University of Colorado at Boulder — Problem set 9 — ECEN 4632, Fall 2016

Due Friday, November 6, 2016

The goal of this homework is to design a bandstop filter to cancel the noise in an audio signal. This is a MATLAB only homework, and therefore each student needs to turn in her/his own lab report and own programs. In order to receive a high grade for this assignment your report should include all the items described below.

1. A brief theoretical description of each filter method, and its connection to the course material.
2. When applicable, a detailed description of the choice of parameters.
3. An in-depth discussion of the experimental results. You should compare the experimental results with the advertised properties of the filter.
4. The MATLAB code, with comments.

## Noise cancellation

Noise-cancellation headphones measure the external ambient noise with a microphone, and generate a signal that matches the noise, but with the opposite amplitude. This "anti-noise" is added to the original music signal and effectively interfere destructively with the ambient noise. The success of the approach relies on the fact that the noise is narrow band. This assumption is a realistic assumption for the vibration of the engine of an airplane, but will fail to apply for the cry of a baby in the same airplane.

In this project you will tackle the problem of removing a narrow-band noise created by the engine of an airplane. You will use a passive approach, and design a narrow-band bandstop filter to remove the frequencies that match the frequency range of the noise. The audio signal that you will process has been sampled at 11,025 Hz. You are told that the noise is in the frequency range [1,600-1,700] Hz.

You will design two band stop filters remove the noise in this narrow frequency range, while leaving the rest of the signal intact.

```
In order to minimize the amount of time spent on programming, I wrote an
appendix with a summary of the design procedure, and examples of designs of
bandstop filters using the Kaiser window, and the Parks-McClellan algorithm.

Please read this appendix carefully.  All the MATLAB examples are in the
archive: http://ecee.colorado.edu/~fmeyer/class/ecen4632/homework9.zip
```

1. Download the audio files, and the MATLAB examples:
   http://ecee.colorado.edu/~fmeyer/class/ecen4632/homework9.zip Load the noisy audio file using the following MATLAB command:

   ```
   >> [song,fs,nb]= audioread('noisy.wav');
   ```

   This is a (low-quality) mono signal with sampling frequency of 11,025 Hz. Listen to the signal with the MATLAB command

   ```
   >> sound(song,fs)
   ```

Can you discern any voice? the goal of the project is to remove the noise and discover the voice!

2. You will design a bandstop filter with the following specifications:

   - passbands: $[0 - 1400]$Hz and $[1,900 - 5,512.5]$Hz
   - stopband $[1,600 - 1,700]$Hz,
   - tolerance in the passband: $0.5$ dB,
   - Maximum gain in the stopband: $-100$ dB.

   You will implement the filter using each of the following design methods:

   - Kaiser,
   - Parks-McClellan.

   For each of the designs, you will compute

   (a) the order of the filter.

   (b) the number of add/multiply operations per input sample required to implement the filter. Be sure to explain the structure you assume;

   (c) the magnitude response (in dB) using `myfreqz`;

   (d) a magnified plot of the magnitude response, focusing on the passband ripple (linear scale);

   (e) the group delay (in samples) using `grpdelay`;

   (f) the pole-zero diagram;

   (g) the impulse response using `filter` and `stem` for 100 samples.

3. Filter the noisy signal using each of your de-noising filters. Listen to the filtered and original files. How do they compare? you will need to re-calibrate the denoised signal so that its dynamic range is in $[-1, 1]$ before playing it with the command `sound`.

4. (Extra credit.) After filtering the signal still contains some minute amounts of noise. Unfortunately, part of the song in the range $[1,600 - 1,700]$Hz has also been removed. Suggest a method to improve the filtered signal by removing more noise while adding back some of the signal.

Appendix

# 1 Design of FIR by windowing

## 1.1 Impulse-response truncation method

Given a desired ideal filter $H_d(e^{j\omega})$ we can compute its pulse response $h_d[n]$. Unfortunately $h_d[n]$ is infinite and noncausal. A Finite Impulse Response (FIR) filter can be constructed by neglecting $h_d[n]$ when $n$ is large. We define the impulse-response truncation method as follows.

