ADDIS ABABA UNIVERSITY

School of Graduate Studies

School of Electrical and Computer Engineering

Optimal Design of Low Pass Finite Impulse Response Filter Using Hybrid Population Based Algorithm

By Tsehay Damte

Masters of Since in Electrical Engineering

March 2014

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

School of Electrical and Computer Engineering

Optimal Design of Low Pass Finite Impulse Response Filter Using Hybrid Population Based Algorithm

By Tsehay Damte

A thesis submitted to the school of graduate studies of Addis Ababa University in partial fulfillment of the requirement of the degree of

Masters of Science in Electrical Engineering

March 2014

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
School of Electrical and Computer Engineering

Optimal Design of Low Pass Finite Impulse Response Filter Using Hybrid Population Based Algorithm

By Tsehay Damte

March 2014

Advisor
Dr. Eneyew Adugna

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

Optimal Design of Low Pass Finite Impulse Response Filter Using Hybrid Population Based Algorithm

By

Tsehay Damte

Faculty of Technology

Approval by Board of Examiners

_____	_____
Chairman Department Graduate Committee	Signature
_____	_____
Advisor	
_____	_____
Co-advisor	
_____	_____
Internal Examiner	
_____	_____
External Examiner	

DECLARATION

I, **TsehayDamte,** declare that this master's research thesis entitled, "**Optimal Design of Low Pass Finite Impulse Response (FIR) Filter Using Hybrid Population Based Algorithm"**, submitted for the award of the Master of Degree in communication Engineering of Addis Abeba University is my original work and it has not been presented for the award of any other Degree, Diploma, Fellowship or other similar titles of any other university or institution.

Name of the student---------- ID: ---------- Signature: ---------- Date: ----

ABSTRACT

This research presents an optimal design of linear phase digital low pass finite impulse response (FIR) filter using hybrid population based algorithm (the hybrid of genetic algorithm and particle swarm optimization). The basis behind this is that such a hybrid approach is expected to have merits of particle swarm optimization (PSO) with those of genetic algorithm (GA). Applying crossover operation on the PSO, information can be swapped between two particles to have the ability to fly to the new search area. Also the purpose of applying mutation to PSO is to increase the diversity of the population and the ability to have the PSO to avoid the local maxima and hence the solution quality is improved. In the design process, the filter length, passband and stopband frequencies, feasible passband ripple (the difference between an ideal filter and designed approximate filter in the passband reign) and stopband ripple (the difference between an ideal filter and designed approximate filter in the stopband reign) sizes are specified. Evolutionary algorithms like GA, PSO, and hybrid of genetic algorithm and particle swarm optimization (HGAPSO) have been used in this work for the design of linear phase FIR lowpass (LP) filter. In this research filter of order 20, 30, 40, and 50 have been realized using HGAPSO and the simulations clearly indicate that HGAPSO have best performance in terms of passband ripple for orders higher than 20. The results justify that the HGAPSO outperforms GA and PSO in terms of minimum stopband ripple and maximum attenuation.

Key-Words: - FIR Filter, RGA, PSO, HGAPSO, Parks and McClellan (PM) Algorithm, Evolutionary Optimization, Low Pass Filter

ACKNOWLEDGEMENTS

First and foremost I would like to thank almighty GOD for being with me throughout my life. I would also like to extend my deepest gratitude to my advisors **Dr. Eneyew Adugna** for his invaluable comments, encouragements and guidance at various stage of this study.

My special gratitude goes to **Menor Tekeba** for his precious idea and tireless support throughout the course of my study.

Similarly, my special thanks goes to Mekelle University for giving me the opportunity to pursue the Masters Degree.

At the last but not the least I want to thank all my teachers who earnestly encouraged and morally supported me to keep up with the task.

Contents

LIST OF TABLES

Tables **Page NO**

LIST OF FIGURES

Figures **Page No**

ABBREVIATIONS

ADC	Analog-To-Digital Converter
CPSO	Cultural Particle Swarm Optimization
dB	Decibels
DFT	Discrete Fourier Transform
DSP	Digital Signal Process
FIR	Finite impulse response
FPGA	Field-Programmable gate Array
GA	Genetic Algorithm
GA-PSO	Hybrid Genetic Algorithm with Particle Swarm Optimization
gbest	group best
IDFT	Inverse Discrete Fourier Transform
IIR	Infinite impulse response
IPSO	Improved Particle Swarm Optimization
LP	Lowpass
NPSO	Novel Particle Swarm Optimization
pbest	personal best
PM	Parks and McClellan
PSO	Particle Swarm Optimization
PSO-CFIWA	PSO with Constriction Factor and Inertia Weight Approach
QPSO	Quantum Particle Swarm Optimization
RF	Radio Frequency
SDP	Semi-Definite Programming

WLS	Weighted Least Squares

CHAPTER ONE

1. INTRODUCTION

1.1. General

In signal processing, the function of a filter is to remove unwanted parts of the signal, such as random noise, or to extract useful parts of the signal, such as the components lying within a certain frequency range [9]. A filter alters the amplitude and/or phase characteristics of a signal with respect to frequency. There are two main kinds of filters,

1. Analog Filters
2. Digital Filters

1.1.1. Analog filters

An analog filter has an analog signal at both its input $x(t)$ and its output $y(t)$. Both $x(t)$ and $y(t)$ are functions of a continuous variable, mostly time, *t*. An analog filter uses analog electronic circuits made up from components such as resistors, capacitors and op amps to produce the required filtering effect. Such filter circuits are widely used in such applications as noise reduction, video signal enhancement, graphic equalizers in, and many other areas. At all stages, the signal being filtered is an electrical voltage or current which is the direct analogue of the physical quantity (e.g. a sound or video signal or transducer output) involved.

1.1.2. Digital filters

A digital filter uses a digital processor to perform numerical calculations on sampled values of the signal. The processor may be a general purpose computer such as a PC, or a specialized DSP (Digital Signal Processor) chip. Digital filters are used in a wide variety of signal processing applications, such as spectrum analysis, digital image processing, and pattern recognition. Digital filters eliminate a number of problems associated with their classical analog counter parts and thus are preferably used in place of analog filters. The analog input signal is first sampled and digitized using an ADC (analog to digital converter). The resulting binary signals, representing successive sampled values of the input signal, are transferred to the processor, which carries out numerical calculations on them. In fast DSP processors the hardware requirements relatively simple and compact in comparison with the equivalent analog circuitry [17].

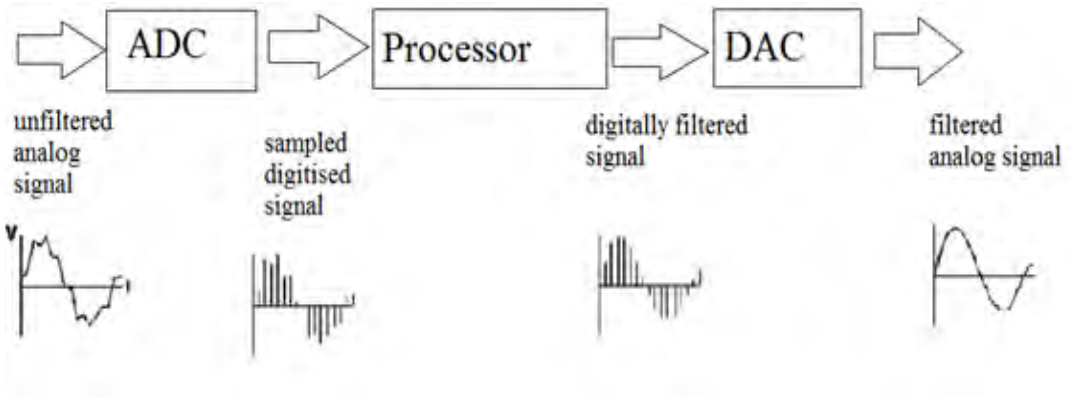


Figure 1.1 Block diagram of digital filter [17].

1.2. Characteristics of an ideal filter

Ideal filters allow a specified frequency range of interest to pass through while attenuating a specified unwanted frequency range. The filters are classified according to their frequency range characteristics [9].

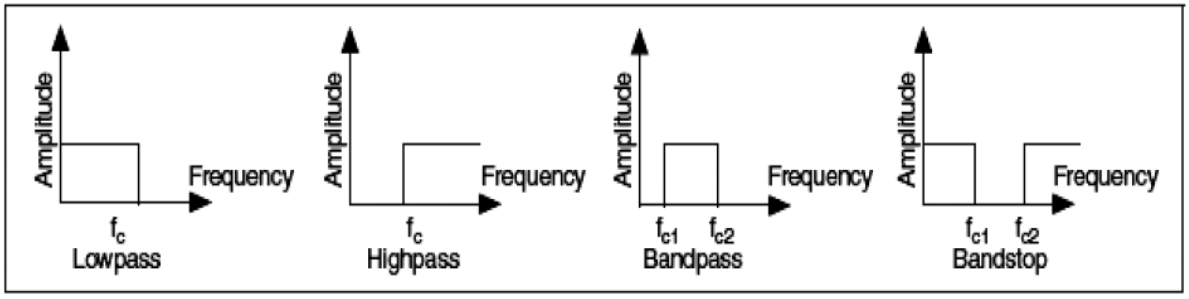


Figure 1.2 Ideal frequency characteristics [9].

In Figure 1.2, the filters exhibit the following behavior:

- The lowpass filter passes all frequencies below f_c.
- The highpass filter passes all frequencies above f_c.
- The bandpass filter passes all frequencies in a frequency band between f_{c1} and f_{c2}.
- The bandstop filter attenuates all frequencies in a frequency band between f_{c1} and f_{c2}.

The frequency points f_c, f_{c1}, and f_{c2} specify the cut off frequencies for the different filters. When designing filters, you must specify the cut off frequencies. The passband of the filter is the

frequency range that passes through the filter. An ideal filter has a gain of one (0 dB) in the passband so the amplitude of the signal neither increases nor decreases.

1.3. Practical (non ideal) filters

In practical applications, ideal filters are not realizable. Ideally, a filter has a unit gain (0 dB) in the passband and a gain of zero (–∞ dB) in the stopband. However, real filters cannot fulfill all the criteria of an ideal filter. In practice, a finite transition band always exists between the passband and the stopband [9].

1.3.1. Transition band

Figure 1.3 shows the passband, the stopband, and the transition band for each type of practical filter. In each plot in Figure 1.3, the *x* axis represents frequency, and the *y* axis represents the magnitude of the filter in dB. Here transition band is in between passband and stopband. [9].

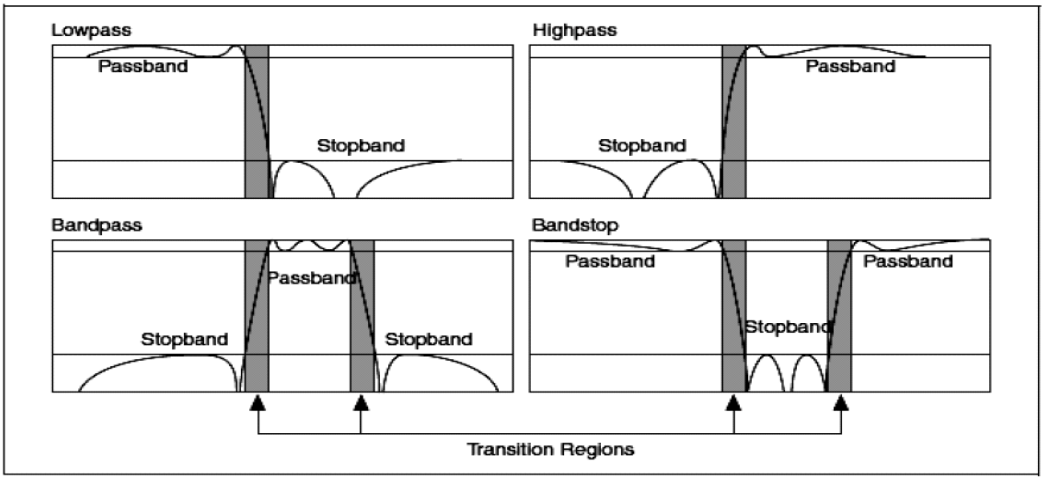


Figure 1.3 Response of Non-ideal filters [9].

1.3.2. Passband ripple and stopband attenuation

In many applications, you can allow the gain in the passband to vary slightly from unity. This variation in the passband is the passband ripple, or the difference between the actual gain and the desired gain of unity. In practice, the stopband attenuation cannot be infinite, and you must specify a value with which you are satisfied. Both the passband ripple and the stopband attenuation can also be expressed in decibels (dB) [9]. For example, a passband ripple (δ_p) of 0.1 db translates to $10^{+0.1/20} - 1 = 0.0115$ linear scale where passband ripple in dB is given

by $\delta_p, dB \equiv A_p = 20\log(\delta_p + 1)$. A stopband ripple ($\delta_s$) of 60 dB on the other hand, equals $10^{-60/20} = 0.001$ on a linear scale where $\delta_s, dB \equiv A_s = -20\,log(\delta_s)$. Thus, a typical passband ripple specification may be 10 times larger than the stopband attenuation.

1.3.3. Sampling rate

The sampling rate is important to the success of a filtering operation. The maximum frequency component of the signal of interest usually determines the sampling rate. In general, choose a sampling rate higher than the highest frequency component of the signal of interest [9].

1.4. Common digital filters

Digital filters can be classified as one of the following types [9].

- Finite impulse response (FIR) filter, also known as non-recursive filters (in a non-recursive filter the current output is calculated solely from the current and previous input values).
- Infinite impulse response (IIR) filter, also known as recursive filters (in a recursive filter the current output is calculated from in addition to the current and previous input values also uses previous output values).

1.4.1. FIR filters

Finite impulse response (FIR) filters are digital filters that have a finite-length impulse response. FIR filters operate only on current and past input values. This can be stated mathematically as [18].

$$h(n) = \begin{cases} 0, & n < n_1 \\ h_n, & n_1 \le n \le n_2 \\ 0, & n > n_2 \end{cases} \tag{1.1}$$

where n_1 and n_2 are finite integers. $h(n)$ denotes the impulse response of the digital filter, n is the discrete time index, and n_1 and n_2 are constants.

The general difference equation for an FIR digital filter (M^{th}-order) is

$$y(n) = \sum_{k=0}^{M} h_k\, x(n-k) \tag{1.2}$$

where $y(n)$ is the filter output at discrete time instance$n$, $h_k$is the k^{th} feed forward tap, or filter coefficient, and $x(n-k)$ is the filter input delayed by *k* samples. The Σ denotes summation from $k = 0\ to\ k = M$ where $M+1$ is the number of feed forward taps in the FIR filter. If a single impulse is present at the input of an FIR filter and all subsequent inputs are zero, the output of an FIR filter becomes zero after a finite time. Therefore, FIR filters have finite impulse response. The time required for the filter output to reach zero equals the number of filter coefficients.

