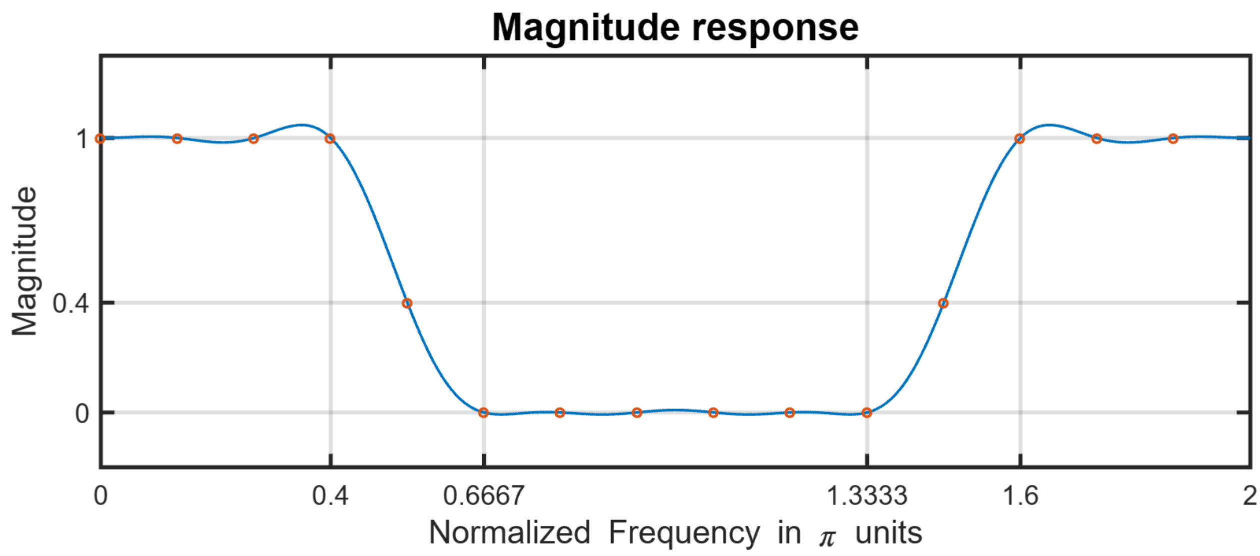
```
M = 14; L = M + 1; alpha = M/2;
Hrs = [ones(1, 4), 0.4, zeros(1,6), 0.4, ones(1, 3)];
angH = -alpha*2*pi/L.*(0:M);
H = Hrs.*exp(1j.*angH); h = real(ifft(H,L));
figure('units','inches','position',[0,0,7,2]);
stem(0:M,h,'filled','linewidth',1,'MarkerSize',3);
xlabel('Sample Index n'); ylabel('Amplitude');
set(gca,'xtick',[0,M/2,M],'ytick',-0.2:0.2:1);
title('impulse Response'); grid;
```



(b) Provide a plot of the amplitude response $H(\omega)$ over $0 \le \omega \le \pi$ and show the samples $H)k)$ as markers in the same plot.

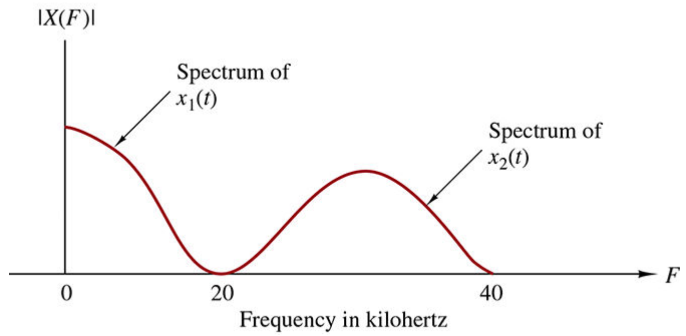
MATLAB script:

```
om = linspace(0, 2, 1001)*pi;
[A, ~, psi] = zerophase(h, 1, om);
figure('units','inches','position',[0,0,8,3]);
plot(om/pi,A,'linewidth',1,'MarkerSize',3)
hold on; plot((0:M)*2/L,Hrs, 'o' ,'linewidth',1,'MarkerSize',3);
xlabel('Normalized Frequency in \pi units');
ylabel('Magnitude'); title('Magnitude response'); axis([0 2 -0.2 1.3])
set(gca,'xtick',[0,3*2/L,5*2/L,10*2/L,12*2/L,2], 'ytick',[0,0.4,1]); grid;
```



Problem 6.4: Text Problem CP 10.1, parts (a), (b), and (c) only (Page 749)

An analog signal $x(t)$ consists of the sum of two components $x_1(t)$ and $x_2(t)$. The spectral characteristics of $x(t)$ are shown in the sketch in Fig. CP10.1. The signal is bandlimited to 40 kHz and it is sampled at a rate of 100 kHz to yield the sequence $x(n)$.



It is desired to suppress the signal $x_2(t)$ by passing the sequence $x(n)$ through a digital lowpass filter. The allowable amplitude distortion on $|X_1(f)|$ is $\pm 2\%$ $(\delta_1 = 0.02)$ over the range $0 \le |F| \le 15$ kHz. Above 20 kHz, the filter must have an attenuation of at least 40 dB $(\delta_2 = 0.01)$.

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

(a) Use the Parks-McClellan's algorithm to design the *minimum*-order linear-phase FIR filter that meets the specifications above. From the plot of the magnitude characteristic of the filter frequency response, provide the actual specifications achieved by the filter.

MATLAB script for computation and plot:

```
wp = 15/100*2*pi; ws = 20/100*2*pi;
```

```
delta1 = 0.02; delta2 = 0.01;
[M, f, m, weights] = firpmord([wp, ws]/pi, [1, 0], [delta1, delta2]);
[h,delta] = firpm(M, f, m, weights); M, delta, delta1
```

```
M = 35
delta = 0.0231
delta1 = 0.0200
```

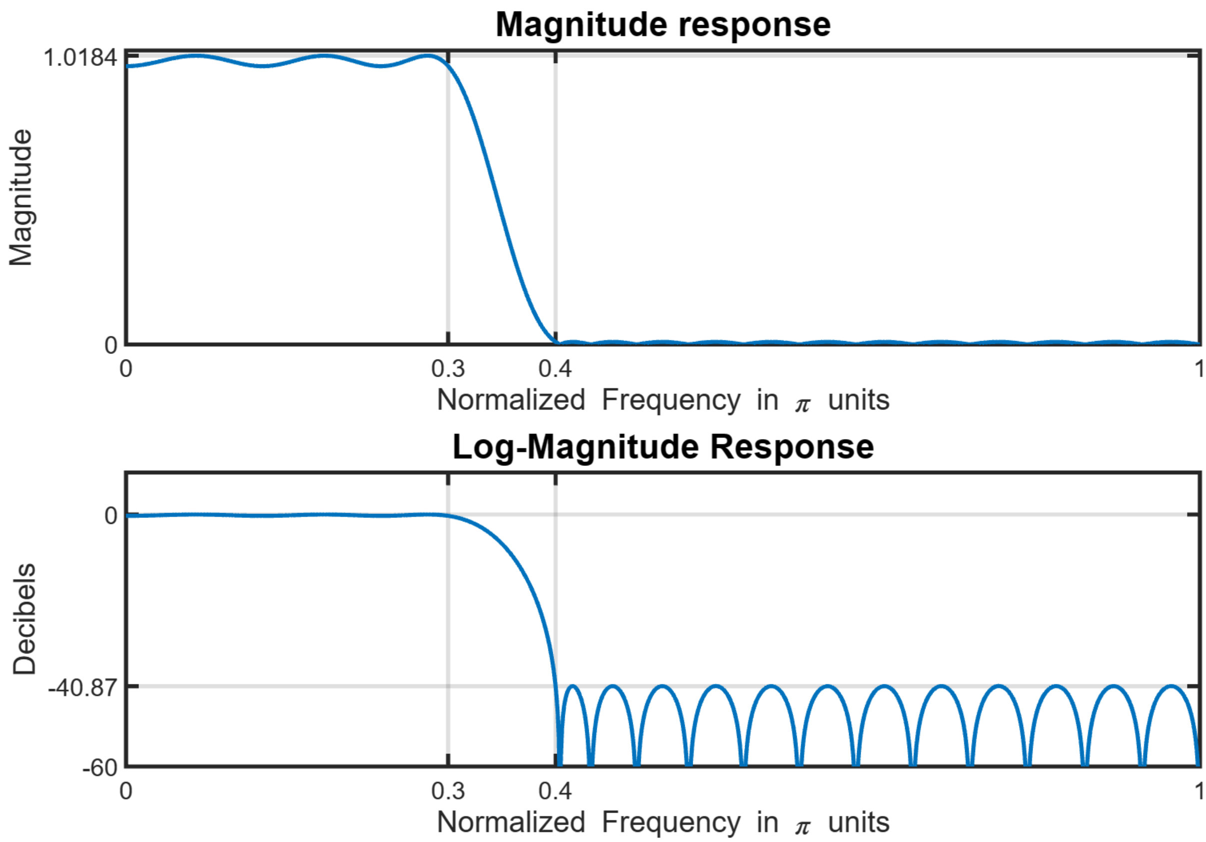
```
M = M+1; [h,delta] = firpm(M, f, m, weights); M, delta, delta1
```

```
M = 36
delta = 0.0212
delta1 = 0.0200
```

```
M = M+1; [h,delta] = firpm(M, f, m, weights); M, delta, delta1
```

```
M = 37
delta = 0.0184
delta1 = 0.0200
```

```
om = linspace(0,1,1001)*pi; H = freqz(h,1,om);
Hmag = abs(H); HdB = 20*log10(Hmag/max(Hmag));
deltap = delta; % achieved passband ripple
As = -20*log10((deltap/2)/(1+deltap)); % achieved stopband attenuation in dB
figure('units','inches','position',[0,0,7,4.5]);
subplot(2,1,1); % magnitude plot
plot(om/pi, Hmag,'LineWidth',1.5); axis([0,1,0,1+2*deltap]);
xlabel('Normalized Frequency in \pi units');
ylabel('Magnitude'); title('Magnitude response'); grid on
set(gca,'xtick',[0,0.3,0.4,1],'ytick',[0,1+deltap]);
subplot(2,1,2); % log-magnitude plot
plot(om/pi, HdB,'LineWidth',1.5); axis([0,1,-60,10])
xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude Response'); grid on
set(gca,'xtick',[0,0.3,0.4,1],'ytick',[-60,-round(As,2),0]);
```