**Impulse-response truncation method**

Input:

- desired amplitude response $A_d(\omega)$
- initial phase: $\phi_0 = 0$ or $\pi/2$
- filter's order $N$

the ideal frequency response is given by

$$H_d(e^{j\omega}) = A_d(\omega)e^{j\phi_0 - N\omega/2}$$

1. Compute the impulse response of ideal desired filter

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega \tag{1}$$

2. Truncate the impulse response by taking

$$\begin{cases} h[n] = h_d[n] & \text{if } 0 \le n \le N \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

### 1.1.1 Optimality of the impulse-truncation method

If we compare the Fourier transform of the ideal desired filter $H_d(e^{j\omega})$ and the Fourier transform of the FIR filter $H(e^{j\omega})$ using the mean squared error, then the impulse-truncation method is optimal.

**Theorem 1**. *For a given frequency response $H_d(e^{j\omega})$ and a given filter order $N$, the filter obtained with the impulse-truncation method has the minimum mean squared error in the frequency domain among all other filters of the same order.*

## 1.2 Rectangular window

Let us denote by $w[n]$ the window that is the indicator of the set $\{0, \ldots, N\}$. Then the impulse response truncation method is given by

$$h[n] = w[n]h_d[n]. \tag{3}$$

This new notations allows us to compute $H(e^{j\omega})$the frequency response of the FIR $h[n]$. We have,

$$H(e^{j\omega}) = \frac{1}{2\pi} H_d \star W(e^{j\omega}) \tag{4}$$

But

$$W(e^{j\omega}) = \sum_{n=0}^{N} e^{-j\omega n} = \frac{1 - e^{-j\omega(N+1)}}{1 - e^{-j\omega}} = e^{-j\omega N/2} \frac{\sin\left((N+1)\omega/2\right)}{\sin(\omega/2)}. \tag{5}$$

The Fourier transform of the rectangular window is composed of two parts:

1. a linear phase shift $e^{-j\omega N/2}$,
2. an amplitude $\frac{\sin((N+1)\omega/2)}{\sin(\omega/2)}$

The function

$$D_N(\omega) = \frac{\sin\left((N+1)\omega/2\right)}{\sin(\omega/2)} \tag{6}$$

is called the Dirichlet kernel. The main properties of $D_N(\omega)$ are

1. $\sup_{\omega \in (-\pi, pi]} D_N(\omega) = D_N(0) = N$
2. $D_N(2\pi k/N) = 0, k = -N/2, \ldots, N/2$
3. $|D_N(\pm 3\pi/N)| \approx 2N/3\pi$
4. $|D_N(\pm 3\pi/N)|/D_N(0) \approx 2/3\pi$

The effect of the convolution between $W(e^{j\omega})$ and $H_d(e^{j\omega})$ is to blur the sharp edge of $H_d(e^{j\omega})$. As a result the filter $H(e^{j\omega})$ is only a poor approximation of $H_d(e^{j\omega})$.

### 1.2.1 Gibbs phenomenon

The Gibbs phenomenon is a direct consequence of the fact that the side-lobes of the Dirichlet kernel do not converge to 0 as $N \longrightarrow \infty$. Indeed, one can show that the approximation error, $\sup_\omega |H(e^{j\omega}) - H_d(e^{j\omega})|$ does not depend on N. We have

$$\forall\ N, \quad \sup_\omega |H(e^{j\omega}) - H_d(e^{j\omega})| \approx 0.0895. \tag{7}$$

The quality of $H(e^{j\omega})$ can only be improved by replacing the rectangular window by another window that is longer and smoother. In fact, because our goal is to design a filter that performs according to $H_d(e^{j\omega})$, we need to choose $W(e^{j\omega})$ so that

$$2\pi H_d \star W(e^{j\omega}) \approx H_d(e^{j\omega}). \tag{8}$$

In other words, we need to have
$$W(e^{j\omega}) \approx 2\pi\delta(\omega). \tag{9}$$
Obviously, we cannot have $W(e^{j\omega}) = 2\pi\delta(\omega)$, since this would entail $w[n] = 1$ for all $n$, and $h[n]$ would not be FIR.

