# EECE5666 (DSP) : Homework-6 Solutions

## Table of Contents



## Default Plot Parameters

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

## Problem 6.1

### Text Problem 10.21 (Page-615)

```
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**: Provide your function code below in the code example for the TA to grade and enter your function at the end of the file.

```
function [A,B] = spec_convert(C,D,typein,typeout)
%  typein: 'abs' or 'rel' or 'ana'
% typeout: 'abs' or 'rel' or 'ana'
%     C,D: input specifications
%     A,B: output specifications

% Enter your function code below


end
```

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

$$A_p = 0.5 \text{ dB and } A_s = 40 \text{ dB}$$

**(i) Absolute specs:**

```
Ap = 0.5; As = 40;
[delp,dels] = spec_convert(Ap,As,'rel','abs'); display(delp); display(dels);
```

```
delp = 0.0288
dels = 0.0103
```

**(ii) Analog specs:**

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

```
epsi = 0.3493
A = 100
```

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

$$\delta_{\rm p} = 0.01 \text{ and } \delta_{\rm s} = 0.001$$

**(i) Relative specs:**

```
clc; close all; clear;
delp = 0.01; dels = 0.001;
[Ap,As] = spec_convert(delp,dels,'abs','rel'); display(Ap); display(As);
```

```
Ap = 0.1737
As = 60.0864
```

**(ii) Analog specs:**

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

```
epsi = 0.2020
A = 1.0100e+03
```

## Problem 6.2

### Text Problem 10.28 (Page 616)

Design a lowpass FIR filter to satisfy the specifications: $\omega_{\rm p} = 0.3\pi$, $A_{\rm p} = 0.5$dB, $\omega_{\rm s} = 0.5\pi$, and $A_{\rm s} = 50$ dB.

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

**(a)** Use an appropriate fixed window to obtain a minimum length linear-phase filter. Provide a plot similar to Figure~10.12 in the textbook. Do not use the **`fir1`** function.

**Solution:** Window design provides equal amount of ripple values in both passband and stopband. Therefore, we must design for the minimum of $\delta_{\rm p}$ and $\delta_{\rm s}$.

```
ws = 0.5*pi; wp = 0.3*pi; As = 50; Ap = 0.5;
[deltap, deltas] = spec_convert(Ap,As,'rel','abs');
delta = min([deltap,deltas]); A_reqd = -20*log10(delta)
```

```
A_reqd = 49.7536
```

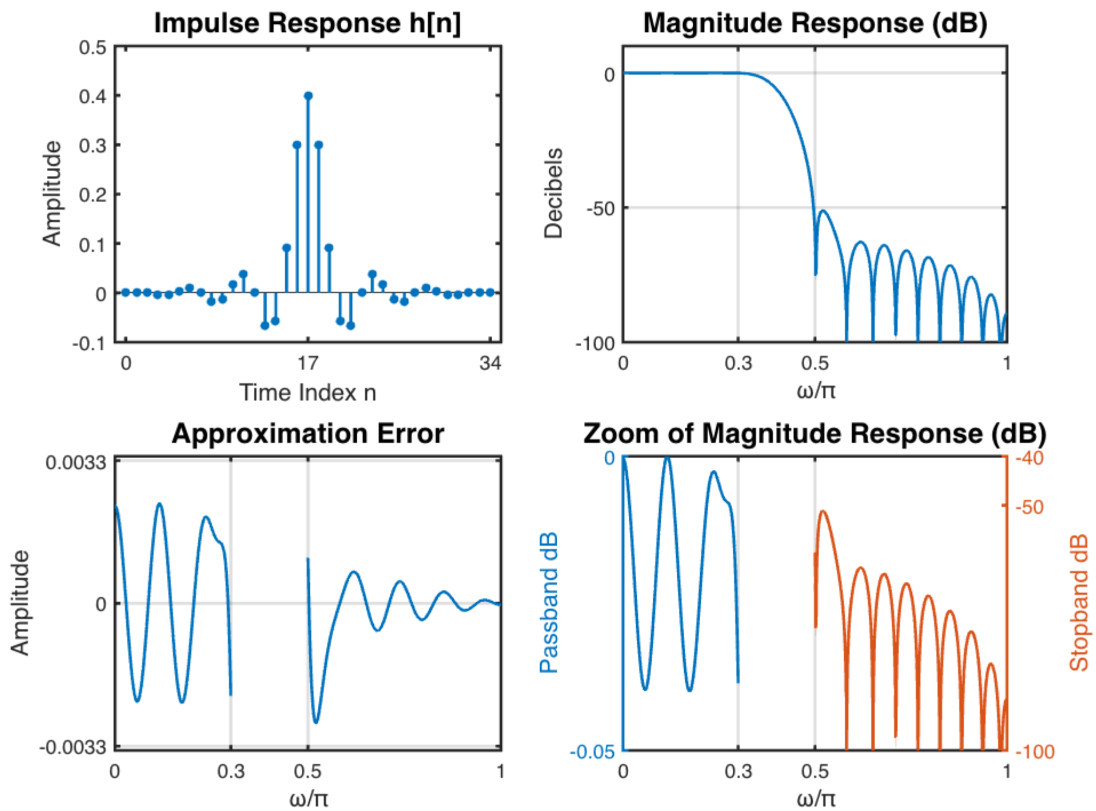
Since the required $A_{\rm reqd} = 49.7536$ dB, the appropriate fixed windows are Hamming and Blackman windows. However, the Hamming window will give the smaller (or minimum) length hence we will use Hamming window. The required design is obtained using the following script.

```
wc = (ws+wp)/2; dw = ws-wp; L = ceil(6.6*pi/dw)+2; M = L-1;
if mod(M,2) == 1
```

```
    M = M+1; % Forces Type-1
    L = M+1; % Length of the filter
end
fprintf('Hamming window length = %d\n',L);
```

```
Hamming window length = 35
```

```
hd = ideallp(wc,M); % Ideal impulse response
h = hd.*hamming(M+1); % Windowed impulse response
om = linspace(0,1,1001)*pi; delta = round(delta,4);
H = freqz(h,1,om); Hmag = abs(H); % Magnitude response
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
Hr = zerophase(h,1,om); % Amplitude response
aperr = nan(1,length(om)); % Approximation error array initialization
magz1 = nan(1,length(om)); % Zoomed log-mag (dB) array intialization (PB)
magz2 = nan(1,length(om)); % Zoomed log-mag (dB) array intialization (SB)
ind = om <= wp; aperr(ind) = Hr(ind)-1; magz1(ind) = Hdb(ind);
ind = om >= ws; aperr(ind) = Hr(ind); magz2(ind) = Hdb(ind);
figure('position',[0,0,8,6]*72);
subplot(2,2,1); stem(0:M,h,'filled','markersize',3,'LineWidth',1.5);
xlabel('Time Index n'); ylabel('Amplitude'); title('Impulse Response h[n]');
set(gca,'xtick',[0,M/2,M]); axis([-1,L,-0.1,0.5]);
subplot(2,2,2); plot(om/pi,Hdb,'linewidth',1.5); axis([0,1,-100,10]);
xlabel('\omega/\pi'); ylabel('Decibels'); title('Magnitude Response (dB)');
set(gca,'ytick',[-100,-50,0],'xtick',[0,wp,ws,pi]/pi); grid;
subplot(2,2,3); plot(om/pi,aperr,'linewidth',1.5);
delta0 = delta+0.0001; axis([0,1,-delta0,delta0]);
xlabel('\omega/\pi'); ylabel('Amplitude'); title('Approximation Error');
set(gca,'ytick',[-delta,0,delta],'yticklabel',{'-0.0033','0','0.0033'});
set(gca,'xtick',[0,wp,ws,pi]/pi); grid on;
subplot(2,2,4); yyaxis left; plot(om/pi,magz1,'linewidth',1.5);
ylabel('Passband dB'); ylim([-0.05,0]); set(gca,'ytick',[-0.05,0]); grid;
yyaxis right; plot(om/pi,magz2,'linewidth',1.5);  ylim([-100,-40]);
set(gca,'ytick',[-100,-50,-40],'ygrid','on'); ylabel('Stopband dB');
xlabel('\omega/\pi'); title('Zoom of Magnitude Response (dB)');
set(gca,'xtick',[0,wp,ws,pi]/pi,'xgrid','on');
```




**(b)** Repeat (a) using the Kaiser window and compare the lengths of the resulting filters. Use the **`fir1`** function for the design.

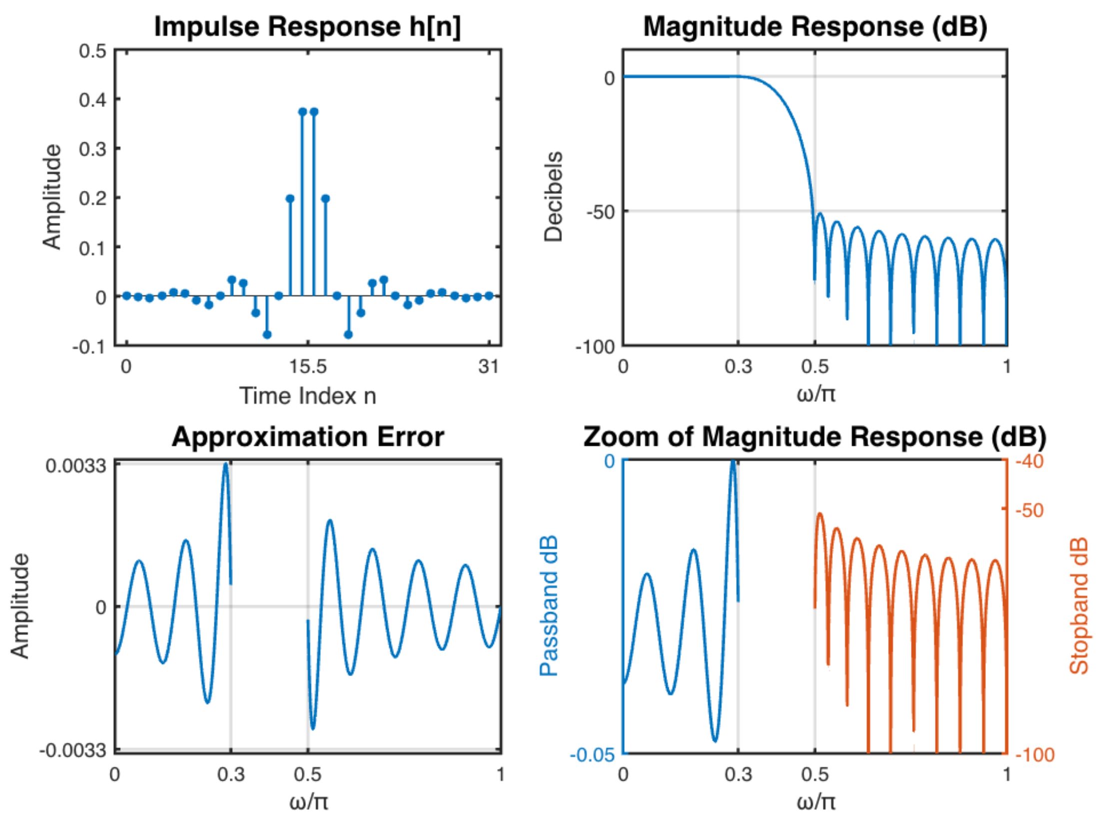
**Solution:** We will use the combination of **`kaiserord`** and **`fir1`** functions for this design.

