Digital Filters Design using Matlab with Graphical User Interface (GUI)

Adnan Affandi[1], Abdullah M. Dobaie[2] and Mubashshir Husain[3]

Professor, Dept., of Elect.& Comp. Eng., Faculty of Eng. King Abdul Aziz University Jeddah, KSA[1]
Assistant Professor, Dept., of Elect.& Comp. Eng., Faculty of Eng. King Abdul Aziz University Jeddah, KSA[2]
Lecturer, Dept., of Elect.& Comp. Eng., Faculty of Eng. King Abdul Aziz University Jeddah, KSA[3]
mubashshir.husain.delhi@gmail.com

Abstract: Digital filtering occupies an extremely important position in the digital signal processing.This paper introduces the new concept of using Matlab with Graphical User Interface in designing FIR(Finite Impulse Response) digital filters and IIR(Infinite Impulse Response) digital filters. Matlab, which is a high-performance numerical calculation program and provides a powerful function of graphical display. Matlab is widely used in engineering calculation, numerical analysis,etc. This paper introduces the definition and basic principles of FIR & IIR digital filters. In this paper we have designed Graphical User Interface consists of almost all types of IIR filters and FIR filters. User simply have to insert filter specifications on GUI and get magnitude response, phase response, etc of required the filter.
[Adnan Affandi, Abdullah M. Dobaie and Mubashshir Husain. **Digital Filters Design using Matlab with Graphical User Interface (GUI).** *Life Sci J* 2014;11(5):336-348]. (ISSN:1097-8135). http://www.lifesciencesite.com. 45

Keywords: FIR digital filter,IIR digital filters Matlab,Graphical User Interface,etc.

1. Introduction

A fundamental aspect of signal processing is filtering. Filtering involves the manipulation of the spectrum of a signal by passing or blocking certain part of the spectrum, depending on the frequency of those parts. Filters are designed according to what kind of manipulation of the signal is required for a particular application. Digital filters are implemented using three fundamental building blocks: an adder, a multiplier, and a delay element and they represent the Capacitor, Inductors and Resistance in the analog filtering.

With these basic building blocks, the two different filter structures can easily be implemented. These two structures are Infinite Impulse Response (IIR) and Finite Impulse Response (FIR), depending on the form of the system's response to a unit pulse input. IIR filters are commonly implemented using a feedback (recursive) structure, while FIR filters usually require no feedback (non-recursive). The design process of a digital filter is long and some way is a kind of routine if done by hand. With the aid of computer programs performing filter design algorithms, designing and optimizing filters can be done relatively quickly.

A filter with linear phase response is desirable in many applications such as image processing and data transmission. One of the desirable characteristics of FIR filters is that they can be designed very easily to have linear phase.[1]

2. Designing of fir digital filters

A. Designing an fir (finite impulse response) filters

FIR filters are one of two primary types of digital filters used in Digital Signal Processing (DSP) applications, the impulse response is "finite" because there is no feedback in the filter as in the second type of filters (It will explained in the IIR filters part).A useful designing model for the design specifications in FIR design is to think of each specification as one of the angles in a triangle as shown fig.1.

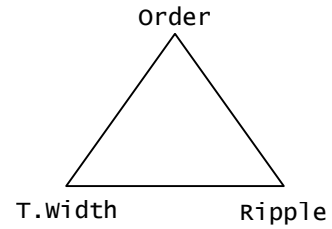

Fig.1 FIR triangle model

The model in Fig 2.2 is used to understand the degrees of freedom available when considering a filter specification. Because the sum of the angles is fixed, we can at most select the values of two of the specifications. The third specification will be determined by the design algorithm utilized. Moreover, as with the angles in a triangle, if we make one of the specifications larger/smaller, it will impact one or both of the other specifications.

B. Optimal fir designs with fixed transition width and filter order

Truncated-and-windowed impulse response de-sign algorithm is very simple and reliable; it is not optimal in any sense. The designs are generally

inferior to one of the order or the transition width or the passband/stopband ripples, the exceeded value of any of them is typically undesirable in the Optimal designs are computed by minimizing some measure of the deviation between the filter to be designed and the ideal filter. The most common optimal FIR design algorithms are based on fixing the transition width and the order of the filter. The deviation from the ideal response is measured only by the passband/stopband ripples. This deviation or error can be expressed mathematically as

$$E(\omega)=H_{\alpha}(\omega)-H_{LP}(e^{j\omega}) \qquad \omega\in\Omega \tag{2.1}$$

Where $H_a(\omega)$ is the zero-phase response of the designed filter and $\omega=[0,\omega_{pass}][\omega_{stop},1]$. It is still necessary to define a measure to determine "the size" of $E(\omega)$ (the quantity we want to minimize as a result of the optimization)

The most often used measures are the L-norm (L∞ or L2) .In order to allow for different peak ripples in the passband and stopband, a weighting function W (w) is usually introduced

$$E_W(\omega)=W(\omega)\left[H_{\alpha}(\omega)-H_{LP}(e^{j\omega})\right] \qquad \omega\in\Omega \tag{2.2}$$

The most famous two filter kinds in this field are the Equiripple and the Least Square Filter and they will described later

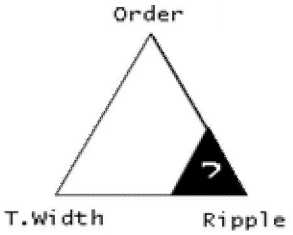

Fig.2 FIR triangle model for fixed transition width and filter order

C. Optimal fir designs with fixed transition width and peak passband/ stopband ripple

Fixed Transition width and passband/stopband ripple allow us to reach an optimum filter with a minimum number of tabs (order). The equations are even more dramatic when the passband ripple and stopband ripple specifications are different (unlike the equiripple filters) . The reason is that the truncated-and windowed impulse response methods always give a result with approximately the same passband and stopband peak ripple. Therefore, always the stricter peak ripple will cause in exceeding (possibly significantly) all other ripple constraints at the expense of unnecessarily large filter order. To illustrate this, we turn to a different equiripple design in which both the peak ripples and the transition width are fixed. In minimum-phase designs with fixed transition width and peak passband/stopband ripple the same procedure can be used to design minimum-phase filters with fixed transition width and peak passband/ stopband ripple. In this case, rather than obtaining smaller ripples, the benefit is meeting the same transition width and peak passband/ stopband ripples with a reduced filter order.

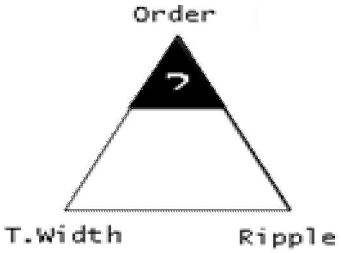

Fig3. FIR triangle model for fixed transition width and peak passband/stopband ripple

D. Optimal fir designs with fixed peak ripple and filter order

Fixing the filter order and the peak ripple values should result in a smaller transition width. In minimum-phase designs with fixed peak ripple and filter order, once again, if linear-phase is not a requirement, a minimum-phase filter can be designed that is better in some sense to a comparable linear phase filter. In this case, for the same filter order and peak ripple value, a minimum-phase design results in a smaller transition width than a linear-phase design.