From the plots, with filter order $M = 37$, the actual passband ripple is $\delta_p = 0.0184$ and the stopband attenuation is $A_s = 40.87$ dB, The passband and stopband edge frequencies are exact at $\omega_p = 0.3\pi$ and $\omega_s = 0.4\pi$, respectively.

(b) Compare the order M obtained in part (a) with the approximate formulas given in equations (10.2.89) and (10.2.90) of the textbook.

Solution:

$$\hat{M} = \frac{-20\log_{10}(\sqrt{\delta_1\delta_2}) - 13}{14.6(w_s - w_p)/2\pi} + 1 = 33.8626 \approx 34$$

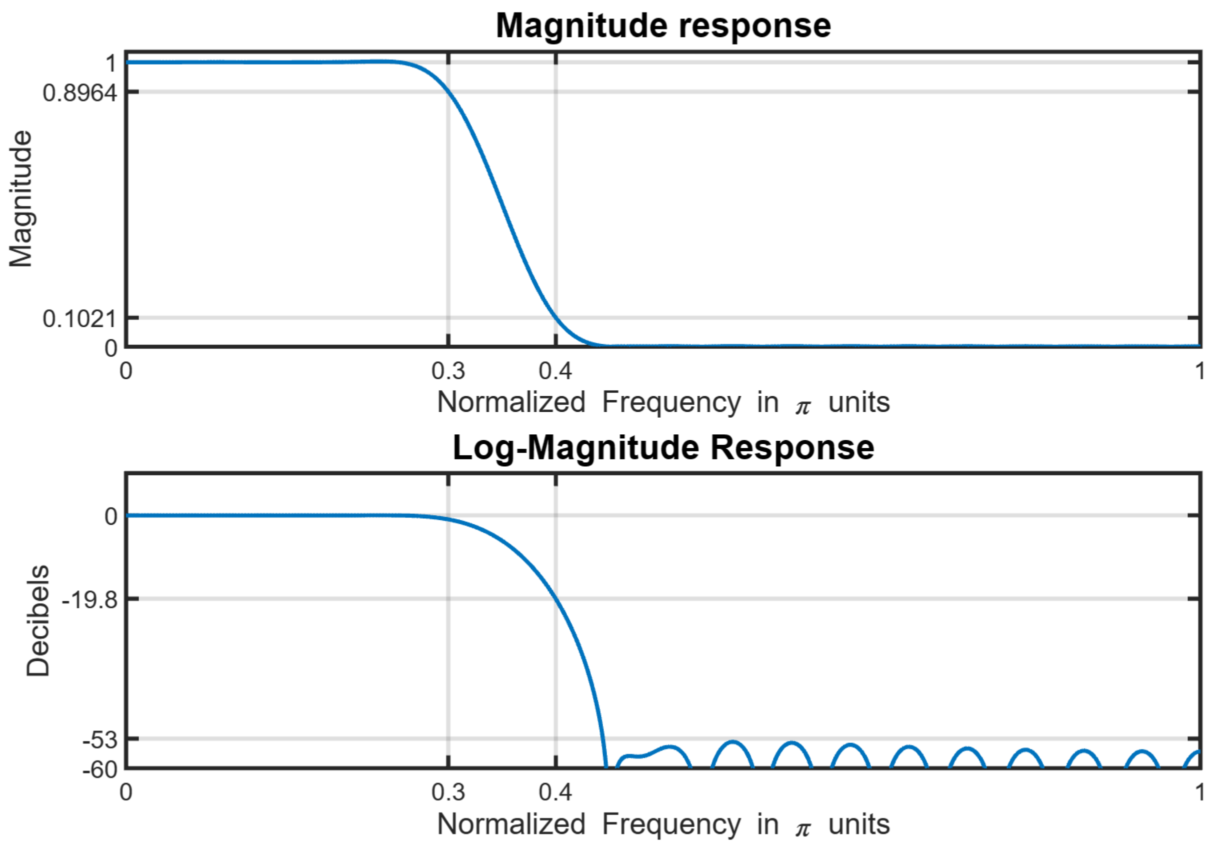
$$\hat{M} = \frac{D_\infty(\delta 1, \delta 2) - f(\delta 1, \delta 2)((w_s - w_p)/2\pi)^2}{(w_s - w_p)/2\pi} + 1 = 35.1 \approx 35$$

Clearly, the second formula due to Herrman is more accurate since the actual order is 37.

(c) For the order M obtained in part (a), design an FIR digital lowpass filter using the window technique and the Hamming window. Compare the frequency response characteristics of this design with those obtained in part (a).

MATLAB script for design and plot:

```
M = 37; wc = mean([15/100*2*pi, 20/100*2*pi]);
w_hann = hamming(M)'; hd = ideal_lp(wc, M); h = hd.*w_hann;
H = freqz(h,1,om); Hmag = abs(H); HdB = 20*log10(Hmag/max(Hmag));
deltapW = 1-Hmag(301); % achieved passband ripple by Hamming design
deltasW = Hmag(401); % achieved stopband ripple by Hamming design
AsW = -20*log10(deltasW); % achieved stopband attenuation in dB
figure('units','inches','position',[0,0,7,4.5]);
subplot(2,1,1); % Magnitude plot
plot(om/pi, Hmag,'LineWidth',1.5); axis([0,1,0,1+2*deltap])
xlabel('Normalized Frequency in \pi units');
ylabel('Magnitude'); title('Magnitude response');
set(gca,'xtick',[0,0.3,0.4,1],'ytick',[0,deltasW,1-deltapW,1]); grid on
subplot(2,1,2); % Kog-magnitude plot
plot(om/pi, HdB,'LineWidth',1.5);
xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude Response');
set(gca,'xtick',[0,0.3,0.4,1],'ytick',[-60,-53,-round(AsW,1),0]);
grid on; ylim([-60, 10]);
```



For the given filter order of $M = 37$, the above plots show that the achieved passband ripple is $0.1036 < 0.01$, which is the required one. Similarly, even though the Hamming window provides 53 db of attenuation, this filter obtained only 19.8 dB of stopband attenuation, well below the required 40 dB attenuation.

Conclusion: The filter designed with hamming window doesn't satisfy the required frequency-domain specifications.

Problem 6.5: Text Problem CP10.10 (Page 753)

A highpass FIR filter has the following specifications:

stopband edge frequency: $\omega_s = 0.6\pi$, passband edge frequency: $\omega_s = 0.75\pi$, passband ripple: 0.5dB, stopband attenuation: 50dB.