```
[M,wn,beta,ftype] = kaiserord([0.3 0.5],[1 0],[deltas,deltap]);
 M = M+1; % Needed an additional increase to satisfy requirements
 L = M+1; % Length of the filter
 fprintf('Kaiser window length = %d\n',L);
```

```
Kaiser window length = 32
```

```
 h = fir1(M,wn,ftype,kaiser(M+1,beta));
 H = freqz(h,1,om); Hmag = abs(H); % Magnitude response
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
Hr = zerophase(h,1,om); % Amplitude response
aperr = nan(1,length(om)); % Approximation error array initialization
magz1 = nan(1,length(om)); % Zoomed log-mag (dB) array intialization (PB)
magz2 = nan(1,length(om)); % Zoomed log-mag (dB) array intialization (SB)
ind = om <= wp; aperr(ind) = Hr(ind)-1-0.0011; magz1(ind) = Hdb(ind);
ind = om >= ws; aperr(ind) = Hr(ind); magz2(ind) = Hdb(ind);
figure('position',[0,0,8,6]*72);
subplot(2,2,1); stem(0:M,h,'filled','markersize',3,'LineWidth',1.5);
xlabel('Time Index n'); ylabel('Amplitude'); title('Impulse Response h[n]');
set(gca,'xtick',[0,M/2,M]); axis([-1,L,-0.1,0.5]);
subplot(2,2,2); plot(om/pi,Hdb,'linewidth',1.5); axis([0,1,-100,10]);
xlabel('\omega/\pi'); ylabel('Decibels'); title('Magnitude Response (dB)');
```

```
set(gca,'ytick',[-100,-50,0],'xtick',[0,wp,ws,pi]/pi); grid;
subplot(2,2,3); plot(om/pi,aperr,'linewidth',1.5);
delta0 = delta+0.0001; axis([0,1,-delta0,delta0]);
xlabel('\omega/\pi'); ylabel('Amplitude'); title('Approximation Error');
set(gca,'ytick',[-delta,0,delta],'yticklabel',{'-0.0033','0','0.0033'});
set(gca,'xtick',[0,wp,ws,pi]/pi); grid on;
subplot(2,2,4); yyaxis left; plot(om/pi,magz1,'linewidth',1.5);
ylabel('Passband dB'); ylim([-0.05,0]); set(gca,'ytick',[-0.05,0]);
yyaxis right; plot(om/pi,magz2,'linewidth',1.5); ylim([-100,-40]);
set(gca,'ytick',[-100,-50,-40],'ygrid','on'); ylabel('Stopband dB');
xlabel('\omega/\pi'); title('Zoom of Magnitude Response (dB)');
set(gca,'xtick',[0,wp,ws,pi]/pi,'xgrid','on');
```




**(c)** Provide a comparison between the two designs

**Comparison**: The Kaiser window design gives length-32 FIR filter while the Hamming window design gives length-35 FIR filter. Both provide minimum stopband attenuation of 50 dB, however, the Kaiser window provides the required passband ripple of 0.0033 while the Hamming window does not, that the passband error is not maximally distributed.

## Problem 6.3

**Text Problem 7.33 (Page 617)** This is a revised version.

A bandpass filter is given by the specifications: $\omega_{s_1} = 0.2\pi$, $A_{s_1} = 40$ dB, $\omega_{p_1} = 0.3\pi$, $\omega_{p_2} = 0.5\pi$, $A_p = 0.2$ dB, $\omega_{s_2} = 0.65\pi$, and $A_{s_2} = 40$ dB.

**(a)** Choose $L = 45$ so that there are two samples in the transition band. using a raised-cosine transition band samples obtain the filter response. Provide a plot of the log-magnitude and impulse responses. Does this design satisfy the given specifications?

**Solution:** There is no independent control over passband and stopband ripple specification in the frequency sampling design. So we will have to verify the obtained ripple factors after design is complete.

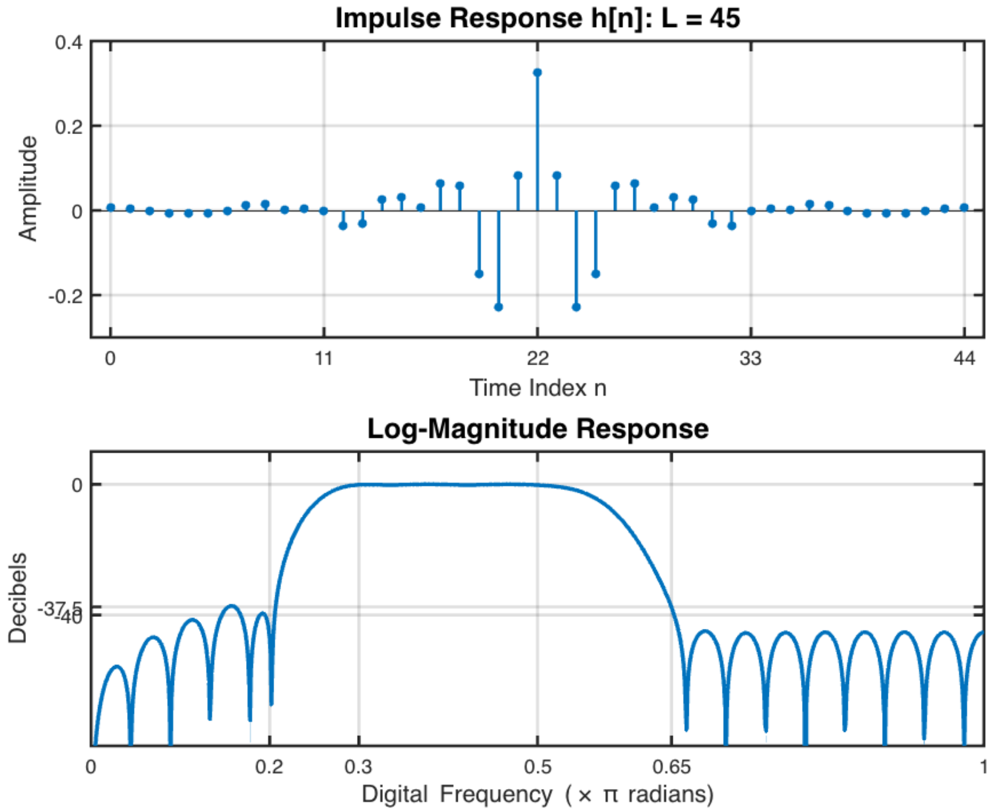
**MATLAB script**:

```
clc; close all; clear;
% Given specifications
fs1 = 0.2; fp1 = 0.3; fp2 = 0.5; fs2 = 0.65; % Cutoff freq in pi rad
As1 = 40; Ap = 0.2; As2 = 40; % Log-magnitude ripples in dB
[delP,delS] = spec_convert(Ap,max(As1,As2),'rel','abs');
del = min(delP,delS); As = -20*log10(del/(1+del));
fprintf('Filter will be designed for %g dB stopband attenuation\n',...
    round(As));
```

```
Filter will be designed for 40 dB stopband attenuation
```

```
L = 45; M = L-1; alpha = M/2; n = 0:M;
df = 2/L; fk = (0:M)*df; omk = fk*pi;
k1 = floor(fs1/df); % index for fs1
k2 = ceil(fp1/df);  % index for fp1
k3 = floor(fp2/df); % index for fp2
k4 = ceil(fs2/df);  % index for fs2
k = k1+1:k2-1;      % index range between fs1 and fp1
A = 0.5-0.5*cos(pi*(fk(k+1)-fs1)/(fp1-fs1)); % Raised-cosine samples
k = k3+1:k4-1;      % index range between fp2 and fs2
B = 0.5+0.5*cos(pi*(fk(k+1)-fp2)/(fs2-fp2)); % Raised-cosine samples
Ad = [zeros(1,k1+1),A,ones(1,k3-k2+1),B,zeros(1,M/2-k4+1)];
% figure('position',[0,0,8,6]*72);
% plot(fk(1:length(Ad)),Ad,'o'); xlim([0,1]);
% set(gca,'xtick',[0,fs1,fp1,fp2,fs2,1]);
Ad = [Ad,fliplr(Ad(2:end))]; % L samples of the amplitude response
psid = -alpha*(omk); % Linear phase response
Hd = Ad.*exp(1j*psid); % L samples of the frequency response
hd = real(ifft(Hd)); h = hd.*rectwin(L)'; % Impulse response
om = linspace(0,1,1001)*pi;
H = freqz(h,1,om); Hmag = abs(H);
HdB = 20*log10(Hmag/max(Hmag)); % Log-Magnitude response in dB
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Impulse response
stem(n,h,'filled','markersize',3,'linewidth',1.5);
axis([-1,L,-0.3,0.4]);
xlabel('Time Index n'); ylabel('Amplitude');
title('Impulse Response h[n]: L = 45'); grid;
set(gca,'xtick',(0:11:44),'ytick',[-0.2:0.2:0.4]);
subplot(2,1,2); % Log-magnitude response
plot(om/pi,HdB,'linewidth',2); axis([0,1,-80,10]);
xlabel('Digital Frequency (\times \pi radians)');
```

```
ylabel('Decibels'); title('Log-Magnitude Response')
set(gca,'xtick',[0,fs1,fp1,fp2,fs2,1],'ytick',[-40,-37.5,0]); grid;
```



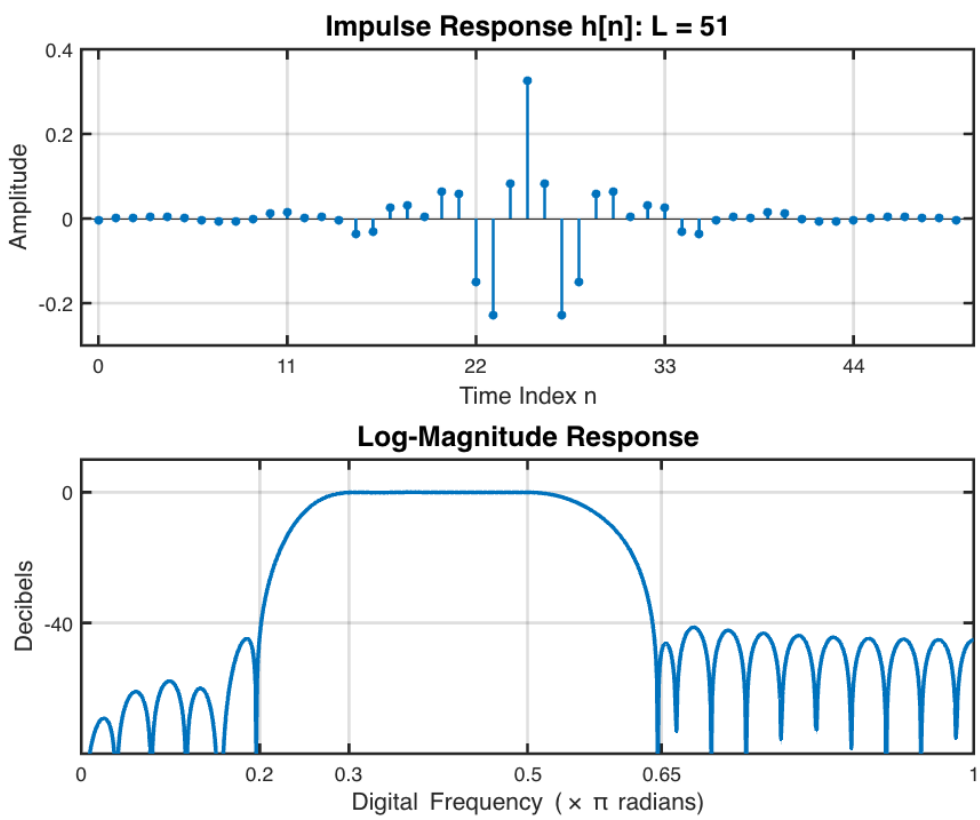

**Observation:** From the log-magnitude plot it is clear that the minimum stopband attenuation is $37.5$ dB and not $40$ dB. Clearly, the design specifications are not satisfied. We may need to increase the filter order and may have more samples in the transition bands.

**Satisfactory Design**: *This part of the solution was not required*.

The optimum filter length is $L = 51$ that still has 2 samples in the first transition band. It was obtained by increasing the length $L$ by two each time and checking the resulting minimum stopband attenuation.

