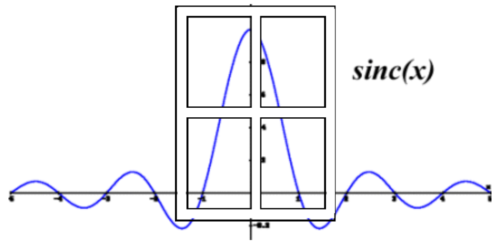
EE 419 - Project 7
FIR Filter Design Methods



Names: Chris Adams & Aiku Shintani **Lab Date: 2/26/19**

Bench #: 9 **Section: 2**

1) FIR Filter Design by Windowing Method – Hand Calculations

Design Specifications:
FIR low pass filter
-6 dB cutoff frequency f_c = 3200 Hz
Sampling rate of f_s =16 KHz

Design Parameters:
Filter length M: 9
Window: Hamming

Filter Design Steps:

Fc: **0.2** cyc/samp

n:									
Ideal $h[n]_{Ideal}$	**-0.0757**	**-0.0624**	**0.0935**	**0.3027**	**0.4**	**0.3027**	**0.0935**	**-0.0624**	**-0.0757**
Window w[n] (Hamming	**0.08**	**0.2147**	**0.54**	**0.8653**	**1**	**0.8653**	**0.54**	**0.2147**	**0.08**
Filter h[n]	**-0.00605**	**-0.0134**	**0.0505**	**0.2619**	**0.4**	**0.2619**	**0.0505**	**-0.0134**	**-0.00605**

Equations for the

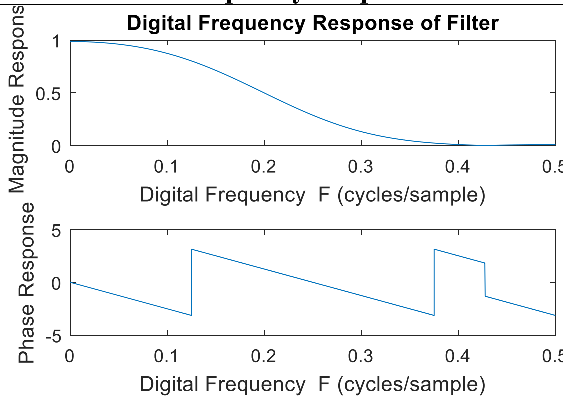
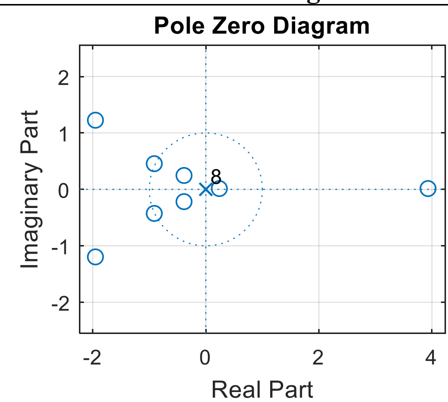
IDEAL FILTER h[n]: sin(2*pi*n*Fc)/pi/n

HAMMING WINDOW w[n]: 0.54 + 0.46cos(2*pi*n/(M-1))

DIFFERENCE EQUATION:
y[n] = -0.00605x[n] - 0.0134x[n-1] + 0.0505x[n-2] + 0.2619x[n-3] + 0.4x[n-4] + 0.2619x[n-5] + 0.0505x[n-6] - 0.0134x[n-7] - 0.00605x[n-8]

Filter Performance:

a) Using your Matlab analysis programs, **plot the frequency response (linear scale vs. digital frequency) and pole/zero diagram** for your filter.

Frequency Response	**Pole/Zero Diagram**
	

b) **Based on the pole/zero diagram** and pole/zero locations (complex values), **explain** how can you tell that this filter is:

i. FIR : all poles are at the origin, #poles = #zeros so causal

ii. **Linear Phase:**

Report the zero locations in polar form: |radius| ∠ angle

Radius		Angle (degrees)	1 / Radius
3.9452	∠	0	0.2535
0.2535	∠	0	3.9452
2.2854	∠	148.0011	0.4376
2.2854	∠	-148.0011	0.4376
1	∠	153.8422	1
1	∠	-153.8422	1
0.4376	∠	148.0011	2.2854
0.4376	∠	-148.0011	2.2854

Explain how these confirm the linear phase behavior: **Each of the zeros are matched with their complex conjugate pair. Each of the zeros not on the unit circle are matched with their reciprocal radius (1/r) zero pair (at the same angle). Symmetrical h[n].**

iii. Low-Pass: Based off of the digital frequency response plot above, the magnitude response is unity gain at DC and the most attenuation occurs at F = 0.5. This response is low pass in nature. Based off of the pole/zero plot above, the system can be confirmed to be lowpass as well. The zeros are closest to the unit circle at high frequencies and furthest to the unit circle at low frequencies.

c) From the Matlab filter analysis plots, report the following **actual performance measures** for your filter:

a. Magnitude response at DC = **0.9859 (linear units)**

b. Stop-band attenuation (in dB) at the Nyquist Freq. = **41.83dB @ F = 0.5 cycles/sample**

c. Magnitude response at the designed Cutoff Freq. = **0.499 (-6.038dB) @ F_c = 0.2 cycles/sample**

2) FIR Filter Design by Windowing Matlab Program

```
function hn_lp = FIR_Filter_By_Window (M,Fc,window(M))
         M = the filter length (odd)
         Fc = filter cutoff digital frequency (-6dB) (cycles/sample)
         window = the Matlab window function name
         hn_lp = windowed impulse response values for the Low-pass FIR filter
```