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

(a) Design a minimum length linear-phase FIR filter using the Parks-McClellan's algorithm. Do not use the **`fir1`** function. Provide a **`stem`** plot of the impulse response.

MATLAB script for design and plot:

```
ws = 0.6*pi; wp = 0.75*pi; Rp = 0.5; As =50;
[delta1, delta2] = db2delta(Rp, As);
[N, f, m, weights] = firpmord([ws, wp]/pi, [0,1], [delta2, delta1]);
N
```

```
N = 26
```

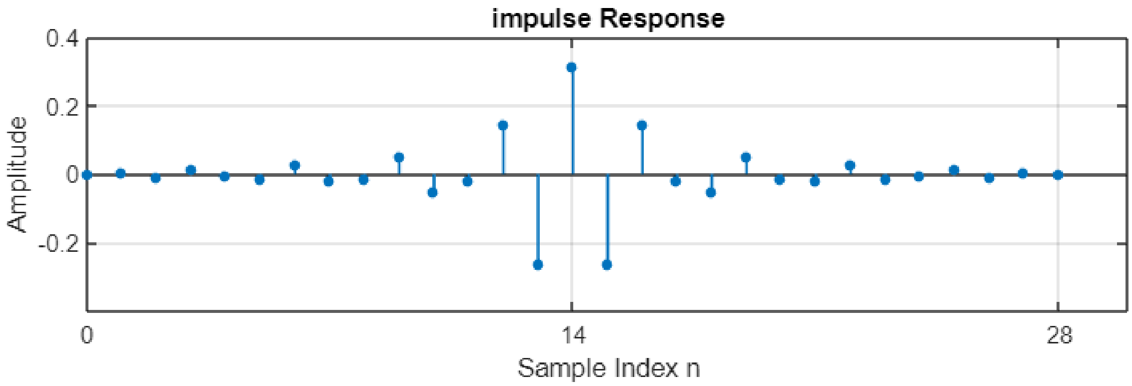
```
h = firpm(N, f, m, weights);
[db, mag, pha, grd, w] = freqz_m(h, 1);
wsi = ws/(2*pi/1e3); Asd = -max(db(1:wsi))
```

```
Asd = 49.5918
```

```
N = N+2;h = firpm(N, f, m, weights);
[db, mag, pha, grd, w] = freqz_m(h, 1);
wsi = ws/(2*pi/1e3); Asd = -max(db(1:wsi))
```

```
Asd = 50.2253
```

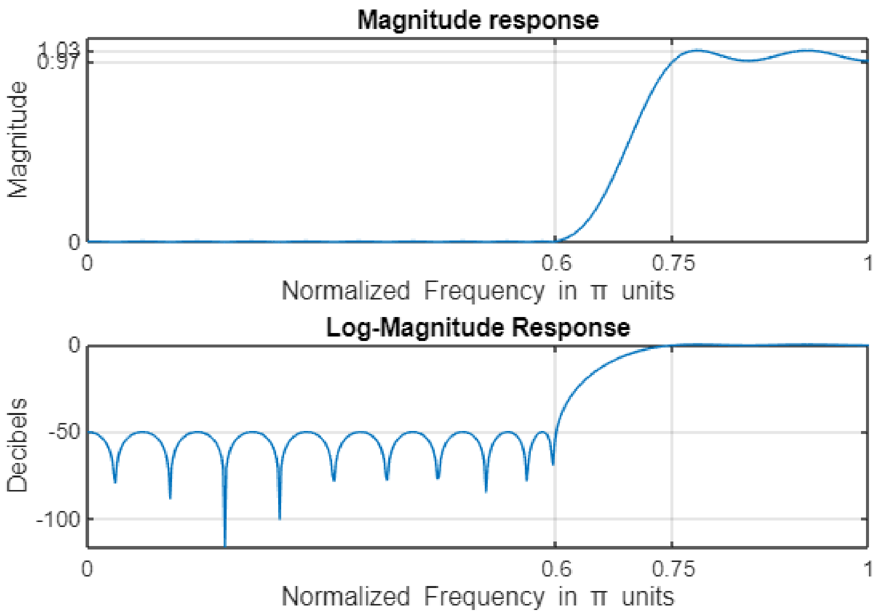
```
figure('units','inches','position',[0,0,7,2]);
stem(0:N,h,'filled','linewidth',1,'MarkerSize',3);
xlabel('Sample Index n'); ylabel('Amplitude');
set(gca,'xtick',[0,N/2,N],'ytick',-0.2:0.2:1);
title('impulse Response'); grid;
```



(b) Provide plots of the magnitude and log-magnitude (dB) responses in two rows and one column subplots over $0 \le \omega \le \pi$.

MATLAB script for plots:

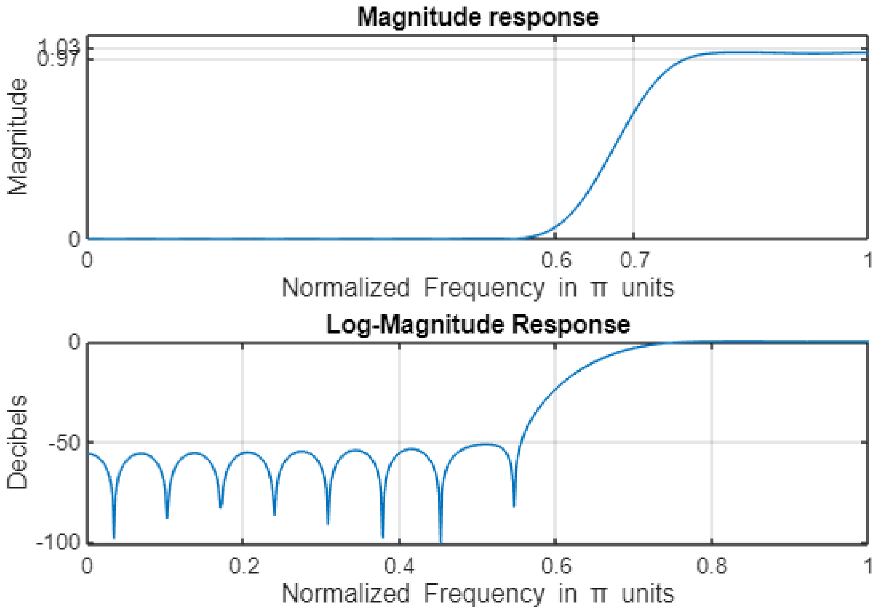
```
om = linspace(0, 1, 501)*pi;
figure('PaperPosition',[0,0,7,4.5]*72,'position',[0,0,7,4.5]*72);
subplot(2,1,1); % stem plots of magnitudes
plot(om/pi, mag); xlabel('Normalized Frequency in \pi units');
ylabel('Magnitude'); title('Magnitude response');
set(gca,'xtick',[0,0.6, 0.75,1], 'ytick',[0,0.97, 1.03]); grid;
axis([0 1 0 1.1])
subplot(2,1,2); % stem plots of magnitudes
plot(om/pi, db); xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude Response'); grid on;
set(gca,'xtick',[0,0.6,0.75,1],'ytick',[-100,-50,0]);
```



(c) Verify your design using the **`fir1`** function.

MATLAB script for design and plots:

```
h = fir1(28,mean([ws, wp]/pi), 'high');
[db, mag, pha, grd, w] = freqz_m(h, 1);
figure('PaperPosition',[0,0,7,4.5]*72,'position',[0,0,7,4.5]*72);
subplot(2,1,1); % stem plots of magnitudes
plot(om/pi, mag); xlabel('Normalized Frequency in \pi units');
ylabel('Magnitude'); title('Magnitude response');
set(gca,'xtick',[0,0.6, 0.7,1], 'ytick',[0,0.97, 1.03]); grid;
axis([0 1 0 1.1])
subplot(2,1,2); % stem plots of magnitudes
plot(om/pi, db); xlabel('Normalized Frequency in \pi units');
ylabel('Decibels'); title('Log-Magnitude Response'); grid on
```



Problem 6.6: Bandstop filter using the frequency-sampling technique

Design a bandstop filter using the frequency-sampling technique. The specifications are: $\omega_{p_l} = 0.3\pi$, $\delta_{p_l} = 0.02$, $\omega_{s_l} = 0.4\pi$, $\omega_{s_u} = 0.6\pi$, $\delta_s = 0.032$, $\omega_{p_u} = 0.7\pi$, and $\delta_{p_u} = 0.02$.

(a) Choose the length of the filter so that there are approximately **two samples** in the transition band. Obtain the impulse response of the filter using raised-cosine values for the transition-band samples. Plot the impulse response.