```
L = 51; M = L-1; alpha = M/2; n = 0:M;
df = 2/L; fk = (0:M)*df; omk = fk*pi;
k1 = floor(fs1/df); % index for fs1
k2 = ceil(fp1/df);  % index for fp1
k3 = floor(fp2/df); % index for fp2
k4 = ceil(fs2/df);  % index for fs2
k = k1+1:k2-1;      % index range between fs1 and fp1
A = 0.5-0.5*cos(pi*(fk(k+1)-fs1)/(fp1-fs1)); % Raised-cosine samples
k = k3+1:k4-1;      % index range between fp2 and fs2
B = 0.5+0.5*cos(pi*(fk(k+1)-fp2)/(fs2-fp2)); % Raised-cosine samples
Ad = [zeros(1,k1+1),A,ones(1,k3-k2+1),B,zeros(1,M/2-k4+1)];
% figure('position',[0,0,8,6]*72);
% plot(fk(1:length(Ad)),Ad,'o'); xlim([0,1]);
% set(gca,'xtick',[0,fs1,fp1,fp2,fs2,1]);
Ad = [Ad,fliplr(Ad(2:end))]; % L samples of the amplitude response
psid = -alpha*(omk); % Linear phase response
```

```
Hd = Ad.*exp(1j*psid); % L samples of the frequency response
hd = real(ifft(Hd)); h = hd.*rectwin(L)'; % Impulse response
om = linspace(0,1,1001)*pi;
H = freqz(h,1,om); Hmag = abs(H);
HdB = 20*log10(Hmag/max(Hmag)); % Log-Magnitude response in dB
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Impulse response
stem(n,h,'filled','markersize',3,'linewidth',1.5);
axis([-1,L,-0.3,0.4]);
xlabel('Time Index n'); ylabel('Amplitude');
title('Impulse Response h[n]: L = 51'); grid;
set(gca,'xtick',(0:11:44),'ytick',[-0.2:0.2:0.4]);
subplot(2,1,2); % Log-magnitude response
plot(om/pi,HdB,'linewidth',2); axis([0,1,-80,10]);
xlabel('Digital Frequency (\times \pi radians)');
ylabel('Decibels'); title('Log-Magnitude Response')
set(gca,'xtick',[0,fs1,fp1,fp2,fs2,1],'ytick',[-40,0]); grid;
```



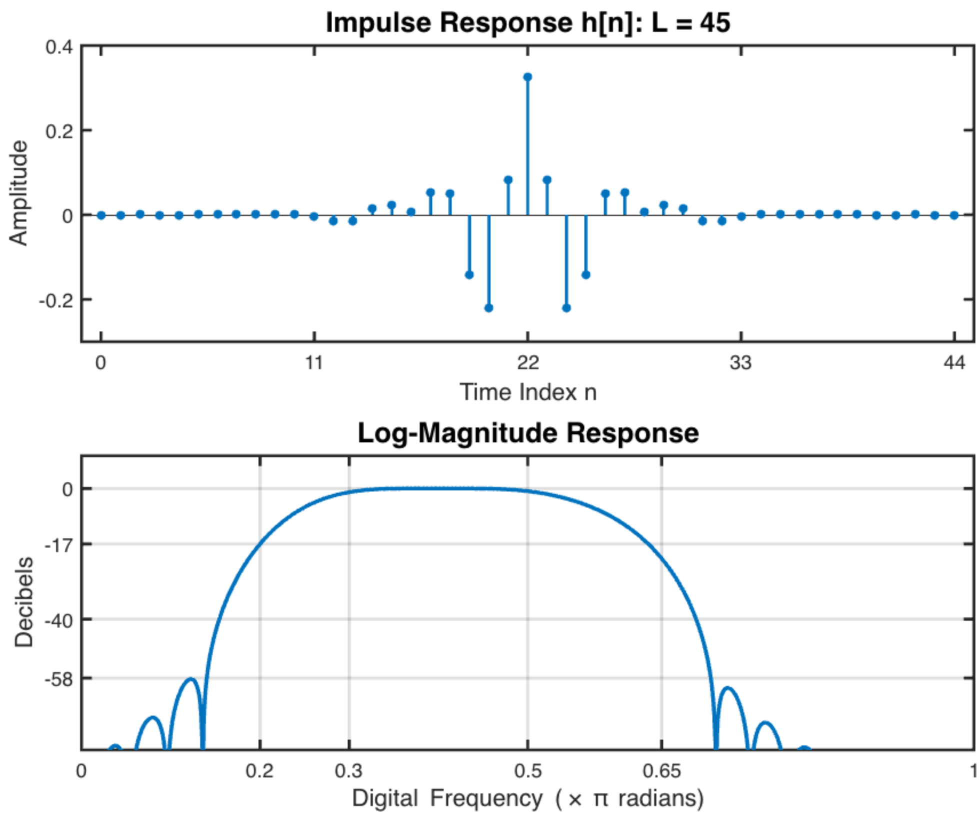

**(b)** Repeat (a) using the **fir2** function and the Hann window. Does this design satisfy the given specifications?

**Solution:** We already have the necessary frequency band-edge array. We only need a desired magnitude array at those frequencies.

**MATLAB script**:

```
fs1 = 0.2; fp1 = 0.3; fp2 = 0.5; fs2 = 0.65; % Cutoff freq in pi rad
```

```
f = [0,fs1,fp1,fp2,fs2,1]; m = [0,0,1,1,0,0];
L = 45; M = L-1; n = 0:M; h = fir2(M,f,m,hann(L));
H = freqz(h,1,om); Hmag = abs(H);
HdB = 20*log10(Hmag/max(Hmag)); % Log-Magnitude response in dB
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Impulse response
stem(n,h,'filled','markersize',3,'linewidth',1.5);
axis([-1,L,-0.3,0.4]);
xlabel('Time Index n'); ylabel('Amplitude');
title('Impulse Response h[n]: L = 45');
set(gca,'xtick',(0:11:44),'ytick',[-0.2:0.2:0.4]);
subplot(2,1,2); % Log-magnitude response
plot(om/pi,HdB,'linewidth',2); axis([0,1,-80,10]);
xlabel('Digital Frequency (\times \pi radians)');
ylabel('Decibels'); title('Log-Magnitude Response')
set(gca,'xtick',[0,fs1,fp1,fp2,fs2,1],'ytick',[-58,-40,-17,0]); grid;
```



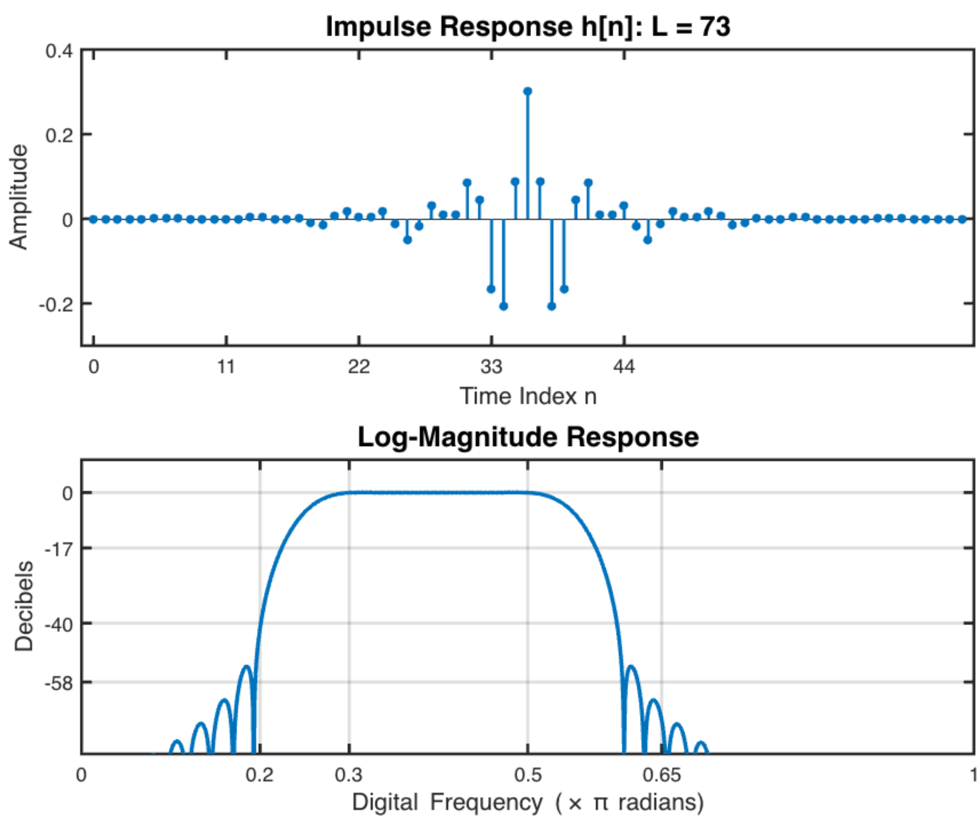

**Observation:** In this design, the minimum sidelobe attenuation is $58$ dB well below the $40$ dB level that we desire. However, the transition bandwidths are much wider than we desire. This is because the Hann window increases the transition bandwidth. The stopband attenuation is $-17$ dB at $\omega_{s_1} = 0.2\pi$.

Hence this design is also not satisfactory.

**Satisfactory Design**: *This part of the solution was also not required.*

We will need to adjust both transition bandwidths (since the Hann window increases bandwidth) and/or use a larger filter order to satisfy the given specs. The smallest value that satisfied the specification was found to be $L = 73$ with zero transition widths. Again, it was obtained by incresing filter length $L$ in steps of two and checking the resulting minimum stopband attenuation.