Test Case:

FIR low pass filter
-6 dB cutoff frequency f_c = 3200 Hz
Sampling rate of f_s =16 KHz
Filter length M: 9
Window: Hamming

Test Results:

Matlab function call used for the test case: `FIR_Filter_By_Window(9, 0.2, hamming(9))`

Filter Coefficients determined by your program for the test case:

B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
-0.006055	-0.013392	0.05052	0.2619	0.4	0.2619	0.05052	-0.013392	-0.006055

Listing of your Function m-file Code:

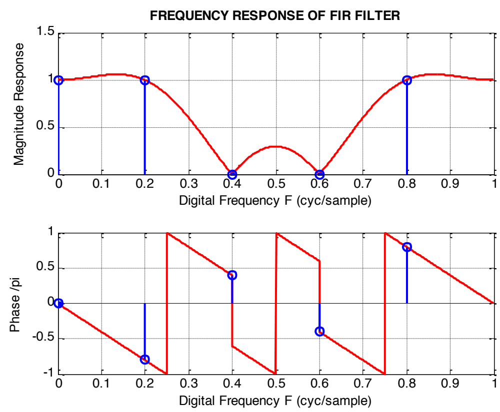
```
function hn_lp = FIR_Filter_By_Window(M, Fc, window)
% This function takes three arguments: M is the filter length, Fc is the
% desired cutoff frequency, and window is the window type that is specified
% by the user as window_name(M).

%test this func. using: hn_lp = FIR_Filter_By_Window(9, 0.2, hamming(9))

n = -(M-1)/2:(M-1)/2;
h_ideal = sin(2*pi.*n*Fc)/pi./n;
h_ideal(((M-1)/2)+1) = 2*Fc;
h_window = window.';
hn_lp = h_ideal.*h_window;

end
```

3) [Matlab] FIR Filter Design by Frequency Sampling

```
function [hn,HF,F]=FIR_Filter_By_Freq_Sample(HF_mag_samples,figurenum)
```

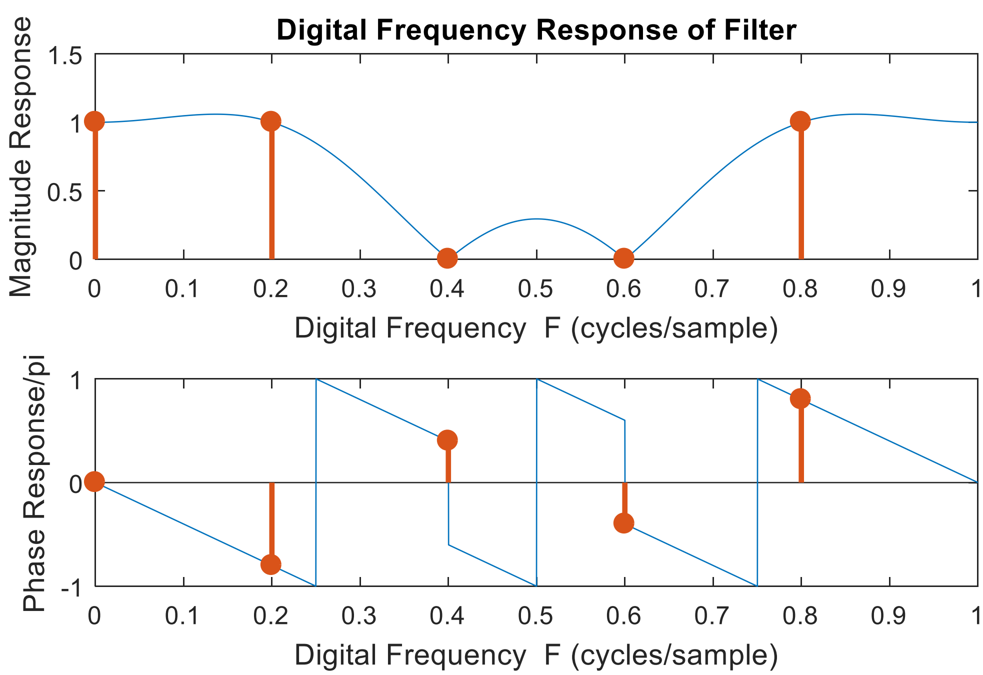
% hn - impulse response of filter (same length as HF_mag_samples)
% HF - complex frequency response of filter
% F – digital frequency values corresponding to the estimated H(F)
% HF_mag_samples – H[k] Magnitude response samples for desired filter
% figurenum - Figure # to plot frequency responses

Test Case: [hn, HF, F] = FIR_Filter_By_Freq_Sample([1 1 0 0 1], 100);