The goal of FIR design using windows is to find windows that are concentrated around $\omega = 0$ as possible in order to approximate the Dirac impulse. The concentration is achieved by having a very fast decay of $W(e^{j\omega})$. We know from Fourier analysis that the fast decay can only be achieved if the function $w[n]$ is very smooth in the time domain. Of course, the window $w[n]$ is a discrete sequence, but we may still want to think of $w[n]$ as the sampled version of a continuous time function $w(t)$ that should be smooth.

## 1.3 Hat (Bartlett) window

The rectangular window is not even continuous in time, and therefore decays extremely slowly in frequency ($\sim 1/\omega$). If we make the window continuous we can expect to experience a decay of order $1/\omega^2$ in frequency, which is better. The Bartlett window achieves this by convolving two rectangular windows with one another,
$$w[n] = \begin{cases} 1 - \frac{|2n-N+1|}{N+1} & \text{if } 0 \le n \le N-1 \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$
The Fourier transform of the Bartlett window is given by the product of two Dirichlet kernels,
$$W(e^{j\omega}) = 2\frac{e^{-j(N-1)\omega/2}}{N+1}\frac{\sin^2\left((N+1)\omega/4\right)}{\sin^2(\omega/2)} \tag{11}$$

## 1.4 Hann (raised cosine) window

The window $w$ can be made smoother by taking one period of a cosine function and raising it by 1,
$$w[n] = \begin{cases} \frac{1}{2}\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) & \text{if } 0 \le n \le N \\ 0 & \text{otherwise} \end{cases} \tag{12}$$
The Fourier transform of $w$ is exactly composed of three shifted Dirichlet kernels. Their relative alignments is such that the side-lobes of the main kernel is canceled by the main lobes of the two other kernels.

## 1.5 Hamming window

Yet another minor improvement over the Hann window is achieved with the Hamming window,
$$w[n] = \begin{cases} 0.541 - 0.43\cos\left(\frac{2\pi n}{N-1}\right) & \text{if } 0 \le n \le N \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$
The decay in the frequency domain is somewhat faster than the Hann window.

## 1.6 Blackman window

The Blackman window improves upon the Hann window by combining not three Dirichlet kernels, but five. The window achieves a faster decay in the Fourier domain by canceling more of the side-lobes. The window is given by

$$w[n] = \begin{cases} 0.42 - 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.008\cos\left(\frac{4\pi n}{N-1}\right) & \text{if } 0 \le n \le N \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

While one could continue to combine Dirichlet kernels in the Fourier domain to cancel more of the side-lobes, there exists a more optimal approach to window design.

The optimal window design was solved in a series of paper by Slepian in the 1960's. Unfortunately, these windows are rarely used today because there is no closed form to compute them. These windows are prolate spheroidal functions and can be computed as eigenvectors of a matrix. Every time the size of the window changes, they need to be calculated again.

Fortunately, a reasonably good approximation to these optimal windows can be computed using Bessel functions. The expression of the approximation was proposed by Jim Kaiser. Kaiser's window is the solution of the following optimization problem:

- choose $N$, the window size in time,
- minimize the width $2\omega_0$ of the main lobe of the Fourier transform of $w$,
- under the constraint that the energy in the side-lobes does not exceed a fraction $\varepsilon$ of the total energy

$$0 \le \int_{-\pi}^{-\omega_0} |H(e^{j\omega})|^2 d\omega + \int_{\omega_0}^{\pi} |H(e^{j\omega})|^2 d\omega \le \varepsilon \int_{-\pi}^{\pi} |H(e^{j\omega})|^2 d\omega \tag{15}$$