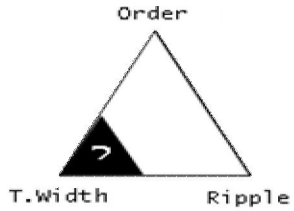

Fig.4 FIR triangle model for fixed peak ripple and filter order

E. Designing optimal fir equiripple filters with fixed transition width and filter order by using graphical user interface

- This linear phase filter can be designed with the function firpm or in minimax concept by firgr.

B=firpm (N, fvector, mvector) or
B=firpm (N, fvector, mvector, wvector) or
B=firgr (N, fvector, mvector) or
B=firgr (N, fvector, mvector, wvector)

- And for Hilbert Transform that have odd symmetry

B=firpm (N, fvector, mvector, 'Hilbert') or

B=firpm (N, fvector, mvector, wvector, 'Hilbert') or

B=firgr (N, fvector, mvector, 'Hilbert') or

B=firgr (N, fvector, mvector, wvector, 'Hilbert')

- And for the Differentiator with odd symmetry

B=firpm (N, fvector, mvector, 'differentiator') or

B=firpm (N, fvector, mvector, wvector, 'differentiator') or

B=firgr (N, fvector, mvector, 'differentiator') or

B=firgr (N, fvector, mvector, wvector, 'differentiator')

Where

N is the filter Order (returns a length N+1 tabs)

fvector is the best approximation to the desired frequency response

mvector is the filter magnitude vector in the least -Pth sense.

wvector is the weight error vector [3]

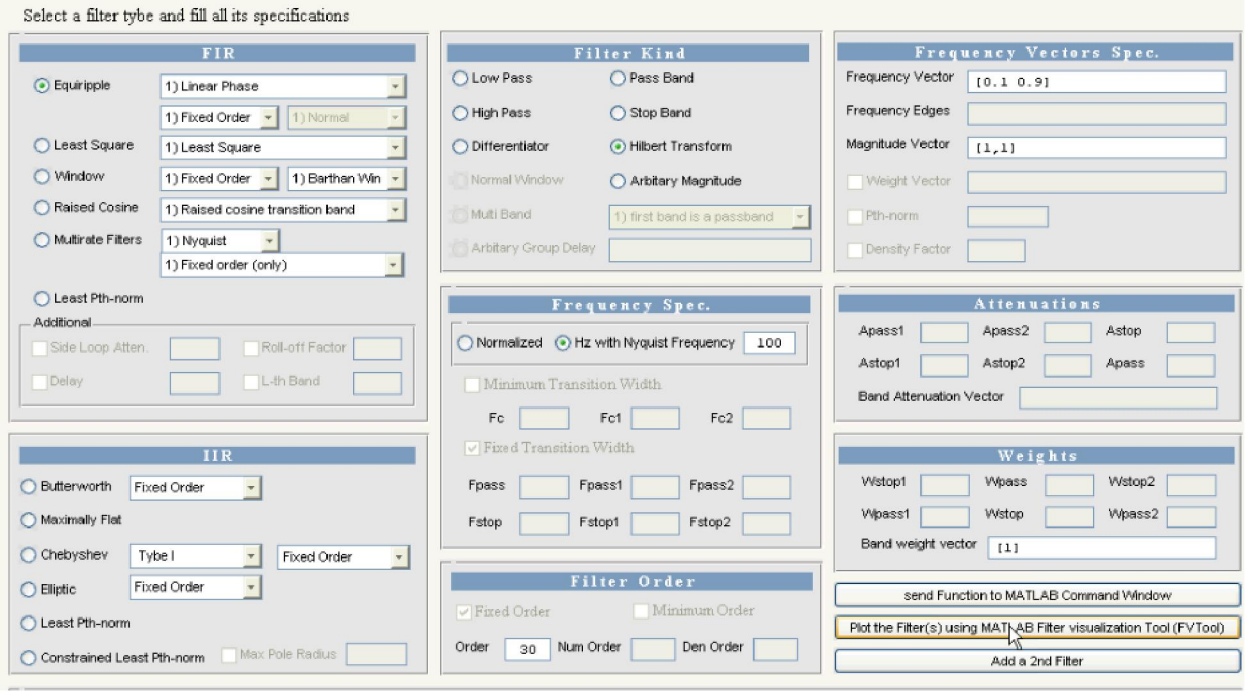

Fig.5 Using Graphical User Interface for designing Hilbert bandpass filter (Fpass=0.1, Fstop=0.9, Order=30).

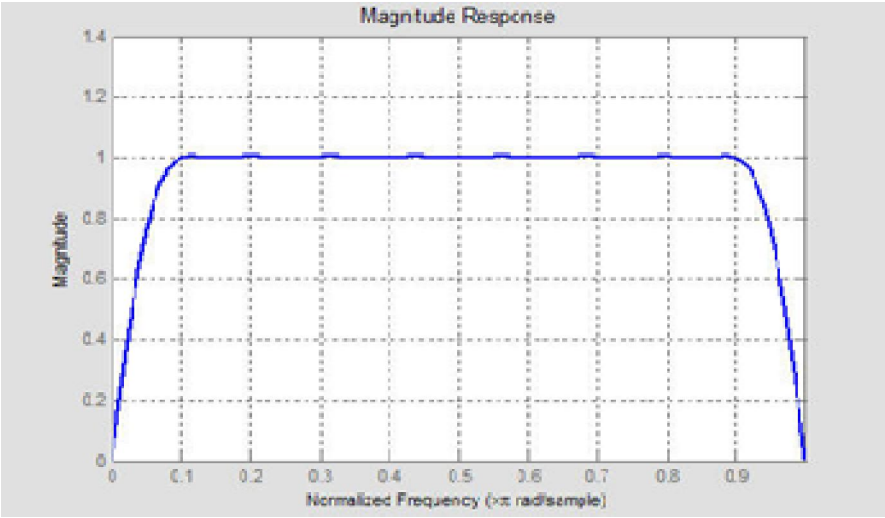

Fig.6 Showing Magnitude Response of Hilbert bandpass filter (Fpass=0.1, Fstop=0.9 & Order=30).

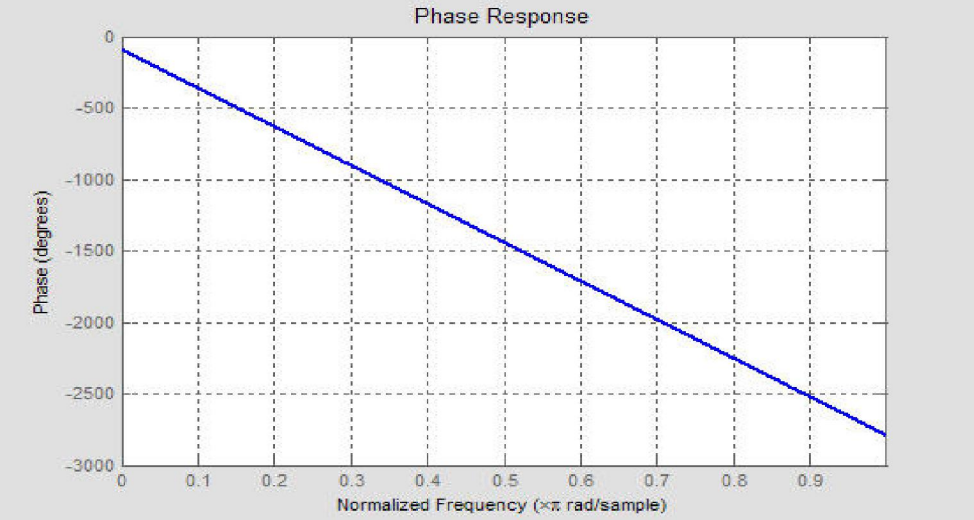

Fig.7 Phase response of a bandpass filter

F. Fir least-squares and fir constrained least-squares filters

Equiripple designs may not be desirable if we want to minimize the energy of the error (between ideal and actual filter) in the passband /stopband. Consequently, if we want to reduce the energy of a signal as much as possible in a certain frequency band, least-squares designs are preferable.