```
fs1d = 0.2; fp1d = 0.3; fp2d = 0.5; fs2d = 0.65; % Given cutoff freq in pi rad
fs1 = 0.25; fp1 = 0.25; fp2 = 0.55; fs2 = 0.55; % Adjusted cutoff freq in pi rad
f = [0,fs1,fp1,fp2,fs2,1]; m = [0,0,1,1,0,0];
L = 73; M = L-1; n = 0:M; h = fir2(M,f,m,hann(L));
H = freqz(h,1,om); Hmag = abs(H);
HdB = 20*log10(Hmag/max(Hmag)); % Log-Magnitude response in dB
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Impulse response
stem(n,h,'filled','markersize',3,'linewidth',1.5);
axis([-1,L,-0.3,0.4]);
xlabel('Time Index n'); ylabel('Amplitude');
title('Impulse Response h[n]: L = 73');
set(gca,'xtick',(0:11:44),'ytick',[-0.2:0.2:0.4]);
subplot(2,1,2); % Log-magnitude response
plot(om/pi,HdB,'linewidth',2); axis([0,1,-80,10]);
xlabel('Digital Frequency (\times \pi radians)');
ylabel('Decibels'); title('Log-Magnitude Response')
set(gca,'xtick',[0,fs1d,fp1d,fp2d,fs2d,1],'ytick',[-58,-40,-17,0]); grid;
```




**Conclusion**: The raised-cosine interpolation approach, Part (a), gives the smaller filter order.

# Problem 6.4

### Text Problem 7.35 (Page 617)

Design of a length $L = 50$ FIR differentiator.

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

---

**(a)** Design the above-described differentiator using the frequency sampling method and provide graphs of impulse and amplitude responses of the designed differentiator in one figure window using two rows and one column.

**Solution:** The frequency response of an ideal digital differentiator is given by

$$H_{\mathrm{d}}(e^{j\omega}) = \begin{cases} +j\omega\, e^{-j\alpha\omega}, & 0 < \omega \le \pi \\ -j\omega\, e^{+j\alpha\omega}, & -\pi < \omega < 0 \end{cases}$$

where $\alpha = L/2$ is the delay. The $L$ samples from $0 \le k \le L-1$ of $H_{\mathrm{d}}(e^{j\omega})$ are given by

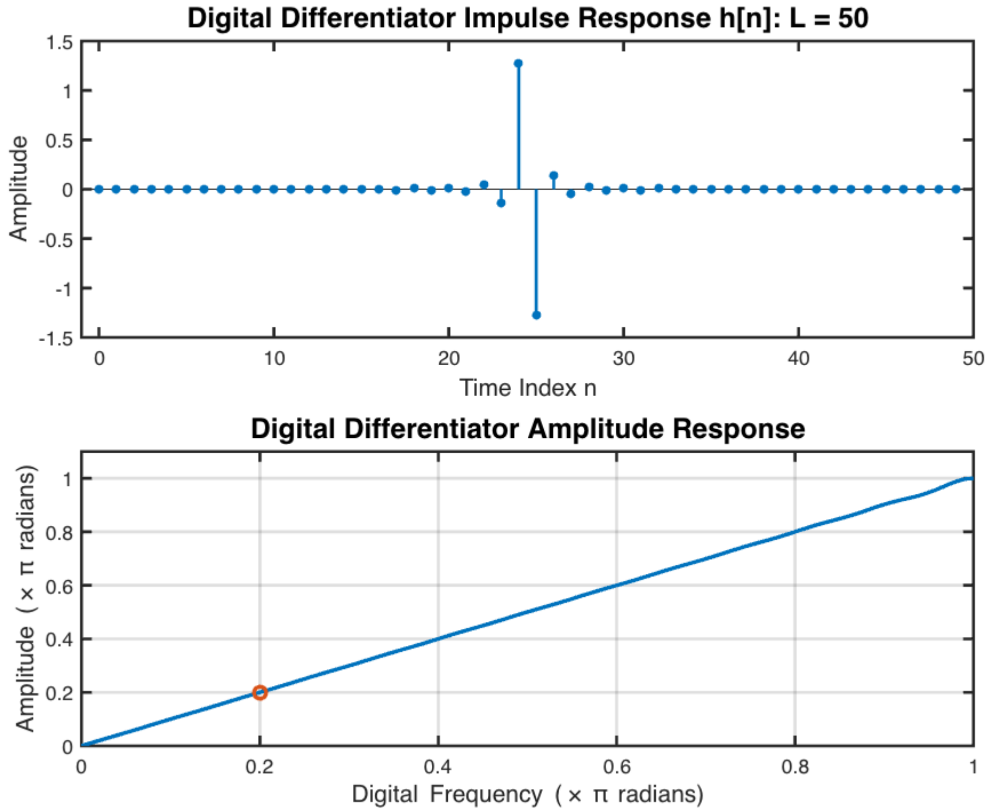
$$H_{\mathrm{d}}[k] = \begin{cases} +j\dfrac{2\pi}{L}k\exp\left(-j\dfrac{M}{L}\pi k\right), & k = 0, 1, \dots, \lfloor M/2 \rfloor \\ -j\dfrac{2\pi}{L}(L-k)\exp\left(+j\dfrac{M}{L}\pi(L-k)\right), & k = \lfloor M/2 \rfloor + 1, \dots, M \end{cases}$$

where $M = L - 1$. Now we can obtain the impulse response of the digital differentiator as

$$h_{\mathrm{d}}[n] = \mathrm{IDFT}\big(H_{\mathrm{d}}[k]\big) \quad \text{and} \quad h[n] = h_{\mathrm{d}}[n]w[n].$$

**MATLAB script**:

```
L = 50; M = L-1; n = 0:M; K = floor(M/2);
k1 = 0:K; k2 = K+1:M;
Hd = [(1j*(2*pi/L)*k1).*exp(-1j*M/L*pi*k1),...
    (-1j*(2*pi/L)*(L-k2)).*exp(1j*M/L*pi*(L-k2))];
h = real(ifft(Hd,L));
om = linspace(0,1,1001)*pi; A = zerophase(h,1,om);
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Impulse Response
stem(n,h,'filled','markersize',3,'linewidth',1.5);
axis([-1,L,-1.5,1.5]);
xlabel('Time Index n'); ylabel('Amplitude');
title('Digital Differentiator Impulse Response h[n]: L = 50');
set(gca,'xtick',(0:10:50),'ytick',(-1.5:0.5:1.5));
subplot(2,1,2); % Amplitude response
plot(om/pi,A/pi,'linewidth',2); axis([0,1,0,1.1]); hold on;
plot(0.2,0.2,'o','linewidth',2);
xlabel('Digital Frequency (\times \pi radians)');
ylabel('Amplitude (\times \pi radians)');
title('Digital Differentiator Amplitude Response')
set(gca,'xtick',(0:0.2:1),'ytick',(0:0.2:1)); grid;
```

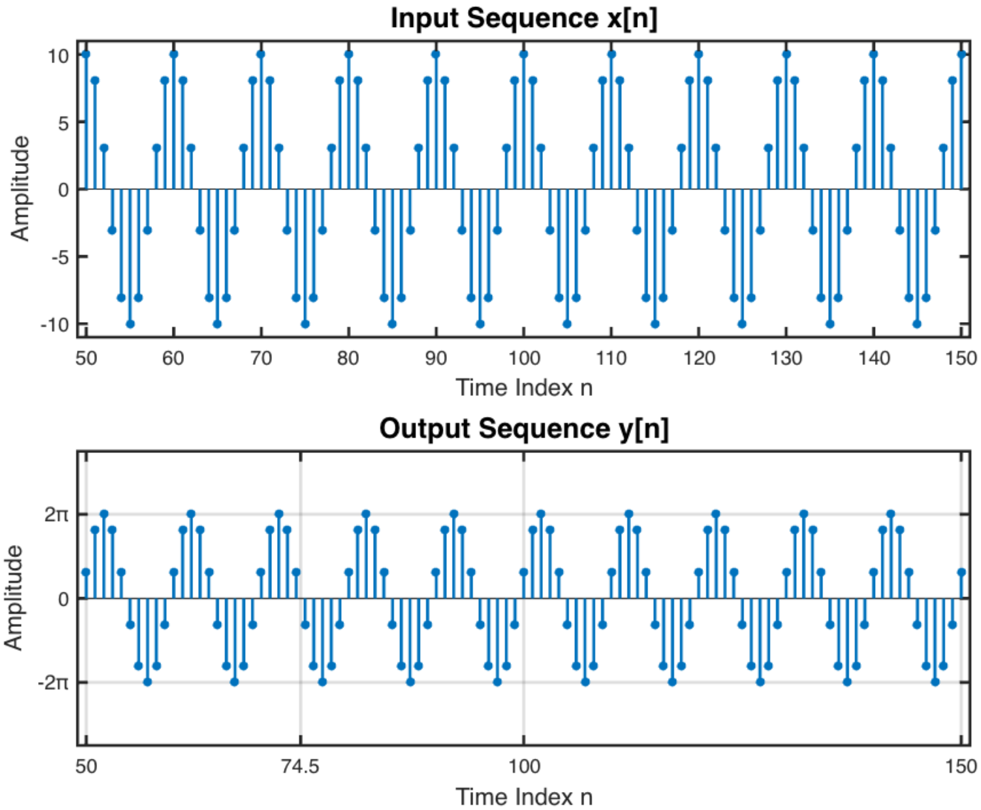



**(b)** Generate $151$ samples of the signal $x[n] = 10\cos(0.2\pi n),\ 0 \le n \le 150$ and process them through the differentiator designed in (a) to obtain $y[n]$. Provide stem plots of both $x[n]$ and $y[n]$ for $50 \le n \le 100$ as $(2\times 1)$ subplots in one figure.

**Solution:** The required script is given below.

**MATLAB script**:

```
n = 0:150; xn = 10*cos(0.2*pi*n);
yn = filter(h,1,xn);
figure('position',[0,0,8,6]*72);
subplot(2,1,1); % Inout sequence
stem(n(51:151),xn(51:151),'filled','markersize',3,'linewidth',1.5);
axis([49,151,-11,11]); xlabel('Time Index n');
ylabel('Amplitude'); title('Input Sequence x[n]');
subplot(2,1,2); % Output sequence
stem(n(51:151),yn(51:151),'filled','markersize',3,'linewidth',1.5);
axis([49,151,-11,11]); xlabel('Time Index n');
ylabel('Amplitude'); title('Output Sequence y[n]');
set(gca,'xtick',[50,50+M/2,100,150],'ytick',[-2*pi,0,2*pi]); grid;
set(gca,'yticklabel',{'-2\pi','0','2\pi'});
```




**(c)** Can you confirm that $y[n]$ corresponds to samples of the derivative of the signal whose samples are given by $x[n]$?

**Solution:** To understand the operation of digital differentiator assume that the input sampled signal $x[n] = 10\cos(0.2\pi n)$ is obtained by sampling analog sinusoidal signal

$$x_c(t) = 10\cos(0.2\pi t)$$

at $F_s = 1$ or at $t = n$. Now let $y_c(t)$ be the derivative of $x_c(t)$, that is,

$$y_c(t) = \frac{\mathrm{d}x_c(t)}{\mathrm{d}t} = 10\big(-\sin(0.2\pi t)\big)(0.2\pi) = -2\pi\sin(0.2\pi t)$$

which after ideal sampling with $F_s = 1$ becomes

$$y[n] = y_c(t)\big|_{t=n} = -2\pi\sin(0.2\pi n).$$

Since the designed digital differentiator is a linear-phase FIR filter with delay of $M/2 = 24.5$ samples, the above output $y[n]$ will appear after $24.5$ samples delay. To visualize this delay, the above output plot has a tick mark (and a grid line) at $74.5$ sample. If we take the peak of the input $x[n]$ at $n = 50$ as the starting point then the differentiated input should start coming out of the filter at $74.5$ sample. As expected, this output is $2\pi$ times the negative of the sine waveform.

Thus we can confirm that the designed FIR filter performs as a differentiator which is a stable system. Analog differentiators are unstable systems due to a pole on the imaginary axis.

# Problem 6.5

### Text Problem 10.49 (Page 619)

Specifications of a bandstop filter are: $\omega_{p_1} = 0.4\pi,\ A_{p_1} = 0.5$dB, $\omega_{s_1} = 0.55\pi,\ \omega_{s_2} = 0.65\pi,\ A_s = 55$dB, $\omega_{p_2} = 0.75\pi$, and $A_{p_2} = 1$dB.

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

**(a)** Design a minimum length linear-phase FIR filter using the Kaiser window. Provide a plot similar to Figure 10.17. Do not use the **fir1** function.

**Solution**: In this part we will obtain the required filter using Kaiser window design equations.

**MATLAB script**:

```
% Given Specifications
fp1 = 0.4; Ap1 = 0.5; % Lower passband specifications
fs1 = 0.55; fs2 = 0.65; As = 55; % Stopband specifications
fp2 = 0.75; Ap2 = 1; % Lower passband specifications
% Required transition bandwidth Df for design
Df1 = fs1-fp1; Df2 = fp2-fs2; Df = min(Df1,Df2)
```

```
Df = 0.1000
```

```
% Required stopband attenuation As for design
Ap = min(Ap1,Ap2);
[deltap,deltas] = spec_convert(Ap,As,'rel','abs');
delta = min(deltap,deltas);
As = -20*log10(delta/(1+delta))
```

```
As = 54.7695
```

```
% Kaiser window design equations
beta = 0.1102*(As-8.7); % Eq (10.84) in the text
M = ceil((As-8)/(2.285*Df*pi)); % Eq (10.85) in the text
beta, M
```