1.4.2. IIR filters

Infinite impulse response (IIR) filters, also known as recursive filters, operate on current and past input values and past output values. Theoretically, the impulse response of an IIR filter cannot reach zero and is an infinite response. The expression for a recursive filter therefore contains not only terms involving the input values $(x_n,\ x_{n-1},\ x_{n-2},\cdots)$ but also terms involving the past output value$(y_n,\ y_{n-1},\ y_{n-2},\cdots)$. The following general difference equation characterizes IIR filters [18].

$$y_n = \sum_{j=0}^{M_b} b_j\, x_{n-j} - \sum_{k=1}^{M_a} a_k\, y_{n-k} \tag{1.3}$$

Where b_j is the set of forward coefficients, M_b+1 is the number of forward coefficients, a_k is the set of reverse coefficients, and M_a is the number of reverse coefficients. x_n is the current input, x_{n-j} is the past inputs, and y_{n-k} is the past outputs.

1.5. Comparing FIR and IIR filters

Because designing digital filters involves making compromises to emphasize a desirable filter characteristic over a less desirable characteristic, comparing FIR and IIR filters can help in selecting the appropriate filter design for a particular application [9]

IIR filters can achieve the same level of attenuation as FIR filters but with far fewer coefficients. Therefore, an IIR filter can provide a significantly faster and more efficient filtering operation than an FIR filter. FIR filters provide a linear phase response but IIR filters provide a nonlinear phase response. FIR filters are used for applications that require linear phase responses like high

quality audio systems. IIR filters are used for applications that do not require phase information, such as signal monitoring applications.

1.6. Frequency-Domain Description of FIR Filters

Digital filters with a finite-duration impulse response (FIR) have characteristics that make them useful in many applications. They can achieve exactly linear phase and the design methods are generally linear. They can be efficiently realized on general or special-purpose hardware. Because of the method of implementation, the FIR filter is also called a convolution filter. From the time-domain view of this operation, the FIR filter is sometimes called a moving-average filter.

The transfer function of an M^{th}– order FIR filter is given by the z transform of $h(n)$ as [23]

$$H(z) = \sum_{n=0}^{M} h(n) z^{-n} \tag{1.4}$$

The frequency response of an M^{th}– order FIR filter is found by setting z = $e^{j\omega}$ which gives the form

$$H(\omega) = \sum_{n=0}^{M} h(n) e^{-j\omega n} \tag{1.5}$$

This frequency response function is complex valued and consists of a magnitude and a phase. Even though the impulse response is a function of the discrete variable n, the frequency response is a function of the continuous-frequency variable ω and is periodic with period 2π. This periodicity is easily shown as follows.

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

$$= \sum_{n=0}^{M} h(n) e^{-j\omega n}\ e^{-j2\pi n} \tag{1.6}$$

where $e^{-j2\pi n} = \cos 2\pi n - j \sin 2\pi n = cos\ 2\pi n = 1$

$H(\omega + 2\pi) = H(\omega)$

Frequency is denoted by ωin radians per sample or by f in hertz (cycle per second). These quantities are related by

$$\omega = \frac{2\pi f}{f_s} \tag{1.7}$$

This can be used to evaluate the frequency response at certain frequencies. The N-point DFT of the length-L impulse response $h(n)$is defined as

$$C(k) = \sum_{n=0}^{L-1} h(n) e^{-\frac{j2\pi nk}{N}}, \qquad k = 0, 1, \cdots, N-1 \tag{1.8}$$

When compared to (1.5), (1.8) gives

$$C(k) = H(\omega)_{|\omega=\frac{2\pi k}{N}} = H\left(\frac{2\pi k}{N}\right), \quad k = 0, 1, \cdots, N-1 \tag{1.9}$$

This states that the DFT of $h(n)$ gives N samples of the frequency response function $H(\omega)$. This sampling at N points may not give enough detail; therefore, more samples are needed. This method is often useful when an accurate picture of all of $H(\omega)$is required.

1.7. Linear Phase FIR Filters

Now we consider the special types of FIR filters in which the coefficients $h(n)$ of the transfer function

$$H(z) = \sum_{n=0}^{M} h(n) z^{-n} \tag{1.10}$$

is assumed to be symmetric or anti-symmetric. Since the order of the polynomial in each of these two types can be either odd or even, there are four types of filters with different properties [13].

Type I. The coefficients are symmetric$[i.e., h(n) = h(M - n)]$, and the order M is even.

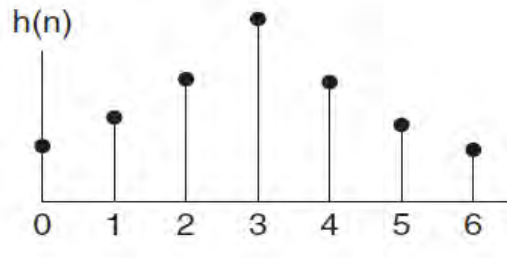


Fig 1.4 (a) Type I $M = 6$

In the general case, $H(e^{j\omega})$can be expressed in a few other forms, for example

$$H(e^{j\omega}) = \sum_{n=0}^{M} h(n)e^{-jn\omega}$$

$$= h(0) + h(1)e^{-j\omega} + h(2)e^{-j2\omega} + \ldots + h(M)e^{-jM\omega} \quad (1.11)$$

$$= e^{-j\left[\left(\frac{M}{2}\right)\omega\right]}\left\{2h(0)\cos\left(\frac{M\omega}{2}\right) + 2h(1)\cos\left(\left(\frac{M}{2}-1\right)\omega\right) + 2h(2)\cos\left(\left(\frac{M}{2}-2\right)\omega\right) + \ldots + h\left(\frac{M}{2}\right)\right\}$$

Put it in a more compact form:

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

$$= e^{j\theta(\omega)}\{H_R(\omega)\} \quad (1.12)$$

Where $H_R(\omega) = h\left(\frac{M}{2}\right) + 2\sum_{n=1}^{M/2} h\left[\frac{M}{2} - n\right]\cos(n\omega)$

and $\theta(\omega) = -\frac{M}{2}\omega$, $\tau = -d\frac{\theta(\omega)}{\omega}$

The total group delay $\tau = M/2$ for a type I FIR filter

Type II. The coefficients are symmetric$[i.e., h(n) = h(M - n)]$, and the order *M* is odd.

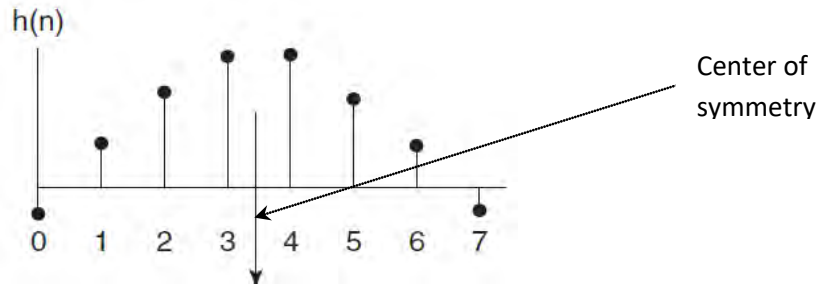


Fig 1.4(b) Type II M=7

In the general case of type II filter, we obtain

$$H(e^{j\omega}) = \sum_{n=0}^{M} h(n)e^{-jn\omega} = e^{j\theta(\omega)}\{H_R(\omega)\}$$

$$= e^{-j\left(\frac{M}{2}\right)\omega}\left\{\sum_{n=1}^{\frac{M+1}{2}} 2h\left[\frac{M+1}{2} - n\right]\cos\left(n-\frac{1}{2}\right)\omega\right\} \quad (1.13)$$

where

$$H_R(\omega) = \sum_{n=1}^{\frac{M+1}{2}} 2h\left[\frac{M+1}{2} - n\right]\cos\left(n-\frac{1}{2}\right)\omega, \quad \theta(\omega) = -\left(\frac{M}{2}\right)\omega$$

The total group delay $\tau = M/2$ for a type II FIR filter

Type III. The coefficients are anti-symmetric$[i.e., h(n) = -h(M-n)]$, and the order M is even.

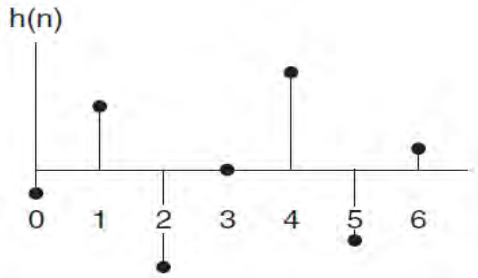


Fig 1.4(c) Type III M= 6

In the general case, it can be shown that

$$H(e^{j\omega}) = e^{-j(M\omega-\pi)/2}\left\{2\sum_{n=1}^{M/2} h\left[\frac{M}{2}- n\right] sin(n\omega)\right\} \quad (1.14)$$

$$H_R(\omega) = 2\sum_{n=1}^{M/2} h\left[\frac{M}{2}- n\right] sin(n\omega), \quad \theta(\omega) = -(M\omega - \pi)/2$$

The total group delay $\tau = M/2$ for a type III FIR filter

Type IV. The coefficients are anti-symmetric [i. e., $h(n) = -h(M - n)$], and the order Mis odd.

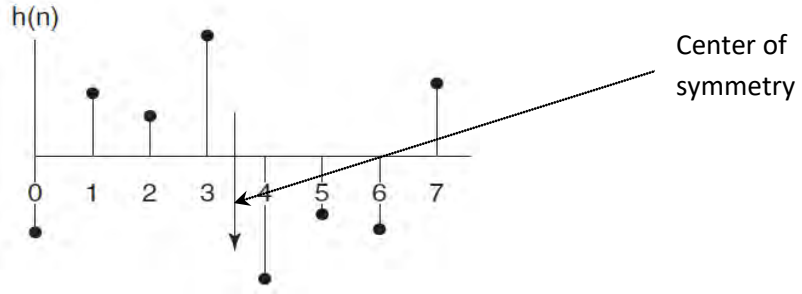


Fig 1.4 (d) Type IV M = 7

The transfer function of the type IV linear phase filter in general is given by

$$H(e^{j\omega}) = e^{-j(M\omega-\pi)/2}\left\{2\sum_{n=1}^{(M+1)/2} h\left[\frac{M+1}{2}- n\right] \sin\left(\left(n-\frac{1}{2}\right)\omega\right)\right\} \quad (1.15)$$

where

$$H_R(\omega) = 2\sum_{n=1}^{(M+1)/2} h\left[\frac{M+1}{2}- n\right] \sin\left(\left(n-\frac{1}{2}\right)\omega\right)$$

,

$$\theta(\omega) = -(M\omega - \pi)/2$$

The total group delay $\tau = M/2$ for a type IV FIR filter

1.7.1. Properties of Linear Phase FIR Filters

The four types of FIR filters discussed above have shown us that FIR filters with symmetric or anti-symmetric coefficients provide linear phase (or equivalently constant group delay); these coefficients are samples of the unit impulse response. The following observations about these typical magnitude responses will be useful in making proper choices in the early stage of their design, as will be [13].

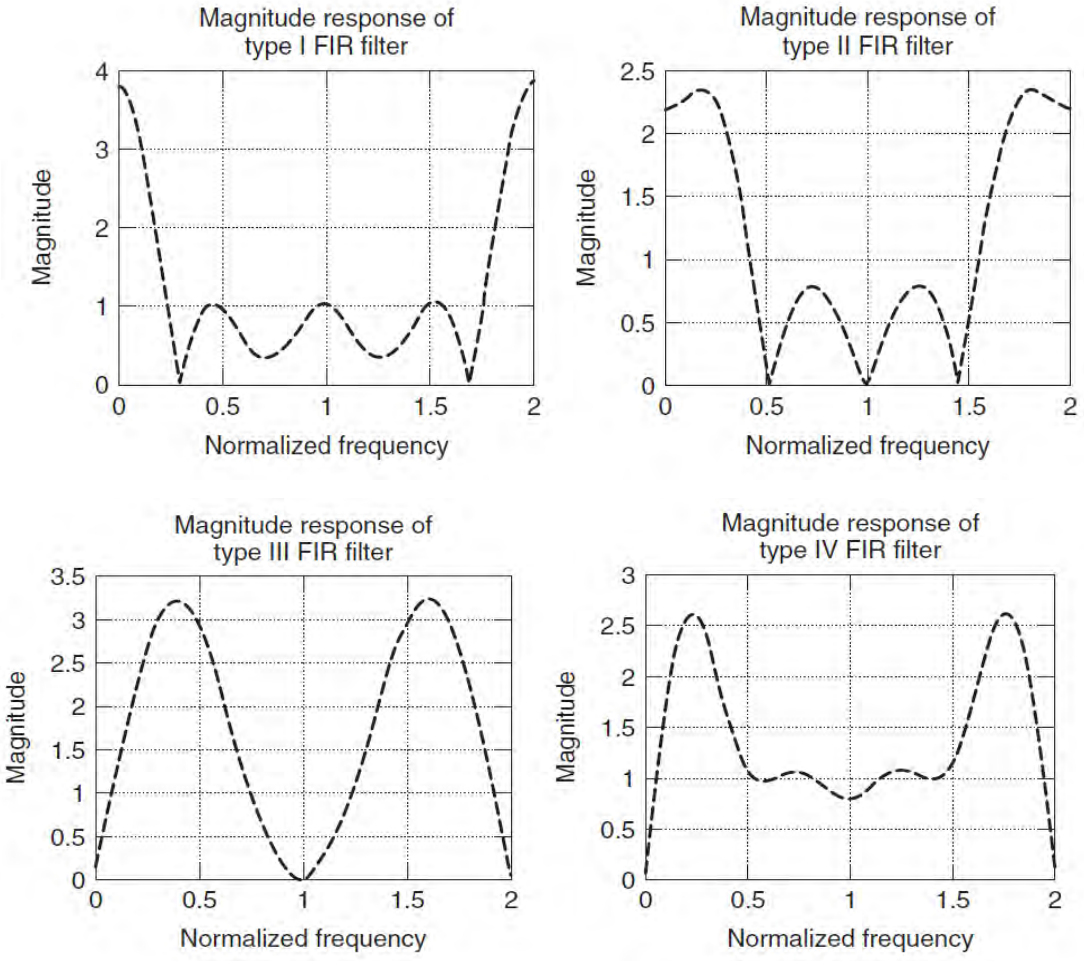


Fig 1.5 Magnitude responses of the four types of linear phase FIR filters.

For example, type I filters have a nonzero magnitude at $\omega = 0$ and also a nonzero value at the normalized frequency $\omega/\pi = 1$ (which corresponds to the Nyquist frequency), whereas type II filters have nonzero magnitude at $\omega = 0$ but a zero value at the Nyquist frequency. So it is obvious that these filters are not suitable for designing bandpass and highpass filters, whereas both of them are suitable for lowpass filters. The type III filters have zero magnitude at $\omega = 0$ and also at$\omega/\pi = 1$, so they are suitable for designing bandpass filters but not lowpass and bandstop filters. Type IV filters have zero magnitude at ω = 0 and a nonzero magnitude at $\omega/\pi = 1$, so they are not suitable for designing lowpass and bandstop filters but are candidates for bandpass and highpass filters [13].