Solution: The minimum (normalized) transition bandwidth is

$$f_{\rm tr} = \min\Big(\big(\omega_{s_l} - \omega_{p_l}\big)/\pi, \big(\omega_{p_u} - \omega_{s_u}\big)/\pi\Big) = \min(0.1, 0.1) = 0.1.$$

Within this bandwidth, we want two samples. Assuming that we can have frequency samples on the band edges, the maximum interval between two consecutive samples should be $\Delta f = f_{\rm tr}/3 = 0.1/3$ (since two samples create three intervals). Let $L = M - 1$ be the length of the filter where M is the filter order. Then the total number of frequency samples on the full normalized band, $(2\pi)/\pi = 2$, is also equal to L, that is, $L \cdot \Delta f = 2$ or $L = 2/\Delta f = (2)\Big(\frac{3}{0.1}\Big) = 60$. Since this is a bandstop filter, it must not be a type-2 linear-phase FIR filter, that is, L must be an odd integer. Hence, we select $L = 61$.

MATLAB script and plots:

Given specifications

```
clear; close all; clc;
ompl = 0.3*pi; omsl = 0.4*pi; omsu = 0.6*pi; ompu = 0.7*pi;
deltapl = 0.02; deltas = 0.032; deltapu = 0.02;
```

Computed parameters

```
fpl = ompl/pi; fsl = omsl/pi; fsu = omsu/pi; fpu = ompu/pi;
ftr = min(fsl-fpl,fpu-fsu); % normalized transition bandwidth
Df = ftr/3; % max interval between two consecutive frequency samples
L = 2/Df; L = 2*floor(L/2)+1; % Make length an odd integer
M = L-1; % Filter order
Df = 2/L; % Delta f for new (odd) L
deltap = min(deltapl,deltapu); % will design for minimum PB ripple
[Ap,As] = speConversion(deltap,deltas,'abs','rel'); % decibel specs
fprintf('Passband ripple = %4.2f dB\n',Ap);
```

```
Passband ripple = 0.35 dB
```

```
fprintf('Stopband Attenuation = %g dB',round(As));
```

```
Stopband Attenuation = 30 dB
```

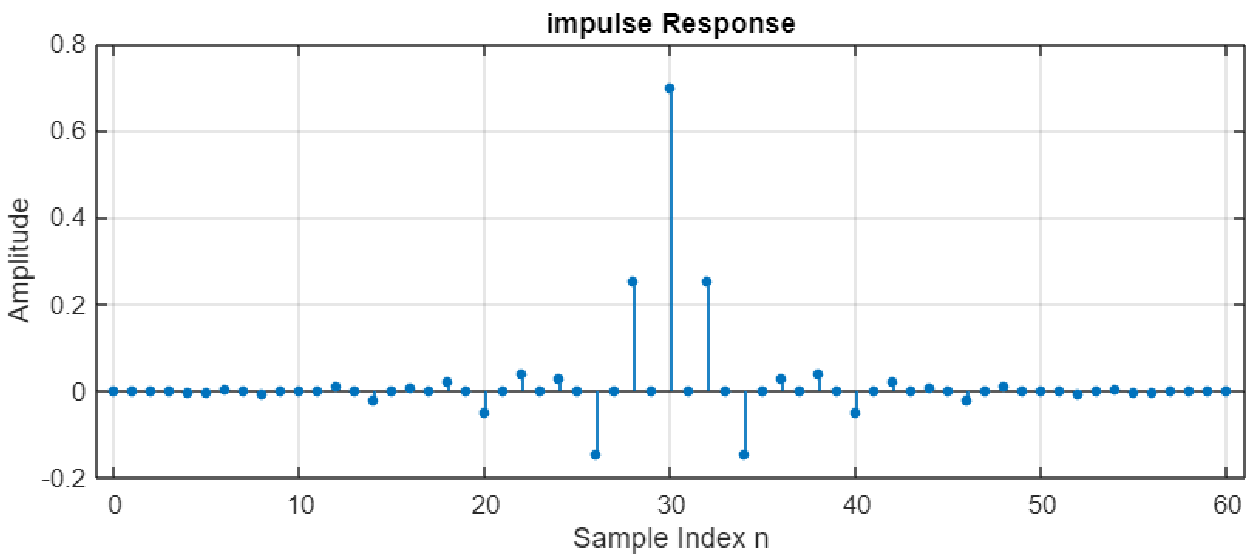
Sample desired amplitude/phase responses: Cosine transition bands

```
f1 = fpl; f2 = fsl; f3 = fsu; f4 = fpu; % band edges in [0,1]
f5 = 2-f4; f6 = 2-f3; f7 = 2-f2; f8 = 2-f1; % band edges in [1,2]
k1 = floor(f1/Df); k2 = ceil(f2/Df); % indexes of band edges
k3 = floor(f3/Df); k4 = ceil(f4/Df); % indexes of band edges
k5 = floor(f5/Df); k6 = ceil(f6/Df); % indexes of band edges
k7 = floor(f7/Df); k8 = ceil(f8/Df); % indexes of band edges
ktr1 = k1+1:k2-1; ktr2 = k3+1:k4-1; % sample indexes in transition bands
ktr3 = k5+1:k6-1; ktr4 = k7+1:k8-1; % sample indexes in transition bands
ftr1 = ktr1*Df; ftr2 = ktr2*Df; % frequencies in transition bands
ftr3 = ktr3*Df; ftr4 = ktr4*Df; % frequencies in transition bands
Ad = [ones(1,k1+1),...
    0.5+0.5*cos(pi*((ftr1-f1)/(f2-f1))),...
```

```
    zeros(1,k3-k2+1),...
    0.5+0.5*cos(pi*((f4-ftr2)/(f4-f3))),...
    ones(1,k5-k4+1),...
    0.5+0.5*cos(pi*((ftr3-f5)/(f6-f5))),...
    zeros(1,k7-k6+1),...
    0.5+0.5*cos(pi*((f8-ftr4)/(f8-f7))),...
    ones(1,L-k8)];
Q = floor(M/2); % last index of half array
psid = [-(M/2)*Df*pi*(0:Q),(M/2)*Df*pi*(L-(Q+1:M))];
```

Desired filter's frequency response samples and impulse response

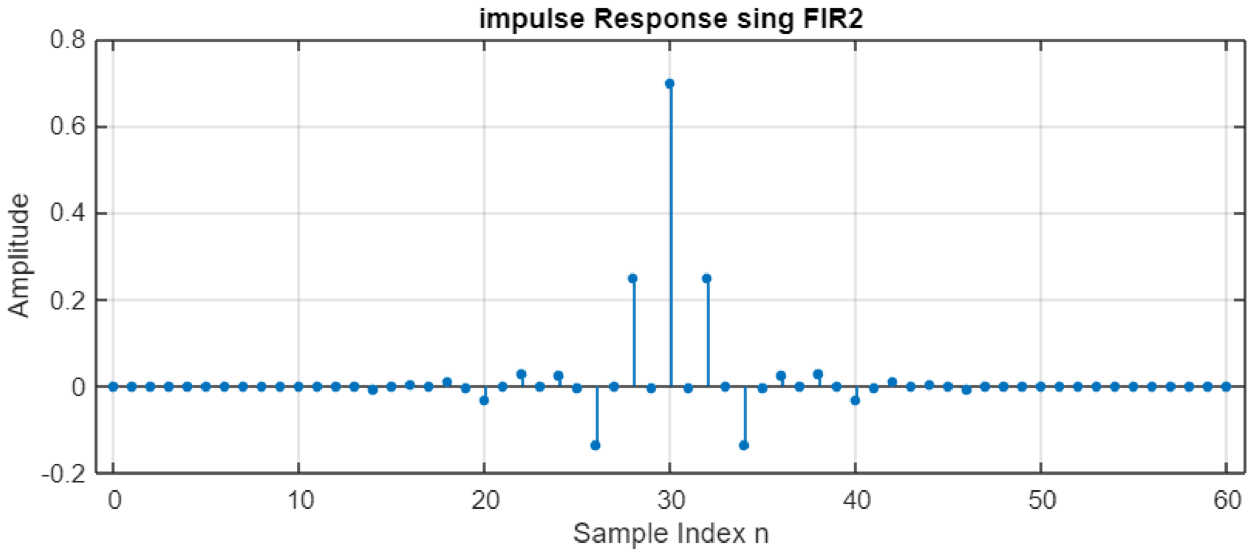
```
Hd = Ad.*exp(1j*psid);
hfs = (real(ifft(Hd))).*rectwin(L)';
figure('units','inches','position',[0,0,8,3]);
stem(0:M,hfs,'filled','linewidth',1,'MarkerSize',3); axis([-1,L,-0.2,0.8]);
xlabel('Sample Index n'); ylabel('Amplitude');
title('impulse Response'); grid;
```



(b) Determine and plot the impulse response of the bandstop filter (with the given specification) using the **`fir2`** function (choose an appropriate window).

