Dissertation

on

Performance Analysis of Low Pass FIR Filter Design using Dynamic and Adjustable Particle Swarm Optimization Techniques

Submitted in partial fulfillment

for the award of degree of

MASTER OF TECHNOLOGY

in

Electronic Circuits and Systems

Submitted By:

Kaushal Kishor Tripathi

(1900104190)

Under the Guidance of

Dr. Mohd. Suhaib Kidwai **Dr. Imran Ullah Khan**

Department of Electronics & Communication Engineering.

INTEGRAL UNIVERSITY, LUCKNOW

May-2022

UNDERTAKING FROM THE CANDIDATE

This is to certify that I Kaushal Kishor Tripathi have completed the M.Tech Dissertation work on the topic **"Performance Analysis of Low Pass FIR Filter Design using Dynamic and Adjustable Particle Swarm Optimization Techniques"** under the supervision of Dr. Mohd. Suhaib Kidwai and Dr. Imran Ullah Khan for the partial fulfillment of the requirement for the Master of Technology (M.Tech.) in Electronic Circuits and Systems from Department of Electronics and Communication Engineering, Integral University, Lucknow. This is an original piece of work & I have not submitted it earlier elsewhere.

Date: 14/05/2022 Signature

Place: Lucknow Kaushal Kishor Tripathi

EnrollmentNo.:1900104190

DECLARATION BY THE CANDIDATE

I, Kaushal Kishor Tripathi, certify that the work embodied in this Dissertation is my own bonafide work carried out by me under the supervision of Dr. Mohd. Suhaib Kidwai (Supervisor) & Dr. Imran Ullah Khan (Co-Supervisor) at Integral University, Lucknow. The matter embodied in this Dissertation has not been submitted elsewhere for the award of any other degree/diploma. I declare that I have not willfully lifted up some other's work, para, text, data, results, etc. reported in the journals, books, magazines, reports, dissertations, thesis, etc., or available at web-sites.

Date: 14/05/2022 Signature

Place: Lucknow Kaushal Kishor Tripathi

Enrollment No.:1900104190

CERTIFICATE FROM THE SUPERVISOR(S)

This is to certify that **Kaushal Kishor Tripathi** has carried out the research work presented in the dissertation entitled **"Performance Analysis of Low Pass FIR Filter Design using Dynamic and Adjustable Particle Swarm Optimization Techniques"** for the award of Master of Technology (M.Tech.) in Electronic Circuits and Systems from Department of Electronics and Communication Engineering, Integral University, Lucknow under my supervision. To the best of my knowledge, the contents of this dissertation have not been submitted to any other institute or university for the award of any degree.

Signature of Supervisor

Full Name: Dr. Mohd. Suhaib Kidwai

Designation: Assistant Professor

Address: Deptt. of ECE, Integral University

Date: 14/05/2022

Place: Lucknow

Signature of Co-Supervisor

Full Name: Dr. Imran Ullah Khan

Designation: Associate Professor (L-II)

Address: Deptt. of ECE, Integral University

Date:14/05/2022

Place: Lucknow

ACKNOWLEDGEMENT

"He is the source of all light and his light is diffused throughout the universe God has given the man intelligent speech, power of expression, and capacity to understand clearly the relations of things."

First and foremost I would thank the Almighty whose infinite grace make all things possible. I am grateful to my parents who have inculcated in me good values and culture and because of their immense contribution; this work is dedicated to them as a token of love. We are living in the age of electronics and this field has witnessed radical changes over the last decades. We have observed vast application of electronics in almost every walk of life especially in the field of medical science. This fascinated me to do something in this field as a student of electronics science.

In my project work great many people have contributed time and efforts in bringing the project to a conclusion. I feel great pleasure in expressing my sense of gratitude and sincere thanks to my Project Guide Dr. Mohd. Suhaib Kidwai and Co Guide Dr Imran Ullah Khan, Department of Electronics and Communication Engineering for his consistent and careful guidance throughout this project work. I feel proud to get the opportunity to work under him which has enlightened my thoughts and enriched my experience. I am grateful to Dr Syed Hasan Saeed, Professor and Head, Department of Electronics and Communication Engineering for providing necessary facilities in the department. I am also thankful to Dr Shailendra Kumar for his constant support throughout this semester.