1.8. Filter Specifications

Before a filter can be designed, a set of filter specifications must be defined. For example, suppose that we would like to design a low-pass filter with a cutoff frequencyω_c. The frequency response of an ideal low-pass filter with linear phase and a cutoff frequency ω_c is

$$H_i(e^{j\omega}) = \begin{cases} 1, & 0 \le |\omega| \le \omega_c \\ 0, & \omega_c \le |\omega| \le \pi \end{cases} \tag{1.16}$$

As illustrated in Fig 1.6. Thus, the specifications include the passband cutoff frequency, ω_p, the stopband cutoff frequency, ω_s,the passband deviation, δ_p, and the stopband deviation, δ_s.

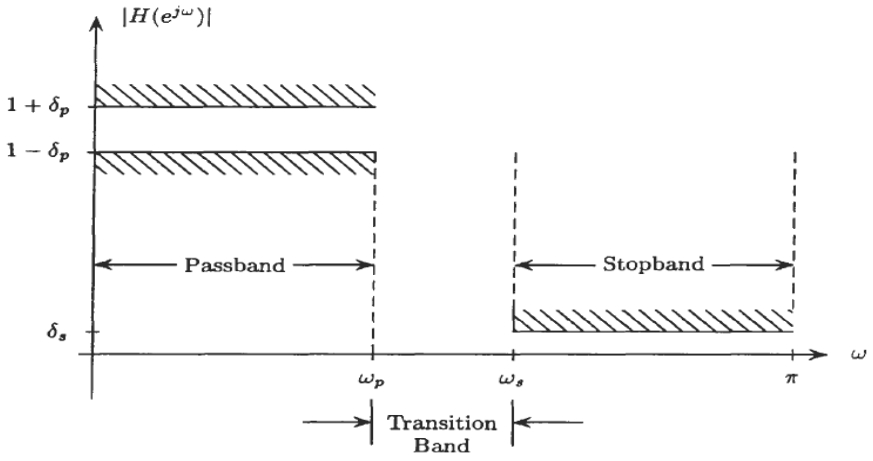


Fig 1.6 Filter specifications for a lowpass filter,

The design parameters are defined as follows:

- δ_s, stop-band ripple ($\leq$ 0.001= 60 and 0.01 = 40 dB is common)

$$\delta_s, dB = A_s = -20\, log_{10}(\delta_s) \quad (1.17)$$

- δ_p, passband ripple ($\leq$ 0.1dB typical)

$$\delta_p, dB = A_p = 20\, log_{10}(1 + \delta_p) \quad (1.18)$$

- ω_s, stopband edge frequency
- ω_p, passband edge frequency
- TW: transition width = $\omega_s - \omega_p$
- SBA: stopband attenuation = $-20\, log_{10}(\delta_s)$

1.9. Digital FIR Filter Design Methods

The objective of most FIR design methods is to obtain values of $h(n)$ such that the resulting filter meets the design specifications, such as amplitude v_s frequency response. Several classical methods are available for obtaining $h(n)$. The window and frequency sampling methods are the most commonly used [22].

1.9.1. Classical Design Methods

1.9.1.1. Window Method

Applying the window sequence to the filter coefficients gives [17]

$$h_w(n) = h(n) * w(n) \quad (1.19)$$

where $w(n)$ designates the window function. Common window functions used in the FIR filter design are as follows.

(a) Rectangular window:

$$w_{rec}(n) = \begin{cases} 1, & -M \leq n \leq M \\ 0, & otherwise \end{cases} \quad (1.20)$$

(b) Triangular (Bartlett) window:

$$w_{tri}(n) = 1 - \frac{|n|}{M}, \qquad -M \leq n \leq M \tag{1.21}$$

(c) Hanning window:

$$w_{han}(n) = 0.5 + 0.5\cos\left(\frac{n\pi}{M}\right), \qquad -M \leq n \leq M \tag{1.22}$$

(d) Hamming window:

$$w_{ham}(n) = 0.54 + 0.46\cos\left(\frac{n\pi}{M}\right), \qquad -M \leq n \leq M \tag{1.23}$$

(e) Blackman window:

$$w_{black}(n) = 0.42 + 0.5\cos\left(\frac{n\pi}{M}\right) + 0.08\cos\left(\frac{2n\pi}{M}\right), \qquad -M \leq n \leq M \tag{1.24}$$

The design procedure includes three steps. The first step is to obtain the impulse response $h(n)$, where $-M \leq n \leq M$; then we multiply the obtained sequence $h(n)$ by the selected window data sequence to yield the windowed non-causal FIR filter coefficients $h_w(n)$; and the final step is to delay the windowed non-causal sequence $h_w(\mathrm{n})$ by M samples to achieve the causal FIR filter coefficients, $b_n = h_w(n - M)$. The design procedure of the FIR filter via windowing is summarized as follows [17]:

(a) Obtain the FIR filter coefficients $h(n)$ via the inverse Fourier transform method.

(b) Multiply the generated FIR filter coefficients by the selected window sequence

$$h_w(n) = h(n) * w(n), \quad n = -M, \ldots 0, 1, \ldots, M \tag{1.25}$$

where $w(n)$ is chosen to be one of the window functions listed in Eq. (1.19) to (1.24).

(c) Delay the windowed impulse sequence $h_w(\mathrm{n})$ by M samples to get the desired windowed FIR filter coefficients:

$$b_n = h_w(n-M), \quad for \quad n = 0,1,\ldots,2M \tag{1.26}$$

1.9.1.2. Frequency Sampling Design Method

In addition to windowing method discussed in the previous section, frequency sampling is another alternative. The key feature of frequency sampling is that the filter coefficients can be calculated based on the magnitudes of the desired filter frequency response specified uniformly in frequency domain [17].

Then, according to the definition of the inverse DFT (IDFT), we can calculate the FIR coefficients:

$$h(n) = \frac{1}{M}\sum_{k=0}^{M} H(k)\, W_N^{-kn}, \quad for \quad n = 0,1,\ldots,M \tag{1.27}$$

$$W_N = e^{\frac{j2\pi}{N}} = cos\left(\frac{2\pi}{N}\right) - j\, sin\left(\frac{2\pi}{N}\right) \tag{1.28}$$

Assume that the FIR filter has linear phase and the number of taps M = 2N + 1. Eq. (1.27) can be significantly simplified as

$$h(n) = \frac{1}{2N+1}\left\{H_0 + 2\sum_{k=1}^{N} H_k\, cos\left(\frac{2\pi k(n-N)}{2N+1}\right)\right\}, \quad for\ n = 0,1,\ldots,2N, \tag{1.29}$$

where H_k for $k = 0,1,\ldots,2N$, represents the magnitude values specifying the desired filter frequency response sampled at $\Omega_K = \frac{2\pi k}{(2N+1)}$.

The design procedure is therefore simply summarized as follows [17]:

1. Given the filter length of $2N + 1$, specify the magnitude frequency response for the normalized frequency range from 0 to π

$$\mathrm{H}_k \text{ at } \Omega_K = \frac{2\pi k}{(2N+1)} \qquad for\ k = 0, 1, \ldots, N, \tag{1.30}$$

2. Calculate FIR filter coefficient

$$h(n) = \frac{1}{2N+1}\left\{H_0 + 2\sum_{k=1}^{N} H_k \cos\left(\frac{2\pi k(n-N)}{2N+1}\right)\right\} \tag{1.31}$$

$for\ n = 0, 1, \ldots, N,$

3. Use the symmetry (linear phase requirement) to determine the rest of the coefficients:

$$h(n) = h(2N - n) \qquad for\ n = N + 1, \ldots, 2N \tag{1.32}$$

Like most other engineering problems, the design of digital filters involves multiple, often conflicting, design criteria and specifications, and finding an optimum design is, therefore, not a simple task. Classical methods such as windowing and frequency sampling usually lead to sub-optimal designs. Consequently, there is a need for optimization-based methods that can be used to design digital filters that would satisfy prescribed specifications [26]

However, optimization problems for the design of digital filters are often complex, highly nonlinear, and multimodal in nature. The problems usually exhibit many local minima. Ideally, the optimization method should lead to the global optimum of the objective function with a minimum amount of computation. Classical optimization methods are generally fast and efficient, and have been found to work reasonably well for the design of digital filters [26]. These methods are very good in locating local minima but unfortunately, they are not designed to discard inferior local solutions in favor of better ones. Therefore, they tend to locate minima in the locale of the initialization point. In general disadvantage of classical methods are generally sub optimality and lack of precise control of critical frequencies and transition width.

Another optimal design methods are Parks-McClellan algorithm (proposed by Parks and McClellan) and population based algorithm are used for the design of exact linear phase filter [1].

1.9.2. Optimal Design Method

1.9.2.1. Parks-McClellan Algorithm

The FIR filter design using the Parks-McClellan algorithm is developed based on the idea of minimizing the maximum approximation error to the desired filter amplitude frequency response. Given an ideal frequency response, the approximation error $E(\omega)$ is defined as

$$E(\omega) = W\left[H\left(e^{j\,\omega T}\right) - H_d\left(e^{j\,\omega T}\right)\right] \tag{1.33}$$

where $H\left(e^{j\,\omega T}\right)$ is the ideal frequency response, $H_d\left(e^{j\,\omega T}\right)$ is the frequency response of the filter to be designed, and W is the weight function for emphasizing certain frequency bands over others during the optimization process. This process is designed to minimize the error shown in Eq. (1.34)

$$min\left(max|\,E(\omega)|\,\right) \tag{1.34}$$

over the set of FIR coefficients. With the help of Remez exchange algorithm, we can obtain the best FIR filter whose magnitude response has an equiripple approximation to the ideal magnitude response. The achieved filters are optimal in the sense that the algorithms minimize the maximum error between the desired frequency response and the actual frequency response. These are often called $minimax$ filters.

Optimal FIR Filter Design Procedure for Parks-McClellan

1. Specify the band edge frequencies such as the passband and stopband frequencies, passband ripple, stopband attenuation, filter order, and sampling frequency of the DSP system.
2. Normalize band edge frequencies to the Nyquist limit (folding frequency= $fs/2$) and specify the ideal magnitudes.

3. Calculate absolute values of the passband ripple and stopband attenuation if they are given in terms of dB values:

$$\delta_p = 10^{\left(\frac{A_p}{20}\right)} - 1$$

$$\delta_s = 10^{\left(\frac{A_s}{20}\right)}$$

Then calculate the ratio and put it into a fraction form:

$$\frac{\delta_p}{\delta_s} = fraction\ form = \frac{numerator}{denominator} = \frac{W_s}{W_p}$$

Next, set the error weight factors for pass-band and stop-band, respectively:

W_p = denominator

W_s = numerator

4. Apply the Remez algorithm to calculate filter coefficients.

5. If the specifications are not met, increase the filter order and repeat steps 1 to 4.

The major limitation of this approach is that the relative values of the amplitude error in the frequency bands are specified by means of the weighting functions and not by the deviations themselves. The program has to be iterated many times in order to meet the filter specifications in terms of stopband attenuation, cut-off frequency and filter length.

1.9.2.2. Optimal design based on population based algorithms

In recent years, a variety of population based algorithms have been proposed for global optimization including stochastic algorithms [27]. Stochastic algorithms involve randomness and/or statistical arguments and in some instances are based on analogies with natural processes. The algorithms based on the mechanics of natural selection and genetics have come to be known collectively as evolutionary algorithms (EAs) [26]. Well-known examples of such algorithms are genetic algorithms, and particle swarm optimization. The population base algorithm methods

have been implemented for the design of optimal digital filters with better control of parameters and highest stopband attenuation.

Genetic algorithms: -are stochastic search methods that can be used to search for an optimal solution to the evolution function of an optimization problem. Holland proposed genetic algorithms in the early seventies as computer programs that mimic the natural evolutionary process. GAs manipulates a population of individuals in each generation (iteration) where each individual, termed as the chromosome, represents one candidate solution to the problem. Within the population, fit individuals survive to reproduce and their genetic materials are recombined to produce new individuals as offspring. The genetic material is modeled by some data structure, most often a finite-length of attributes. As in nature, selection provides the necessary driving mechanism for better solutions to survive. Each solution is associated with a fitness value that reflects how good it is, compared with other solutions in the population. The recombination process is simulated through a crossover mechanism that exchanges portions of data strings between the chromosomes. New genetic material is also introduced through mutation that causes random alterations of the strings. The frequency of occurrence of these genetic operations is controlled by certain pre-set probabilities. The selection, crossover, and mutation processes constitute the basic GA cycle or generation, which is repeated until some pre-determined criteria are satisfied. Through this process, successively better and better individuals of the species are generated [34].

Particle Swarm Optimization (PSO):- is a flexible, robust population-based stochastic search / optimization technique. The PSO algorithm works by simultaneously maintaining several candidate solutions in the search space. During each iteration of the algorithm, each candidate solution is evaluated by the objective function being optimized, determining the fitness of that solution. Each candidate solution can be thought of as a particle “flying” through the fitness landscape finding the maximum or minimum of the objective function.

Initially, the PSO algorithm chooses candidate solutions randomly within the search space. It should be noted that the PSO algorithm has no knowledge of the underlying objective function, and thus has no way of knowing if any of the candidate solutions are near to or far away from a local or global maximum. The PSO algorithm simply uses the objective function to evaluate its candidate solutions, and operates upon the resultant fitness values. Each particle maintains its

position, composed of the candidate solution and its evaluated fitness, and its velocity. Additionally, it remembers the best fitness value it has achieved thus far during the operation of the algorithm, referred to as the *individual best fitness*, and the candidate solution that achieved this fitness, referred to as the *individual best position* or individual best candidate solution. Finally, the PSO algorithm maintains the best fitness value achieved among all particles in the swarm, called the *global best fitness*, and the candidate solution that achieved this fitness, called the *global best position* or *global best candidate solution*.

The PSO algorithm consists of just three steps, which are repeated until some stopping condition is met.

1. Evaluate the fitness of each particle
2. Update individual and global best fatnesses and positions
3. Update velocity and position of each particle

Fitness evaluation is conducted by supplying the candidate solution to the objective function. Individual and global best fitness and positions are updated by comparing the newly evaluated fitness against the previous individual and global best fitness, and replacing the best fitness and positions as necessary [35].

1.10. Objective of thesis

General Objective

The general objective of this thesis work is to design an optimal low pass FIR (finite impels response) digital filter using hybrid population based algorithms.

Specific Objective

The thesis embodies following objectives:

- To investigate different population based algorithms.
- To select better algorithms and integrate the selected algorithms to develop a hybrid one.
- To evolve the design parameters of a digital FIR filter based on the developed hybrid algorithm.
- To analyze, evaluate the performance of the filter, and make comparison of the design with other similar research works.

1.11. Organization of thesis

Chapter 2 discusses different literatures for the optimization of digital filter coefficients using different population based algorithms. This includes design of linear phase digital low pass finite impulse response (FIR) filter using Novel Particle Swarm Optimization (NPSO), designing a linear phase digital high pass FIR filter using Improved Particle Swarm Optimization (IPSO) algorithm and the design of linear phase digital low pass FIR filter using Particle Swarm Optimization with Constriction Factor and Inertia Weight Approach (PSO-CFIWA).

Chapter 3 discusses about problem formulation for of linear, time-invariant FIR filter. In this chapter different error fitness functions is discussed and select the batter error fitness function to be minimizing using hybrid genetic algorithm and particle swarm optimization.

Chapter 4 discusses about the optimization techniques employed in this thesis. The techniques are genetic algorithm, particle swarm optimization and hybrid genetic algorithm and particle swarm optimization.