Solution: Since the minimum stopband attenuation is $50\,\mathrm{dB}$, we will use the Hamming window.

```
hfsHam = fir2(M,[0,f1,f2,f3,f4,1],[1,1,0,0,1,1]);
figure('units','inches','position',[0,0,8,3]);
stem(0:M,hfsHam,'filled','linewidth',1,'MarkerSize',3);
axis([-1,L,-0.2,0.8]);
xlabel('Sample Index n'); ylabel('Amplitude');
title('impulse Response sing FIR2'); grid;
```

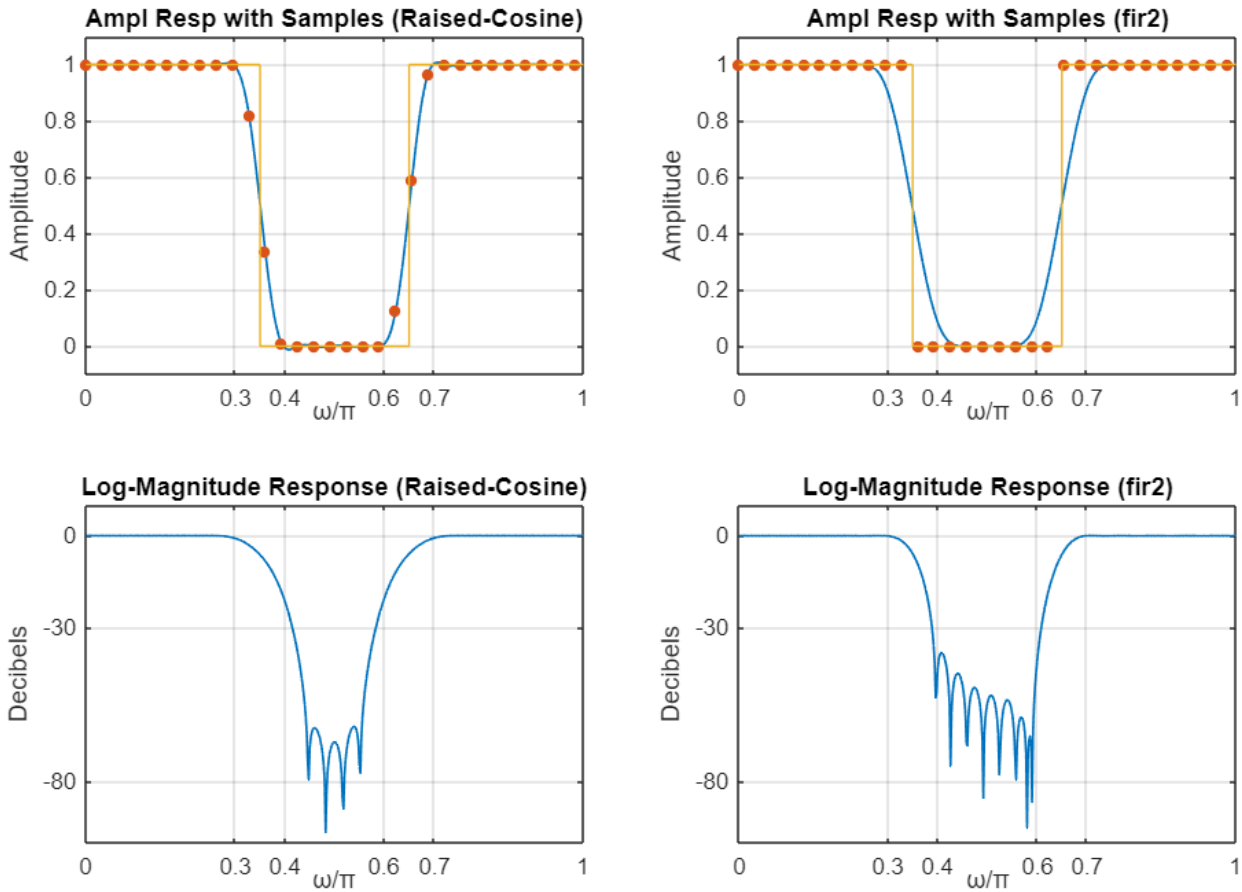


(c) Provide a plot of amplitude and log-magnitude (dB) responses using two rows and one column subplots.

MATLAB script and plot:

```
N = 500; om = linspace(0,1,N+1)*pi; % freq array for plots
f12 = (f1+f2)/2; f34 = (f3+f4)/2; % transition band cutoff freq for plotting
k12 = floor(f12/Df); k34 = floor(f34/Df);
Adideal = [ones(1,k12+1),zeros(1,k34-k12),ones(1,M/2-k34)];
Ha = freqz(hfsHam,1,om); magHa = abs(Ha); % Magnitude response
magadb = 20*log10(magHa/max(magHa)); % Log-magnitude response in dB
[Hra,om] = zerophase(hfs,1,om); % Amplitude response
Hb = freqz(hfs,1,om); magHb = abs(Hb); % Magnitude response
magbdb = 20*log10(magHb/max(magHb)); % Log-magnitude response in dB
[Hrb,om] = zerophase(hfsHam,1,om); % Amplitude response
figure('units','inches','position',[0,0,9,6]);
subplot(2,2,1); % Plot of Amplitude response with frequency samples in (a)
plot(om/pi,Hra,'linewidth',1); hold on; axis([0,1,-0.1,1.1]);
plot((0:M/2)*Df,Ad(1:M/2+1),'.','markersize',15);
plot([0,f12,f12,f34,f34,1],[1,1,0,0,1,1],'linewidth',0.5);
xlabel('\omega/\pi','VerticalAlignment','middle'); ylabel('Amplitude');
title('Ampl Resp with Samples (Raised-Cosine)')
set(gca,'xtick',[0,f1,f2,f3,f4,1]); grid; hold off;
subplot(2,2,2); % Plot of Amplitude response with frequency samples in (b)
plot(om/pi,Hrb,'linewidth',1); hold on; axis([0,1,-0.1,1.1]);
plot((0:M/2)*Df,Adideal,'.','markersize',15);
plot([0,f12,f12,f34,f34,1],[1,1,0,0,1,1],'linewidth',0.5);
xlabel('\omega/\pi','VerticalAlignment','middle'); ylabel('Amplitude');
title('Ampl Resp with Samples (fir2)');
set(gca,'xtick',[0,f1,f2,f3,f4,1]); grid; hold off;
subplot(2,2,3); % Plot of log-magnitude response for filter in (a)
plot(om/pi,magadb,'linewidth',1); axis([0,1,-100,10]);
xlabel('\omega/\pi','VerticalAlignment','middle'); ylabel('Decibels');
```

```
title('Log-Magnitude Response (Raised-Cosine)');
set(gca,'xtick',[0,f1,f2,f3,f4,1],'ytick',[-80,-30,0]); grid;
subplot(2,2,4); % Plot of log-magnitude response for filter in (b)
plot(om/pi,magbdb,'linewidth',1); axis([0,1,-100,10]);
xlabel('\omega/\pi','VerticalAlignment','middle'); ylabel('Decibels');
title('Log-Magnitude Response (fir2)');
set(gca,'xtick',[0,f1,f2,f3,f4,1],'ytick',[-80,-30,0]); grid;
```



(d) Provide an assessment of your results based on the above plot.

Answer: The frequency sampling design using raised cosine did not satisfy the stopband attenuation specification but the design using **`fir2`** along with Hamming window did.

Problem 6.7: Amplitude Response of Type-III FIR Filter

Consider the type-III linear-phase FIR filter characterized by antisymmetric impulse response and even order M:

$$h(n) = -h(M-n), \quad 0 \le n \le M; \qquad M \sim \text{ even} \qquad (6.7.1)$$

The frequency response of the above filter can be expressed as $H(\omega) = A(\omega)\,e^{\jmath\Psi(\omega)}$, where $A(\omega)$ is the amplitude response and $\Psi(\omega) = \beta - \alpha\omega$ is the generalized linear phase response.