```
beta = 5.0769
M = 66
```

```
L = M+1; w_kai = kaiser(L,beta)';
% Ideal impulse response for the bandstop filter
fL = (fp1+fs1)/2; % Lower cutoff frequency
fU = (fs2+fp2)/2; % Upper cutoff frequency
fC = (fU-fL)/2;    % Ideal lowpass filter cutoff frequency
f0 = (fU+fL)/2;    % Carrier frequency
n = 0:M; alpha = M/2; nnd = n-alpha;
hBP = 2*fC*sinc(fC*nnd).*cos(f0*pi*nnd); % Eq (5.72) in the text
hBS = (nnd == 0) - hBP; % Eq (5.74) in the text
% Impulse response of the designed filter
h = hBS.*w_kai;
```

### Design Validation

```
K = 1000; f = linspace(0,1,K+1); % Dense frequency samples
H = freqz(h,1,f*pi); Hmag = abs(H); % Magnitude response
```

```
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
ind1 = fs1*K; ind2 = fs2*K-1;
Ap_attained = -max(Hdb(ind1+1:ind2-1)); display(Ap_attained)
```

```
Ap_attained = 53.1868
```

Clearly this is not sufficient so increase $M$ to $M+2$ (Type-I) and repeat the process

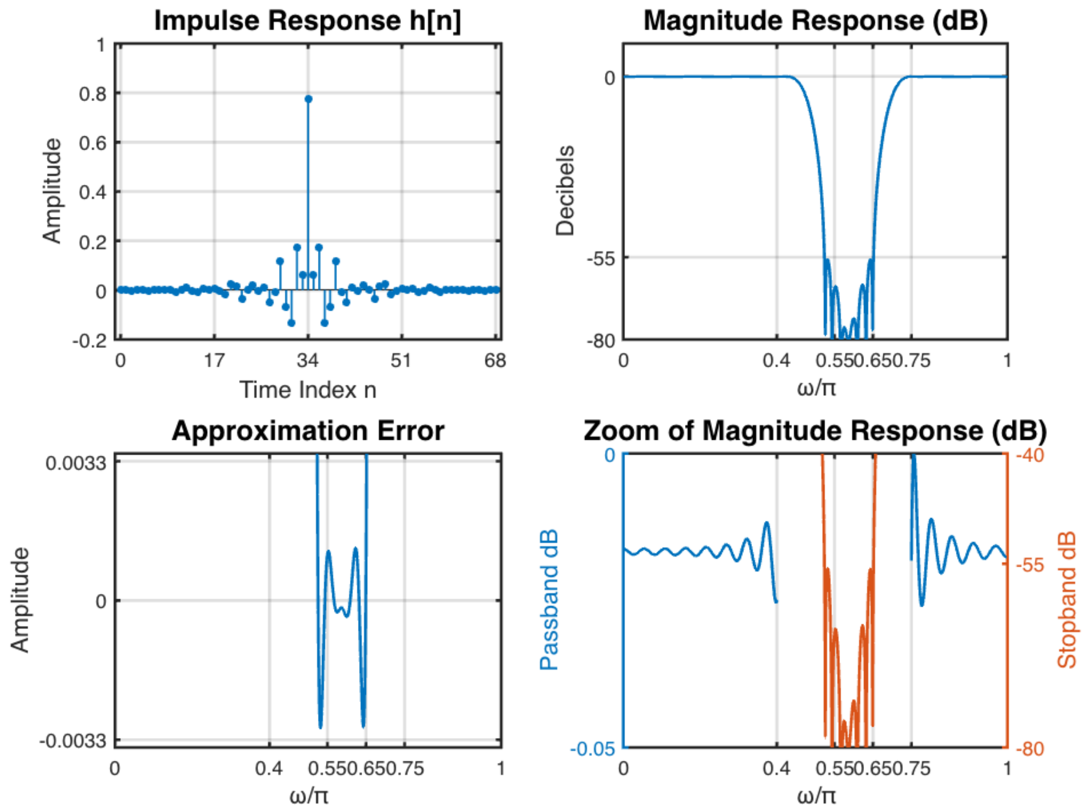
```
M = M+2; n = 0:M; alpha = M/2; nnd = n-alpha;
L = M+1; w_kai = kaiser(L,beta)';
hBP = 2*fC*sinc(fC*nnd).*cos(f0*pi*nnd); % Eq (5.72) in the text
hBS = (nnd == 0) - hBP; % Eq (5.74) in the text
% Impulse response of the designed filter
h = hBS.*w_kai;
% Design validation
K = 1000; f = linspace(0,1,K+1); % Dense frequency samples
H = freqz(h,1,f*pi); Hmag = abs(H); % Magnitude response
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
ind1 = fs1*K; ind2 = fs2*K-1;
Ap_attained = -max(Hdb(ind1+1:ind2-1)); display(Ap_attained)
```

```
Ap_attained = 55.7766
```

### Filter Response Plots

```
% Filter response plots
delta = round(delta,4);
H = freqz(h,1,f*pi); Hmag = abs(H); % Magnitude response
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
Hr = zerophase(h,1,f*pi); % Amplitude response
aperr = nan(1,length(f)); % Approximation error array initialization
magz1 = nan(1,length(f)); % Zoomed log-mag (dB) array intialization (PB)
magz2 = nan(1,length(f)); % Zoomed log-mag (dB) array intialization (SB)
ind = (f <= fp1) | (f >= fp2); % Indices in two passbands
aperr(ind) = Hr(ind)-1; % Error in passbands
magz1(ind) = Hdb(ind); % Log-magnitude (dB) in passbands
ind = (f >= fs1) | (f <= fs2); % Indices in the stopband
aperr(ind) = Hr(ind); % Log-magnitude (dB) in stopband
magz2(ind) = Hdb(ind); % Error in stopband
figure('position',[0,0,8,6]*72);
subplot(2,2,1); stem(0:M,h,'filled','markersize',3,'LineWidth',1);
xlabel('Time Index n'); ylabel('Amplitude'); title('Impulse Response h[n]');
set(gca,'xtick',[0:M/4:M]); axis([-1,L,-0.2,1]); grid;
subplot(2,2,2); plot(f,Hdb,'linewidth',1.5); axis([0,1,-80,10]);
xlabel('\omega/\pi'); ylabel('Decibels'); title('Magnitude Response (dB)');
set(gca,'ytick',[-80,-55,0],'xtick',[0,fp1,fs1,fs2,fp2,1]); grid;
subplot(2,2,3); plot(f,aperr,'linewidth',1.5);
delta0 = delta+0.0001; axis([0,1,-delta0,delta0]);
xlabel('\omega/\pi'); ylabel('Amplitude'); title('Approximation Error');
set(gca,'ytick',[-delta,0,delta],'yticklabel',{'-0.0033','0','0.0033'});
set(gca,'xtick',[0,fp1,fs1,fs2,fp2,1]); grid on;
subplot(2,2,4); yyaxis left; plot(f,magz1,'linewidth',1.5);
ylabel('Passband dB'); ylim([-0.05,0]); set(gca,'ytick',[-0.05,0]); grid;
```

```
yyaxis right; plot(f,magz2,'linewidth',1.5);  ylim([-80,-40]);
set(gca,'ytick',[-80,-55,-40],'ygrid','on'); ylabel('Stopband dB');
xlabel('\omega/\pi'); title('Zoom of Magnitude Response (dB)');
set(gca,'xtick',[0,fp1,fs1,fs2,fp2,1],'xgrid','on');
```




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

**Solution:** In this part we will use the **`kaiserord`** function to determine Kaiser window parameters and **`fir1`** to design the filter.

**MATLAB script**:

```
% Given Specifications
fp1 = 0.4; Ap1 = 0.5; % Lower passband specifications
fs1 = 0.55; fs2 = 0.65; As = 55; % Stopband specifications
fp2 = 0.75; Ap2 = 1; % Lower passband specifications
% Required transition bandwidth Df for design
Df1 = fs1-fp1; Df2 = fp2-fs2; Df = min(Df1,Df2)
```

```
Df = 0.1000
```

```
% Required stopband attenuation As for design
Ap = min(Ap1,Ap2);
[deltap,deltas] = spec_convert(Ap,As,'rel','abs');
delta = min(deltap,deltas);
As = -20*log10(delta/(1+delta))
```

```
As = 54.7695
```

```
[M,fn,beta,ftype] = kaiserord([fp1,fs1,fs2,fp2],[1,0,1],[delta,delta,delta])
```

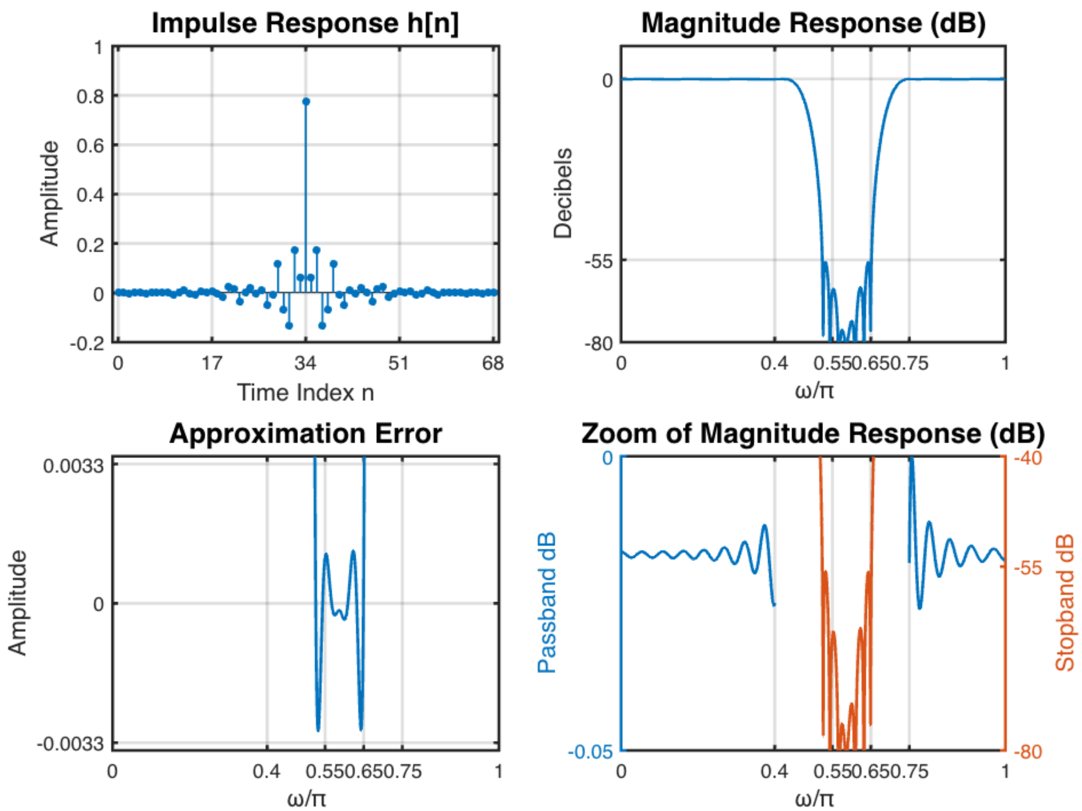
```
M = 66
fn = 2×1
    0.4750
    0.7000
beta = 5.0751
ftype = 'stop'
```

Note that these are the same $M$ and $\beta$ values that we obtained before. We will increase $M$ to $M+2$ and run the **fir1** function.