G. Designing Optimal Fir Least Square Filters With Fixed Transition Width And Filter Order

This filter can be designed with the function firls as follows

B=firls (N, fvector, mvector) or

B=firls (N, fvector, mvector, bwvector)

And for Hilbert Transform that have odd symmetry

B= firls (N, fvector, mvector, 'Hilbert') or

B= firls (N, fvector, mvector, bwvector, 'Hilbert')

And for the Differentiator with odd symmetry

B= firls (N, fvector, mvector, 'differentiator') or

B= firls (N, fvector, mvector, bwvector, 'differentiator')

Where

N is the filter Order (returns a length N+1 tabs)

fvector is the best approximation to the desired frequency response

mvector is the filter magnitude vector in the least -Pth sense.

bwvector is the weight per band vector [3].

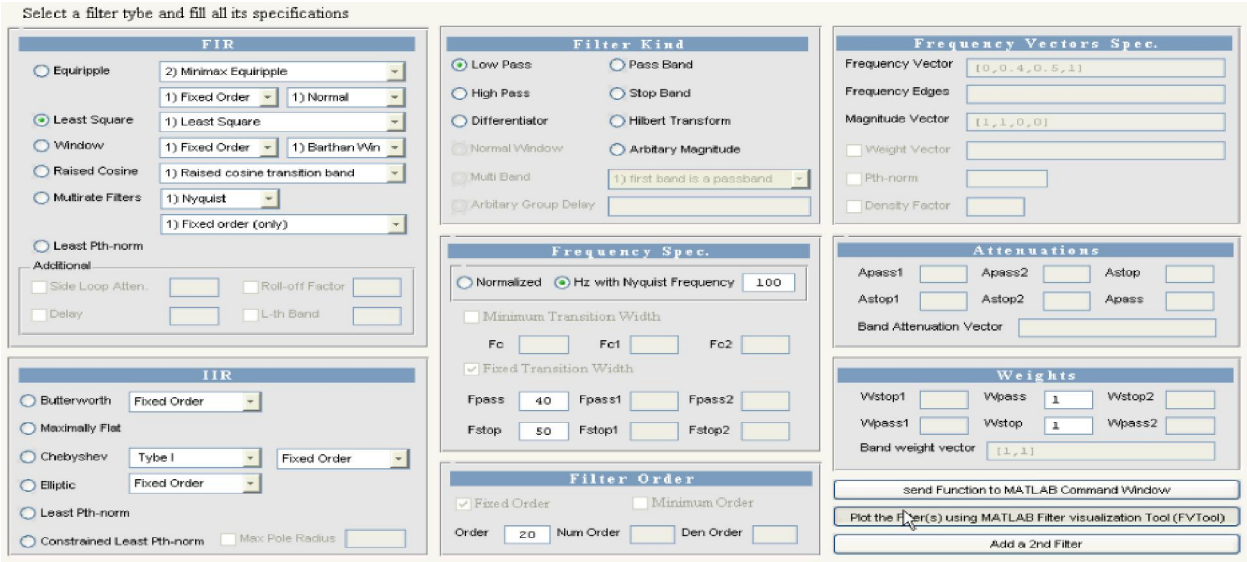

Fig.8 Designing A lowpass Least Square filter with order=20 and band edges at $f_1 = 0.4$ and $f_2 = 0.5$ (normalized) by using GUI.

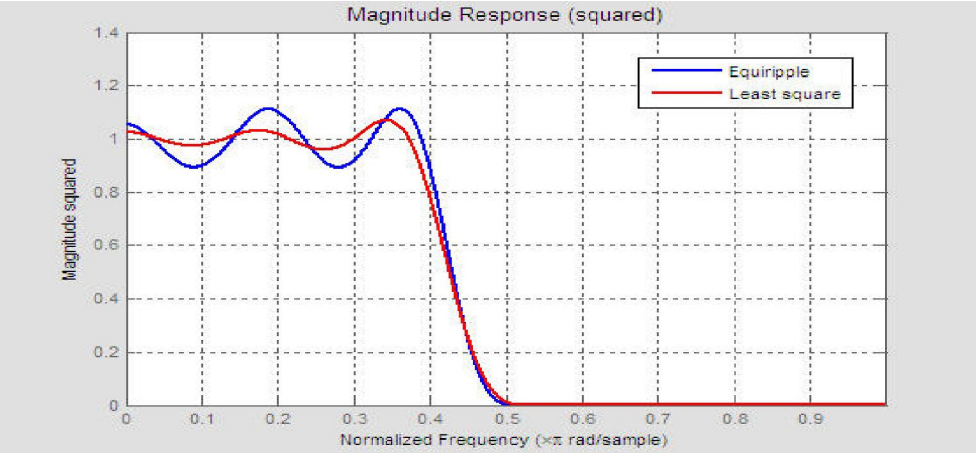

Fig.9 A lowpass Least Square filter with order=20 and band edges at $f_1 = 0.4$ and $f_2 = 0.5$ (normalized), compared with an Equiripple filter with the same specifications

An equiripple filter designed with firpm exhibits equiripple behavior. And a least square filter designed with firls filter has a better response over most of the passband and stopband, but at the band edges (f = 0.4 and f = 0.5), the response is further away from the ideal than the firpm filter. This shows that the firpm filter's maximum error over the passband and stopband is smaller and, in fact, it is the smallest possible for this band edge configuration and filter length.

H. Fir windowing

I. Kaiser window design technique

May be it is quite important to describe at least one of the window kind and we will take the Kaiser window as an example. The main problem with the window design method is that it is very difficult to trade-off between attenuation and transition bandwidth. Kaiser developed a window function and a design formula that will usually result in a filter length less than those designed by using other window methods.Given a lowpass filter, the passband region is from 0 to ω_p and the stopband region from ω_s to π as described in the following fig

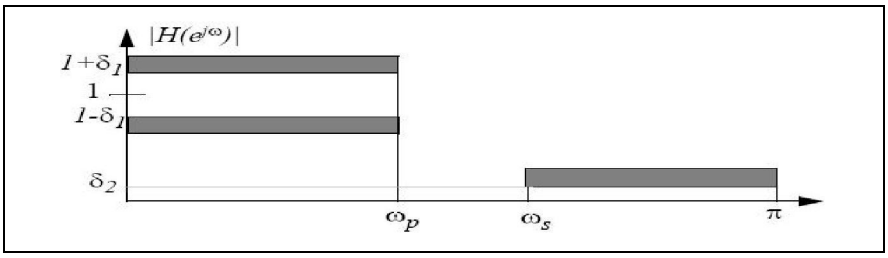

Fig.10 the windowing lowpass filter components

$$\Delta\omega = \omega_S - \omega_P \tag{3.6}$$

$$A = -20\log_2 \delta \tag{3.7}$$

$$K(n) = \frac{I_O\left(\beta\sqrt{1-\left(1-\frac{2n}{N}\right)^2}\right)}{I_O(\beta)}, for \quad 0 \le n \le N \tag{3.8}$$

Where N controls the transition bandwidth and β controls the sidelobe attenuation.