Date: 14/05/2022 Signature

Place: Lucknow Kaushal Kishor Tripathi

Enrollment No.: 1900104190

ABSTRACT

Digital filters are used for applications like control systems, audio systems, video processing, and communication, medical applications to name just a few. They are designed in hardware or software and worked in both real-time and recorded signals. In hardware form they can routinely perform tasks that were almost exclusively performed by analog systems in the past whereas software digital filters can be implemented using low-level or user-friendly high-level programming languages. Nowadays they are used to perform many filtering tasks which in the not so distant past were performed almost exclusively by analog filters and are replacing the traditional analog filters in many applications. Along with the advantages, such as, high accuracy and reliability, small physical size and reduced sensitivity to component tolerances or drift, digital implementations helps to achieve certain characteristics not possible with analog designs such as exact linear phase and multi rate operation. They are applied to very low frequency signals, such as those occurring in biomedical and seismic applications very efficiently. In addition, the characteristics of digital filters can be changed or adapted by simply changing the content of a finite number of registers, thus multiple filtering tasks can be performed by one programmable digital filter without the need to replicate the hardware. With the ever increasing number of applications involving digital filters, the variety of requirements that have to be met by them is increased. This thesis is based on the design of digital filters problem that involves multiple, often conflicting, design criteria and specifications. Finding an optimum filter design is the main objective in this work. Analytic or simple iterative methods 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. We have performed the work for an optimal design of linear phase digital low pass finite impulse response (FIR) filter using Particle Swarm Optimization and its modification related to dynamic adjustable PSO. The dynamic adjustable PSO an improved particle swarm optimization (PSO) that proposes a velocity vector and swarm updating and hence the solution quality is improved. In the design process, the filter length, pass band and stop band frequencies, feasible pass band and stop band ripple sizes are specified.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLE

LIST OF ABBREVIATION

S. No.	Abbreviation	Name
1	FIR	Finite impulse response
2	IIR	Infinite impulse response
3	PSO	Particle swarm optimization
4	GA	Genetic algorithm
5	pbest	Personal best
6	gbest	Global best
7	EC	Evolutionary computation
8	ES	Evolution strategies
9	EP	Evolutionary programming
10	DE	Differential evolution
11	FEP	Fast evolutionary programming
12	DAPSO1	Dynamic and adjustable particle swarm optimization1
13	DAPSO2	Dynamic and adjustable particle swam optimization 2

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

This chapter introduces digital filter design as an optimization problem and discusses various methods applied in the design of digital filters traditionally and currently using the Computational intelligence techniques.

1.2 DIGITAL FILTER

A filter is a frequency selective circuit that allows a certain frequency to pass while attenuating the others. Filters could be analog or digital. Analog filters use electronic components such as resistor, capacitor, transistor etc. to perform the filtering operations. These are mostly used in communication for noise reduction, video/audio signal enhancement etc. In contrast, digital filters use digital processors which perform mathematical calculations on the sampled values of the signal in order to perform the filter operation. A computer or a dedicated digital signal processor may be used for implementing digital filters. Filters mostly find their use in communication for noise reduction, audio/video signal enhancement etc.

Any time varying signal $C = x(t)$ sampled at a sampling interval of h has input signals $x_0, x_1, x_2, x_3, ..., x_n$ in intervals $0, h, 2h, 3h, ..., nh$. These inputs have corresponding outputs $y_0, y_1, y_2, y_3, ..., y_n$ depending upon the kind of operation performed. Thus, the order of the filter is determined by the number of the previous input terms used to calculate the current output. The a_0, a_1, a_2 terms appearing in the following equations are called the filter coefficients to determine the operation of the filter and determine the characteristics of the filter. Various filter parameters which come into picture are the stop band and pass band normalized frequencies (ω_s, ω_p), the pass band and stop band ripple (δ_p) and (δ_s), the stop band attenuation and the transition width. This has been shown in Fig. 1.1.