Chapter 5 contains the magnitude response result of simulation using MATLAB of low pass FIR filter of order 20 and 30 within a given specifications, followed by discussion of the obtained results.

CHAPTER TWO

2. LITERATURE REVIEW

Different kinds of approaches in modeling and designing of digital filter were used. The paper, [1], proposed and presented optimal design of linear phase digital lowpass finite impulse response (FIR) filter using Novel Particle Swarm Optimization (NPSO). NPSO is an improved particle swarm optimization (PSO) that proposes a new definition for the velocity vector and swarm updating and hence the solution quality is improved.

The inertia weight has been modified in the PSO to enhance its search capability that leads to a higher probability of obtaining the global optimal solution. The key feature of the presented modified inertia weight mechanism is to monitor the weights of particles, which were linearly, decreased in general applications, to avoid storing too many similar particles at the end of the optimization process.

Evolutionary algorithms like genetic algorithm (GA), particle swarm optimization (PSO), and the novel particle swarm optimization (NPSO) have been used in work for the design of linear phase FIR low pass (LP) filter. Comparison of the results of PM, GA, PSO, and NPSO algorithm has been made. The simulation results clearly indicate that NPSO demonstrates the best performance in terms of magnitude response, minimum stopband ripple and maximum stopband attenuation with the narrowest transition width [1].

Furthermore, in the literature, [19], presented a novel approach for designing a linear phase digital high pass FIR filter using Improved Particle Swarm Optimization (IPSO) algorithm.

Filter of order 20 has been realized using GA, PSO as well as with the proposed IPSO algorithm. Extensive simulation results justify that the proposed algorithm outperforms GA and classical PSO in the accuracy of the magnitude response of the filter as well as in the convergence speed.

Furthermore, [3] presented design of recursive digital filter which minimizes both the magnitude and group delay simultaneously under the multi-objective optimization. Butterworth low pass filter was designed by optimizing magnitude and group delay. Multi-Objective problem of magnitude and group delay was solved using Multi objective Genetic Algorithm that operates on a complex, continuous search space and optimized by statistically determining the abilities of commonly used genetic operators.

Multi-objective formulations are realistic models for many complex engineering optimization problems. In many real-life problems, objectives under consideration conflict with each other, and optimizing a particular solution with respect to a single objective can result in unacceptable results with respect to the other objectives. In most of the applications, particularly those in design, require the simultaneous optimization of more than one objective function, like printed wiring boards, control system design, the design of induction motors, analog circuits design, modern power systems and for pricing system security in electricity markets. It means to get a best optimum value for design variables some type of compromise is required to get a satisfactory design.

Genetic Algorithm is used to solve multi-objective problem and provide better results as compared to classical methods and other Evolutionary Algorithms. In the experimental study, GA gained better results than the compared state-of-the-art algorithms on Butterworth low and highpass filter design cases with remarkably lower computational cost. GA does not require any limitation of the design objectives or the designable filter class. Butterworth lowpass filter is flat in passband and rolls off to zero in stopband. GA can be implemented to various multidimensional and higher filter orders. Multi-Objective GA can be implemented to optimize bandpass and bandstop filters [3].

The work done in, [4], presented an efficient general technique for designing approximately linear phase complex coefficient FIR digital filters with arbitrary magnitude and group delay responses. The approach was based on a batch back-propagation neural networks algorithm by directly minimizing the real magnitude error and phase error from the linear-phase. The paper developed both the real and complex coefficient FIR digital filters design problems.

The approach is based on a batch back-propagation neural networks algorithm by directly minimizing the real magnitude error and phase error from the linear-phase. The paper develops both the real and complex coefficient FIR digital filters design problems. Several simulations were given and the results show that the proposed design method could achieve better solutions in the passband ripple, stopband attenuation and group delay error comparing with the weighted least squares (WLS) method and these semi-definite programming (SDP) method. Compared with the multiple criterion optimization approach, the filter designed by the proposed method achieves about 35% reduction in the passband ripple and in the group delay error at the expense of a slight small increase in the stopband ripple [4].

In the literature, [6], presented a novel population-based search technique is proposed for the design of FIR digital filters. The key idea behind cultural particle swarm optimization algorithms is to acquire problem-solving knowledge (beliefs) from the evolving in return making use of that knowledge to population and guiding the search.

Compared to quantum particle swarm optimization (QPSO) and the original PSO algorithm, cultural particle swarm optimization (CPSO) was able to converge to the global optima and the fitness was greatly improved. The simulation results of lowpass and bandpass filters demonstrate that CPSO is an efficient and alternative approach for FIR digital filters design and has higher optimizing precision and strong global search capability [6].

The paper, [7], presented an alternative approach for the design of linear phase digital lowpass FIR filter using Particle Swarm Optimization with Constriction Factor and Inertia Weight Approach (PSO-CFIWA).

In the paper, for the given problem, the realization of the FIR filters of different order had been performed. Filters of orders 20 and 30 have been realized using GA as well as PSOCFIWA. Extensive simulation results justify that the proposed algorithm outperforms GA in the accuracy of the magnitude response of the filter as well as in the convergence speed.

In the literature, [9], presented a lowpass, bandpass and highpass FIR filter is implemented on Field-Programmable gate Array (FPGA). Direct-form approach in realizing a digital filter is considered. This approach gives a better performance than the common filter structures in terms of speed of operation, cost, and power consumption in real-time. A fifty three-order lowpass and highpass and seventy-three order bandpass FIR filter is implemented in Spartan-III-xc3s500c-4fg320 FPGA. The Direct form structure of these filters is implemented. This approach gives a better performance than the common filter structures in terms of speed of operation, cost, and power consumption. In the Direct form structure, M Shift Registers, M Adders and $M+1$ multipliers are used to realize the M^{th} order lowpass, highpass and bandpass filter. The designed filters can work for real time processing of any digital signal.

In the paper,[10], presented a new design method of the FIR digital filters that approximate the magnitude and the group delay responses simultaneously. The proposed method solves only a least squares approximation while transforming the desired magnitude and group delay responses to obtain the equiripple characteristics. With this method, FIR digital filters can be easily

designed without a special optimization algorithm. In spite of a very simple algorithm, its results are better than the conventional methods.

CHAPTER THREE

3. PROBLEM FORMULATION

A digital FIR filter is characterized by,

$$H(z) = \sum_{n=0}^{M} h(n) z^{-n}, \tag{3.1}$$

where M is the order of the filter which has $(M+1)$ number of filter's impulse response coefficients, $h(n)$. The values of $h(n)$ are to be determined in the design process and M represents the order of the polynomial function. This research work presents the even order FIR LP filter design with $h(n)$ as even symmetric. Because the $h(n)$ coefficients are symmetrical, the dimension of the problem is halved. Thus, $(M/2+1)$ number of $h(n)$ coefficients are actually optimized, which are finally concatenated to find the required $(M+1)$ number of filter coefficients.

An ideal filter has a magnitude of one in the passband and a magnitude of zero in the stopband. Error fitness function is formed by the errors between the frequency responses of the ideal filter and the designed approximate filter. In each iteration of the optimization algorithm, error fitness values of particle vectors are calculated and used for updating the particle vectors with new coefficients $h(n)$. The final particle vector obtained after a certain number of iterations or after the error fitness is below a certain limit is considered to be the optimal result, yielding an optimal filter. Various filter parameters which are responsible for the optimal filter design are stopband and passband normalized frequencies (ω_s, ω_p) passband and stopband ripples (δ_s, δ_p), stopband attenuation and transition width.

In this research work, evolutionary optimization algorithms like GA, PSO and hybrid GA and PSO are individually applied to obtain the actual designed filter response as close as possible to the ideal response.

Now from Eq.(3.1), the number of particle i.e. the number of filter coefficient vector $\{h_0, h_1, \dots, h_M\}$, which is to be optimized is represented in $(M/2+1)$ dimension instead of $(M+1)$ dimension.

The frequency response of the FIR digital filter can be calculated as,

$$H\left(e^{j\,\omega_k}\right) = \sum_{n=0}^{M} h(n)\, e^{-j\,\omega_k n} \tag{3.2}$$

where $\omega_k = \frac{2\pi k}{M}$; $H(e^{j\,\omega_k})$ is the Fourier transform complex vector. This is the FIR filter's frequency response. This is the FIR filter's frequency response. The frequency is sampled in $[0, \pi]$ with M points.

Different kinds of error fitness functions have been used in different literatures. An error function given by Eq.(3.3) is the approximate error function used in PM algorithm for filter design [14].

$$E(\omega) = W[H_d\left(e^{j\,\omega}\right) - H_i(e^{j\,\omega})] \tag{3.3}$$

Where $H_d\left(e^{j\,\omega}\right)$ is the frequency response of the designed approximate filter; $H_i(e^{j\,\omega})$ is the frequency response of the ideal filter; W is the weighting function used to provide different weights for the approximate errors. For ideal LP filter, $H_i(e^{j\omega})$is given as,

$$H_i\left(e^{j\,\omega_k}\right) = \begin{cases} 1, & for \;\; 0 \le \omega_k \le \omega_c \\ 0, & otherwise \end{cases} \tag{3.4}$$

where ω_c is the cut-off frequency The major drawback of PM algorithm is that the ratio of $\delta_p / \delta_s$is fixed. To improve the flexibility in the error function to be minimized, so that the desired level of δ_p and $\delta_s$may be specified. The error function(J_1) given in Eq.(3.5) has been considered as fitness function in many literatures [15] [16]. The error fitness to be minimized using the evolutionary algorithms, is defined as:

$$J_1 = \max_{\omega \le \omega_p}\left(|E(\omega)| - \delta_p\right) + \max_{\omega \ge \omega_s}(|E(\omega)| - \delta_s) \quad (3.5)$$

where δ_p and δ_s are the ripples in the passband and stopband, respectively, and ω_p and ω_s are passband and stopband normalized cut-off frequencies, respectively.

An error fitness function (J_2) given by Eq.(3.6) has been adopted for digital FIR filter in order to achieve higher stopband attenuation and to have better control on the transition width [1].In this research, an error fitness function given by Eq. (3.6) has been used to find the optimal filter coefficients. By using Eq.(3.2), it is found that the proposed filter design approach results inconsiderable improvement over the PM and other optimization techniques [1].

$$J_2 = \sum abs[abs\,(|H_d(\omega)| - 1) - \delta_p] + \sum abs\,[(|H_d(\omega)| - \delta_s)] \quad (3.6)$$

For the first term of Eq.(3.6), $\omega \in$ passband including a portion of the transition band and for the second term of Eq.(3.6), $\omega \in$stopband including the rest portion of the transition band. The portions of the transition band chosen depend on passband edge and stopband edge frequencies.

The error fitness function given in Eq.(3.6) represents the generalized fitness function to be minimized using the evolutionary algorithms GA, conventional PSO, and the proposed hybrid PSO and GA individually. Each algorithm tries to minimize this error fitness J_2 and thus optimizes the filter performance. Unlike other error fitness functions which consider only the maximum errors, J_2 involves summation of all absolute errors for the whole frequency band, and hence, minimization of J_2yields much higher stopband attenuation and lesser stopband ripples. Transition width is also kept reduced. Since the coefficients of the linear phase filter are matched, the dimension of the problem is halved. This greatly reduces the computational burdens of the algorithms, applied to the optimal design of linear phase even symmetrical FIR filters.

CHAPTER FOUR

4. OPTIMIZATION TECHNIQUES

4.1. Genetic Algorithm (GA)

GAs are stochastic search methods that can be used to search for an optimal solution to the evolution function of an optimization problem Holland proposed in the early seventies [28] as computer programs that mimic the natural evolutionary process.

GAs differ from classical optimization and search methods in several respects. Rather than focusing on a single solution, GAs operate on group of trial solutions in parallel where they manipulate a population of individuals in each generation (iteration) where each individual, termed as the chromosome, represents one candidate solution to the problem. Within the population, fit individuals survive to reproduce and their genetic materials are recombined to produce new individuals as offspring. The genetic material is modeled by some finite-length data structures. As in nature, selection provides the necessary driving mechanism for better solutions to survive. Each solution is associated with a fitness value that reflects how good it is compared with other solutions in the population. The recombination process is simulated through a crossover mechanism that exchanges portions of data strings between chromosomes. New genetic material is also introduced through mutation that causes random alterations of the strings. The frequency of occurrence of these genetic operations is controlled by certain pre-set probabilities. The selection, crossover, and mutation processes constitute the basic GA cycle or generation, which is repeated until some pre-determined criteria are satisfied. Through this process, successively better and better individuals of the species are generated.

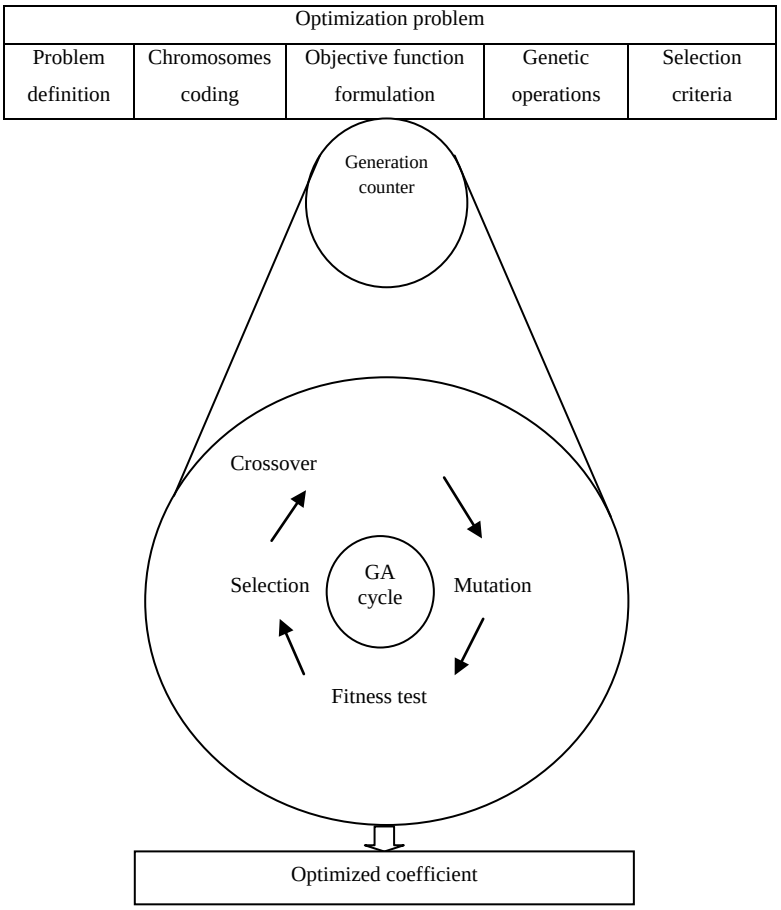


Figure 4.1 Conceptual representation of the optimization process through a genetic algorithm [36].