The window can be expressed in closed form as

$$w[n] = \begin{cases} \frac{I_0\left[\alpha\sqrt{1-(|2n-N+1|/(N-1))^2}\right]}{I_0[\alpha]} & \text{if } 0 \le n \le N \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

The function $I_0$ is the modified Bessel function of order 0, given by the power series,

$$I_0(x) = \sum_{k=0}^{\infty} \left(\frac{x^k}{2^k k!}\right)^2 . \tag{17}$$

The parameter $\alpha$ controls the main lobe width and the side-lobe level. Large values of $\alpha$ lead to a wider main lobe and a lower side-lobe.

## 1.7 Windowed FIR design procedure

**Windowed FIR design**

Input:

- desired amplitude response $A_d(\omega)$
- initial phase: $\phi_0 = 0$ or $\pi/2$
- filter's order $N$

the ideal frequency response is given by

$$H_d(e^{j\omega}) = A_d(\omega)e^{j(\phi_0 - N\omega/2)}$$

1. Compute the impulse response of ideal desired filter

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega \tag{18}$$

2. Mollify the truncated ideal filter, and define the filter

$$\begin{cases} h[n] = w[n]h_d[n] & \text{if } 0 \leq n \leq N \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

We can select the windows according the desired tolerances (see table 1). The order $N$ of the filter only influences the size of the transition band and not the tolerances. As we use more sophisticated windows, we have to increase the order of the filter for the same transition bandwidth.

| window | main lobe width = transition bandwidth | side-lobe level dB | $\delta_p, \delta_s$ | pass-band ripple dB | stop-band attenuation dB |
|---|---|---|---|---|---|
| rectangular | $4\pi/(N+1)$ | -13.5 | 0.09 | 0.75 | 21 |
| Bartlett | $8\pi/(N+1)$ | -27 | 0.05 | 0.45 | 26 |
| Hann | $8\pi/(N+1)$ | -32 | 0.0063 | 0.055 | 44 |
| Hamming | $8\pi/(N+1)$ | -43 | 0.0022 | 0.019 | 53 |
| Blackman | $12\pi/(N+1)$ | -57 | 0.0002 | 0.0017 | 74 |

Table 1: Windows specifications.

The parameters of the Kaiser are chosen as follows. Define

$$A = -20\log_{10}(\min(\delta_s, \delta_p)) \tag{20}$$

then we choose $\alpha$ with

$$\alpha = \begin{cases} 0.1102(A-27) & \text{if } A > 50, \\ 0.5842(A-21)^{0.4} + 0.07886(A-21) & \text{if } 21 < A \leq 50, \\ 0 & \text{if } A \leq 21, \end{cases} \tag{21}$$

and $N$ with

$$N = \frac{Ae - 7.95}{2.285|\omega_s - \omega_p|}. \tag{22}$$

## 2 Equiripple design of FIR filters

The design of FIR filters with windows is usually suboptimal: the filter only matches the specifications once, and the filter is unnecessarily long for the given specifications. The solution consists in changing the optimization criterion. Instead of minimizing the mean squared error,

$$\int_{-\pi}^{\pi} |H(e^{j\omega}) - H_d(e^{j\omega})|^2 d\omega, \tag{23}$$

we search for the filter with a fixed order $N$ that minimizes

$$\max_{\omega \in [-\pi,\pi]} |H(e^{j\omega}) - H_d(e^{j\omega})|. \tag{24}$$

Equivalently, we may want to specify the tolerance in terms of

$$\delta = \max_{\omega \in [-\pi,\pi]} |H(e^{j\omega}) - H_d(e^{j\omega})|. \tag{25}$$

and seek the filter with minimum order $N$.