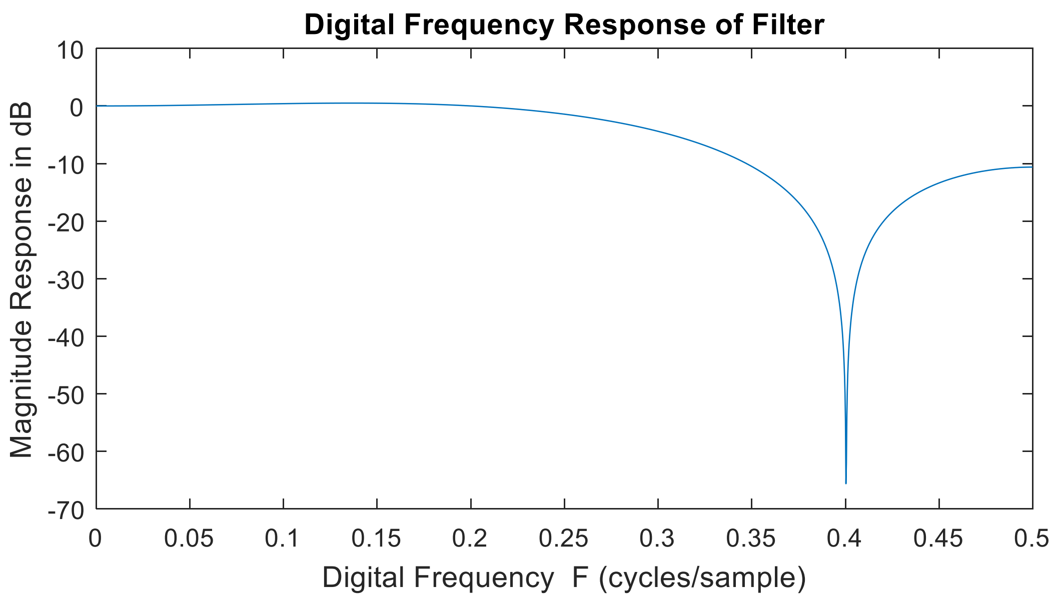


Test Results:

a) **The frequency response plot resulting from the test case** (should match the figure above)**.**
NOTE: It is OK if the Phase Angle values when |H[k]|=0 are plotted as 0 instead of the values show.



b) **The dB Magnitude response plot for the test case**



c) A listing of your m-file.

```
function [hn, HF, F] = FIR_Filter_By_Freq_Sample(HF_mag_samples, figure_num)
%This function takes two input arugments: HF_mag_samples which correspond
%to the H_k magnitude (positive only and for a DC gain of <1 all
%coefficients are <1) response that the user want and the figure #. The
%function returns the corresponding unit sample response (h[n]), the
%frequency response (HF), and the digital freq (F).

% part a
k = 0:length(HF_mag_samples)-1; %# of mag samples is M
M = length(k);                  % M is equal to # k samples for DFT

angle_rad = -pi.*k*(M-1)/M;     %compute angle argument (in radians)

%correct angles arguments so they're within -pi and pi
for x=1:length(angle_rad)

    while (angle_rad(x) < -pi)
        angle_rad(x) = angle_rad(x)+2*pi;
    end

    while (angle_rad(x) > pi)
        angle_rad(x) = angle_rad(x)-2*pi;
    end
end

Hk_angle = exp(j*angle_rad);    %Hk_angles
Hk = HF_mag_samples.*Hk_angle;  %Hk in complex form
hn = real(ifft(Hk));            %get the unit sample response

% part b
HF_no_pad = fft(hn); %for use in low-res FFT ==> DFT, discrete
                     %compute non-padded HF

M_pad = 2^12;        %for use in high-res FFT ==> "DTFT", psuedo continuous
HF = fft(hn, M_pad); %compute padded HF
F = 0:1/(M_pad-1):1; %sample freq. spacing
```

```
Fk = 0:1/M:(M-1)/M;  %for stem plots

HF_mag = abs(HF);                 %compute the magnitude of padded HF
HF_mag_no_pad = abs(HF_no_pad); %compute the magnitude of non-padded HF
HF_ang = angle(HF)/pi;            %compute the angle of padded HF
HF_ang_no_pad = angle_rad/pi;    %compute the angle of non-padded HF

%plot digital frequency response
figure(figure_num)
subplot(2,1,1)
plot(F, HF_mag) %plot magnitude response (linear)
xlabel('Digital Frequency  F (cycles/sample)')
ylabel('Magnitude Response')
title('Digital Frequency Response of Filter')

hold on

%superimpose non-padded DFT magnitude
stem(Fk, HF_mag_no_pad, '.', 'MarkerSize', 20, 'Linewidth', 2);

%plot phase response
subplot(2,1,2)
plot(F, HF_ang)
xlabel('Digital Frequency  F (cycles/sample)')
ylabel('Phase Response/pi')

hold on

%superimpose non-padded DFT phase
stem(Fk, HF_ang_no_pad, '.', 'MarkerSize', 20, 'Linewidth', 2);

%part c, plot magnitude response (in dB this time)
F_c = 0:1/(M_pad-1):0.5;                   %F = 0-0.5 instead of F =0-1 like
before
figure(figure_num + 1)
plot(F_c, 20*log10(HF(0:0.5*(M_pad - 1)))) %only take up to F = 0.5 worth of HF
xlabel('Digital Frequency  F (cycles/sample)')
ylabel('Magnitude Response in dB')
title('Digital Frequency Response of Filter')

end
```

4a) FIR By Frequency Sampling - Matlab & Manual Design

Design Specifications:
FIR Low Pass Filter
-6 dB Cutoff Frequency f_c = 12 KHz
Sampling Rate of f_s =48 KHz

Design Parameters:
Filter length M: 9
Method: Frequency Sampling
Constraint: |H[k]| = 0 or 1

Filter Design:

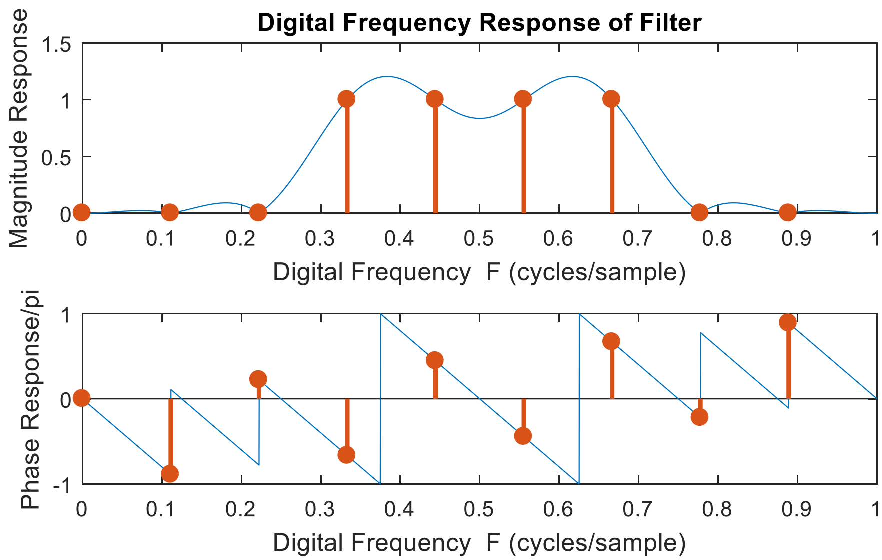
Sample Index k	Digital Frequency F_k (cycl/samp)	Magnitude $\|H_d[k]\|$	Phase Angle $\angle H_d[k]$	Complex H[k]	Unit Sample Response h[n]
0	**0**	**0**	**0**	**0**	**-0.0725**
1	**1/9**	**0**	**-8*pi/9**	**0**	**0.1111**
2	**2/9**	**0**	**-16*pi/9**	**0**	**0.0591**
3	**3/9**	**1**	-24*pi/9 +18*pi/9 = **-6*pi/9**	**-0.5 – j0.866**	**-0.3199**
4	**4/9**	**1**	-32*pi/9 + 36*pi/9 = **4*pi/9**	**0.1736 + j0.9848**	**0.4444**
5	**5/9**	**1**	-40*pi/9 + 36*pi/9 = **-4*pi/9**	**0.1736 - j0.9848**	**-0.3199**
6	**6/9**	**1**	-48*pi/9 + 54*pi/9 = **6*pi/9**	**-0.5 + j0.866**	**0.0591**
7	**7/9**	**0**	-56*pi/9 + 72*pi/9 = **16*pi/9**	**0**	**0.1111**
8	**8/9**	**0**	-64*pi/9 + 72*pi/9 = **8*pi/9**	**0**	**-0.0725**

The difference equation for the filter designed:

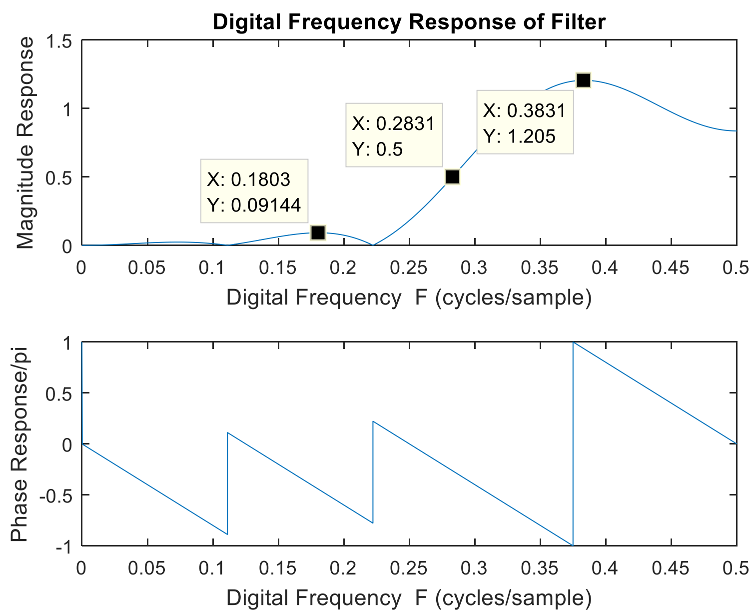
$$y[n] = -0.0725x[n] + 0.1111x[n-1] + 0.0591x[n-2] - 0.3199x[n-3] + 0.4444x[n-4] - 0.3199x[n-5] + 0.0591x[n-6] + 0.1111x[n-7] - 0.0725x[n-8]$$

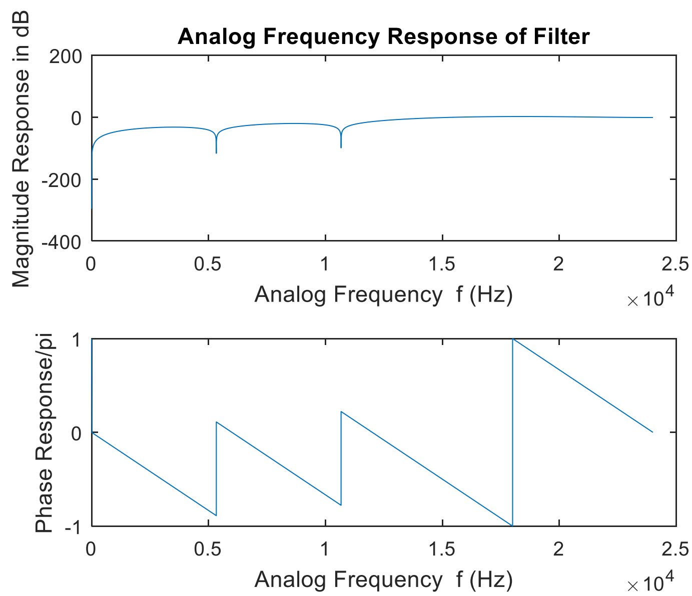
Filter Performance:

Plot of the magnitude and phase of the filter's frequency response (using plot from FIR_Filter_By_Freq_Sample( ))