Zero Order: $y_n = a_0 x_1$ (1.1a)

First Order: $y_n = a_0 x_n + a_1 x_{n-1}$ (1.1b)

Second Order: $y_n = a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2}$ (1.1c)

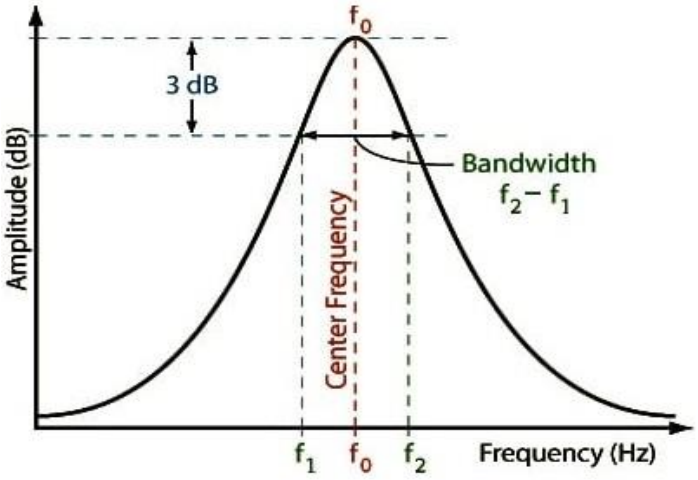


Figure 1.1: Illustration of filter parameters [3]

1.2.1 Finite Impulse Response

Finite Impulse Response filters are those for which the output of the filter depends only on the present inputs. FIR filter, or also called the non-recursive filter can be represented by the following difference equation:

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \cdots + b_n x_{n-m} \quad (1.2a)$$

By introducing a unit delay element z^{-1}, such that $z^{-1}x_n = x_{(n-1)}$, the transfer function of FIR filter can be represented as:

$$\frac{Y(Z)}{X(Z)} = b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_n z^{-m} \quad (1.2b)$$

The FIR filter has following advantages:

- Since FIR filter has its poles located at the origin, it is inherently stable since the poles lie within the unit circle.
- FIR filters can be designed as linear phase filters, making them a better choice in phase sensitive application.

1.2.2 Infinite Impulse Response

Infinite impulse response filters are those for which the output of the filter at any given time depends upon the present inputs and past outputs. The difference equation for a Linear Time Invariant IIR filter can be written as:

$$y_n = b_0x_n + b_1x_{n-1} + b_2x_{n-2} + \cdots b_nx_{n-M} - a_1y_{n-1} \ldots . - a_ny_{n-N} \quad (1.3a)$$

Similar to FIR, introducing a unit delay element z^{-1} such that $z^{-1}x_n = x(n-1)$ and $z^{-1}y_n = y(n-1)$, the transfer function of IIR filter can be represented as:

$$Y_Z\left(1 + a_1z^{-1} + \cdots + a_nz^{-n}\right) = X_Z\left(b_0 + b_1z^{-1} + b_2z^{-2} + \cdots b_nz^{-m}\right) \quad (1.3b)$$

$$\frac{Y(Z)}{X(Z)} = \frac{\sum_{i=0}^{M} b_iZ^{-i}}{\sum_{i=1}^{N}(1+a_iZ^{-i})} \quad (1.3c)$$

The IIR filters have the following advantages over FIR:

- They can achieve much sharper transition region than FIR filters of the same order.
- They require less memory and are computationally less complex for the same length of the filter.

However, due to the feedback element present in the IIR filters, chances of accumulating the rounded errors over summed iterations are higher.

1.2.3 Low-Pass Filter

Low-Pass filters are those that allow the frequencies below a threshold to pass while attenuating the frequencies beyond the threshold. The threshold frequency is called the cut-off frequency. Fig. 1.2 shows the Low-pass filter.