This optimization criterion results in an equiripple design where the frequency response oscillates between the tolerances within each band. There is no closed form expression for the solution, and the filter is obtained using an iterative algorithm that converges toward the optimal solution. The solution is incrementally improved during each iteration.

The algorithm is known as the Parks-McClellan algorithm and relies on a computational procedure known as the Remez exchange algorithm. The design relies on an optimization technique that constructs the polynomial that uniformly approximates a given function over a given interval by minimizing the maximum error over the interval (minimax design).

We will describe the procedure in the case of the design of a lowpass filter. We further assume that the filter is generalized linear phase of type I. The frequency response of $h[n]$ can therefore be written as

$$H(e^{j\omega} = \left( \sum_{k=0}^{L} a_k (\cos\omega)^k \right) e^{-jL\omega} \tag{26}$$

where $L = N/2$ is the group delay, and $N$ is the order of the filter. The goal of the design is to find the order, $N = 2L$, and the coefficients $a_k, k = 0, \ldots, L$ of the polynomial

$$P(x) = \sum_{k=0}^{L} a_k x^k \tag{27}$$

in order to match the filter specifications.

We now write the specifications in terms of a weight functions $W(\omega)$,

$$W(\omega) = \begin{cases} \delta_p/\delta_s & \text{if } 0 \le \omega \le \omega_p \\ 1 & \text{if } \omega_p < \omega \le \pi. \end{cases} \tag{28}$$

The specifications are met if

$$\max_{\omega \in [0,\omega_p] \cup [\omega_s,\pi]} |W(\omega) \left[H(e^{j\omega}) - H_d(e^{j\omega})\right]| = \delta_s \tag{29}$$

Let $D$ be the desired lowpass frequency response. We write $D$ in terms of $\cos\omega$ instead of $\omega$,

$$D(\cos\omega) = \begin{cases} 0 & \text{if } -1 \le \cos\omega \le \cos\omega_s \\ 1 & \text{if } \cos\omega_p \le \cos\omega \le 1. \end{cases} \tag{30}$$

The optimization problem takes the form

$$\min_{a_k,L} \max_{\cos\omega \in [-1,\cos\omega_s] \cup [\cos\omega_p,1]} |W(\omega)\left[P(\cos\omega) - D(\cos\omega)\right]| \tag{31}$$

or

$$\min_{a_k,L} \max_{x \in [-1,\cos\omega_s] \cup [\cos\omega_p,1]} |W(x)\left[P(x) - D(x)\right]| \tag{32}$$

This is now a problem of approximating a function $D(x)$ using an order $L$ polynomial. There exists a necessary and sufficient condition for a polynomial $P$ to minimize the approximation error.

## 2.1 The Alternation Theorem

The solution of the optimization problem is based on the following theorem that tell us when we found the optimal polynomial.

**Theorem 2.** *The polynomial $P(x) = \sum_{k=0}^{L} a_x x^k$ is the unique $L^{th}$-order polynomial that minimizes*

$$\min_{a_k} \max_{x \in F} |W(x)\left[P(x) - D(x)\right]|, \tag{33}$$

*where $F$ is a union of intervals, if and only if there exit at least $L+2$ points $x_k \in F$ where the error*

$$E(x) = W(x)(P(x) - D(x)) \tag{34}$$

*exhibits $L+2$ alternations,*

$$E(x_i) = -E(x_{i+1}) = \cdots = \pm \min_{a_k,L} \max_{x \in F} |W(x)\left[P(x) - D(x)\right]|. \tag{35}$$

We can apply the theorem to the filter design problem, and we obtain the following necessary and sufficient condition:

**Theorem 3.** *The polynomial $P(x) = \sum_{k=0}^{L} a_x x^k$ is the unique $L^{th}$-order polynomial that minimizes*

$$\min_{a_k,L} \max_{\omega \in [0,\omega_p] \cup [\omega_s,\pi]} |W(\omega) \left[H(e^{j\omega}) - H_d(e^{j\omega})\right]| \tag{36}$$