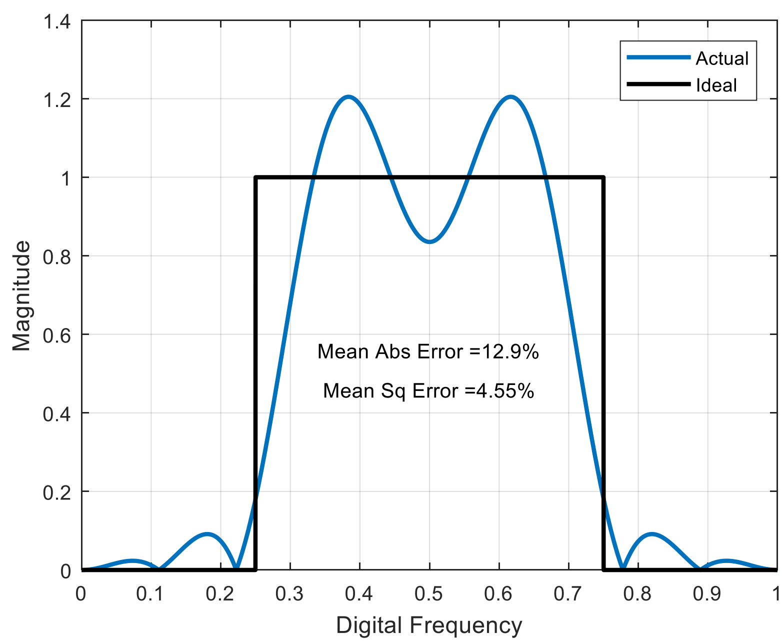


Plots of the magnitude and phase of the filter's frequency response (from your Matlab filter analysis program).





Report the percent errors in magnitude (mean absolute error and mean squared error) between the M=9 filter and an ideal filter (computed by the M-File function: `magnitude_response_error()`)



mean_abs_error = 12.9%   mean_sq_error = 4.55%

Quantify the performance of the filter, based on the frequency response plots.

1) Maximum Passband Ripple = **0.205**

2) Maximum Passband Attenuation A_p = **3.61**dB

3) Cutoff Frequency (-6 dB) $\mathbf{F_c}$ = **0.2831**cycle/sample; f_c = 13.589KHz

4) Maximum Stopband Ripple = **0.09144**

5) Minimum Stopband Attenuation A_s = **20.78**dB

4b) Improved Design By Iteration With Matlab

Design Goals:

- minimize filter's frequency response mean absolute error percentage
- reduce the magnitude response overshoots and ripple
- achieve closer to the desire -6dB cutoff frequency.

Final Filter Design:

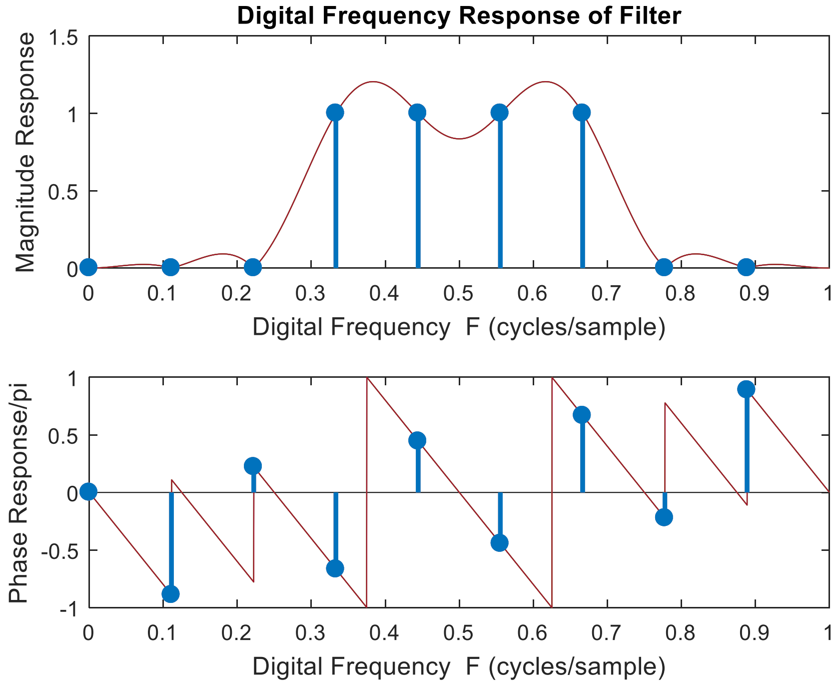
Index **k, n**	0	1	2	3	4	5	6	7	8
Magnitude Samples $\|H_d[k]\|$	**0**	**0**	**0.179**	**0.912**	**1**	**1**	**0.912**	**0.179**	**0**
Unit Sample Response **h[n]**	**-0.0323**	**0.0717**	**0.0315**	**-0.3032**	**0.4647**	**-0.3032**	**0.0315**	**0.0717**	**-0.0323**

The revised difference equation for the improved filter:

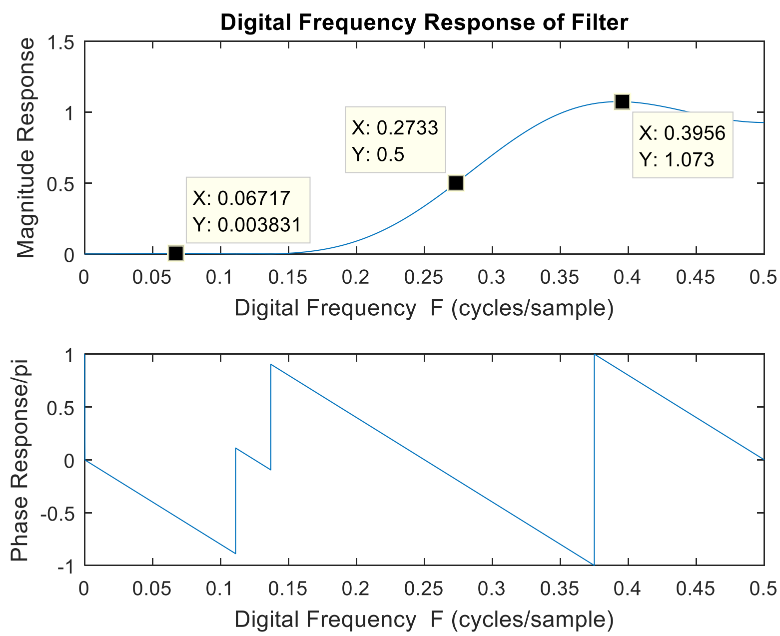
$$y[n] = -0.0323x[n] + 0.0717x[n-1] + 0.0315x[n-2] - 0.3032x[n-3] + 0.4647x[n-4] - 0.3032x[n-5] + 0.0315x[n-6] + 0.0717x[n-7] - 0.0323x[n-8]$$

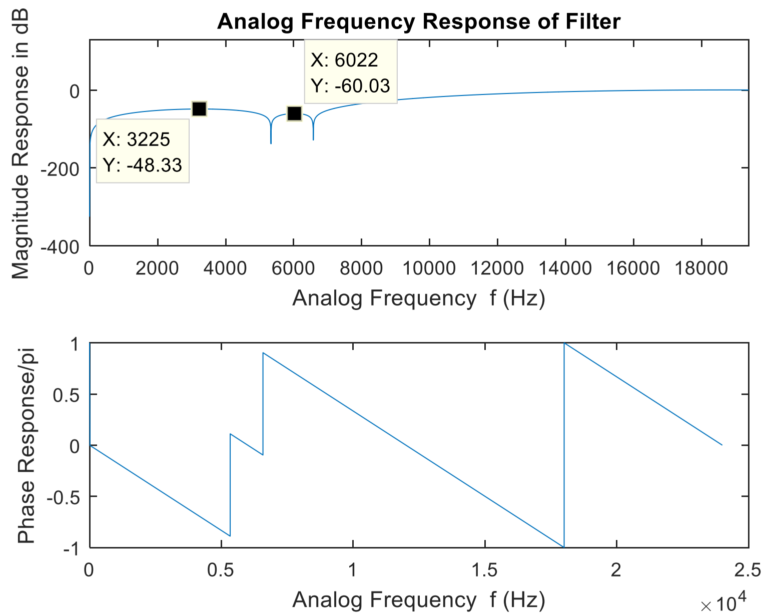
Final Filter Performance:

Plot of the magnitude and phase of the filter's frequency response (using plot from FIR_Filter_By_Freq_Sample( ))

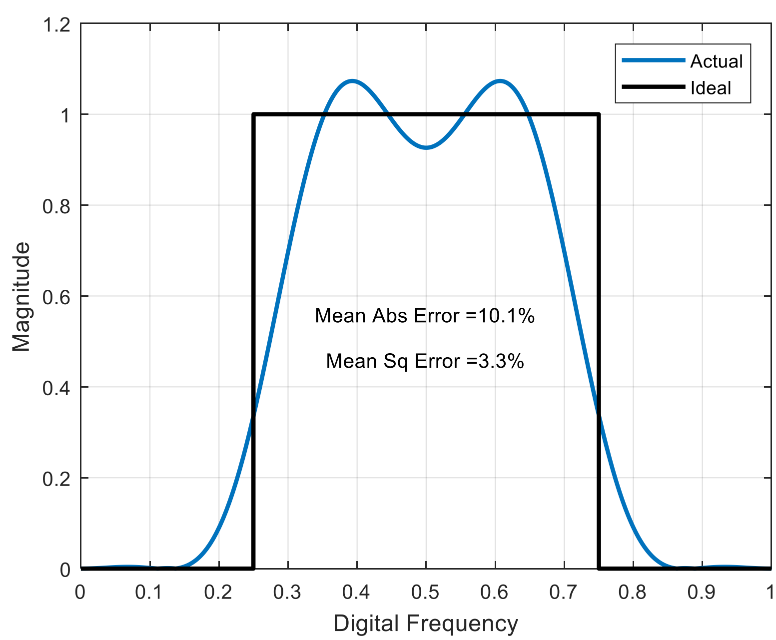


Plots of the magnitude and phase of the filter's frequency response (from your Matlab filter analysis program).





Report the percent errors in magnitude (mean absolute error and mean squared error) between the M=9 filter and an ideal filter (computed by the M-File function: `magnitude_response_error()`)



mean_abs_error = 10.1% **mean_sq_error = 3.3%**

Quantify the performance of the filter, based on the frequency response plots.

1) Maximum Passband Ripple = **0.073**

2) Maximum Passband Attenuation A_p = **1.27**dB

3) Cutoff Frequency (-6 dB) $\mathbf{F_c}$ = **0.2733**cycle/sample; f_c = 13.118KHz

4) Maximum Stopband Ripple = **0.003831**

5) Minimum Stopband Attenuation A_s = **48.33**dB

Revised: 2/26/19

5) [Matlab] FIR Filter Design Comparison

All designs should use the following parameters:
Target Cutoff Frequency (-6 dB) $F_c = 0.20$ cycles/sample
Transition Width $\Delta F_{Transition} = 0.133$ cycles / sample
Filter Length M = 15
Linear Phase Response
Real, Symmetric Filter Coefficients
Sampling Frequency of $f_s = 1$ KHz

The cases to be evaluated are:

a) FIR Design by Windowing (3 cases)
   i) Rectangular Window
   ii) Hamming Window
   iii) Kaiser Window ($\delta_s = \delta_p = 0.03$)

b) FIR Design By Frequency Sampling (No windowing)

c) FIR Optimal Design By Parks-McClellan Algorithm

Alternative Designs:

a) Listing of Matlab commands used to create each design. (Analysis steps do not need to be shown).