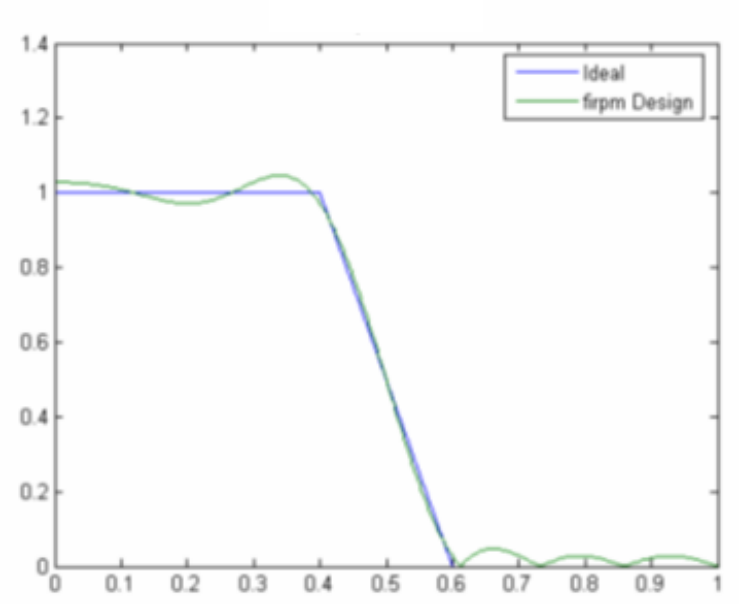


Figure 1.2: Low pass filter [3]

1.2.4 High pass Filter

High pass filters allow the frequencies beyond a threshold frequency to pass while attenuating others. Fig. 1.3 shows the high pass filter.

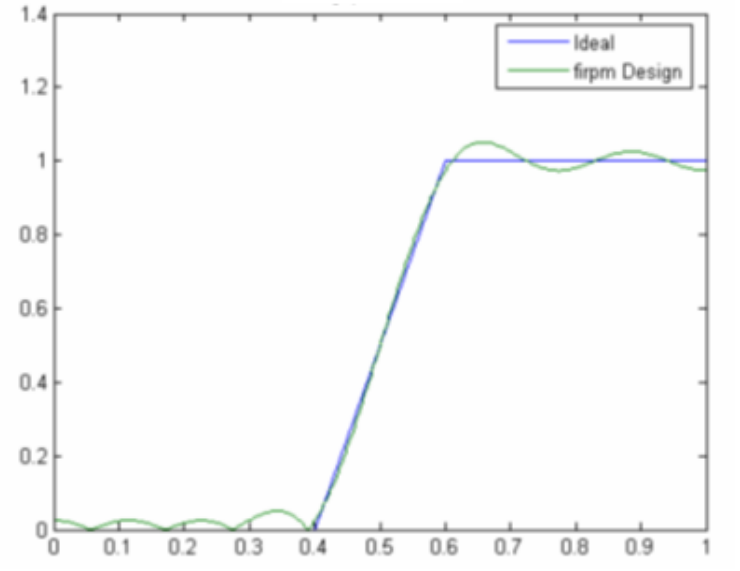


Figure 1.3: High pass filter[3]

1.2.5 Band pass Filter

In a band pass filter, frequencies which lie between a lower cutoff frequency and an upper cutoff frequency are allowed to pass while others are attenuated. The frequency band for which the filter allows to pass is called the pass band and the bands of frequencies which are attenuated are called the stop band frequencies. Fig. 1.4 shows the band pass filter.