*if and only if there exit at least $L+2$ frequencies $\omega_k$ where the error*

$$E(\omega) = |W(\omega) \left[H(e^{j\omega}) - H_d(e^{j\omega})\right]| \tag{37}$$

*exhibits $L+2$ alternations*

$$E(\omega_i) = -E(\omega_{i+1}) = \cdots = \pm \min_{a_k,L} W(\omega) \left[H(e^{j\omega_i}) - H_d(e^{j\omega_i})\right] \tag{38}$$

We note the theorem doesn't yield a closed-form solution for the optimal polynomial of degree $L$. However, we use the theorem to search for the optimal polynomial by trying to create $L+2$ alternations.

We note that the theorem does not tell us how to choose the right order $L$ to match the specifications, so that

$$\min_{a_k, L} \quad \max_{x \in F} |W(x)\,[P(x) - D(x)]\,| = \delta_s \tag{39}$$

Typically the order $L$ can be estimated using the following formula:

$$2L = \frac{-20\log_{10}\sqrt{\delta_p \delta_s} - 13}{2.32|\omega_p - \omega_s|}. \tag{40}$$

We are now left with the problem of iteratively computing as set of coefficients $a_k$, some frequencies $\omega_i$ and an error $\varepsilon$, so that

$$E(\omega_i) = \pm\varepsilon. \tag{41}$$

We observe that if the frequencies $\omega_i$ are known then we can solve for the $L+1$ coefficients $a_k, k = 0, \cdots, L$ and the error $\varepsilon$ using the system of $L+2$ equations,

$$a_o + a_1\cos(\omega_i) + a_2\cos^2(\omega_i) + \cdots + a_L\cos^L(\omega_i) + (-1)^i\varepsilon = D(\omega_i), \qquad i = 0, \ldots, L+1 \tag{42}$$

Evgeny Yakovlevich Remez developed in 1934 a computational algorithm to compute the optimal polynomial $P$. The algorithm is known as the Parks-McClellan algorithm in the signal processing literature.

## 2.2 Remez exchange algorithm

1. Choose $L$ according to the empirical formula (40).
2. Choose a set of initial frequencies $\omega_i, i = 0, \cdots, L+1$.
3. Solve the linear system (42) to compute $\varepsilon$ and the $a_k$.
4. Find the frequencies $\theta_i$ where the error

$$E(\omega) = W(\omega)\left[H(e^{j\omega_i}) - H_d(e^{j\omega_i})\right] \tag{43}$$

   is extremal. If there are more than $L+2$ extrema, keep these with the largest error.

5. Replace each $\omega_i$ with the closest $\theta_i$.
6. If all $|\theta_i\omega_i|$ are smaller than some specified precision, then continue; else go to 3. then go to
7. Solve the linear system (42) to compute $\varepsilon$ and the $a_k$.
8. If $\varepsilon < \delta_s$ then stop; else increase $L$ and go to 2.

## 2.3 Implementation in MATLAB

The function `firpm` implements the Remez exchange algorithm.

## 3 Examples

The frequency of an ideal bandstop filter is given by

$$H(e^{j\omega}) = \begin{cases} 1 & \text{if } 0 \leq |\omega| < \pi/3 \\ 0 & \text{if } \pi/3 \leq |\omega| < 2\pi/3 \\ 1 & \text{if } 2\pi/3 \leq |\omega| \leq \pi \end{cases} \tag{44}$$

1. Design an FIR filter with stopband attenuation of 60dB and a transition band of $\pi/6$ using a Kaiser window, and an equiripple design.