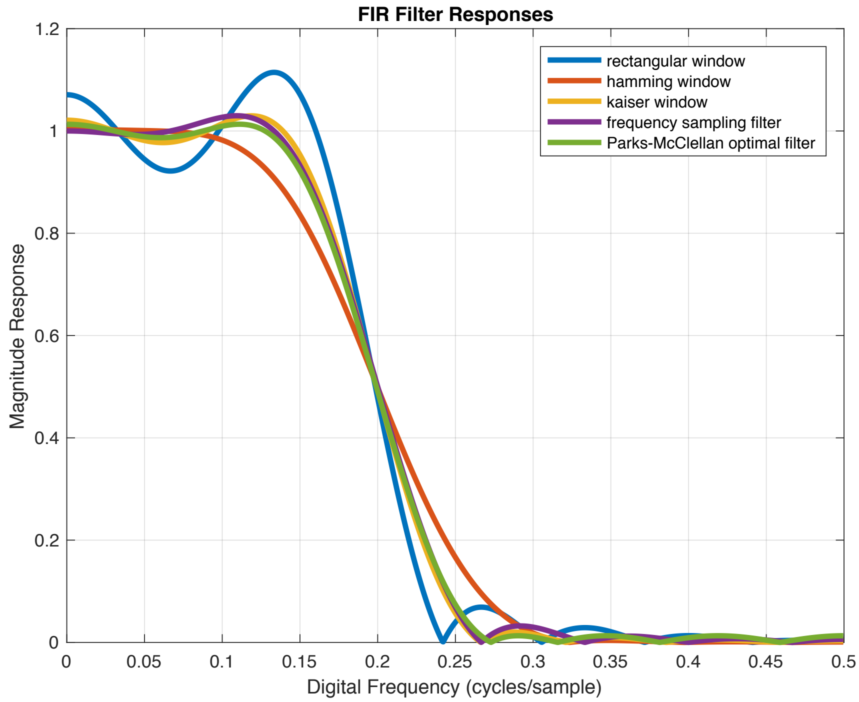
```
M = 15;                    %filter length
Fc = 0.2;                  %digital cutoff
fsample = 1000;            %sampling frequency
beta = 2.180895622;        %beta for kaiser

%rectangular window filter
rect_filter = FIR_Filter_By_Window(M, Fc, rectwin(M));
[h_rect] = freqz(rect_filter,1,1024);

%hamming window filter
hamm_filter = FIR_Filter_By_Window(M, Fc, hamming(M));
[h_hamm] = freqz(hamm_filter,1,1024);

%kaiser window filter
kaiser_filter = FIR_Filter_By_Window(M, Fc, kaiser(M,beta));
[h_kaiser] = freqz(kaiser_filter,1,1024);

%filter by frequency sampling
hn = FIR_Filter_By_Freq_Sample([1 1 1 0.5 0 0 0 0 0 0 0 0 0.5 1 1]  , 1);
[h_sample] = freqz(hn,1,1024);

%Parks-McClellan optimal FIR filter
pm_filter = firpm(M-1, [0 .1335 0.2665 0.5]*2, [1 1 0 0]);
[h_pm,w] = freqz(pm_filter,1,1024);
```

b) Resulting Filter Designs:

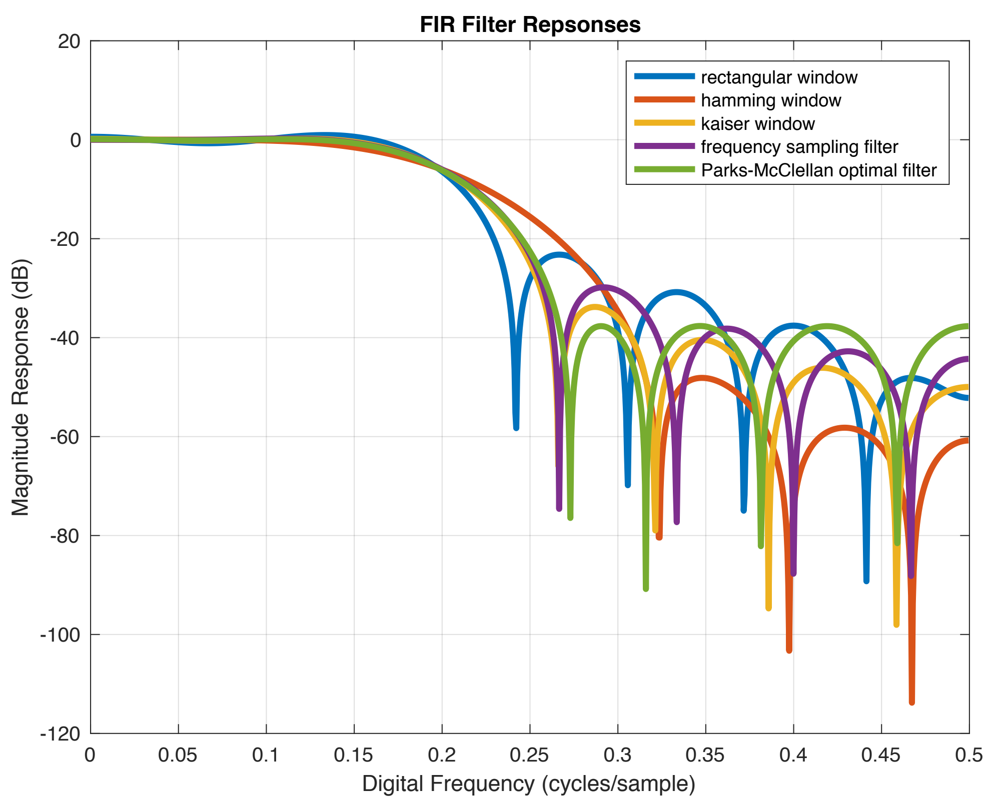
Filter Coefficients	Windowing – Rectangular	Windowing – Hamming	Windowing – Kaiser	Frequency Sampling	Parks - McClellan
B_0	0.0267	0.0021	0.0103	0.0041	0.0061
B_1	0.0505	0.0063	0.0261	0.0206	0.0231
B_2	-0.0000	-0.0000	-0.0000	-0.0000	-0.0017
B_3	-0.0757	-0.0331	-0.0577	-0.0571	-0.0526
B_4	-0.0624	-0.0401	-0.0537	-0.0539	-0.0501
B_5	0.0935	0.0773	0.0876	0.0880	0.0867
B_6	0.3027	0.2889	0.2979	0.2983	0.2958
B_7	0.4000	0.4000	0.4000	0.4000	0.3987
B_8	0.3027	0.2889	0.2979	0.2983	0.2958
B_9	0.0935	0.0773	0.0876	0.0880	0.0867
B_{10}	-0.0624	-0.0401	-0.0537	-0.0539	-0.0501
B_{11}	-0.0757	-0.0331	-0.0577	-0.0571	-0.0526
B_{12}	-0.0000	-0.0000	-0.0000	0.0000	-0.0017
B_{13}	0.0505	0.0063	0.0261	0.0206	0.0231
B_{14}	0.0267	0.0021	0.0103	0.0041	0.0061

Test Results and Performance Comparison:

Composite Magnitude Response Plot (linear scale)



Composite Magnitude Response Plot (log (dB) scale)



Design Method	Maximum Passband Ripple δp	Maximum Stopband Ripple δs	Minimum Stopband Attenuation As (dB)	|H(F)| at Fc = 0.2 cyc/spl	Passband Edge Freq. Fp (cyc/spl)	Stopband Edge Freq. Fs (cyc/spl)	Transition Bandwidth ΔF_T (cyc/spl)	Mean Absolute Error (%)
Windowing (Rectangle)	**0.1146**	**0.0690**	**-23.223**	**-6.3354**	**0.1631**	**0.2373**	**0.0742**	**6.44%**
Windowing (Hamming)	**0.0032**	**0.0039**	**-48.179**	**-6.0125**	**0.1089**	**0.2925**	**0.1836**	**8.09%**
Windowing (Kaiser)	**0.0296**	**0.0124**	**-38.132**	**-6.0851**	**0.1465**	**0.2559**	**0.1094**	**5.84%**
Frequency Sampling	**0.0298**	**0.0322**	**-29.843**	**-5.9674**	**0.1416**	**0.2578**	**0.1162**	**6.19%**
Parks-McClellan	**0.0132**	**0.0131**	**-37.655**	**-6.1301**	**0.1387**	**0.2603**	**0.1216**	**6.18%**

Conclusions of your comparison between methods and their performance. Identify relative strengths / weaknesses of each approach, and the tradeoffs involved in selecting an FIR filter design method.

> **Different FIR design methods have different advantages and disadvantages. FIR filter by windowing is an efficient quick way to develop a FIR filter however you cannot modify the characteristics. Characteristics depend of the type of window for the most part (Kaiser you can control the ripple). The rectangular window has the quickest transition but the worst ripple. Hamming has the least ripple but the worst/slowest transition. Kaiser is the best of the window filters but requires and extra calculation. FIR filter design by frequency sampling is relatively accurate to specs somewhat difficult to tune. Finally, the Parks-McClellan optimal filter is the most accurate to the specifications since it optimizes and tunes the FIR filter.**

Project Conclusions:

Summarize one or two learnings about FIR Filter Design that this project helped you understand better. Also describe any particular challenges that you had to overcome, and at least one suggestion for improvement of this lab in the future.

Name: Aiku Shintani

Conclusions: Through the course of this project, different methods of FIR filter design were investigated. The limitations of the FIR design via windowing and frequency sampling were clear when compared to the Parks McClellan method. The Parks McClellan method takes advantage of statistical theories which optimize the filter design process. The frequency sampling method is relatively useful but requires quite a bit of tuning until the desired response is achieved. It is clear from these observations that in practical FIR design, the Parks McClellan method is the most efficient in achieving quick results. The report mentions that we should perform hand calculations and report them but in terms of the final report the grader cannot tell whether we performed them. I took the time to make our hand calculations neat and structured but since I am not turning them in it was not worth it.

Name: Chris Adams

Conclusions: Throughout this project, the advantages and disadvantages of different FIR filter design techniques were made clear. While it is quick and easy to implement FIR filters by windowing, different windows have different benefits in terms of transition band and ripple. FIR filters by frequency sampling is better for maintaining certain filter specs but the Parks McClellan produces the best result due to the statistical methods it utilizes to optimize the filter. Some of the calculations were tedious but through trial and error, the desired filter response was acquired. Lastly, in future experiments, hand calculations should be emphasized less. The lab was quite long, and while most everything seemed important, more time should be spent on matlab analysis than hand calculations.