The resultant formula by Kaiser:

$$N = \frac{A-8}{2.285\Delta\omega} \tag{3.9}$$

$$\beta = \begin{cases} 0.1102(A-8.7) & A > 50 \\ 0.5842(A-21)^{0.4} + 0.07886(A-21) & 21 \le A \le 50 \\ 0 & A \le 21 \end{cases} \tag{3.10}$$

These formulas can be used to estimate the values of N and β. To design a filter of minimal filter length while

satisfying a given set of specification may require a few iterations to fine-tune the values of N.

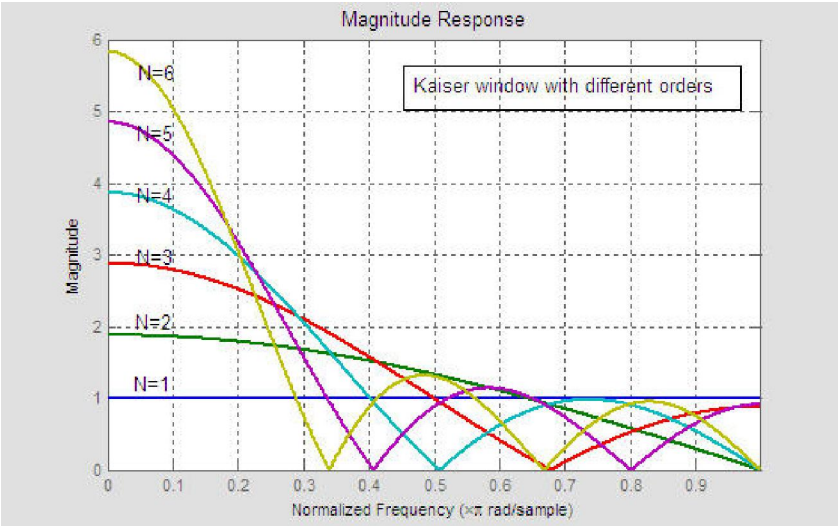

Fig.11 Kaiser window for different orders (order 1 to 6) .The transition bandwidth decreases with the increasing of the order.

J. Matlab and fir raised cosine

The function firrcos can be used to design a raised cosine FIR filter and a square root FIR raised cosine in MATLAB

B=firrcos (N,fc,TW,Fs) or
B=firrcos (N,fc,TW,Fs,'sqrt')
With a rolloff factor:
B=firrcos (N,fc,Fs,'rolloff') or
B=firrcos (N,fc,Fs,'rolloff','sqrt')
With a delay
B=firrcos (N,fc,TW,Fs,'normal',delay) or
B=firrcos (N,fc,TW,Fs,'sqrt',delay)
With a rolloff factor and delay
B=firrcos (N,fc,Fs,'rolloff','normal',delay) or
B=firrcos (N,fc,Fs,'rolloff','sqrt',delay)
Where:
N is the filter Order (returns a length N+1 tabs)
fc is the passband edge frequency
TW transition bandwidth (Fstop – Fpass)
Fs is the sampling frequency (= 2 Nyquist Frequency)
delay is a variable integer delay[3]

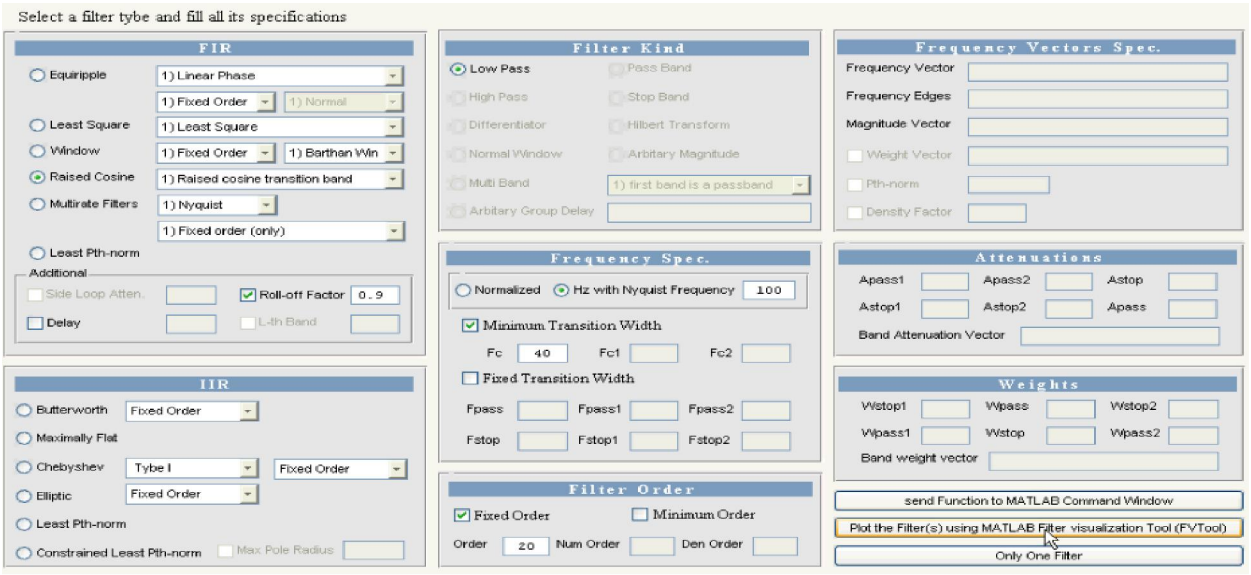

Fig.12 A Raised Cosine filter (order 20) with a cutoff frequency off 0.5 (normalized) and a roll-off factor R=0.1, R=0.3, R=0.5, R=0.7, and R=0.9 by using GUI.

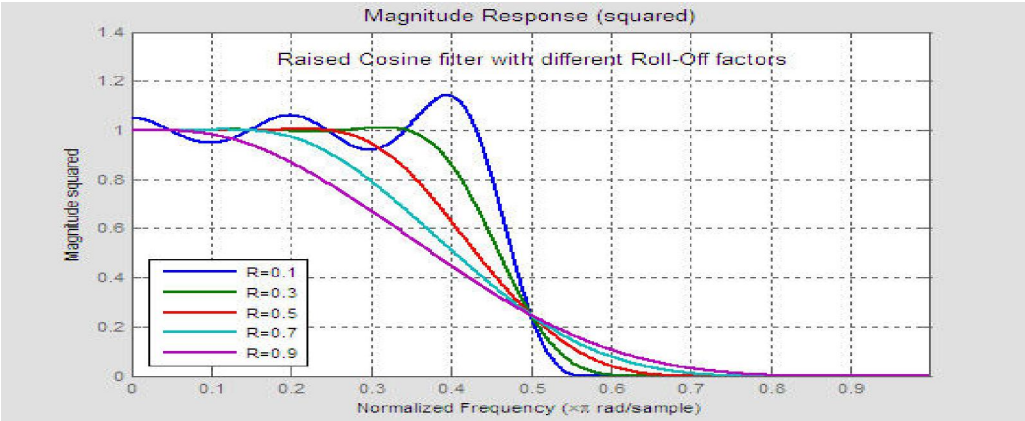

Fig.13 Magnitude Response

Different roll-off factors (R=0.1, R=0.3, R=0.5, R=0.7, R=0.9) with the same order (order=20) and cutoff frequency (fc = 0.5).The figure shows the squared magnitude and the different unity gain regions and the attenuation regions and the raised cosine regions and it easily to figure out the relation between them and the rolloff factor

3- Infinite Impulse Response(IIR) Digital Filter

A. Designing an IIR (Infinite Impulse Response) filters

IIR (Infinite Impulse Response) or Recursive filters are signal processing filters which re-use one or more output(s) of the filter as inputs. This feedback results in an unending impulse response characterized by exponentially growing, decaying, or sinusoidal signal output components.