### 3.1 Solution

```
%
% filter specifications
As = 60;

% cutoff frequencices

wc1 = pi/3;
wc2 = 2*pi/3;

% transition bandwidth

dw = pi/6;

% Kaiser parameter

beta = 0.1102*(As-8.7);
N = ceil((As - 8)/(2.285*dw));
N

%____________________________________________
%
%  Design using a Kaiser window
%
%____________________________________________

w_kai = (kaiser(N+1,beta))';

% band pass = subtraction of low pass

hd = ideal_low(wc1,N) + ideal_low(pi,N) - ideal_low(wc2,N);

% filter design = windowing

h = hd .* w_kai;
```

```
% evaluate the filter performance

[db,mag,pha,grd,w] = myfreqz (h,[1]);

%____________________________________________
%
%   draw some pretty plots
%
%____________________________________________

% time

n=[0:1:N];

figure;

subplot(1,1,1);

subplot(2,1,2);
plot(w/pi,db);

title('Magnitude Response in dB');grid;
xlabel('frequency in pi units'); ylabel('Decibels')

subplot(2,2,1);
stem(n,hd); title('Ideal Impulse Response')
axis([-1 N -0.2 0.8]); xlabel('n'); ylabel('hd(n)')

subplot(2,2,2);
stem(n,w_kai);title('Kaiser Window')
axis([-1 N 0 1.1]); xlabel('n'); ylabel('w(n)')

subplot(2,2,3);
stem(n,h);title('Actual Impulse Response')
axis([-1 N -0.2 0.8]); xlabel('n'); ylabel('h(n)')

subplot(2,2,4);
plot(w/pi,db);
title('Magnitude Response in dB');grid;
xlabel('frequency in pi units'); ylabel('Decibels')


%____________________________________________
%
%   equiripple design
%
```

```
%____________________________________________

f = [0 0.30 0.67 1];
a = [1   0   1];
d = [1e-3  1e-3 1e-3];

[N,f0,a0,w] = firpmord(f,a,d);
N
be = firpm (N,f0,a0,w);
[db,mag,pha,grd,w] = myfreqz (be,[1]);

%____________________________________________
%
%   draw some pretty plots
%
%____________________________________________

% time

n=[0:1:N];

figure;
subplot(2,1,1);

subplot(2,1,1);
stem(n,be);title('Actual Impulse Response')
axis([-1 N -0.2 0.8]); xlabel('n'); ylabel('h(n)')

subplot(2,1,2);
plot(w/pi,db);
title('Magnitude Response in dB');grid;
xlabel('frequency in pi units'); ylabel('Decibels')
```

### 3.2 Function myfreqz.m

```
function [db,mag,pha,grd,w] = myfreqz(b,a);

% INPUT:
%
%   b = numerator polynomial of H(z)   (for FIR: b=h)
%   a = denominator polynomial of H(z) (for FIR: a=[1])
%
% OUTPUT:
%
%  db = Relative magnitude of the frequency response
%          in dB computed over 0 to pi radians
```

```
%
% mag = absolute magnitude computed over 0 to pi radians
%
% pha = Phase response in radians over 0 to pi radians
%
% grd = Group delay over 0 to pi radians
%
%   w = 501 frequency samples between 0 to pi radians
%

[H,w] = freqz(b,a,1000,'whole');

H = (H(1:1:501))';
w = (w(1:1:501))';

mag = abs(H);

db = 20*log10((mag+eps)/max(mag));

pha = angle(H);

grd = grpdelay(b,a,w);
```

### 3.3 Function ideal_low.m

```
function hd = ideal_low(wc,N);

% ideal lowpass filter = sinc ( wc*n)/(pi *n)
%
%
% [hd] = ideal_lp(wc,M)
%
% INPUT:
%              N:  order of the filter
%             wc: cutoff frequency in radians
%
% OUTPUT:
%              hd: ideal impulse response sampled at n = 0.. N
%

alpha = N/2;

n = [0:1:N];

m = n - alpha;
```

```
hd = zeros (1,N);

notz     = find (m ~=0);
m_notz = m(notz);
m0       = find (m ==0);

%
% sinc function

hd(notz) = sin(wc*m_notz) ./ (pi*m_notz);

%
%  fix the value at 0

hd (m0) = wc/pi;
```