(a) Show that the amplitude response $A(\omega)$ and the generalized linear phase response $\Psi(\omega)$ are given by

$$\begin{aligned} A(\omega) &= \sum_{k=1}^{M/2} c(k)\sin(\omega k); \quad \text{where} \quad c(k) = 2h\Big(\frac{M}{2} - k\Big), \\ \Psi(\omega) &= \frac{\pi}{2} - \frac{M}{2}\omega \end{aligned} \tag{6.7.2}$$

Solution: First note from (6.7.1) that $H(M/2) = 0$. Now consider,

$$H(\omega) = \sum_{n=0}^{M} h(n)e^{-\jmath\omega n} = \sum_{n=0}^{(M/2-1)} h(n)e^{-\jmath\omega n} + \sum_{(M/2)+1}^{M} h(n)e^{-\jmath\omega n}$$

The following change of variable

$$n \to (M-n) \Rightarrow \Big(\frac{M}{2}+1\Big) \to \Big(\frac{M}{2}-1\Big),\ M \to 0,\ \text{ and }\ h(n) \to -h(n)$$

in the second sum on the right-hand side above gives

$$\begin{aligned} H(\omega) &= \sum_{n=0}^{(M/2)-1} h(n)e^{-\jmath\omega n} - \sum_{n=0}^{(M/2)-1} h(n)e^{-\jmath\omega(M-n)} \\ &= e^{-\jmath\omega(M/2)} \sum_{n=0}^{(M/2)-1} h(n)\{e^{-\jmath\omega n + \jmath\omega(M/2)} - e^{-\jmath\omega(M-n)+\jmath\omega(M/2)}\} \\ &= e^{-\jmath\omega(M/2)} \sum_{n=0}^{(M/2)-1} h(n)\{e^{+\jmath\omega(M/2-n)} - e^{-\jmath\omega(M/2-n)}\} \\ &= \jmath e^{-\jmath\omega(M/2)} \sum_{n=0}^{(M/2)-1} 2h(n)\sin\Big(\Big[\frac{M}{2} - n\Big]\omega\Big) \end{aligned} \tag{6.7.3}$$

Another change of variable gives

$$\Big(\frac{M}{2} - n\Big) \to k \Rightarrow \begin{cases} n = 0 & \to k = \dfrac{M}{2} \\ n = \dfrac{M}{2} - 1 & \to k = 1 \end{cases} \quad \text{and} \quad \sin\Big(\Big[\frac{M}{2} - n\Big]\omega\Big) \to \sin(\omega k)$$

Substituting in (6.7.3), we obtain

$$H(\omega) = \jmath e^{-\jmath\omega(M/2)} \sum_{k=1}^{M/2} 2h\Big(\frac{M}{2} - k\Big)\sin(\omega k)$$

Define $c(k) \triangleq 2h\Big(\frac{M}{2} - k\Big),\ k = 1, 2, \ldots, M/2$. Then, using $\jmath = e^{\jmath\pi/2}$, we obtain

$$H(\omega) = e^{j\left[\frac{\pi}{2}-\omega\left(\frac{M}{2}\right)\right]} \sum_{k=1}^{M/2} c(k)\sin(\omega k) \quad \Rightarrow \quad A(\omega) = \sum_{k=1}^{M/2} c(k)\sin(\omega k) \text{ and } \Psi(\omega) = \frac{\pi}{2} - \omega\left(\frac{M}{2}\right) \tag{6.7.4}$$

This verifies (6.7.2).

(b) Show that the amplitude response $A(\omega)$ can be further expressed as

$$A(\omega) = \sin(\omega) \sum_{k=0}^{M/2-1} \tilde{c}(k)\cos(\omega k) \tag{6.7.5}$$

with coefficients $\tilde{c}(k)$ given by the recursive algorithm (recursing backwards)

$$\begin{aligned}
\tilde{c}\left(\frac{M}{2}-1\right) &= 2c\left(\frac{M}{2}\right) \\
\tilde{c}\left(\frac{M}{2}-2\right) &= 2c\left(\frac{M}{2}-1\right) \\
\tilde{c}(k) &= 2c(k+1) + \tilde{c}(k+2), \quad k = \frac{M}{2} - 3 : -1 : 1 \\
\tilde{c}(0) &= c(1) + \frac{1}{2}\tilde{c}(2)
\end{aligned} \tag{6.7.6}$$

Solution: To convert (6.7.4) into (6.7.5), we start with (6.7.5) and use the following trigonometric identity

$$\cos(A)\sin(B) = \frac{1}{2}\left[\sin(A+B) - \sin(A-B)\right], \tag{6.7.7}$$

to simplify to obtain a trigonometric polynomial in $\sin(k\omega)$, and then identify the coefficients with those in (6.7.4) to prove (6.7.6). Using (6.7.7) in the right-hand side of (6.7.5), we obtain

$$\begin{aligned}
\sin(\omega) \sum_{k=0}^{M/2-1} \tilde{c}(k)\cos(\omega k) &= \sum_{k=0}^{M/2-1} \tilde{c}(k)\sin(\omega)\cos(\omega k) \\
&= \frac{1}{2} \sum_{k=0}^{M/2-1} \tilde{c}(k)\Big\{ \sin\big((k+1)\omega\big) - \sin\big((k-1)\omega\big) \Big\} \\
\sum_{k=1}^{M/2} c(k)\sin(\omega k) &= \frac{1}{2} \sum_{k=0}^{M/2-1} \tilde{c}(k)\sin\big((k+1)\omega\big) - \frac{1}{2} \sum_{k=0}^{M/2-1} \tilde{c}(k)\sin\big((k-1)\omega\big)
\end{aligned} \tag{6.7.8}$$

Making change of variable $k \to (k-1)$ in the first sum on the right-hand side in (6.7.8) and $k \to (k+1)$ in the second sum on the right-hand side in (6.7.8), we obtain

$$
\begin{aligned}
\sum_{k=1}^{M/2} c(k)\sin(\omega k) &= \frac{1}{2}\sum_{k=1}^{M/2}\tilde{c}(k-1)\sin\left(k\omega\right) - \frac{1}{2}\sum_{k=-1}^{M/2-2}\tilde{c}(k+1)\sin\left(k\omega\right) \\
&= \frac{1}{2}\tilde{c}(0)\sin(\omega) + \frac{1}{2}\sum_{k=2}^{M/2-2}\tilde{c}(k-1)\sin(\omega k) + \frac{1}{2}\tilde{c}\left(\frac{M}{2}-2\right)\sin\left(\omega\left(\frac{M}{2}-1\right)\right) \\
&\quad + \frac{1}{2}\tilde{c}\left(\frac{M}{2}-1\right)\sin\left(\left(\frac{M}{2}\right)\omega\right) - \frac{1}{2}\tilde{c}(0)\sin(-\omega) - \frac{1}{2}\tilde{c}(1)\sin(0) \\
&\quad - \frac{1}{2}\tilde{c}(2)\sin(\omega) - \frac{1}{2}\sum_{k=2}^{M/2-2}\tilde{c}(k+1)\sin(\omega k) \\
\sum_{k=1}^{M/2} c(k)\sin(\omega k) &= \left(\tilde{c}(0) - \frac{1}{2}\tilde{c}(2)\right)\sin(\omega) + \frac{1}{2}\sum_{k=2}^{M/2-2}\left(\tilde{c}(k-1) - \tilde{c}(k+1)\right)\sin(\omega k) \\
&\quad + \frac{1}{2}\tilde{c}(M/2-2)\sin\left(\omega(M/2-1)\right) + \frac{1}{2}\tilde{c}(M/2-1)\sin\left((M/2)\omega\right)
\end{aligned} \qquad (6.7.9)
$$

Now comparing the coefficients of the harmonic terms of $\sin(\omega k)$ on both sides of $(6.7.9)$ we obtain the desired result by recursing backwards, that is, starting with $\tilde{c}[M/2-1]$ and ending with $\tilde{c}[0]$, as follows:

$$
\begin{aligned}
c\left(\frac{M}{2}\right) &= \frac{1}{2}\tilde{c}\left(\frac{M}{2}-1\right) && \Rightarrow \quad \tilde{c}\left(\frac{M}{2}-1\right) = 2c\left(\frac{M}{2}\right), && k = \frac{M}{2}-1 \\
c\left(\frac{M}{2}-1\right) &= \frac{1}{2}\tilde{c}\left(\frac{M}{2}-2\right) && \Rightarrow \quad \tilde{c}\left(\frac{M}{2}-2\right) = 2c\left(\frac{M}{2}-1\right), && k = \frac{M}{2}-2 \\
c(k) &= \frac{1}{2}\left(\tilde{c}(k-1) - \tilde{c}(k+1)\right) && \Rightarrow \quad \tilde{c}(k) = 2c(k+1) + \tilde{c}(k+2), && k = \frac{M}{2}-3:-1:1 \\
c(1) &= \tilde{c}(0) - \frac{1}{2}\tilde{c}(2) && \Rightarrow \quad \tilde{c}(0) = c(1) + \frac{1}{2}\tilde{c}(2), && k = 0
\end{aligned} \qquad (6.7.9)
$$

(c) Let $h[n] = \{\underset{\uparrow}{1}, 2, 3, 4, 0, -4, -3, -2, -1\}$ be the impulse response of a type-III linear-phase FIR filter. Determine the coefficient arrays $\{c[k]\}_{k=1}^{4}$ and $\{\tilde{c}[k]\}_{k=0}^{3}$, using results from parts (a) and (b), respectively. Verify your results by plotting and comparing the two resulting amplitude responses.