4.1.1 Chromosome Representation

In the most basic form, GA works as a function optimizer with a given objective function

$$minimize\ f(\mathrm{a}) \qquad \text{where } a\ = [\ a_1\ a_2\ \dots\ a_M]^T \tag{4.1}$$

In general, GAs operates on a symbolic representation of the design variables known as a chromosome. This requires an encoding function of the form

$$T:\ S_a\ \rightarrow X \tag{4.2}$$

to map the solution space S_a of the problem onto the chromosome space X [29]. By analogy with the biological terminology, the encoded chromosomes are called the genotype representation and the corresponding solutions in the search space are called the phenotype representation. A

chromosome x is the encoded version, i.e., genotype, of a solution with phenotype *a* and they are related by

$$a = T(X) \tag{4.3}$$

where

$$X = [x_1, x_2, \dots, x_M]^T \tag{4.4}$$

Each element x_i in a chromosome *X* is often referred to as a *gene*

4.1.2. Encoding Schemes

GAs use various encoding schemes, such as, binary encoding, integer encoding, and decimal encoding. In binary encoding scheme, each variable is encoded into a bit string of predefined length whereas integers are used as the elements of chromosome vectors in integer encoding. The elements of chromosome vectors are represented using decimal numbers in decimal encoding. This scheme is also called real encoding. The choice of encoding is the most important factor in designing a genetic algorithm. The encoding has profound implications on the performance of the GA. Several strategies have been suggested for selecting an encoding scheme but until there is more rigorous theory on GAs and the different encodings, the best strategy seems to be to choose an encoding that is naturally suited for the problem at hand and then design a genetic algorithm that can handle this encoding [30]. For example, binary encoding is useful if the variables of the problem are discrete whereas decimal encoding might be necessary when high-precision is required.

4.1.3. Population Initialization

The initial population of candidate solutions is usually generated randomly across the search space. However, domain-specific knowledge or other information can be easily incorporated. Once an initial population is created, the main GA cycle can begin.

4.1.4. Fitness Function

The GA produces a succession of populations whose members will have generally improving adaptability to the environment. In order to drive the search of the GA, the fitness levels of the individuals in the population is evaluated by using a fitness function.

The fitness function is usually an objective or cost function but anything will suffice as long as it can successfully quantify the quality of all possible phenotype solutions. The fitness function is dependent on the environment and application of the system that is undergoing the genetic search process, and it is the only connection between the physical problem being optimized and the genetic algorithm itself [31].

Given a population P^t at generation t, the GA iteration starts by evaluating the set

$$F^t = \{ f(1)^t, f(2)^t, \quad \dots \ , f(M)^t \tag{4.5}$$

where

F^t is fitness value of one chromosome at generation t

M is the number of gen in one chromosome

$f(i)^t$ is the fitness value of the gen with $i = 1, 2, \dots, M$

of objective function values associated with the chromosomes $\{X(i)^t\}$ with $i = 1, 2, \dots, M$. The GA then applies the genetic operators and selection to produce population P^{t+1} for the next generation.The objective function for GAs is formulated as in classical optimization algorithms. The mathematical structure of these algorithms is simple and flexible.

4.1.5. Genetic Operators

Evolution from generation to generation is simulated by preserving, redistributing or altering genetic material contained in the chromosome strings of fit individuals. These basic functionalities of the genetic algorithm are provided by the genetic operators. The basic GA operators are crossover and mutation, constitute the main algorithm whereas the population and fitness function can be viewed as external entities. Both crossover and mutation are probabilistic operations and their frequencies of occurrence are controlled by predefined probabilities, p_x and p_m, respectively. As crossover plays the key role in improving the solution, it is assigned a high frequency of occurrence.

Crossover

Crossover recombines genetic material from selected individuals to form one or more offspring where some of the useful traits of the parents are preserved. The goal is to generate new chromosomes that are more fit than their ancestors, thereby contributing to the overall convergence of the population. There are many ways of performing crossover. One-point, two-point, or uniform crossover is used with binary encoding.

During a one-point crossover, two individuals $X(1) = [x_1 x_2 \dots x_M]^T$ and $X(2) = [x_1' x_2' \dots x_M']^T$ selected undergo crossover. Parts of the stings from each individual are swapped at the same location called the crossover point to create two offspring chromosomes $X^c(1)$ and$X^c(2)$ as follow

$$X^c(1) = \left[x_1 x_2 \dots x_i \, x_{i+1}' \dots x_M'\right] \tag{4.6}$$

$$X^c(2) = [\, x_1' \, x_2' \dots x_i' x_{i+1} \dots x_M]$$

The crossover point $i$in Eq. (4.6) is chosen randomly from a set of integers

$$I = \{\, i \in \boldsymbol{R} : 1 \leq i \leq M - 1 \,\}$$

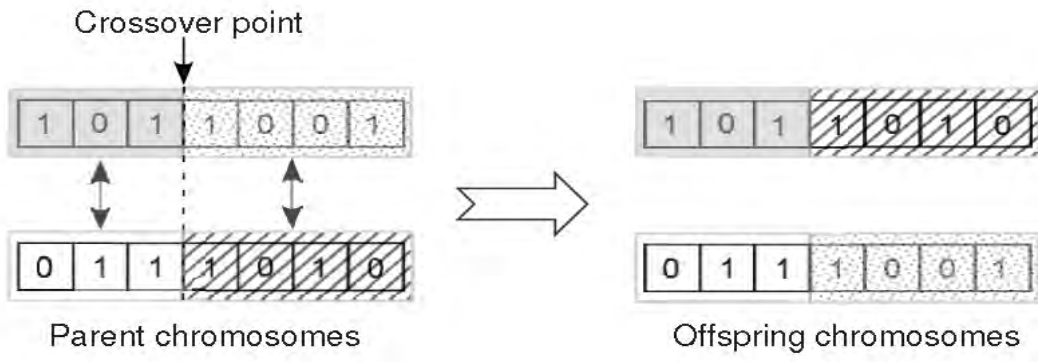


Figure 4.2A typical one-point crossover in binary representation.

Mutation

Mutation randomly changes an offspring after crossover. Mutation is treated as supporting operator for the purpose of restoring lost genetic material. Bit flip mutation is the most common mutation operator for binary-encoded GAs. This is realized by simply inverting one or more bits in the chromosome string based on the probability of mutation.

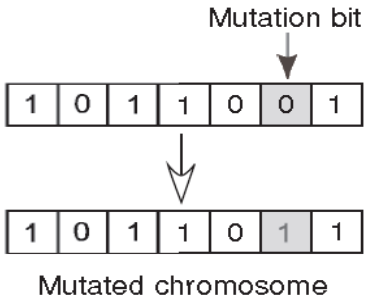


Figure 4.3 Mutation operations in binary representation

4.1.6. Selection Methods

The process of natural section is simulated to achieve a selection mechanism in the GAs. It essentially defines how the algorithms update the population from one generation to the next. In general, chromosomes x are selected from the population based on the requirements imposed on the solutions in terms of objective functions F^t in order to create a new population on the principle of the “survival of the fittest". This can be achieved by using many different selection methods but the most commonly used methods are roulette-wheel, tournament, ranking, and steady-state selection.

In roulette-wheel selection each individual's probability of being selected in the next population is proportional to its fitness value. The probability of survival $p_s(i)$ of a chromosome $x^t(i)$ is calculated by using the normalized fitness value [36].

$$p_s(i) = \frac{f^t(i)}{\sum_{i=1}^{M} f^t(i)} \tag{4.7}$$

with

$$\sum_{i=1}^{M} p_s(i) = 1 \tag{4.8}$$

Since the probability of selection is based on the fitness proportion in the population, this method is also referred to as proportionate selection method.

Rank selection involves ranking the individuals from `best' to `worst' on the basis of their measured fitness values. The fitness rank is used to determine the probability of survivalp_s. New fitness values that are inversely related to their ranking are then assigned to the individuals.

In tournament selection, a group of individuals are chosen iteratively from P^t by holding a tournament and the one with the best fitness value is chosen for P^{t+1} until it is filled with a predetermined number of individuals. The tournament size is typically set to a pair of individuals but it can be up to five.

The steady-state selection method employs a deterministic section procedure. In this method, most of the individuals survive and only a fraction of the population is updated in every generation. A fixed number, say, N_s, of new individuals are created and added to the population of N_p. Then the $N_p + N_s$ chromosomes are sorted according to their fitness values, the least fit N_s chromosomes are discarded and the rest survive to a new generation.

The configurations of GAs are very problem specific. The success of any genetic algorithm largely depends on how well it has been customized for a given application. The customization can be done by choosing proper objective function(s), chromosome encoding scheme, genetic operators, and section methods. Beside these parameters, there are other parameters and conditions that also affect the performance of a GA.

Population size $\mathrm{N_p}$, crossover and mutation probabilities p_x and p_m, respectively, and termination criteria play significant role in a GA's convergence. In spite of many attempts to find the optimal parameter values, systematic trials for specific problems remain the most accepted norm in configuring a GA.

The basic concept and configuration of GAs in general have been introduced in this section. However, GAs does not necessarily follow any strict configuration or specific guidelines. As a result, the number of GA variants with different configurations that exit today is overwhelming. This also brings enormous possibilities in the optimization domain whereby problems which were not within the scope of any existing method can now be solved. With the increasing computing power offered by advancements in integrated circuit technology, the simulation of

evolutionary systems is becoming more and more tractable and GAs are being applied to many real world problems including the design of digital filters.

The steps of GA as implemented for the optimization of $h(n)$ coefficients are [1].

- Initialization of real chromosome strings $h(n)$ of$P$ population, each consisting of a set of $h(n)$ coefficients. Size of the set depends on the number of coefficients in a particular filter design.
- Decoding of strings and evaluation of error fitness value of each string.
- Selection of elite strings in order of increasing error fitness values from the minimum value.
- Copying of the elite strings over the non-selected strings.
- Crossover and mutation to generate off-springs.
- Genetic cycle updating and repeat from the step of evaluation error fitness value of each string.
- The iteration stops when the maximum number of genetic cycles is reached. The grand minimum error fitness value and its corresponding chromosome string or the desired optimal solution vector is finally obtained

4.2. Particle Swarm Optimization (PSO)

Particle swarm optimization (PSO) was originally designed and introduced by Eberhart and Kennedy. The PSO is a population based search algorithm based on the simulation of the social behavior of birds, bees or a school of fishes. Each individual within the swarm is represented by a vector in multidimensional search space. This vector has also one assigned vector which determines the next movement of the particle and is called the velocity vector. The PSO algorithm also determines how to update the velocity of a particle. Each particle updates its velocity based on current velocity and the best position it has explored so far; and also based on the global best position explored by swarm.

The PSO process then is iterated a fixed number of times or until a minimum error based on desired performance index is achieved. It has been shown that this simple model can deal with difficult optimization problems efficiently. The PSO was originally developed for real-valued spaces but many problems are, however, defined for discrete valued spaces where the domain of the variables is finite. In binary PSO, each particle represents its position in binary values which

are 0 or 1. Each particle's value can then be changed (or better say mutate) from one to zero or vice versa. In binary PSO the velocity of a particle defined as the probability that a particle might change its state to one [33].

The PSO formulae define each particle as potential solution to a problem in D dimensional space. The position of particle i is represented as.

$$X_i = (x_{i1}, x_{i2}, \dots\dots\dots\dots\ x_{iD}) \tag{4.9}$$

Each particle also maintains a memory of its previous best position, represented as

$$P_i = (p_{i1}, p_{i2}, \dots\dots\dots\dots\ p_{iD}) \tag{4.10}$$

A particle in a swarm is moving; hence, it has a velocity, which can be represented as

$$V_i = (v_{i1}, v_{i2}, \dots\dots\dots\dots\ v_{iD}) \tag{4.11}$$

Each particle knows its best value so far ($pbest$) and its position. Moreover, each particle knows the best value so far in the group (g$best$) among *pbests*. This information is analogy of knowledge of how the other particles around them have performed. Each particle tries to modify its position using the following information:

• The distance between the current position and the $pbest$

• The distance between the current position and the g$best$

This modification can be represented by the concept of velocity. Velocity of each agent can be modified by the following equation in inertia weight approach (IWA) [1]

$$V_i^{(k+1)} = w \times V_i^k + c_1 \times rand_2 \times (P_i^k - X_i^k) + c_2 \times rand_2 \times (P^k - X_i^k) \tag{4.12}$$

where V_i^k is the velocity of i^{th} particle vector at k^{th} iteration;$w$is called as the inertia factor which controls the influence of previous velocity on the new velocity; $c_1$is a positive constant, called as coefficient of the self-recognition component, c_2 is a positive constant, called as coefficient of the social component.,$rand_1$and $rand_2$are the random numbers, which are used to

maintain the diversity of the population, and are uniformly distributed in the interval [0,1]; X_i^k is thecurrent position of i^{th} particle vector at k^{th} iteration; P_i^k is the personal best of the i^{th} particle at the k^{th} iteration; P^k is the group bestof the group at the k^{th}iteration.

From Eq.(4.12), a particle decides where to move next, considering its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm. In the particle swarm model, the particle searches the solutions in the problem space.
The following inertia factor is usually utilized in [1]

$$w = w_{Max} - (w_{Max} - w_{min}) \frac{iter}{iter_{Max}} \tag{4.13}$$

where
w_{Max} −initial weight,
w_{min} −final weight,
$iter_{Max}$ −maximum iteration number,
$iter$ −current iteration number.

Using the above equation, diversification characteristic is gradually decreased and a certain velocity, which gradually moves the current searching point close to $pbest$ and g$best$ can be calculated.
The searching point in the solution space may be modified by (4.14).

$$X_i^{(k+1)} = X_i^k + V_i^{(k+1)} \tag{4.14}$$

The first term of Eq. (4.12) is the previous velocity of the particle vector. The second and third terms are used to change the velocity of the particle vector. Without the second and third terms, the particle vector will keep on ‘‘flying’’ in the same direction until it hits the boundary. Namely, it corresponds to a kind of inertia represented by the inertia constant,$w$and tries to explore new areas. The steps of conventional PSO as employed for FIR LP filter design is given
Step 1: Initialization: Population (swarm size) of particle vectors, P =10; maximum iteration cycles=800;filter order, $nvar$ =20; c_1 , c_1 =2.05; minimum and maximum values of

filtercoefficients, $h_{min} = -1$, $h_{max} = 1$; number of samples = 100; $\delta_p = 0.1$, $\delta_s = 0.01$; initialization of the velocities of all the particle vectors.

Step 2: Generate initial particle vectors of filter coefficients ($nvar/2 + 1$) randomly with limits; Computation of initial error fitness values of the total population, n_p from Eq. (4.12).

Step 3: Compare the fitness of each particle vector with each $pbest$. If the current solution is better than its $pbest$, then replace it $pbest$ by the current solution.

Step 4: Compare the fitness of each particle vector with each g$best$. If the fitness of any particle vector is better than it g$best$, then replace it by g$best$.

Step 5: Update the velocity and position of all particle vectors according to Eq. (4.13) and Eq. (4.14), respectively.

Step 6: Repeat steps 2-5 until the maximum iteration cycles or the convergence of minimum error fitness values are met; finally, g$best$ is the particle vectorof optimal filter coefficients ($nvar/2 + 1$); Formcomplete ($nvar + 1$) coefficients by copying (becausethe filter has linear phase) before getting the optimal frequency spectrum.