In digital IIR filters, the output feedback is immediately apparent in the equations defining the output. Note that unlike with FIR filters, in designing IIR filters it is necessary to carefully consider "time zero" case in which the outputs of the filter have not yet been clearly defined.

To start a theoretical IIR we start with the difference equation which defines how the input signal is related to the output signal

$$y(n)=b_0x(n)+b_1x(n-1)+\cdots+b_Px(n-P)+a_1y(n-1)+a_2y(n-2)+\cdots+a_Qy(n-Q) \quad (2.3)$$

where P is the forward filter order, b_i are the forward filter coefficients, Q is the feedback filter order, a_i are the feedback filter coefficients, x(n) is the input signal and y(n) is the output signal. A more condense form of the difference equation is

$$y(n)=\sum_{i=0}^{P} b_i x(n-i)+\sum_{k=1}^{Q} \alpha_k y(n-k) \quad (2.4)$$

To find the impulse response we set

$$x(n)=\delta(n) \quad (2.5)$$

Where $\delta(n)$ is the delta impulse. The impulse response for an IIR filter follows as

$$h(n)=\sum_{i=0}^{P} b_i \delta(n-i)+\sum_{k=1}^{Q} \alpha_k h(n-k) \quad (2.6)$$

The Z-transform of the impulse response yields the transfer function of the IIR filter

$$H(z)=Z\{h(n)\}=\sum_{n=-\infty}^{\infty} h(n)z^{-n} \quad (2.7)$$

We note that Z {δ (n)} = 1 then with the definition of the impulse response and the time shift property of the Z-transform follows

$$H(z)=\sum_{i=0}^{P} b_i z^{-i}+\sum_{k=1}^{Q} \alpha_k z^{-k} H(Z) \quad (2.8)$$

Isolating H(z) on the left hand side leads to the desired format of the transfer function

$$H(z)=\frac{\sum_{i=0}^{P} b_i z^{-i}}{1-\sum_{k=1}^{Q} \alpha_k z^{-k}} \quad (2.9)$$

The transfer function allows us to judge whether or not a system is bounded-input, bounded-output (BIBO) stable. To be specific, the BIBO stability criteria require all poles of the transfer function to have an absolute value smaller than one. In other words, all poles must be located within a unit circle in the z-plane. To find the poles of the transfer function we have to extend it with $\frac{z^O}{z^O}$ (or mathematically multiply by $\frac{z^O}{z^O}$)

Where O = max (P, Q) and arrive at

$$H(z) = \frac{\sum_{i=0}^{P} b_i z^{O-i}}{z^{O} - \sum_{k=1}^{Q} \alpha_k z^{O-k}} \quad (2.10)$$

The poles of the IIR filter transfer function are the zeros of the denominator polynomial of the transfer function. The poles are evaluated as

$$z^{O} - \sum_{k=1}^{Q} \alpha_k z^{O-k} = 0 \quad (2.11)$$

Clearly, if $\alpha_k \neq 0$ then the poles are not located on the origin of the z-plane. This is in contrast to the FIR filter where all poles are located on the origin of z-plane.

The primary advantage of IIR filters over FIR filters is that they typically meet a given set of specifications with a much lower filter order than a corresponding FIR filter. Although IIR filters have nonlinear phase.

Data processing within MATLAB is commonly performed off-line, That is, the entire data sequence is available before filtering. This allows for a noncausal, zero-phase filtering approach, which eliminates the nonlinear phase distortion of an IIR filter.

Design of digital IIR filters is heavily dependants on that of their analog counterparts which is because they are well studied, and have rich resources, while that, MATLAB toolbox provide some new function which are designed directly at the Z-domain and they have special features.[2]

B. IIR butterworth filter

The Butterworth filter is the most known theoretical IIR filter .it is designed to have a frequency response which is as flat as mathematically possible in the passband.

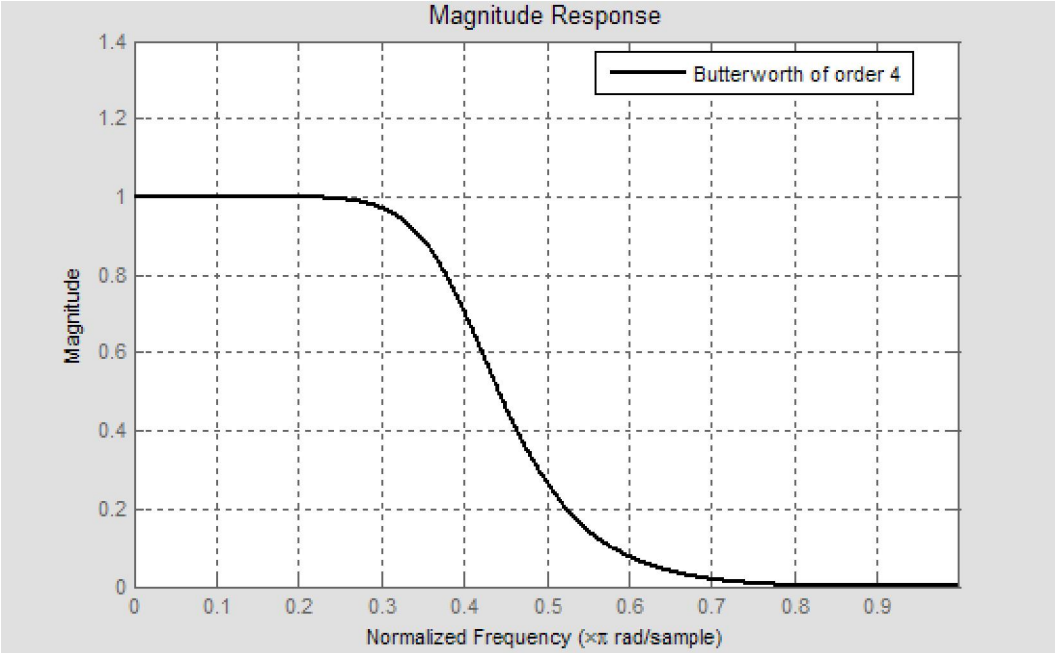

Figure.14 Butterworth of order 4, clearly it has a flat passband (no ripples) and rolls off towards zero in the stopband

C. Butterworth characteristic and response

The frequency response of the Butterworth filter is maximally flat (has no ripples) in the passband, and rolls off towards zero in the stopband. When viewed on a logarithmic scale the response linearly slopes towards negative infinity. The Butterworth is the only filter that maintains this same shape for higher orders (but with a sharper slope in the stopband).Compared with a Chebyshev Type I/Type II filter or an elliptic filter, the Butterworth filter has a slower roll-off, and thus will require a higher order to implement a particular stopband specification. However, Butterworth filter will have a more linear phase response in the passband than the Chebyshev Type I or II and elliptic filters.