Solution:

```
clc; close all; clear;
h = [1:4,0,-4:-1]; M = length(h)-1;
c = 2*h(M/2:-1:1); disp(c); % part (a) coefficients
```

```
     8     6     4     2
```

Thus, the amplitude response can be expressed as

$$
A(\omega) = 8\sin(\omega) + 6\sin(2\omega) + 4\sin(3\omega) + 2\sin(4\omega). \qquad (6.7.10)
$$

To use the backward recursing algorithm in $(6.7.9)$, we use the following code fragment to implement $(6.7.9)$

```
ctilde = zeros(1,M/2); % initialize ctilde array
ctilde(M/2) = 2*c(M/2);
```

```
ctilde(M/2-1) = 2*c(M/2-1);
for k = M/2-2:-1:2
    ctilde(k) = 2*c(k) + ctilde(k+2);
end
ctilde(1) = c(1)+0.5*ctilde(3);
disp(ctilde); % part (b) coefficients
```

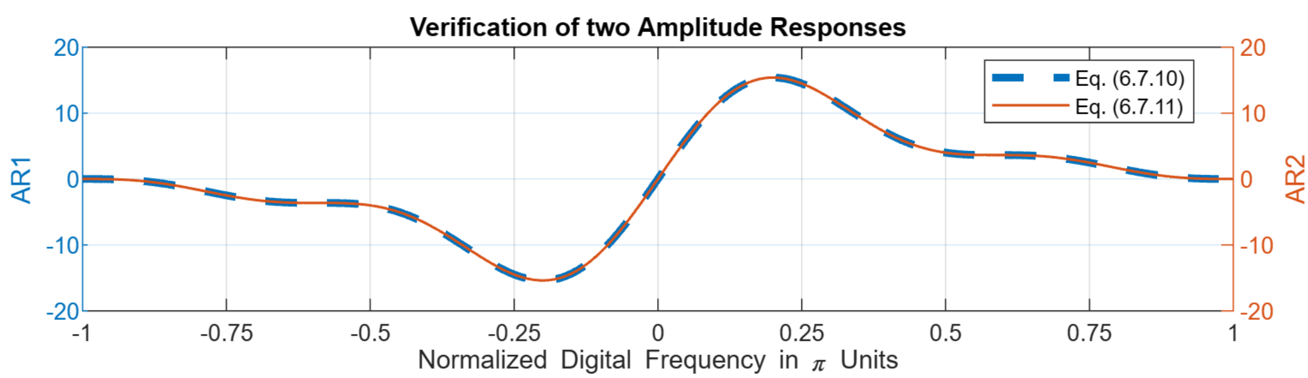
```
    12    16     8     4
```

Thus, the amplitude response can also be expressed as

$$A(\omega) = \sin(\omega)\big[12 + 16\cos(\omega) + 8\cos(2\omega) + 4\cos(3\omega)\big]. \qquad (6.7.11)$$

Verification: Amplitude response plots for the expressions in (6.7.10) and (6.7.11) are done using the following script.

```
f = linspace(-1,1,501); om = f*pi;
AR1 = c(1)*sin(om)+c(2)*sin(2*om)+c(3)*sin(3*om)+c(4)*sin(4*om);
AR2 = ctilde(1)+ctilde(2)*cos(om)+ctilde(3)*cos(2*om)+ctilde(4)*cos(3*om);
AR2 = sin(om).*AR2;
figure('units','inches','Position',[0,0,8,2]);
yyaxis left, plot(f,AR1,'--','LineWidth',3); hold on;
axis([-1,1,-20,20]); ylabel('AR1');
yyaxis right; plot(f,AR2,'LineWidth',1); ylabel('AR2');
xlabel('Normalized Digital Frequency in \pi Units');
set(gca,'xtick',(-1:0.25:1)); grid;
legend('Eq. (6.7.10)','Eq. (6.7.11)','location','best');
title('Verification of two Amplitude Responses');
```



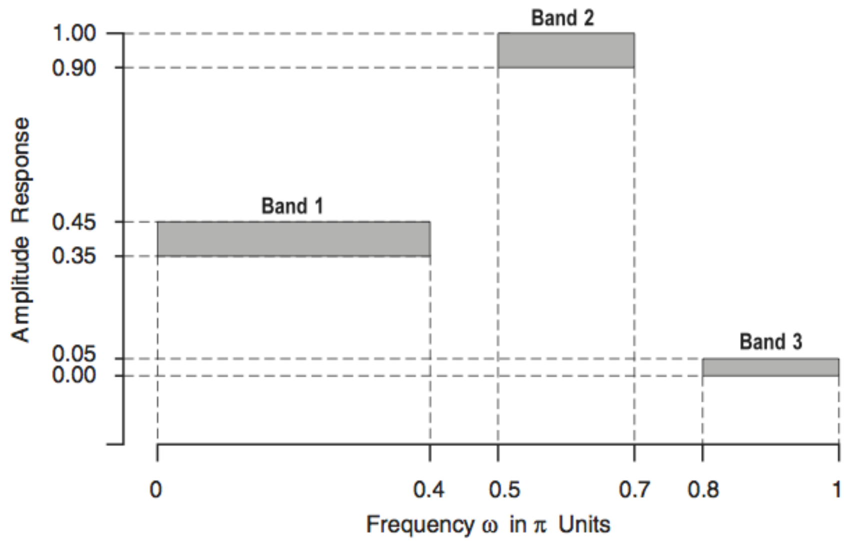
Thus, our calculations are correct. Alternatively, we can expand the expression in (6.7.11) and use the appropriate trigonometric identities.

$$\begin{aligned} A(\omega) &= 12\sin(\omega) + 16\sin(\omega)\cos(\omega) + 8\sin(\omega)\cos(2\omega) + 4\sin(\omega)\cos(3\omega) \\ &= 12\sin(\omega) + 16(0.5)\sin(2\omega) + 8(0.5)\big[\sin(3\omega) - \sin(\omega)\big] + 4(0.5)\big[\sin(4\omega) - \sin(2\omega)\big] \\ &= 8\sin(\omega) + 6\sin(2\omega) + 4\sin(3\omega) + 2\sin(4\omega) \end{aligned}$$

which agrees with (6.7.10). Hence, our calculations are correct.

Problem 6.8: Multiband FIR Filter Design

Specifications of an FIR filter are given below:



(a) Design a lowest-order equiripple linear-phase FIR filter to satisfy the above specifications.

Solution: From the given figure specifications, we need to obtain the desired responses $\{A_i\}_{i=1}^{3}$ and ripples $\{\delta_i\}_{i=1}^{3}$ in each band. The $\{A_i\}$ are halfway between the two tolerance limits while $\{\delta_i\}$ are half of the tolerance width in each band. Then we will use the **`firpmord`** and **`firpm`** functions to obtain the desired filter.