4.3. Hybrid of GA and PSO (HGAPSO)

The drawback of PSO is that the swarm may prematurely converge. The underlying principle behind this problem is that, for the global best PSO, particles converge to a single point, which is on the line between the global best and the personal best positions. This point is not guaranteed for a local optimum. Another reason for this problem is the fast rate of information flow between particles, resulting in the creation of similar particles with a loss in diversity that increases the possibility of being trapped in local optima [33].

A further drawback is that stochastic approaches have problem-dependent performance. This dependency usually results from the parameter settings in each algorithm. The different parameter settings for a stochastic search algorithm result in high performance variances. In general, no single parameter setting can be applied to all problems. Increasing the inertia weight (w) will increase the speed of the particles resulting in more exploration (global search) and less exploitation (local search) or on the other hand, reducing the inertia weight will decrease the speed of the particles resulting in less exploration of searching space. Thus finding the best value for the parameter is not an easy task and it may differ from one problem to another. Therefore, from the above, it can be concluded that the PSO performance is problem-dependent. The

problem-dependent performance can be addressed through hybrid mechanism. It combines different approaches to be benefited from the advantages of each approach [33].

To overcome the limitations of PSO, hybrid algorithms with GA are proposed. The basis behind this is that such a hybrid approach is expected to have merits of PSO with those of GA. One advantage of PSO over GA is its algorithmic simplicity. Another clear difference between PSO and GA is the ability to control convergence. Crossover and mutation rates can subtly affect the convergence of GA, but these cannot be analogous to the level of control achieved through manipulating of the inertia weight. In fact, the decrease of inertia weight dramatically increases the swarm's convergence. The main problem with PSO is that it prematurely converges to stable point, which is not necessarily maximum. To prevent the occurrence, position update of the global best particles is changed. The position update is done through some hybrid mechanism of GA. The idea behind GA is due to its genetic operators such as crossover and mutation. By applying crossover operation, information can be swapped between two particles to have the ability to fly to the new search area. The purpose of applying mutation to PSO is to increase the diversity of the population and the ability to have the PSO to avoid the local maxima [33].

In this research, the hybrid of GA and PSO named HGAPSO is used for optimal design of FIR lowpass filter. This algorithm consists of four major operators: enhancement, selection, crossover, and mutation. The flowchart of the HGAPSO is shown in Fig 4.4.

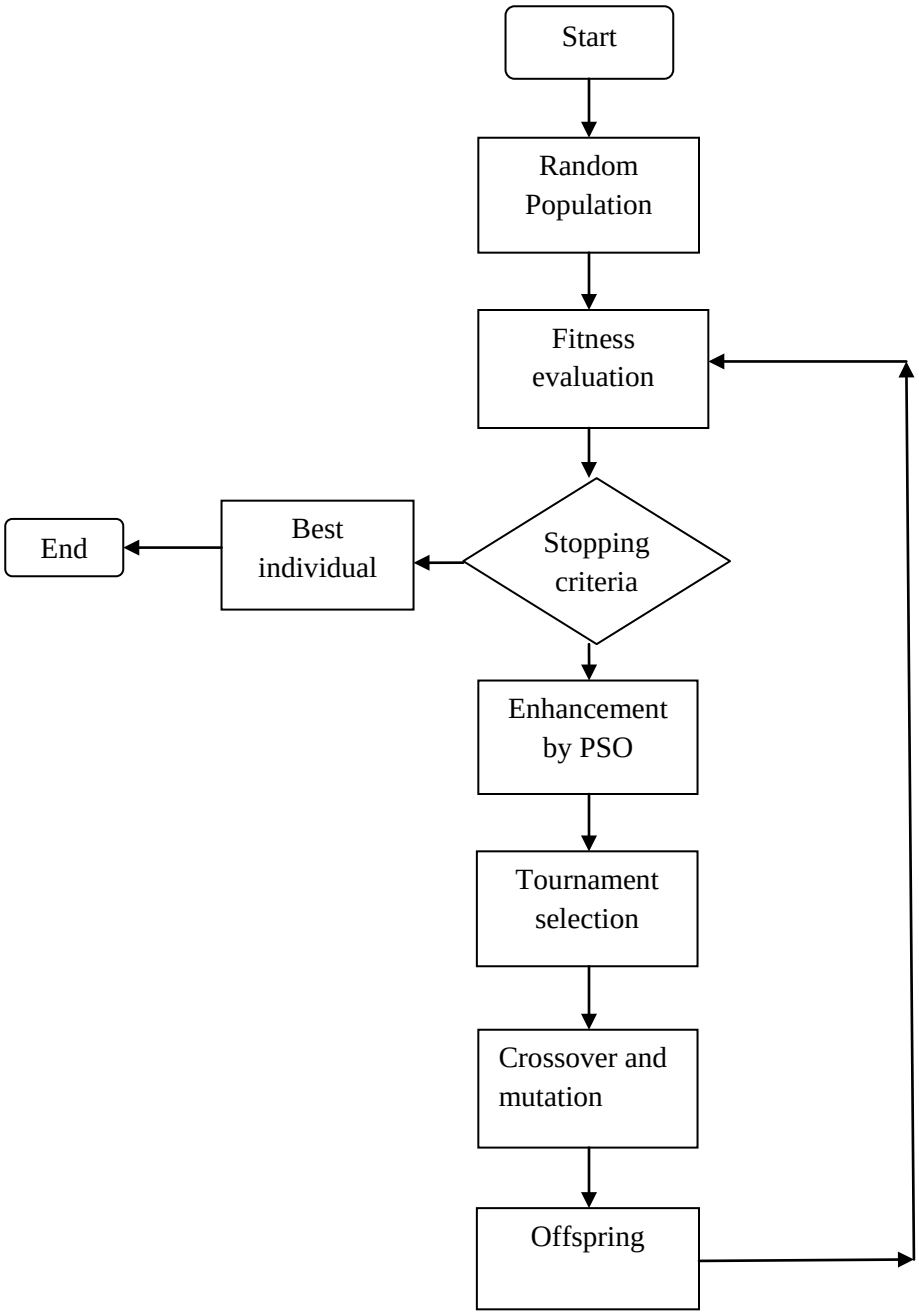


Fig. 4.4 The flow chart of the HGAPSO

CHAPTER FIVE

5. RESULTS AND DISCUSSIONS

5.1. Analysis of Magnitude Response of Lowpass FIR Filter

This part of the research presents the simulations undertaken in the MATLAB. The MATLAB simulation has been performed extensively to realize lowpass FIR filters with orders of 20 and 30, which result in the number of coefficients as 21 and 31. Each algorithm is run for 40 times to obtain its best results.

In this research population size is selected to be 10 because if the population size is large the algorithm takes much time to get optimal solution but not much improvement on solution. The iteration cycle is 800 because increasing the iteration cycle more than 800 has no much significant effect on the solution. The selection method used for GA and HGAPSO in this research is tournament selection; the reason is tournament selection uses error fitness value to select the fittest chromosome (parents) to undergo crossover.

Table 5.1 shows the best chosen parameters for GA, PSO, and hybrid GA and PSO, respectively

Parameters	GA	PSO	HGAPSO
Population size	10	10	10
Iteration Cycle	800	800	800
Crossover rate	1	-	1
Mutation rate	0.02	-	0.02
Selection Probability	0.5	-	0.5
Selection	Tournament	-	tournament
C_1	-	2.00	2.00
C_2	-	2.00	2.00
w_{Max}	-	1	1
w_{min}	-	0.4	0.4

The parameters of the filter to be designed using GA, PSO and hybrid GA and PSO are: passband ripple (δ_p) = 0.1 (normalized), stopband ripple (δ_s) = 0.01 (normalized). For the LP

filter, passband (normalized) edge frequency(ω_p) = 0.45; stopband (normalized) edge frequency (ω_s) = 0.55; transition width=0.1 and the filter has orders 20 and 30.

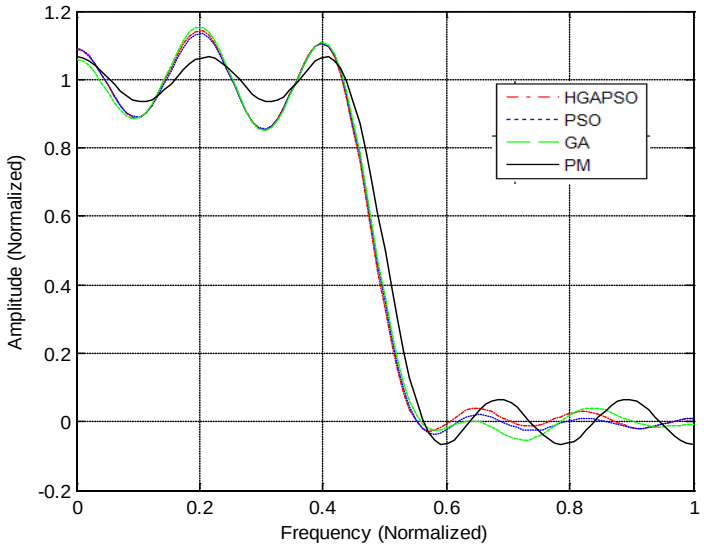


Fig 5.1 Normalized amplitude plots for the FIR LP filter of order 20

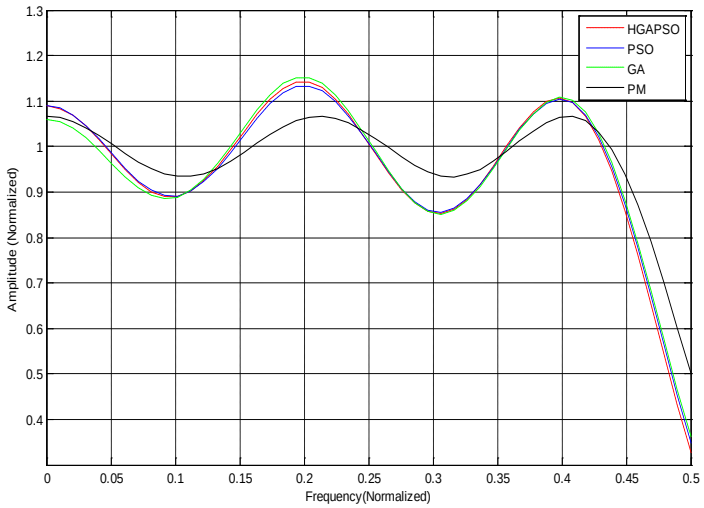


Fig 5.2 Normalized amplitude Pass band ripple plots for the FIR LP filter of order 20

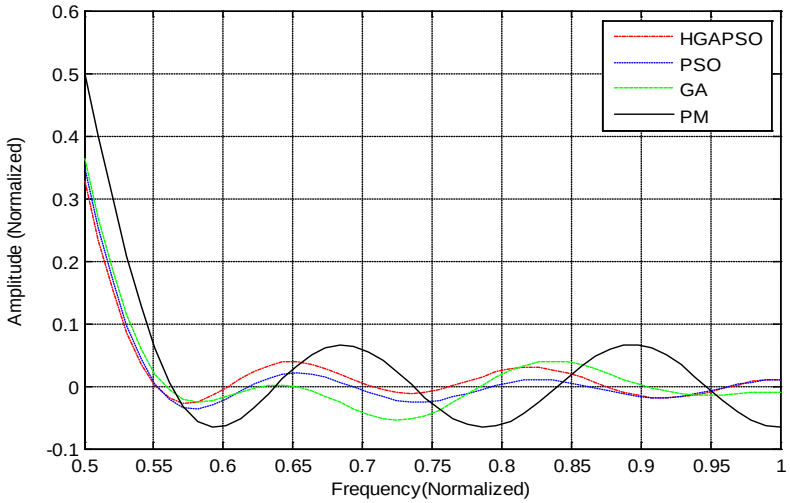


Fig 5.3 Normalized amplitude stop band ripple plots for the FIR LP filter of order 20

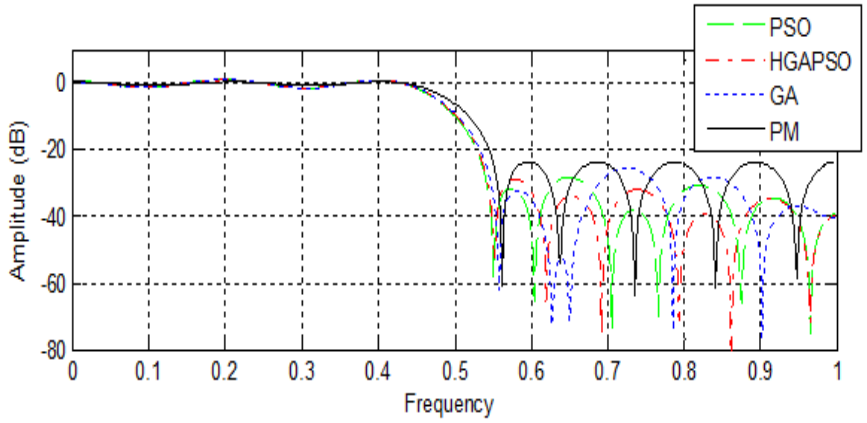


Fig 5.4 dB plots for the FIR LP filter of order 20

Table 5.2 Optimized coefficients of the FIR LP filter of order 20

h(n)	GA	PSO	HGAPSO
h(1)=h(21)	0.0362	0.0349	0.0379
h(2)=h(20)	0.0432	0.0432	0.0427
h(3)=h(19)	-0.0101	-0.0069	-0.0093
h(4)=h(18)	-0.0396	-0.0433	-0.0429
h(5)=h(17)	-0.0066	0.0034	0.0043
h(6)=h(16)	0.0526	0.0538	0.0532
h(7)=h(15)	-0.0104	-0.0105	-0.0151
h(8)=h(14)	-0.1058	-0.1010	-0.0981
h(9)=h(13)	0.0111	0.0122	0.0134
h(10)=h(12)	0.3169	0.3174	0.3147
h(11)	0.4850	0.4841	0.4877

Table 5.3 Other comparative results of performance parameters of all algorithms for the FIR LP filter of order 20

FIR LP filter of order 20				
Algorithm	Transition width (normalized)	Maximum stopband ripple (normalized)	Maximum passband ripple (normalized)	Maximum stopband Attenuation (dB)
PM	0.102	0.06594	0.066	23.617
GA	0.0918	0.03981	0.151	28.000
PSO	0.0908	0.03449	0.133	29.246
HGAPSO	0.0857	0.02625	0.141	31.617