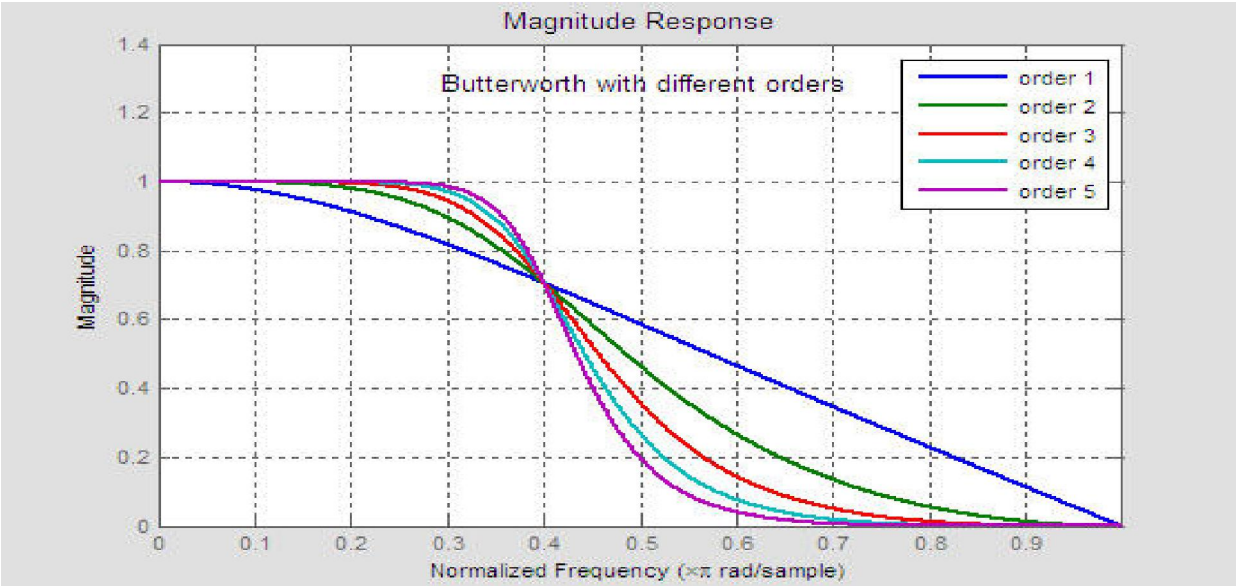

Figure.15 Lowpass Butterworth with different orders (order 1 to 5), as the order increased the transition width slowly reduced, and thus it require a high order to meet a specific requirement,

D. Matlab and maxflat

MATLAB toolbox provide a generalized low pass butterworth function as follows

[B,A] = maxflat (num,den,fc)

Where

num is the numerator order

den is the denominator Order

fc is the cut-off frequency at which the filter's magnitude response is equal to $1/\sqrt{2}$

Note : butter(N,Wn) = maxflat(N,N,Wn) except in the zeros and poles

(example for N=20)

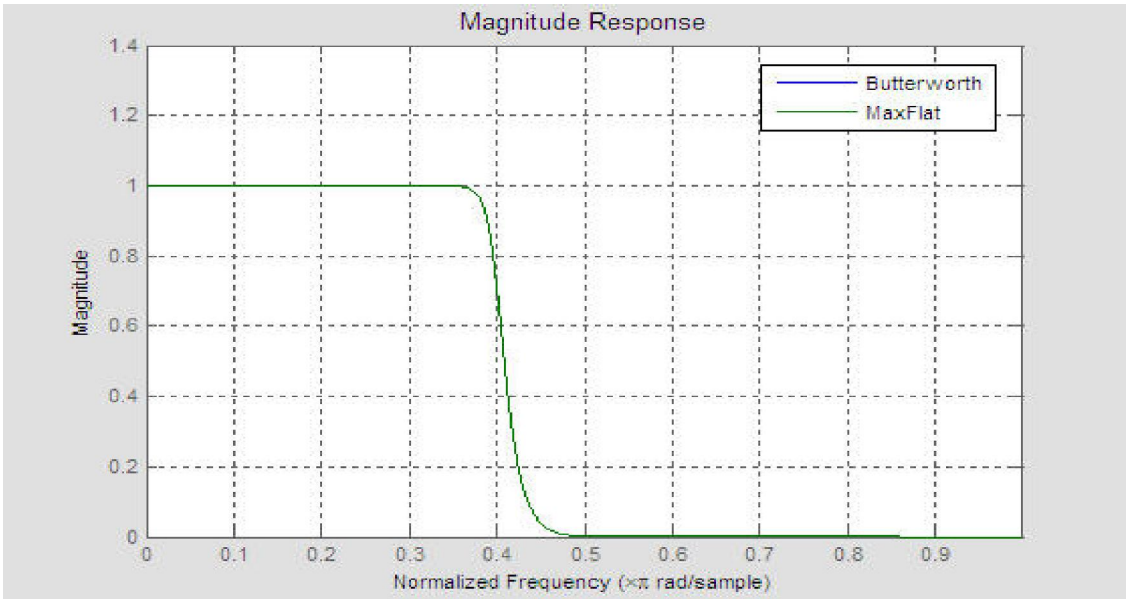

Figure.16 Magnitude response for a Butterworth filter and a maxflat filter, the two filter gives the same response shape

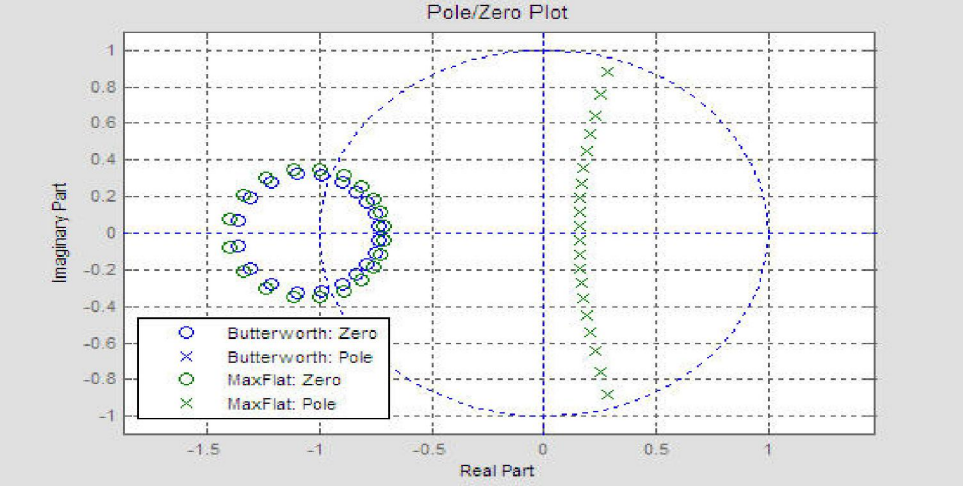

Figure.17 Pole/Zero response for a Butterworth filter and a maxflat filter,the two filter gives the same response shape but with some difference in the pole/zero plane

E. Matlab and chebyshev

MATLAB provide two Chebyshev functions cheby1 and cheby2 and they contains all design needs.