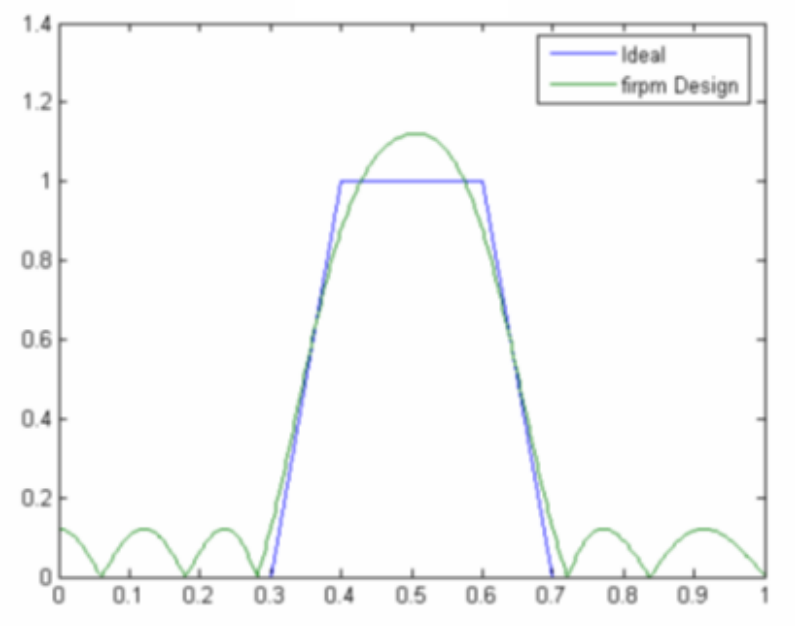


Figure 1.4: Band pass filter [3]

1.2.6 Band stop Filter

In a band stop filter, the frequencies between two cutoff frequencies are attenuated while the others are allowed to pass. Fig. 1.5 shows the band stop filter.

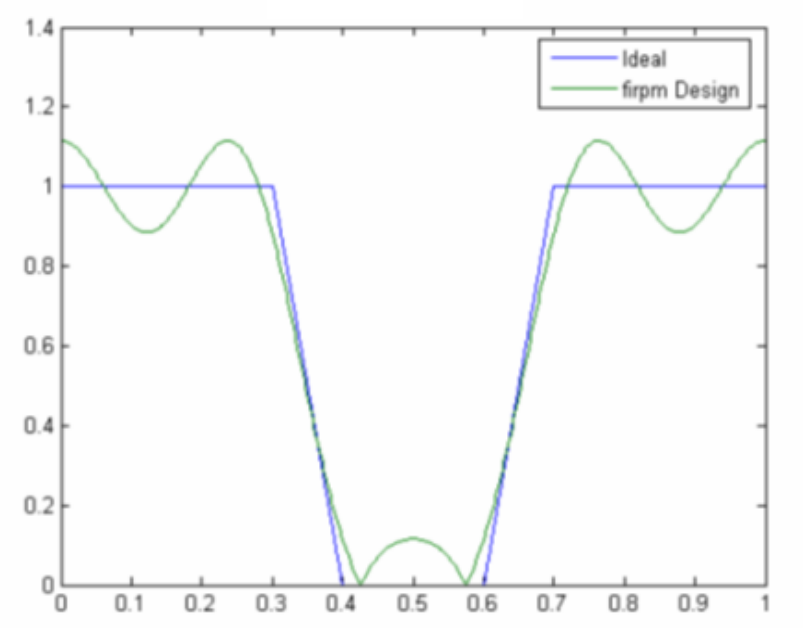


Figure 1.5: Band stop filter [3]

1.3 FILTER DESIGN TECHNIQUES

1.3.1 Traditional Techniques

Traditionally, different techniques exist for the design of digital filters. Of these, windowing method is the most popular. In this method, ideal impulse response is multiplied with a window function. There are various kinds of window functions (Butterworth, Chebyshev, Kaiser etc.), depending on the requirements of ripples on the pass band and stop band, stop band attenuation and the transition width. These various windows limit the infinite length impulse response of ideal filter into a finite window to design an actual response. But windowing methods do not allow sufficient control of the frequency response in the various frequency bands and other filter parameters such as transition width. Designer always has to compromise on one or the other of the design specifications. So, computational intelligence techniques have been implemented in the design of digital filters to design with better parameter control and to better approximate the ideal filter. Since population based stochastic

search methods have proven to be effective in multidimensional nonlinear environment, all of the constraints of filter design can be effectively taken care of by the use of these algorithms.