```
M = M+2; L = M+1;
h = fir1(M,fn,ftype,kaiser(L,beta),'scale');
```

### Filter Response Plots

```
% Filter response plots
delta = round(delta,4);
H = freqz(h,1,f*pi); Hmag = abs(H); % Magnitude response
Hdb = 20*log10(Hmag./max(Hmag)); % Log-magnitude response in dB
Hr = zerophase(h,1,f*pi); % Amplitude response
aperr = nan(1,length(f)); % Approximation error array initialization
magz1 = nan(1,length(f)); % Zoomed log-mag (dB) array intialization (PB)
magz2 = nan(1,length(f)); % Zoomed log-mag (dB) array intialization (SB)
ind = (f <= fp1) | (f >= fp2); % Indices in two passbands
aperr(ind) = Hr(ind)-1; % Error in passbands
magz1(ind) = Hdb(ind); % Log-magnitude (dB) in passbands
ind = (f >= fs1) | (f <= fs2); % Indices in the stopband
aperr(ind) = Hr(ind); % Log-magnitude (dB) in stopband
magz2(ind) = Hdb(ind); % Error in stopband
figure('position',[0,0,8,6]*72);
subplot(2,2,1); stem(0:M,h,'filled','markersize',3,'LineWidth',1);
xlabel('Time Index n'); ylabel('Amplitude'); title('Impulse Response h[n]');
set(gca,'xtick',(0:M/4:M)); axis([-1,L,-0.2,1]); grid;
subplot(2,2,2); plot(f,Hdb,'linewidth',1.5); axis([0,1,-80,10]);
xlabel('\omega/\pi'); ylabel('Decibels'); title('Magnitude Response (dB)');
set(gca,'ytick',[-80,-55,0],'xtick',[0,fp1,fs1,fs2,fp2,1]); grid;
subplot(2,2,3); plot(f,aperr,'linewidth',1.5);
delta0 = delta+0.0001; axis([0,1,-delta0,delta0]);
xlabel('\omega/\pi'); ylabel('Amplitude'); title('Approximation Error');
set(gca,'ytick',[-delta,0,delta],'yticklabel',{'-0.0033','0','0.0033'});
set(gca,'xtick',[0,fp1,fs1,fs2,fp2,1]); grid on;
subplot(2,2,4); yyaxis left; plot(f,magz1,'linewidth',1.5);
ylabel('Passband dB'); ylim([-0.05,0]); set(gca,'ytick',[-0.05,0]); grid;
yyaxis right; plot(f,magz2,'linewidth',1.5);  ylim([-80,-40]);
set(gca,'ytick',[-80,-55,-40],'ygrid','on'); ylabel('Stopband dB');
xlabel('\omega/\pi'); title('Zoom of Magnitude Response (dB)');
set(gca,'xtick',[0,fp1,fs1,fs2,fp2,1],'xgrid','on');
```




## Problem 6.6

### Bandpass filter Specifications:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Lower stopband edge | : | $0.3\pi$, | Upper stopband edge | : | $0.6\pi$; | minimum stopband attenuation | : 50dB |
| Lower passband edge | : | $0.4\pi$, | Upper passband edge | : | $0.5\pi$; | maximum passband ripple | : 0.5dB |

**(a)** Design a bandpass filter to satisfy above specifications using the Hamming window technique. Plot the impulse response and the magnitude response in dB of the designed filter using $(2 \times 1)$ subplots in one figure. Do not use the **`fir1`** function.

**MATLAB script**:

```
clc; close all; clear;
% Specifications:
ws1 = 0.3*pi; % lower stopband edge
wp1 = 0.4*pi; % lower passband edge
wp2 = 0.5*pi; % upper passband edge
ws2 = 0.6*pi; % upper stopband edge
Rp = 0.5;     % passband ripple
As = 50;      % stopband attenuation
%
% Select the min(delta1,delta2) since delta1=delta2 in window design
[delta1,delta2] = spec_convert(Rp,As,'rel','abs');
if (delta1 < delta2)
   delta2 = delta1; disp('Delta1 is smaller than delta2')
```

```
    [Rp,As] = delta2db(delta1,delta2)
end
%
tr_width = min((wp1-ws1),(ws2-wp2));
M = ceil(6.6*pi/tr_width); M = 2*floor(M/2); L = M+1, % choose evrn M
```

```
L = 67
```

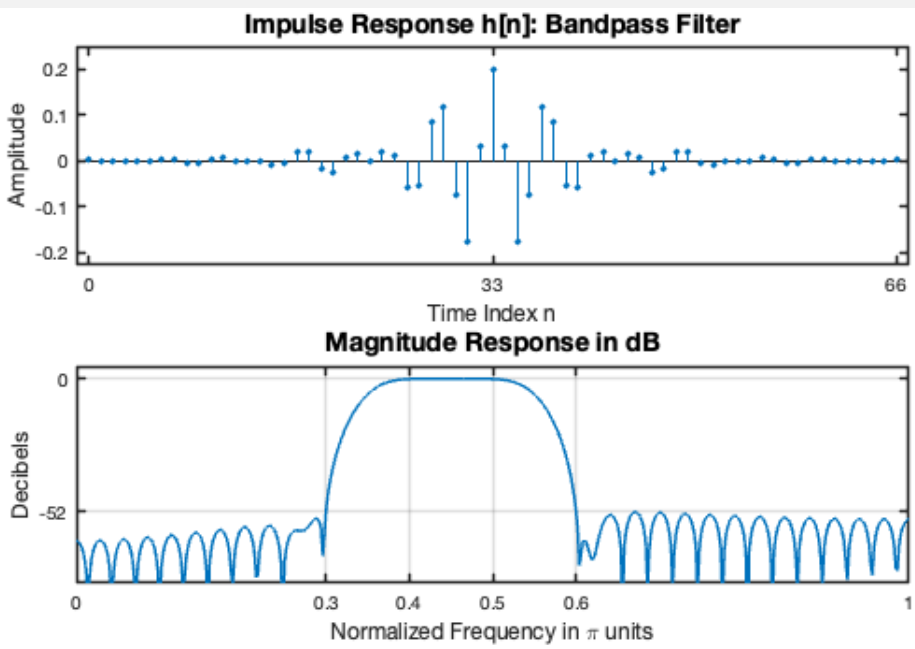
```
n = 0:L-1; w_ham = (hamming(L))';
wc1 = (ws1+wp1)/2; wc2 = (ws2+wp2)/2; hd = ideal_lp(wc2,L)-ideal_lp(wc1,L);
h = hd .* w_ham; [db,~,~,~,w] = freqz_m(h,1); delta_w = pi/500;
Asa = ceil(-max(db((1:floor(ws1/delta_w)+1))));          % Actual SB Attn
fprintf('Obtained stopband attenuation: %g dB',Asa);
```

```
Obtained stopband attenuation: 52 dB
```

```
Rpa = -min(db(ceil(wp1/delta_w)+1:floor(wp2/delta_w)+1)); % Actual PB ripple
fprintf('Obtained stopband attenuation: %g dB',round(Rpa,4));
```

```
Obtained stopband attenuation: 0.0488 dB
```

```
%% Filter Response Plots
figure('position',[0,0,8,5]*72);
subplot(2,1,1); stem(n,h,'filled','markersize',3,'LineWidth',1);
title('Impulse Response h[n]: Bandpass Filter');
axis([-1,L,min(hd)-0.05,max(hd)+0.05]);
xlabel('Time Index n'); ylabel('Amplitude')
set(gca,'XTickMode','manual','XTick',[0;M/2;M])
subplot(2,1,2); plot(w/pi,db,'LineWidth',1.5);
title('Magnitude Response in dB'); axis([0,1,-As-30,5]);
xlabel('Normalized Frequency in \pi units'); ylabel('Decibels')
set(gca,'XTickMode','manual','XTick',[0;0.3;0.4;0.5;0.6;1])
set(gca,'XTickLabelMode','manual','XTickLabels', ...
    [' 0 ';'0.3';'0.4';'0.5';'0.6';' 1 ']);
set(gca,'YTickMode','manual','YTick',[-Asa;0]); grid;
```



**(b)** Repeat part (a) using the **fir1** function.

**MATLAB script**:

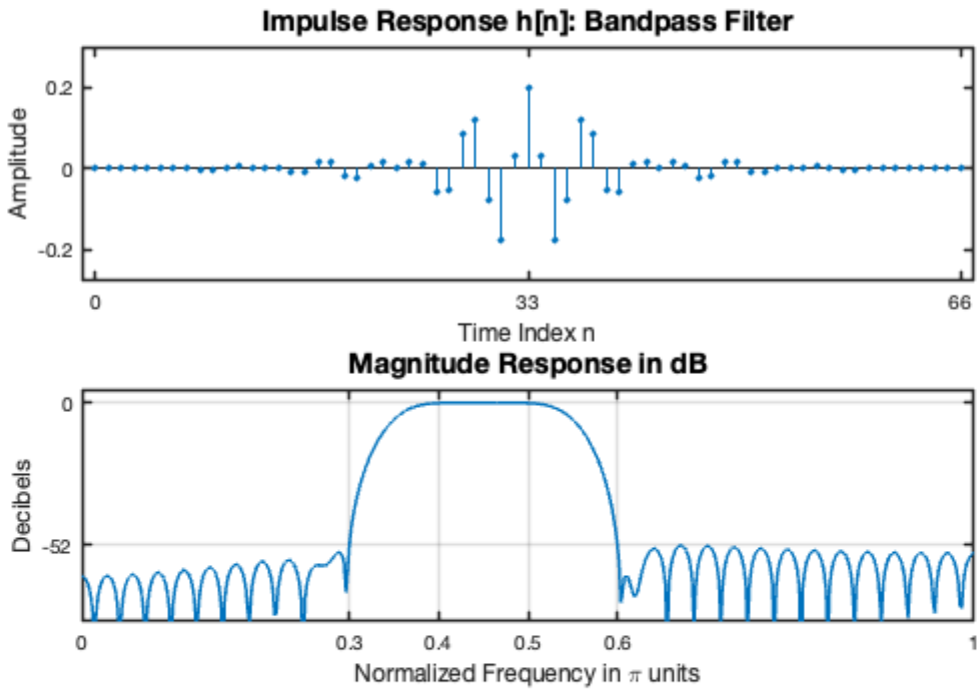
```
h = fir1(M,[wc1,wc2]/pi,'bandpass',w_ham);
[db,mag,pha,grd,w] = freqz_m(h,1);
delta_w = pi/500;
Asa = ceil(-max(db((1:floor(ws1/delta_w)+1))));          % obtained SB Attn
fprintf('Obtained stopband attenuation: %g dB',Asa);
```

```
Obtained stopband attenuation: 52 dB
```

```
Rpa = -min(db(ceil(wp1/delta_w)+1:floor(wp2/delta_w)+1)); % obtained PB ripple
fprintf('Obtained stopband attenuation: %g dB',round(Rpa,4));
```