For Lowpass Chebyshev:

[B,A] = cheby1(N,Apass,fc)

[B,A] = cheby2(N,Astop,fc)

For Highpass Chebyshev:

[B,A] = cheby1(N,Apass,fc,'high')

[B,A] = cheby2(N,Astop,fc,'high')

For passband Chebyshev

[B,A] = cheby1(N,Apass,[fc1,fc2])

[B,A] = cheby2(N,Astop,[fc1,fc2])

For stopband Chebyshev

[B,A] = cheby1(N,Apass,[fc1,fc2],'stop')

[B,A] = cheby2(N,Astop,[fc1,fc2],'stop')

Where

N is the filter order

fc is the cutoff frequency (0 < fc < 1)

fc1 and fc2 is the bandpass and stopband frequencies for the Bandpass or stopband filters

Apass is the passband attenuation (dB)

Astop is the stopband attenuation (dB)[3]

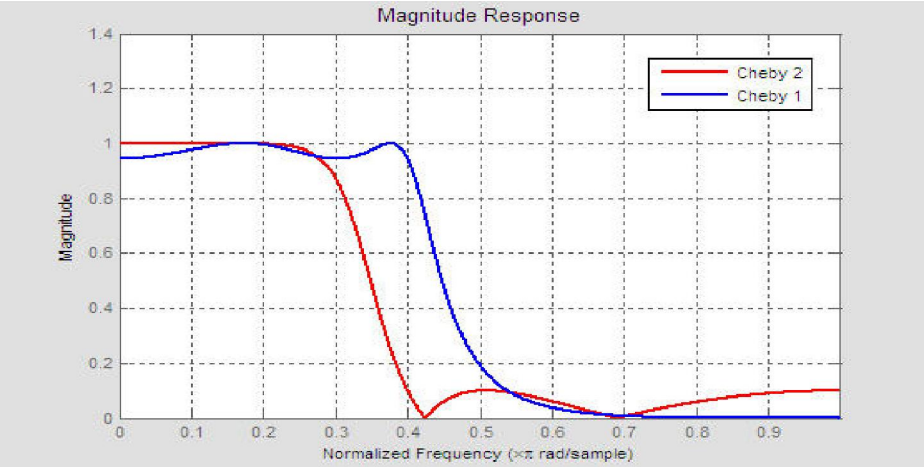

Fig18. Magnitude Response

Chebyshev type I and Chebyshev type II with the same order (order 4) and same cutoff frequency (fc=0.4 normalized) the first one have some ripples in the passband but also smooth at the stopband,the second filter is the opposite, and it is clear that Chebyshev type II has a slower transition off than the Chebyshev type I.

F. MATLAB AND IIR ELLIPTIC FILTER

- Lowpass Elliptical filter:

[B,A] = ellip(N,Apass,Astop,fc)

- For Highpass Elliptical filter:

[B,A] = ellip(N,Apass,Astop,fc,'high')

- For passband Elliptical filter:

[B,A] = ellip(N,Apass,Astop,[fc1,fc2])

- For stopband Elliptical filter:

[B,A] = ellip(N,Apass,Astop,[fc1,fc2],'stop')

Where

N is the filter order

fc is the cutoff frequency (0 < fc < 1)

fc1 and fc2 is the bandpass and stopband frequencies for the Bandpass or Stopband filters

Apass is the passband attenuation (dB)

Astop is the stopband attenuation (dB) [3]

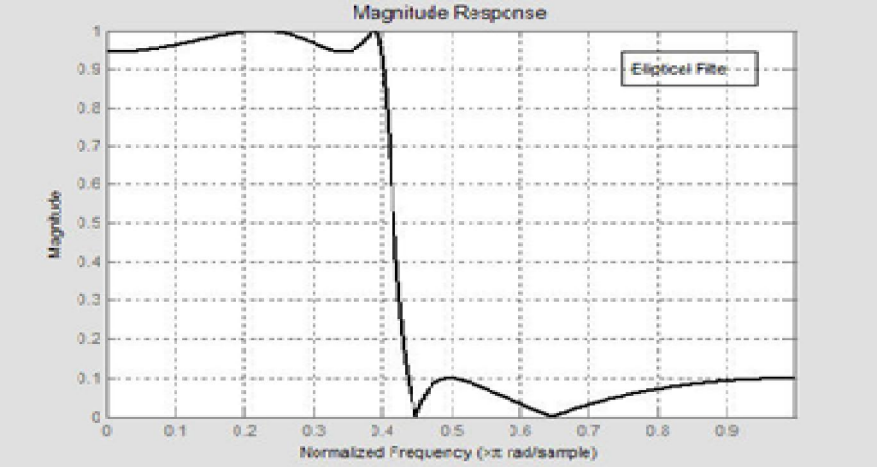

Fig.19 An elliptical filter of order 4,it has a sharp transition bandwidth but with some ripples in the passband and stopband

G. Comparison with other linear filters

The next figure contains the last four iir filters magnitude response, the four filters designed with the same order and cutoff frequency.

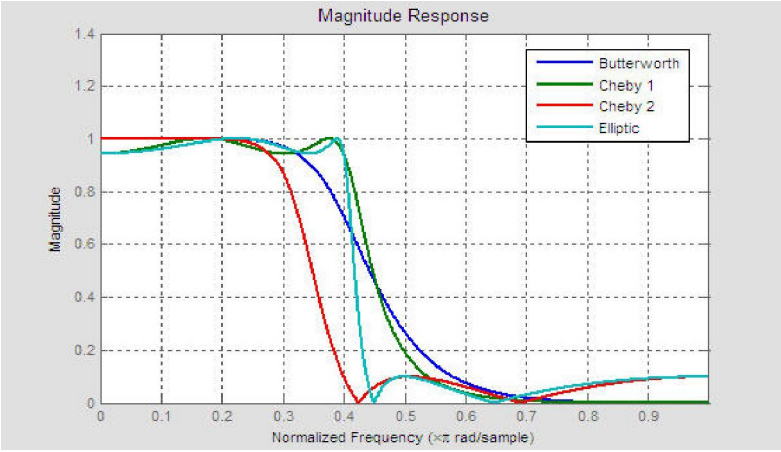

Fig.20 Magnitude Response

Figure (20) Four filters for the same order and cutoff frequency: Butterworth, Chebyshev type I,Chebyshev type II and a elliptical filter, The elliptic filter has a sharper transition band than all the others, but also it has ripples on the whole bandwidth. The butter worth has a flat passband, and the two chebyshevs are in between.