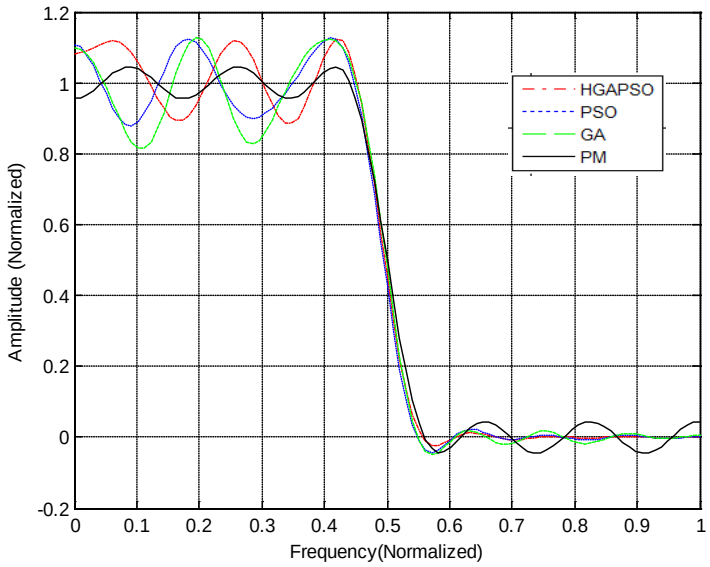


Fig 5.5 Normalized amplitude plots for the FIR LP filter of order 30

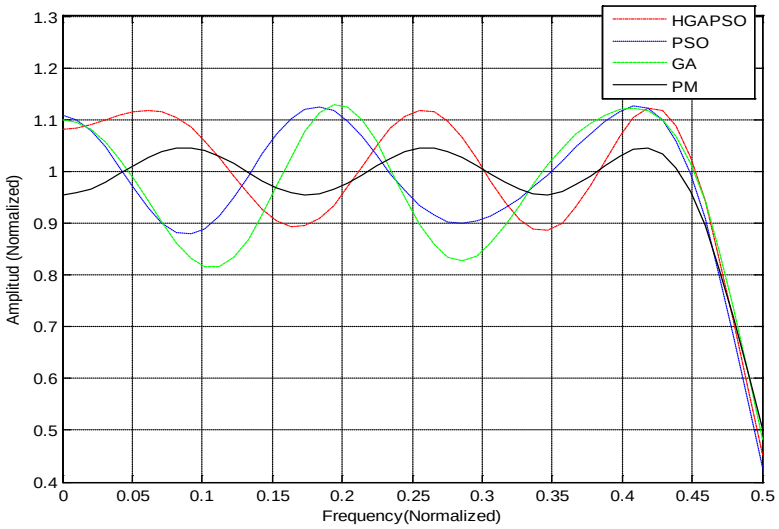


Fig 5.6 Normalized amplitude Pass band ripple plots for the FIR LP filter of order 30

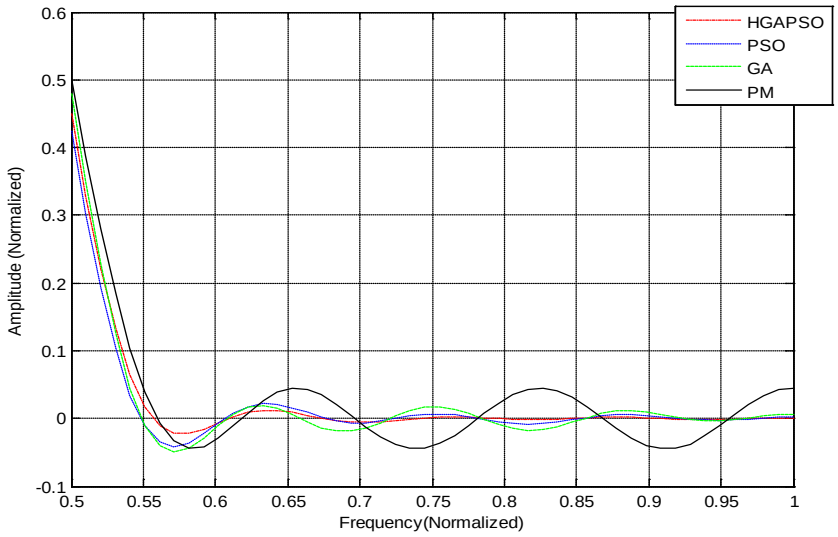


Fig 5.7 Normalized amplitude Stop band ripple plots for the FIR LP filter of order 30

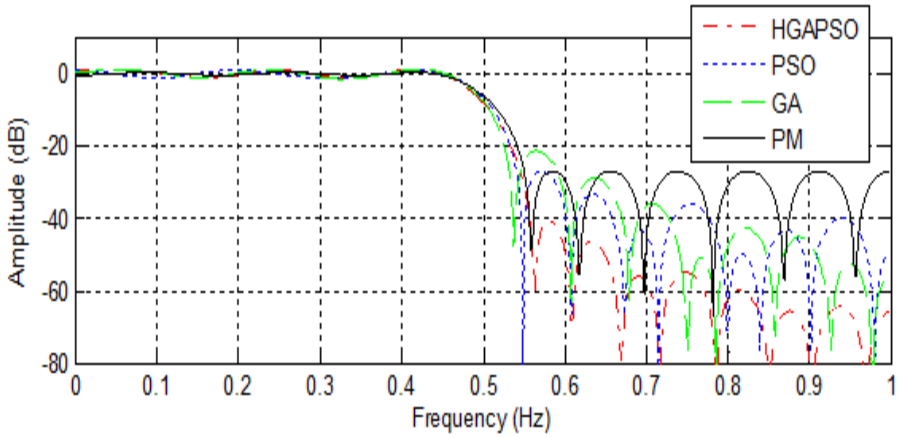


Fig 5.8 dB plots for the FIR LP filter of order 30

Table 5.4 Optimized coefficients of the FIR LP filter of order 30

h(n)	GA	PSO	HGAPSO
h(1)=h(31)	-0.014	-0.0050	-0.0008
h(2)=h(30)	-0.0071	0.0040	0.0049
h(3)=h(29)	0.011	0.0159	0.0032
h(4)=h(28)	0.0059	0.0053	-0.0224
h(5)=h(27)	0.0043	-0.0008	-0.0356
h(6)=h(26)	0.0321	0.0274	0.0022
h(7)=h(25)	0.0502	0.0433	0.0397
h(8)=h(24)	0.0010	-0.0073	0.0096
h(9)=h(23)	-0.0489	-0.0513	-0.0325
h(10)=h(22)	-0.0003	-0.0004	0.0080
h(11)=h(21)	0.0680	0.0620	0.0678
h(12)=h(20)	0.0057	-0.0021	0.0068
h(13)=h(21)	-0.1067	-0.1055	-0.0971
h(14)=h(21)	-0.0064	0.0015	0.0075
h(15)=h(21)	0.3095	0.3176	0.3255
h(16)	0.4914	0.4980	0.5079

Table 5.5 Other comparative results of performance parameters of all algorithms for the FIR LP filter of order 30

FIR LP filter of order 30				
Algorithm	Transition width (normalized)	Maximum stopband ripple (normalized)	Maximum passband ripple (normalized)	Maximum stopband attenuation (dB)
PM	0.102	0.04492	0.045	26.95
GA	0.0816	0.05018	0.129	25.989
PSO	0.0816	0.04221	0.126	27.492
HGAPSO	0.0765	0.0099	0.118	40.087

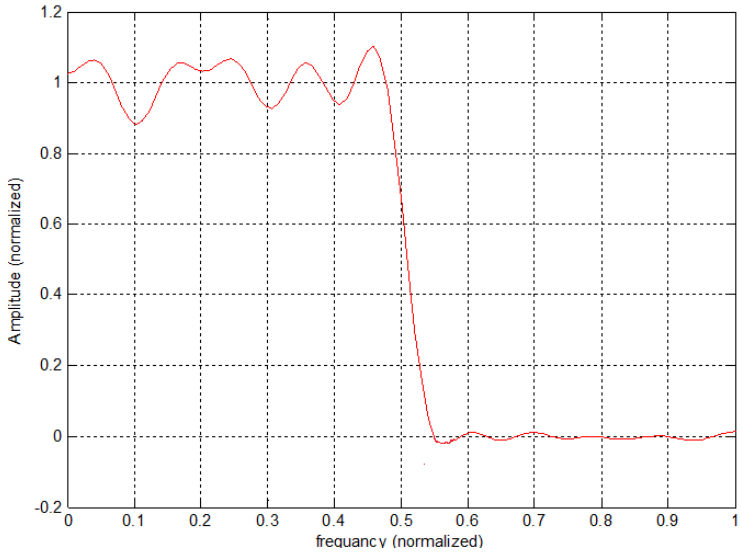


Fig 5.9 Normalized amplitude plots for HGAPSO FIR LP filter of order 40

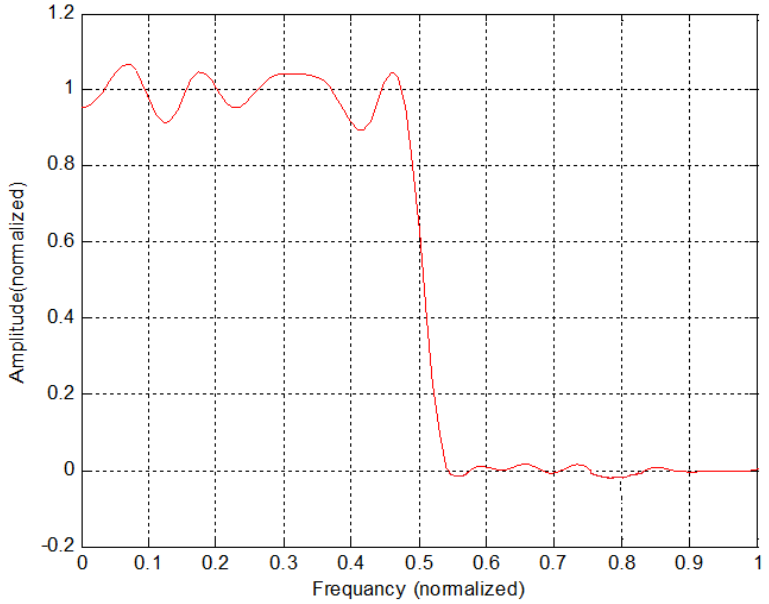


Fig 5.10 Normalized amplitude plots for HGAPSO FIR LP filter of order 50

Table 5.6 Optimized coefficients of the FIR LP filter of orders 40 and 50 using HGAPSO

h(n)	HGAPSO order 40
h(1)=h(41)	-0.0065
h(2)=h(40)	-0.0186
h(3)=h(39)	-0.0062
h(4)=h(38)	0.0135
h(5)=h(37)	0.0064
h(6)=h(36)	-0.0190
h(7)=h(35)	-0.0094
h(8)=h(34)	0.0189
h(9)=h(33)	0.0126
h(10)=h(32)	-0.0170
h(11)=h(31)	0.0011
h(12)=h(30)	0.0422
h(13)=h(29)	0.0177
h(14)=h(28)	-0.0405
h(15)=h(27)	-0.0104
h(16)=h(26)	0.0598
h(17)=h(25)	0.0096
h(18)=h(24)	-0.1058
h(19)=h(23)	-0.0113
h(20)=h(22)	0.3199
h(21)	0.5119

h(n)	HGAPSO order 50
h(1)=h(51)	0.0029
h(2)=h(50)	0.0037
h(3)=h(49)	-0.0035
h(4)=h(48)	0.0032
h(5)=h(47)	0.0107
h(6)=h(46)	-0.0018
h(7)=h(45)	-0.0133
h(8)=h(44)	-0.0100
h(9)=h(43)	0.0075
h(10)=h(42)	-0.0020
h(11)=h(41)	-0.0279
h(12)=h(40)	-0.0089
h(13)=h(39)	0.0205
h(14)=h(38)	0.0107
h(15)=h(37)	0.0240
h(16)=h(36)	-0.0140
h(17)=h(35)	0.0295
h(18)=h(34)	0.0076
h(19)=h(33)	-0.0363
h(20)=h(32)	-0.0010
h(21)=h(31)	0.0576
h(22)=h(30)	0.0057
h(23)=h(29)	-0.1044
h(24)=h(28)	-0.0064
h(25)=h(27)	0.3182
h(26)	0.5020

Table 5.7 Other comparative results of performance parameters of HGAPSO for the FIR LP filter of orders 20, 30, 40 and 50

HGAPSO	Maximum stopband ripple (normalized)	Maximum passband ripple (normalized)	Maximum stopband attenuation (dB)
Order 20	0.02625	0.141	31.617
Order 30	0.0099	0.118	40.087
Order 40	0.00995	0.103	40.04
Order 50	0.00993	0.067	40.06

Tables 5.3 & 5.5 show the comparative results of filter orders 20 and 30, respectively, and the performance parameters in terms of maximum stopband ripple (normalized), maximum passband ripple (normalized), maximum stopband attenuation and transition width (normalized) for LP filter using PM, GA, PSO, and HGAPSO, respectively.

Tables 5.3 & 5.5 show the comparison of the maximum stopband attenuations achieved for LP filter of orders 20 and 30 using PM, GA, PSO, HGAPSO, respectively. The maximum stopband attenuation achieved for the LP filter using the HGAPSO is 31.617 dB and 40.087dB respectively. It is observed from Tables 5.3 & 5.5 that HGAPSO achieves the best stopband attenuation, as compared to those of PM, GA and PSO.

Tables 5.3 & 5.5 also summarizes maximum passband ripple (normalized) for LP FIR filter with orders 20 and 30 using PM, GA, PSO, and the HGAPSO, respectively. From Tables 5.3 & 5.5, it is observed that the maximum passband ripple (normalized) obtained using HGAPSO is 0.141 and 0.118 for orders 20 and 30 respectively. But the HGAPSO has maximum passband ripple and is less efficient for orders 20 than 30 compared to GA, PSO and PM. This indicates that HGAPSO is more efficient for higher order. PM has a minimum passband ripple but has maximum stopband ripple compared to all algorithms used in this research so, PM is good for passband ripple but not good for stopband ripple.

Tables 5.3 & 5.5 also show maximum stopband ripple and transition width for LP FIR filter with orders 20 and 30 using PM, GA, PSO, HGAPSO, respectively. From the table it is observed that

the HGAPSO has a minimum stopband ripple of 0.02625 and 0.0099 and a shorter transition width about 0.0857 and 0.0765 for both orders 20 and 30 respectively.

Table 5.7 shows the maximum passband ripple, maximum stopband ripple and maximum attenuation for LP FIR filter with orders 40 and 50 using HGAPSO from the Table it is observed that the passband ripple is decreases if the order is increases so, the performance of HGAPSO algorithm increase if the order of the filter is also increased.

Figures 5.9 & 5.10 shows the Amplitude (normalized) and frequency plot of LP FIR filter with orders 40 and 50. From the Fig lower order has a high passband ripple than higher order.

Figures 5.1-5.8 show the amplitude response of the LP filter order of 20 & 30 using PM, GA, PSO, HGAPSO. The amplitude response in dB is plotted in Figure 5.4 & 5.8 for order 20 & 30 respectively for lowpass filter. The normalized amplitude response is shown in Figures 5.1-5.3 for order 20 and Figures 5.5-5.7 for order 30. Figures 5.2 & 5.6 show the normalized passband ripple for FIR LP filter of orders 20 & 30. Figures 5.3 & 5.7 show the plots of normalized stopband ripple. From the above Figures and Tables, it is observed that HGAPSO has a better amplitude response (dB), normalized amplitude response and normalized stop band ripple for LP filter, as compared to PM, RGA and PSO algorithms.