```
Obtained stopband attenuation: 0.0488 dB
```

```
%% Filter Response Plots
figure('position',[0,0,8,5]*72);
subplot(2,1,1); stem(n,h,'filled','markersize',3,'LineWidth',1);
title('Impulse Response h[n]: Bandpass Filter');
axis([-1,L,min(hd)-0.1,max(hd)+0.1]);
xlabel('Time Index n'); ylabel('Amplitude')
set(gca,'XTickMode','manual','XTick',[0;M/2;M])
subplot(2,1,2); plot(w/pi,db,'LineWidth',1.5);
title('Magnitude Response in dB'); axis([0,1,-As-30,5]);
xlabel('Normalized Frequency in \pi units'); ylabel('Decibels')
set(gca,'XTickMode','manual','XTick',[0;0.3;0.4;0.5;0.6;1])
set(gca,'XTickLabelMode','manual','XTickLabels', ...
    [' 0 ';'0.3';'0.4';'0.5';'0.6';' 1 ']);
set(gca,'YTickMode','manual','YTick',[-Asa;0]); grid;
```



# Problem 6.7

### Modified version of the Text Problem 10.22 (Page 615)

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

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

**(a)** Show that the amplitude response $A(e^{j\omega})$ is given by $(10.34)$ with coefficients $c[k]$ given in $(10.35)$, that is

$$A(e^{j\omega}) = \sum_{k=1}^{M/2} c[k]\sin(\omega k); \qquad c[k] = 2h\left[\frac{M}{2}-k\right], \quad k = 1, 2 \ldots, \frac{M}{2}. \qquad (6.7.2)$$

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

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

The following change of variable

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

in the second sum on the right hand side above gives

$$\begin{aligned}
H(e^{j\omega}) &= \sum_{n=0}^{(M/2)-1} h[n]e^{-j\omega n} - \sum_{n=0}^{(M/2)-1} h[n]e^{-j\omega(M-n)} \\
&= e^{-j\omega(M/2)} \sum_{n=0}^{(M/2)-1} h[n]\{e^{-j\omega n + j\omega(M/2)} - e^{-j\omega(M-n)+j\omega(M/2)}\} \\
&= e^{-j\omega(M/2)} \sum_{n=0}^{(M/2)-1} h[n]\{e^{+j\omega(M/2-n)} - e^{-j\omega(M/2-n)}\} \\
&= je^{-j\omega(M/2)} \sum_{n=0}^{(M/2)-1} 2h[n]\sin\left(\left[\frac{M}{2}-n\right]\omega\right) \qquad (6.7.3)
\end{aligned}$$

Another change of variable gives

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

Substituting in $(6.7.3)$, we obtain

$$H(e^{j\omega}) = je^{-j\omega(M/2)} \sum_{k=1}^{M/2} 2h\left[\frac{M}{2}-k\right]\sin(\omega k)$$

Define $c[k] \triangleq 2h\left[\frac{M}{2}-k\right]),\ k = 1, 2, \ldots, M/2$. Then,

$$H(e^{j\omega}) = je^{-j\omega(M/2)} \sum_{k=1}^{M/2} c[k]\sin(\omega k) \quad \Rightarrow \quad A(e^{j\omega}) = \sum_{k=1}^{M/2} c[k]\sin(\omega k). \qquad (6.7.4)$$

This verifies $(6.7.2)$.

**(b)** Show that the amplitude response $A(e^{j\omega})$ can be further expressed as in $(10.36)$, that is

$$A(e^{j\omega}) = \sin(\omega) \sum_{k=0}^{M/2-1} \widetilde{c}[k] \cos(\omega k) \qquad (6.7.5)$$

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

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

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

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

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

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

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

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

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 $\widetilde{c}[M/2-1]$ and ending with $\widetilde{c}[0]$, as follows:

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

**(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

$$A(e^{j\omega}) = 8\sin(\omega) + 6\sin(2\omega) + 4\sin(3\omega) + 2\sin(4\omega). \tag{6.7.10}$$

```
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
```

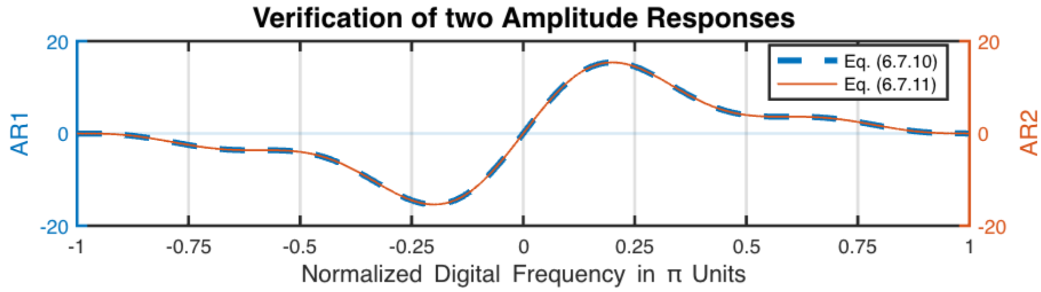
```
    12    16     8     4
```

Thus, the amplitude response can also be expressed as

$$A(e^{j\omega}) = \sin(\omega)\left[12 + 16\cos(\omega) + 8\cos(2\omega) + 4\cos(3\omega)\right]. \tag{6.7.11}$$

**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('Position',[0,0,8,2]*72);
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 will expand the expression in (6.7.11) and use the appropriate trigonometric identities.

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

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

## Problem 6.8

### Text Problem 10.37 (Page 617)

A lowpass FIR filter is to be designed using the specifications: $\omega_p = 0.3\pi$, $A_p = 0.5$ dB, $\omega_s = 0.5\pi$, and $A_s = 50$ dB.

**(a)** Use the Parks-McClellan algorithm to obtain a minimum length linear-phase filter impulse response. Indicate the length of your designed filter.

**MATLAB script**:

```
clc; close all; clear;
newblue = [0.000,0.447,0.741]; % New blue color
fp = 0.3; Ap = 0.5; fs = 0.5; As = 50; % Given specs in normalized frequencies
[deltap,deltas] = spec_convert(Ap,As,'rel','abs');
[M,f0,A0,w] = firpmord([fp,fs],[1,0],[deltap,deltas]);
[~,delta0] = firpm(M,f0,A0,w);
M, delta0, deltap
```

```
M = 18
delta0 = 0.0482
deltap = 0.0288
```

Since `delta0` > `deltap`, we increase `M` → `M+1` and check again.

```
M = M+1; [~,delta0] = firpm(M,f0,A0,w);
M, delta0, deltap
```

```
M = 19
delta0 = 0.0359
deltap = 0.0288
```

Since `delta0` > `deltap`, we again increase `M` → `M+1` and check again.

```
M = M+1; [h,delta0] = firpm(M,f0,A0,w);
M, delta0, deltap
```

```
M = 20
delta0 = 0.0243
deltap = 0.0288
```

Since `delta0` < `deltap`, we can stop in this iteration. Just to be sure, let us increase `M` → `M+1` and check.

```
[~,delta0] = firpm(M+1,f0,A0,w);
delta0, deltap
```

```
delta0 = 0.0229
deltap = 0.0288
```

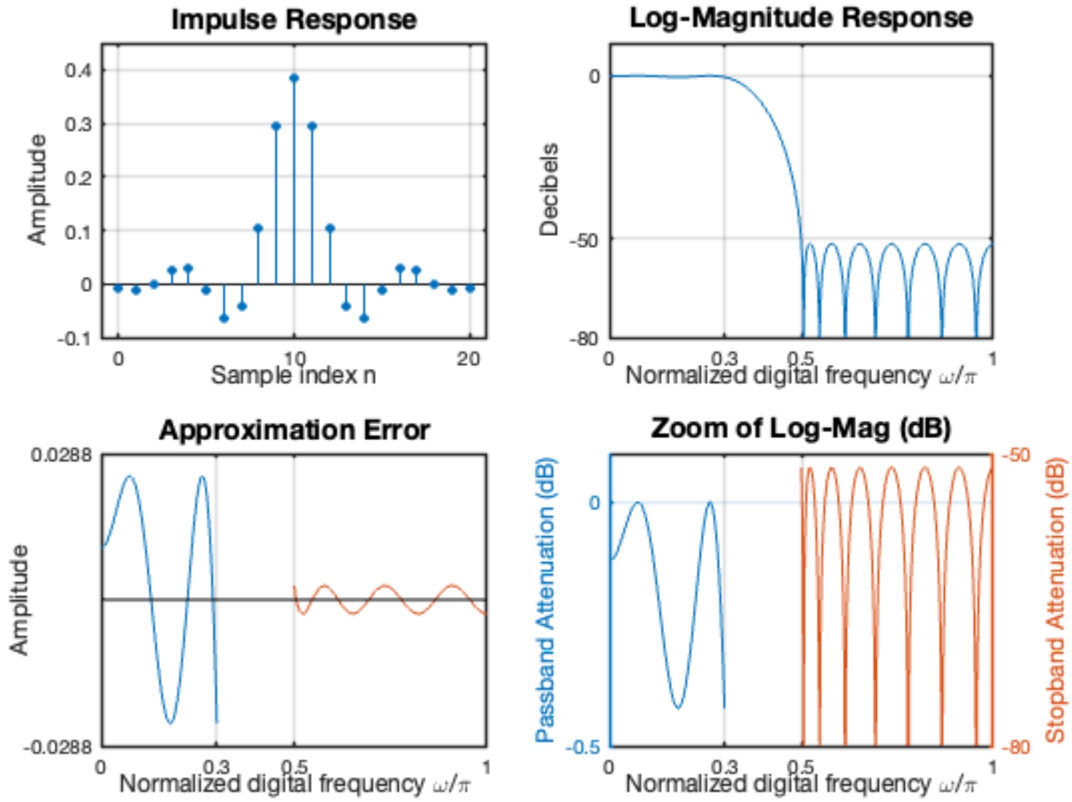
Clearly, $M = 20$ is the optimum filter order. The corresponding filter length is $L = M + 1 = 21$.

---

**(b)** Provide a plot figure similar to Figure 10.12 in the text with appropriate labels and tick marks.

**MATLAB script**:

```
n = 0:M; alpha = M/2;
N = 500; om = linspace(0,1,N+1)*pi; df = 1/N; % freq array for plots
Ip = floor(fp/df); % index of PB edge
Is = ceil(fs/df); % index of SB edge
H = freqz(h,1,om); Hmag = abs(H); % Magnitude response
HdB = 20*log10(Hmag/max(Hmag)); % Log-magnitude response in dB
[Hr,om] = zerophase(h,1,om); % Amplitude response
% Performance Plots
figure('position',[0,0,8,6]*72);
subplot(2,2,1); % Stem plot of impulse response
stem(n,h,'filled','linewidth',1,'MarkerSize',4); axis([-1,M+1,-0.1,0.45]);
set(gca,'ytick',(-0.1:0.1:0.4),'xtick',[0,alpha,M]); ylabel('Amplitude');
xlabel('Sample index n','VerticalAlignment','middle');
title('Impulse Response'); grid;
subplot(2,2,2); % Plot of log-magnitude response in dB
plot(om/pi,HdB,'linewidth',1); axis([0,1,-80,10]);
set(gca,'xtick',f0,'ytick',[-80,-As,0]); grid;
xlabel('Normalized digital frequency \omega/\pi','VerticalAlignment','middle');
ylabel('Decibels'); title('Log-Magnitude Response');
subplot(2,2,3); % Plot of approximation error
plot(om(1:Ip+1)/pi,Hr(1:Ip+1)-1,'linewidth',1,'color',newblue); hold on;
plot(om(Is+1:end)/pi,Hr(Is+1:end),'linewidth',1);
plot([0,1],[0,0],'k','linewidth',0.5); axis([0,1,-deltap,deltap]);
title('Approximation Error'); ylabel('Amplitude','VerticalAlignment','middle');
xlabel('Normalized digital frequency \omega/\pi','VerticalAlignment','middle');
set(gca,'xtick',f0,'ytick',[-deltap,deltap]); grid; hold off;
subplot(2,2,4); % Plot of zoomed log-magnitude response in dB
yyaxis left % Plot passband attenuation
plot(om(1:Ip+1)/pi,HdB(1:Ip+1),'linewidth',1,'color',newblue);
ylabel('Passband Attenuation (dB)');
set(gca,'ylim',[-Ap,0.1],'ytick',[-Ap,0],'xlim',[0,1]);
yyaxis right % Plot stopband attenuation
plot(om(Is+1:end)/pi,HdB(Is+1:end));
ylabel('Stopband Attenuation (dB)'); set(gca,'ylim',[-80,-50]);
title('Zoom of Log-Mag (dB)'); set(gca,'ytick',[-80,-As]);
set(gca,'Xtick',f0); grid;
xlabel('Normalized digital frequency \omega/\pi','VerticalAlignment','middle');
```




## Problem 6.9

### Text Problem 10.58 (Page 621)

Specifications of bandstop digital filter are given below:

$$
\begin{aligned}
|H(e^{j\omega})| &= 1 \pm 0.01, && 0.00\pi \le |\omega| \le 0.35\pi \\
|H(e^{j\omega})| &= 0 \pm 0.004, && 0.45\pi \le |\omega| \le 0.55\pi \\
|H(e^{j\omega})| &= 1 \pm 0.01, && 0.65\pi \le |\omega| \le \pi
\end{aligned}
$$

**(a)** Design an FIR filter using the Parks-McClellan algorithm to obtain a minimum length linear-phase filter impulse response. Indicate the length of your designed filter.

**MATLAB script**: We will use the **pmfirord** and **pmfir** function to obtain this design.