MATLAB script:

```
clc; close all; clear;
% Given Specifications:
f1 = 0.4; f2 = 0.5; f3 = 0.7; f4 = 0.8;           % Band-edge frequencies
A1 = (0.35+0.45)/2; A2 = (0.9+1)/2; A3 = 0.05/2; % Desired responses
delta1 = (0.45-0.35)/2;                           % Band-1 ripple
delta2 = (1-0.9)/2;                               % Band-2 ripple
delta3 = 0.05/2;                                  % Band-3 ripple
% Estimated Filter order using FIRPMORD function
[M,fo,ao,W] = firpmord([f1,f2,f3,f4],[A1,A2,A3],[delta1,delta2,delta3]);
fprintf('Filter order M = %i\n',M);
```

```
Filter order M = 18
```

```
fprintf('Band-weights: [%3.1f, %3.1f, %3.1f]\n',W)
```

```
Band-weights: [8.0, 19.0, 1.0]
```

The band weights reported by **`firpmord`** are incorrect. The correct weights are:

```
W = [delta1/delta1,delta1/delta2,delta1/delta3]
```

```
W = 1×3
    1.0000    1.0000    2.0000
```

Now we will monitor δ_1 in the **`firpm`** function.

```
% Filter Design using FIRPM function:
[~,delta] = firpm(M,fo,ao,W);
fprintf('Required ripple: %g, Obtained ripple: %g',delta1,delta);
```

```
Required ripple: 0.05, Obtained ripple: 0.104688
```

Thus we will increase the order M by two in each iteration until the obtained $\delta \leq \delta_1$. This is because the desired response at $\omega = \pi$ is not zero (although we can accept it to be zero but MATLAB won't allow it).

```
M = M+2; [~,delta] = firpm(M,fo,ao,W);
fprintf('Required ripple: %g, Obtained ripple: %g',delta1,delta);
```

```
Required ripple: 0.05, Obtained ripple: 0.0759648
```

```
M = M+2; [~,delta] = firpm(M,fo,ao,W);
fprintf('Required ripple: %g, Obtained ripple: %g',delta1,delta);
```

```
Required ripple: 0.05, Obtained ripple: 0.0693988
```

```
M = M+2; [~,delta] = firpm(M,fo,ao,W);
fprintf('Required ripple: %g, Obtained ripple: %g',delta1,delta);
```

```
Required ripple: 0.05, Obtained ripple: 0.0545163
```

```
M = M+2; [h,delta] = firpm(M,fo,ao,W);
fprintf('Required ripple: %g, Obtained ripple: %g',delta1,delta);
```

```
Required ripple: 0.05, Obtained ripple: 0.0441745
```

```
disp(M)
```

```
    26
```

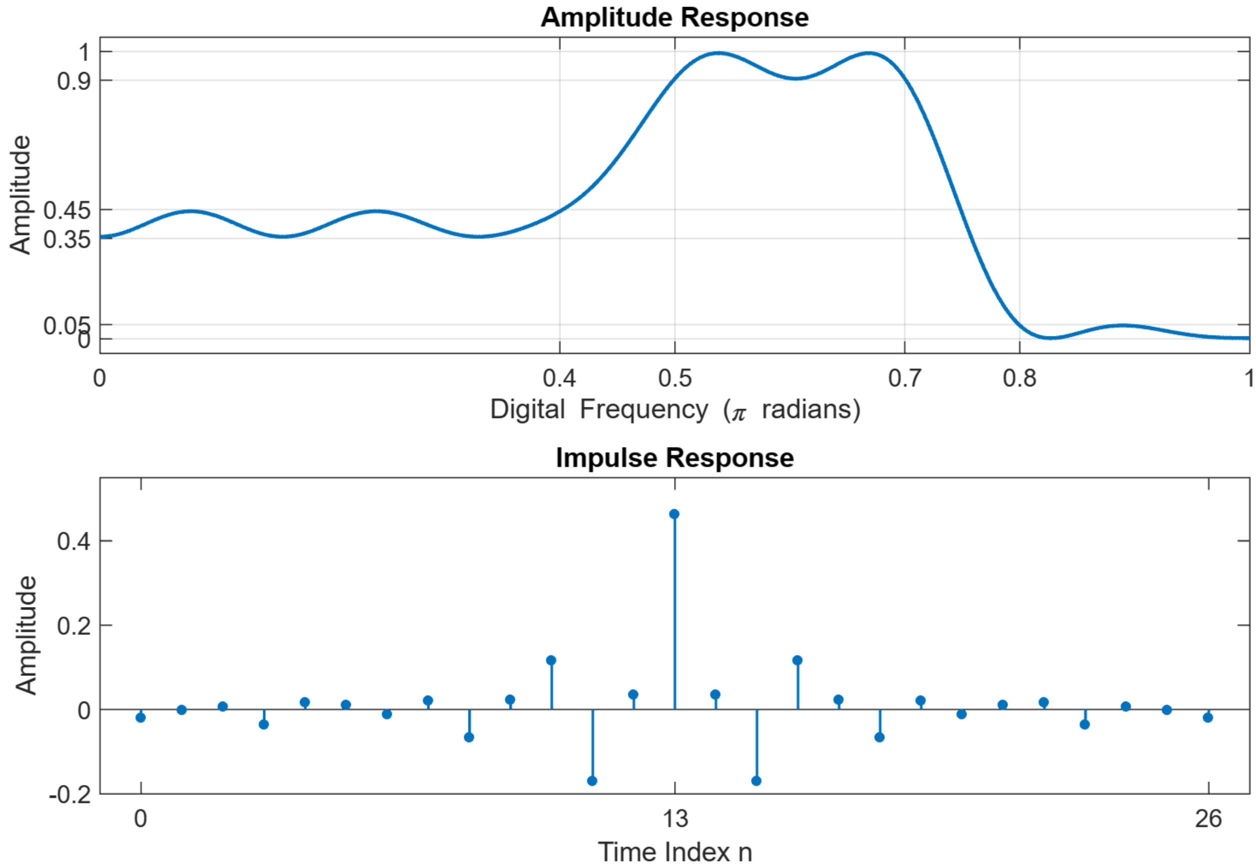
Now the design is complete that results in a 26th-order filter or length 27 impulse response.

(b) Provide a plot of the amplitude response and a plot of the impulse response in one figure using two rows and one column.

MATLAB script:

```
f = linspace(0,1,1001); A = zerophase(h,1,f*pi);
figure('units','inches','position',[0,0,8,5]);
subplot(2,1,1); % Amplitude response
plot(f,A,'linewidth',1.5); axis([0,1,-0.05,1.05]);
xlabel('Digital Frequency (\pi radians)'); ylabel('Amplitude');
title('Amplitude Response');
set(gca,'xtick',[0,f1,f2,f3,f4,1],'ytick',[0,0.05,0.35,0.45,0.9,1]); grid;
subplot(2,1,2); % Impulse response
stem(0:M,h,'filled','markersize',3,'linewidth',1);
xlabel('Time Index n'); ylabel('Amplitude');
```

```
title('Impulse Response'); axis([-1,M+1,-0.2,0.55]);
set(gca,'xtick',[0,M/2,M],'ytick',-0.2:0.2:0.6);
```



Observation: We now have a proper design in which δ_1, δ_2, and δ_3 ripples are within limits as required and are also uniformly distributed.

The `speConvert` function

```
function [A,B] = speConversion(C,D,typein,typeout)
% typein:  'abs' or 'rel' or 'ana'
% typeout: 'abs' or 'rel' or 'ana'
% C, D:     input specifications
% A, B:     output specifications
%
typein = lower(typein); typeout = lower(typeout);
% Error check
if ~(strcmp(typein,'abs') || strcmp(typein,'rel') || strcmp(typein,'ana'))
    error('typein not recognized')
end
if ~(strcmp(typeout,'abs') || strcmp(typeout,'rel') || strcmp(typeout,'ana'))
    error('typeout not recognized')
end
```

```
% When "typein" = "typeout", no conversion
if strcmp(typein,typeout)
    A = C; B = D;
end
% When "typein' is 'abs'
if strcmp(typein,'abs') && strcmp(typeout,'rel')
    A = 20*log10((1+C)/(1-C)); B = 20*log10((1+C)/D);
elseif strcmp(typein,'abs') && strcmp(typeout,'ana')
    A = 20*log10((1+C)/(1-C)); B = 20*log10((1+C)/D);
    A = sqrt(10^(A/10)-1); B = 10^(B/20);
end
% When 'typein" is 'rel'
if strcmp(typein,'rel') && strcmp(typeout,'abs')
    A = (10^(C/20)-1)/(10^(C/20)+1); B = (1+A)/(10^(D/20));
elseif strcmp(typein,'rel') && strcmp(typeout,'ana')
    A = sqrt(10^(C/10)-1); B = 10^(D/20);
end
% When 'typein" is 'ana'
if strcmp(typein,'ana') && strcmp(typeout,'rel')
    A = 20*log10(sqrt(1+C^2)); B = 20*log10(D);
elseif strcmp(typein,'ana') && strcmp(typeout,'abs')
    A = 20*log10(sqrt(1+C^2)); B = 20*log10(D);
    A = (10^(A/20)-1)/(10^(A/20)+1); B = (1+A)/(10^(B/20));
end
end
```