H. Matlab and least pth-norm or constraines least pth-norm

- For the least -Pth norm (The function iirlpnorm)
- [B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector) or
- [B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector) or
- [B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector, radius, pthnorm)
- [B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector, radius, pthnorm, DENS)

For the Constrained Least -Pth norm (The function iirlpnormc)

[B, A] = iirlpnormc (num, den, fvector, edgesvector, mvector) or

[B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector) or

[B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector,pthnorm)or

[B, A]= iirlpnorm (num, den, fvector, edgesvector, mvector, wvector, pthnorm, DENS)

Where:

num is the filter numerator order

den is the filter denominator order

fvector is the best approximation to the desired frequency response

mvector is the filter magnitude vector in the least -Pth sense.

edgesvector specifies the band-edge frequency points where a frequency band starts/stops and a don't care regions stops/starts.

wvector is the weight error vector

Pthnorm is a two-element vector [Pmin Pmax] allows for the specification of the minimum and maximum values of P used in the least -Pth algorithm

DENS specifies the grid density used in the optimization

radius is the maximum pole radius [3]

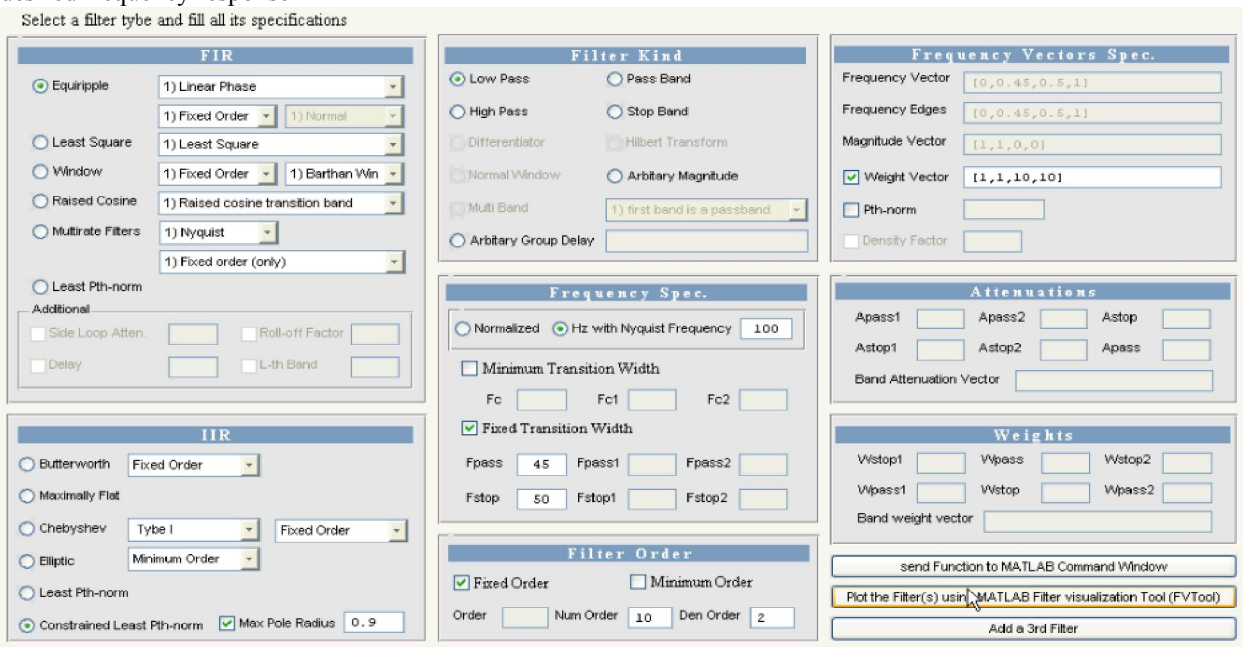

Fig.21 The two Lowpass filters with following specifications

[B1,A1]=iirlpnorm(10,2,[0,0.45,0.5,1],[0,0.45,0.5,1],[1,1,0,0],[1,1,10,10]);
[B2,A2]=iirlpnormc(10,2,[0,0.45,0.5,1],[0,0.45,0.5,1],[1,1,0,0],[1,1,10,10],0.9);
are designed with iirlpnorm (Least Pth-norm) and iirlpnormc (constrained Least Pth-norm by using GUI.

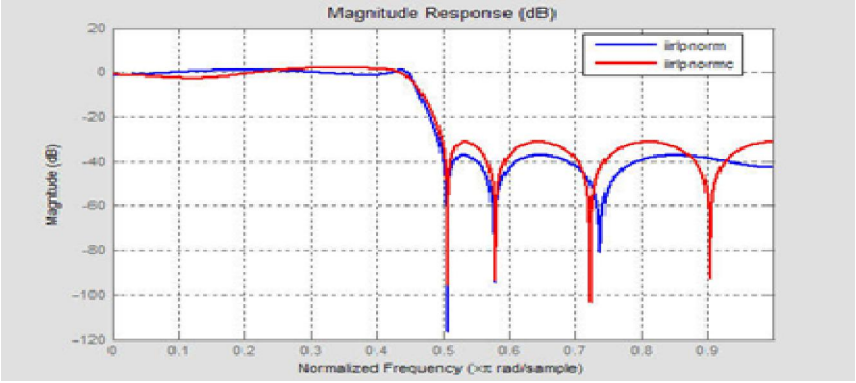

Fig.22 Magnitude Response

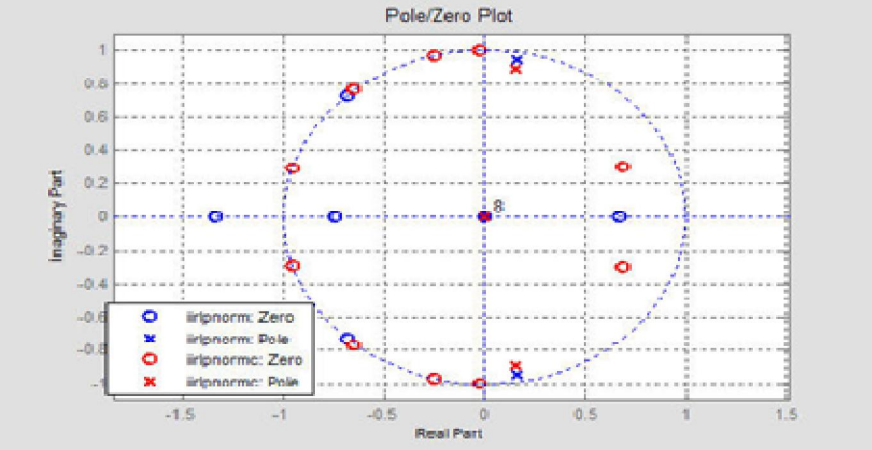

Fig.23 Pole/Zero Plot

The figure shows the pole/zero plot for the iirlpnorm (Least Pth-norm) and iirlpnormc (Constrained Least Pth-norm). The un-constrained design cause a zero outside the unit circle and the constrained design prevents this problem.

Conclusion

The ability of using this advanced computer aided design methods were demonstrated by a specially developed GUI program for an accurate design to choose the best kind suitable digital filters using MATLAB techniques.

Introduction and application of recursive and non-recursive filters were demonstrated and have been introduced to the design of inter- digitized computer. A Typical real examples were given and a demonstrated tests were achieved for many different and wide specifications of digital filter design. Finally, the main objective of this program was to help and guide the experienced and the non-experienced user in order to achieve an optimum design of digital filters.

Acknowledgment

This article was funded by Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah. The authors, therefore, acknowledge with thanks DSR technical and financial support. We would like to sincerely thank Mr. Adil Ameer Ahmed Hafez for his efforts during the work.

References

1. Chonghua Li " Design and Realization of FIR Digital Filters Based on MATLAB ", IEEE. PP.101-104, 2010.
2. Xueling Bai Hongmei Zhang " Design of Digital Filter Based on VB and Matlab ", IEEE. PP.4-85 – 4-88, 2009.
3. MATLAB Signal Processing Toolbox, The MathWorks,2013.

3/15/2014