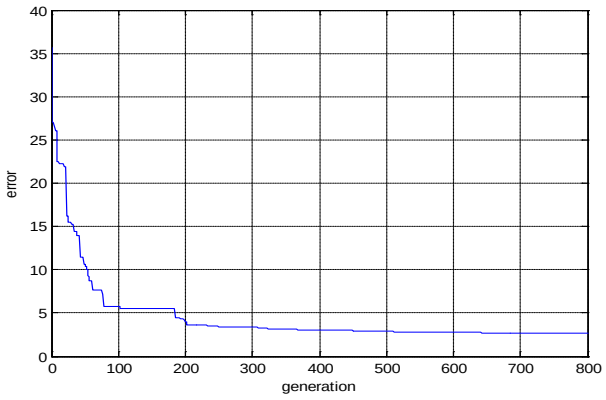


Fig 5.11 Convergence Profile for GA in case of FIR LP Filter of Order 20

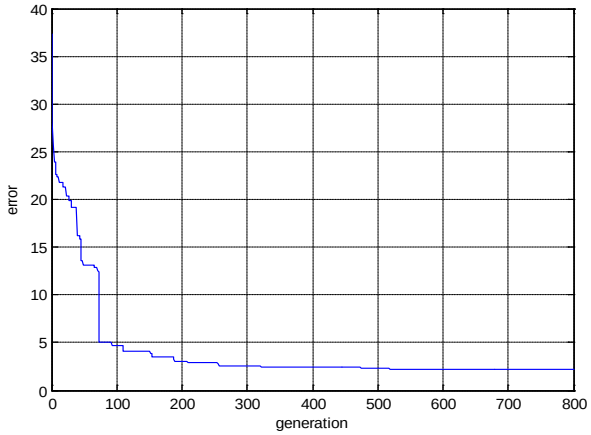


Fig 5.12 Convergence Profile for PSO in case of FIR LP Filter of Order 20

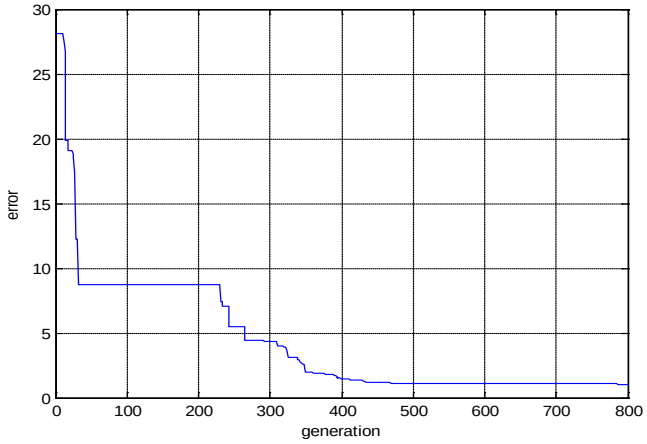
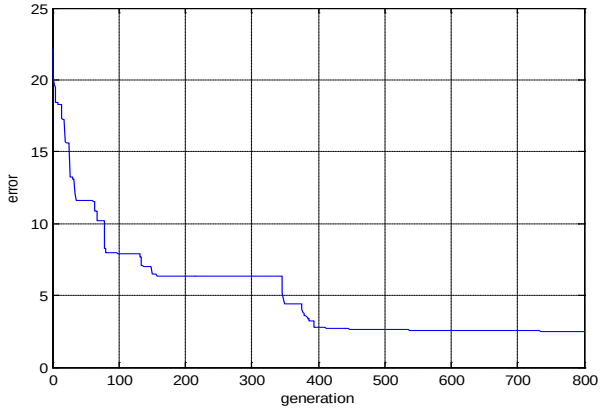


Fig 5.13 Convergence Profile for hybrid GA and PSO in case of FIR LP Filter of Order 20



5.14 Convergence Profile for GA in case of FIR LP Filter of Order 30

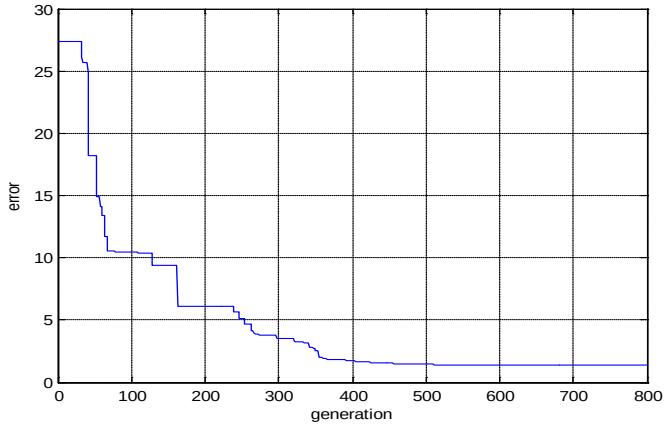


Fig 5.15 Convergence Profile for PSO in case of FIR LP Filter of Order 30

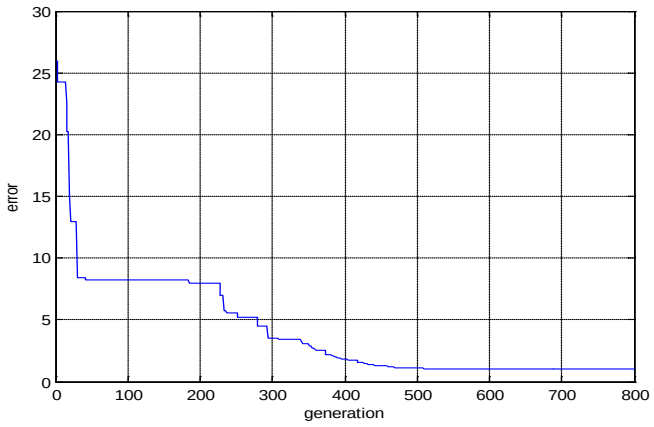


Fig 5.16 Convergence Profile for hybrid GA and PSO in case of FIR LP Filter of Order 30

5.2. Comparative Effectiveness and Convergence Profiles of GA, PSO and hybrid GA and PSO

In order to compare the algorithms in terms of the error fitness value, Figures 5.11-5.16 show the convergences of error fitness values obtained using GA, PSO, and the HGAPSO, respectively. The convergence graph has been shown for the filter orders of 20 and 30. Hybrid of GA and PSO converges to much lower error fitness value as compared to GA and PSO which yield suboptimal higher values of error fitness values. For order 20 and 30 GA converges to the minimum error fitness value of 2.676 and 2.586; PSO converges to the minimum error fitness value of 2.216 and 1.353; whereas, the HGAPSO converges to the minimum error fitness value of 1.039 and 1.011

respectively. For the designed FIR LP filter, HGAPSO converges to the least minimum error fitness value in finding the optimum filter coefficients.

CHAPTERSIX

6. CONCLUSION AND FUTURE SCOPE OF WORK

6.1. Conclusion

The FIR filters are widely used in digital signal processing and can be designed using classical methods. Classical optimization methods cannot optimize such objective functions and cannot converge to the global minimum solution. So, evolutionary optimization methods have been implemented for the design of optimal digital filters with better control of parameters and the highest stopband attenuation.

In this thesis, lowpass FIR filters of order twenty and thirty are designed using GA, PSO and HGAPSO by using MATLAB. The even order FIR LP filters are designed as filter coefficient ($h(n)$) even symmetric and positive initial values taken. Comparison of the results of PM, GA, PSO, and HGAPSO algorithm has been made. It is revealed that HGAPSO has the ability to converge to the best quality near optimal solution and possesses the best convergence characteristics among the algorithms. From the Tables 5.3 & 5.5 it is observed that filter order 30 has minimum stopband ripple and maximum attenuation compared to order 20, so as filter order increases the solution quality is improve.

The simulation results clearly indicate that HGAPSO demonstrates the best performance in terms of amplitude response, minimum stop band ripple and maximum stopband attenuation with the narrowest transition width but not good for maximum passband ripple for lower order. From the simulation indicate that the performance of HGAPSO increases for orders more than 20. Thus, the HGAPSO may be used as a good optimizer for obtaining the optimal filter coefficients for higher order in any practical digital filter design problem of digital signal processing systems.

In this research the exact specifications which are the passband ripple and stopband ripple are not exactly met but approached to the specified value. I cannot realize the exact reason why the specified values are not exactly met. This is not only for this research work but also for other previous research works.

6.2. Future Scope of Work

The future scope of this work could include the following:

1. Investigate the performance of the Hybrid GA and PSO design algorithm for the design of high pass, band pass and pass FIR filters.
2. Investigate the performance of the Hybrid GA and PSO design algorithm for the design of infinite impulse response (IIR) filters.

Reference

[1]. Sangeeta Mondal, S.P.Ghoshal, RajibKar and DurbadalMandal, "*Novel Particle Swarm Optimization for Low Pass FIR Filter Design*", Department of Electronics and Communication Engg. National Institute of Technology, Durgapur INDIA, E-ISSN: 2224-3488, Issue 3, Volume 8, July 2012.

[2]. Yasunori Sugita, NaoyukiAikawa and Toshinori Yoshikawa, "Design Method of FIR Digital Filters With Specified Group Delay Errors Using Successive Projection", International Journal of Innovative Computing, Information and Control, Volume 4, Number 2, February 2008.

[3]. Gurleen Kaur and Ranjit Kaur, "Design of recursive Digital Filters Using Multi objective Genetic Algorithm ",International Journal of Engineering Science and Technology, ISSN : 0975, Vol.3 No.7, July 2011.

[4]. Wang Xiaohua and He Yigang, "Design of complex FIR filters with arbitrary magnitude and group delay responses", Journal of Systems Engineering and Electronics, Vol. 20, No. 5, 2009.

[5]. L.R. Rabiner, "Approximate design relationships for High-pass FIR digital filters", IEEE Trans. Audio Electro acoustics, AU-21, No.5, 1973.

[6]. Zhongkai Zhao and HongyuanGao, "FIR Digital Filters Based on Cultural Particle Swarm Optimization", Proceedings of the 2009 International Workshop on Information Security and Application, Qingdao, China, November 21-22, 2009.

[7]. RajibKar, DurbadalMandal, Dibbendu Roy and Sakti Prasad Ghoshal, "FIR Filter Design using Particle Swarm Optimization with Constriction Factor and Inertia Weight Approach", ACEEE Int. J. on Electrical and Power Engineering, Vol. 02, No. 02, August 2011.

[8]. SheenuThapar, Parminder Kaur and Neha Aggarwal ,"A Low Pass FIR Filter Design Using Genetic Algorithm Based Artificial Neural Network", International Journal of Computer Technology and Electronics Engineering (IJCTEE) Volume 2, Issue 4, August 2012.

[9]. HarshKumar," FPGAImplementation of Digital FIRFilter", Electronics and Communication Engineering DepartmentThaparUniversity, June-2012.

[10]. Minoru Akazawa, Masaaki Ikehara,"Simultaneous Approximation of Magnitude and Group Delay in FIR Digital Filters"n.d

[11]. R. Eberhart, Y. Shi, "Comparison between Genetic Algorithmsand Particle Swarm Optimization", Proc. 7th Ann. Conf. onEvolutionary Computation, San Diego, 2000.

[12]. Ricardo A. Losada, "Digital Filters with MATLAB", the MathWorks, Inc.,2008.

[13]. B. A. Shenoi,"Introduction to Digital signal Processing and Filter design",a John Wiley & Sons, INC., 2006.

[14]. T.W. Parks, J.H. McClellan, "Chebyshev approximation for non recursive digital filters with linear phase", IEEE Trans. CircuitsTheory, CT-19, 1972, pp. 189–194.

[15]. J.I. Ababneh, M. H. Bataineh, "Linear phaseFIR filter design using particle swarmoptimization and genetic algorithms", DigitalSignal Processing, 18, 657–668, 2008.

[16]. A. Sarangi, R.K. Mahapatra, S.P. Panigrahi, "DEPSO and PSO-QI in digital filter design", Expert Systems with Applications, 2011, vol. 38, No.9, pp.10966-10973.

[17]. Li Tan,"Digital SignalProcessingFundamentals andApplications", 2008.

[18]. John G. Proakis, Dimitris G. Manolakis,"Digital Signal Processing",3rd Edition, 1996.

[19]. SangeetaMandal, S.P.Ghoshal, Purna Mukherjee, DyutiSengupta, RajibKar, DurbadalMandal, "Design of Optimal Linear Phase FIR High Pass Filterusing Improved Particle Swarm Optimization", ACEEE Int. J. on Signal & Image Processing, Vol. 03, No. 01, 2012

[20]. Ifeachor,E.C., Jervis,B.W., "Digital Signal Processing",2nd Edition, Low Price Edition 2007.

[21]. Parhi K K., "A Systematic Approach for Design of Digit-serial Signal Processing Architectures", Circuits and Systems, 1991.

[22]. Proakis,J.G., Manolakis,D.G., "Digital Signal Processing" 3rd Edition, PHI publication 2004.

[23]. T. W. Parks, C. S. Burrus, "Digital Filter Design" 1987 by Texas Instruments Incorporated Published by John Wiley & Sons, Inc.

[24]. D. Mandal, S. P. Ghoshal, and A. K. Bhattacharjee, "Application of Evolutionary Optimization Techniques for Finding the Optimal set of Concentric Circular Antenna Array", Expert Systems with Applications, vol. 38, pp. 2942-2950, 2010.

[25]. O. Herrmann, W. Schussler, "Design of nonrecursive digital filters with linear phase", Electron. Lett., 6, 1970, pp.329–330.

[26]. Sabbir U. Ahmad, “Design of Digital Filters Using Genetic Algorithms” , M.Eng., Nanyang Technological University, Singapore, 2008

[27]. A. Youunis, J. Gu, Z. Dong, and G. Li, \Trends, Features, and Test of Commonand Recently,” Introduced Global Optimization Methods," in Proceedings of the12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conferencevol. 1, Victoria, BC, September 2008, pp. 1-31.

[28]. J. H. Holland, “Adaptation in Natural and Artificial Systems”, Ann Arbor, MI:University of Michigan Press, 1975.

[29]. D. Dumitrescu, B. Lazzerni, L. Jain, and A. Dumitrescu, “Evolutionary Computation”, Boca Raton, FL: CRC Press, 2000.

[30]. M. Mitchell, “An Introduction to Genetic Algorithms”, Cambridge, MA: MITPress, 1996.

[31]. J. M. Johnson and V. Rahmat-Samii, “Genetic algorithms in engineering electromagnetic," IEEE Antennas and Propagation Magazine, vol. 39, no. 4,pp. 1045{9243, August 1997.

[32]. A. Kavehand S. Malakouti Rad, “Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design,” Iranian Journal of Science & Technology, Transaction B: Engineering, Vol. 34, No. B1, pp 15-34 Printed in The Islamic Republic of Iran, 2010

[33]. K. Premalatha and A.M. Natarajan,”Hybrid PSO and GA for Global Maximization,”Int. J. Open Problems Compt. Math., Vol. 2, No. 4, December 2009

[34]. Ranjit Singh and Sandeep K. Arya, “Genetic Algorithm for the Design of Optimal IIR Digital Filters,”Journal of Signal and Information Processing, 2012, 3, 286-292

[35]. James Blondin,”Particle Swarm Optimization: A Tutorial,”September 4, 2009

[36]. Sabbir U. Ahmad,”Design of Digital Filters Using Genetic Algorithms,”M.Eng., Nanyang Technological University, Singapore, 2001