1.3.2 Computational Intelligence Techniques

Computational intelligence based techniques such as particle swarm optimization (PSO) and genetic algorithms (GA) have been implemented in the design of digital filters [1].

Digital filter design is an important aspect of digital signal processing. Although various traditional techniques have been used in the past, digital filter design as an optimization problem can be solved by using computational intelligence based techniques. The use of these intelligent stochastic search approaches tend to produce better results in a short period of time, thus opening grounds for adaptive filter to be designed to use in an online environment.

1.4 PARTICLE SWARM OPTIMIZATION TECHNIQUES

PSO (Particle swarm optimization) is a search technique based on social behavior of bird flocking and fish schooling [10]. There are different kinds of bio and social behavior inspired algorithms. PSO is one of the different swarm based algorithms. In PSO, each particle of the swarm is a possible solution in the multi-dimensional search space. The particles change their positions with a certain velocity in “each” iteration, according to the standard PSO equations, thus moving towards the global best (gbest) solution. Being easy to implement and yet so effective, PSO has been utilized in a wide variety of optimization applications. In this thesis, PSO has been used in system identification and to design digital filters.

1.5 PSO ALGORITHM

Particle swarm optimization is a population based search algorithm and is inspired by the observation of natural habits of bird flocking and fish schooling. In PSO, a swarm of particles moves through a D dimensional search space. The particles in the search process are the potential solutions, which move around the defined search space with some velocity until the error is minimized or the solution is reached, as decided by the fitness function. Fitness function is the measure of particles fitness which is the deviation of the particle from the

required solution. The particles reach to the desired solution by updating their position and velocity according to the PSO equations. In PSO model, each individual is treated as a volume-less particle in the D-dimensional search space with initial random velocity. Each particle has memory which keeps track of its previous best position and fitness, with the position and velocity of ith particle represented as:

$$X_i = (x_{i1}, x_{i2}, \dots x_{iD}) \quad (1.4a)$$

$$V_i = (v_{i1}, v_{i2}, \dots v_{iD}) \quad (1.4b)$$

These particles are randomly distributed over the search space with initial position and velocity. They change their positions and velocity according to these equations where C_1 and C_2 are cognitive and social acceleration constants, rand$_1$ () and rand$_2$ () are two random functions uniformly distributed in the range of [0, 1] and w is the inertia weight introduced to accelerate the convergence speed of PSO. Vector P_i = (P_{i1}, P_{i2},.....,P_{iD}) is the best previous position (the position giving the best fitness value) of particle i called the pbest, and vector P_g= (P_{g1},P_{g2},....., P_{gD}) is the position of the best particle among all the particles in the population and is called the gbest. X_{id}, V_{id}, P_{id} are the dth dimension of vector of X_i, V_i, P_i. The gbest is changed to lbest in local PSO where lbest is the best value in the neighborhood.

$$V_{id}(k+1) = w * V_{id}(k) + C_1 * rand_1 * (P_{id} - X_{id}) + C_2 * rand_2 * (P_s - X_{id}) \quad (1.4c)$$

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k+1) \quad (1.4d)$$

Other variations of PSO equations also exist for discrete and binary PSO. The flowchart in Fig. 1.6 shows the PSO algorithm.