```
clc; close all; clear;
% Given Specifications:
f1 = 0.35; f2 = 0.45; f3 = 0.55; f4 = 0.65;     % Band-edge frequencies
A1 = 1; A2 = 0; A3 = 1;                         % Desired responses
delta1 = 0.01; delta2 = 0.004; delta3 = 0.01;   % Desired ripples
dB1 = round(20*log10((1-delta1)/(1+delta1)),1); % Band-1 ripple in dB
dB2 = round(20*log10(delta2/(1+delta1)),1);     % Band-2 ripple in dB
dB3 = round(20*log10((1-delta3)/(1+delta3)),1); % Band-3 ripple in dB
% 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 = 44
```

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

```
Band−weights: [1.0, 2.5, 1.0]
```

The above weights reported by the **firpmord** function are correct, given by $[\delta_1/\delta_1, \delta_1/\delta_2, \delta_1/\delta_3]$. Thus, we will monitor $\delta_1$ in the **firpm** function.

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

```
Required ripple: 0.01, Designed ripple: 0.0115735
```

The obtained $\delta$ is slightly greater than the required $\delta_1$. Hence we will increase $M \to M+2$ in each iteration because bandstop filter must be a Type-I filter.

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

```
Required ripple: 0.01, Designed ripple: 0.0115771
```

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

```
Required ripple: 0.01, Designed ripple: 0.00625547
```

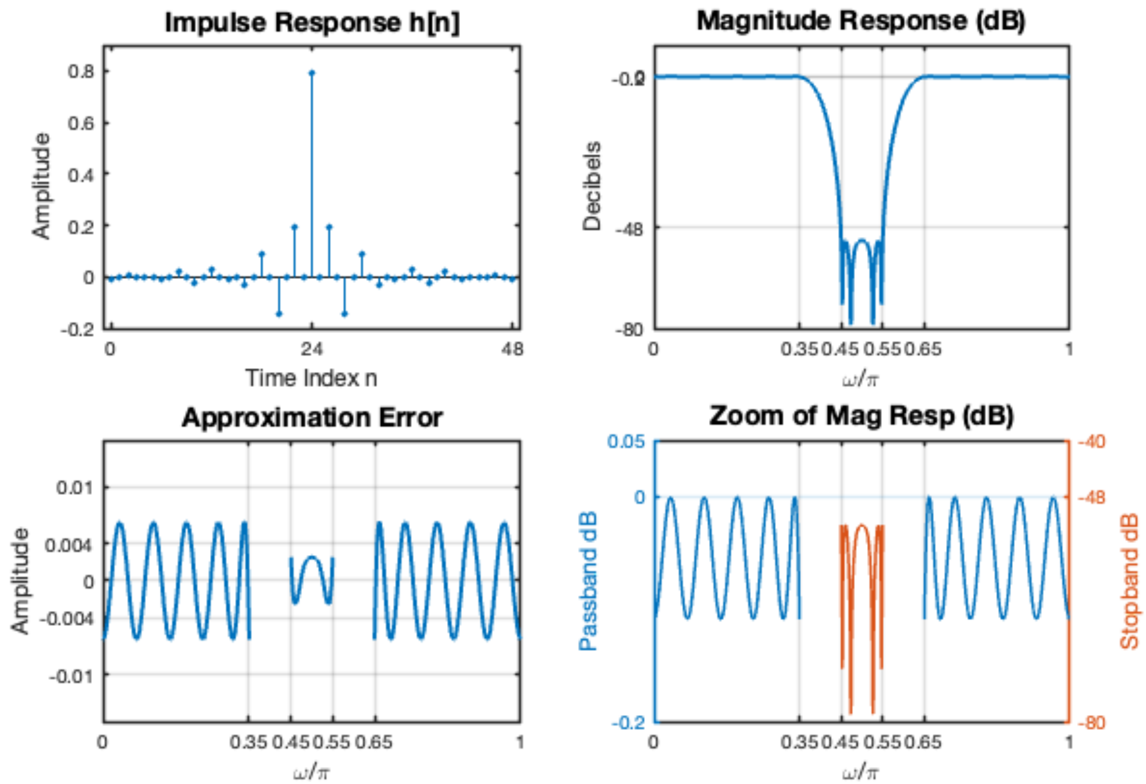
Now the design is complete with a $48$-order filter or length $49$ impulse response.

**(b)** Provide a plot figure similar to Figure 10.32 in the text with appropriate labels and tick marks.

**MATLAB script**:

```
f = linspace(0,1,1001); A = zerophase(h,1,f*pi);
Hmag = abs(A); HdB = 20*log10(Hmag/max(Hmag));
aperr = nan(1,length(f)); magz1 = nan(1,length(f)); magz2 = nan(1,length(f));
ind = f <= f1; aperr(ind) = A(ind)-A1; magz2(ind) = HdB(ind);
ind = f >= f2 & f <= f3; aperr(ind) = A(ind)-A2; magz1(ind) = HdB(ind);
ind = f >= f4; aperr(ind) = A(ind)-A3; magz2(ind) = HdB(ind);
figure('position',[0,0,9,6]*72);
subplot(2,2,1); stem(0:M,h,'filled','markersize',3,'linewidth',1);
xlabel('Time Index n'); ylabel('Amplitude'); title('Impulse Response h[n]');
set(gca,'xtick',[0,M/2,M]); axis([-1,M+1,-0.2,0.9]);
subplot(2,2,2); plot(f,HdB,'linewidth',2); axis([0,1,-80,10]);
xlabel('\omega/\pi'); ylabel('Decibels'); title('Magnitude Response (dB)');
set(gca,'ytick',[-80,dB2,dB1,0],'xtick',[0,f1,f2,f3,f4,1]); grid;
subplot(2,2,3); plot(f,aperr,'linewidth',2);
axis([0,1,-delta1-0.005,delta1+0.005]);
xlabel('\omega/\pi'); ylabel('Amplitude'); title('Approximation Error');
set(gca,'ytick',[-delta1,-delta2,0,delta2,delta1]);
set(gca,'xtick',[0,f1,f2,f3,f4,1]); grid on;
subplot(2,2,4); yyaxis left; % Passband zoom
plot(f,magz2,'linewidth',1.5); ylabel('Passband dB');
set(gca,'ylim',[dB1,0.05],'ytick',[dB1,0,0.05],'ygrid','on');
yyaxis right; % Stopband zoom
plot(f,magz1,'linewidth',1.5); ylabel('Stopband dB');
set(gca,'ylim',[-80,-40],'ytick',[-80,dB2,-40]);
```

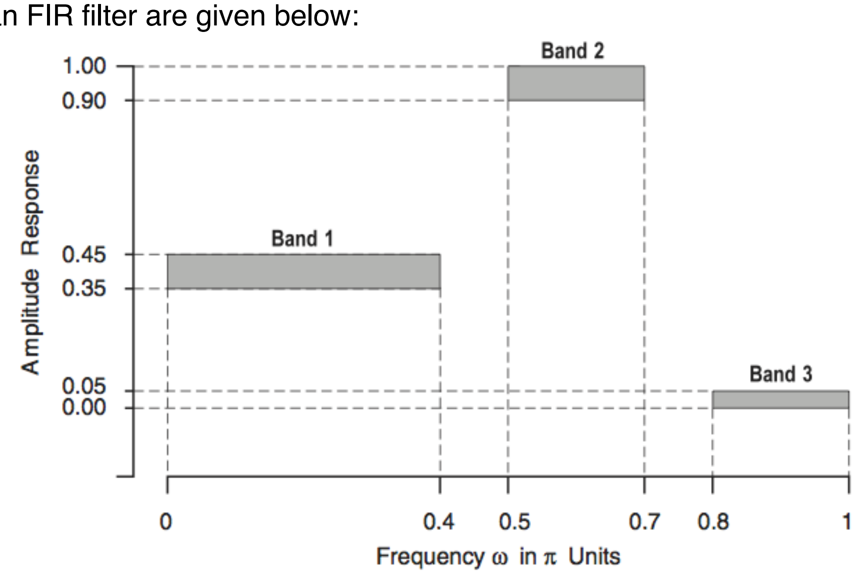
```
xlabel('\omega/\pi'); title('Zoom of Mag Resp (dB)');
set(gca,'xtick',[0,f1,f2,f3,f4,1],'xgrid','on');
```




**Observation:** We now have a proper design in which $\delta_1$, $\delta_2$, and $\delta_3$ ripples are within limits as required and are also uniformly distributed.

## Problem 6.10

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 $\delta_1$ 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)
```

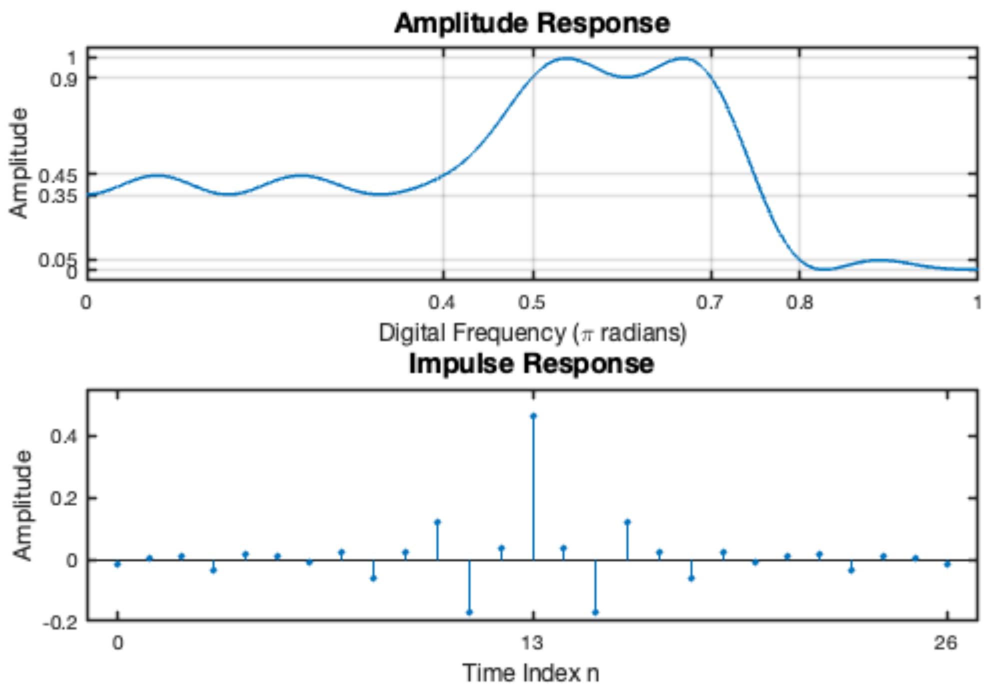
```
    26
```

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('position',[0,0,8,5]*72);
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 $\delta_1$, $\delta_2$, and $\delta_3$ ripples are within limits as required and are also uniformly distributed.

## The `spec_convert` function

```
function [A,B] = spec_convert(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
```