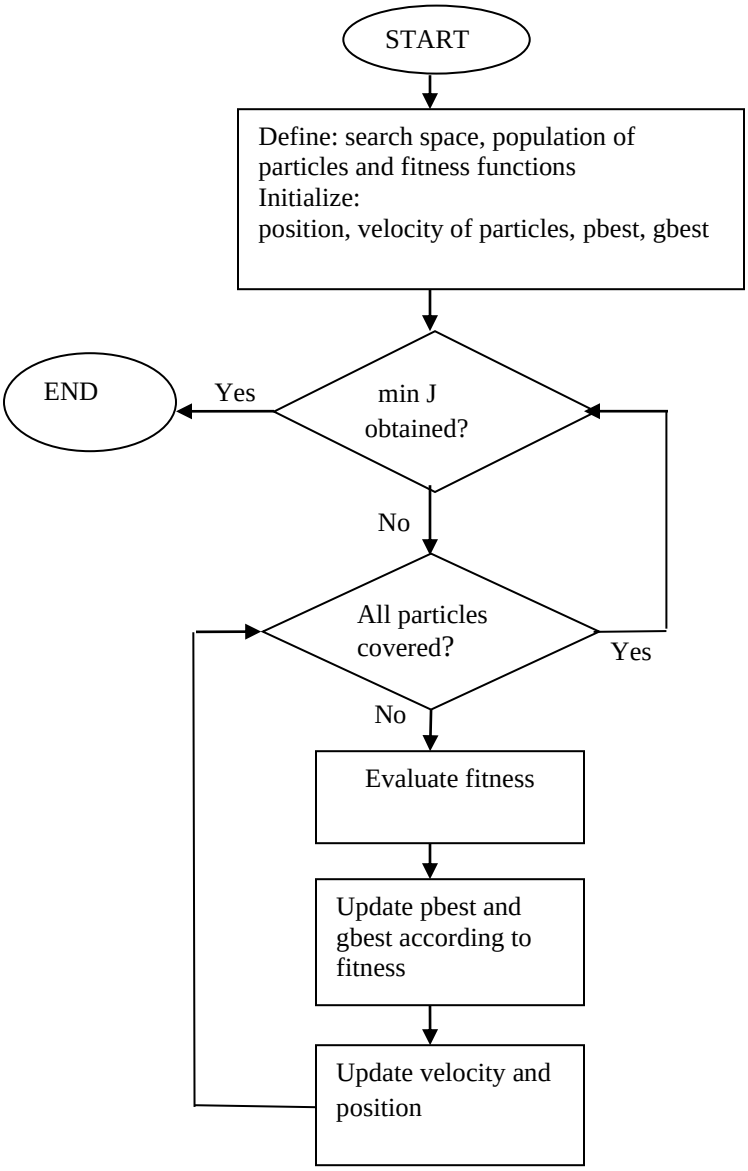


Figure 1.6: Flowchart for PSO

1.5.1 Parameters

PSO equation consists of three parameters. These c1 and c2 are called cognitive and social acceleration constants and help to guide the particles towards the gbest. These constant are equal and have the values from 0 to 2 but studies have shown their values set to 2 gives the best results. Another parameter of PSO is w called the inertia weight. Value of w ranges from 0.2 to 1.2 but is usually considered to be 0.8 for best results in case of Constrained PSO (CPSO). For unconstrained PSO, w is linearly decreasing from 0.9 to 0.4 over iterations.

1.5.2 Topologies

Social Interaction is the key factor in the success of PSO. Individual particles in the population learn from their own and their neighbors past experiences to get to a better fitness value. There are three different topologies for PSO. These control the flow of information within the network. Particles in PSO are capable of communicating within the network but information outside the network is inaccessible. The type of network is determined by the following three topologies:

1.5.2.1 Star Topology

In this topology, each particle is connected with every other particle in the population and shares information among all. It is also called as the global version of PSO and hence the particle which performs the best in the swarm has an influence over the entire population. This is shown in the Fig. 1.7

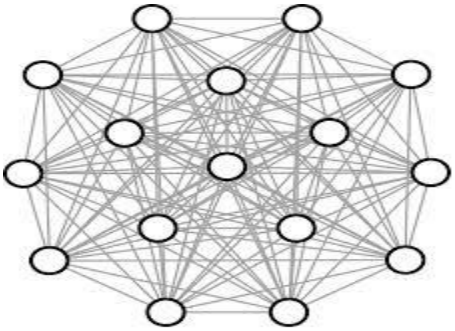


Figure 1.7: Star Topology [10]

1.5.2.2 Ring Topology

In this topology, each particle is connected with its immediate neighbors and information sharing is only between the neighbors. This topology is used in the local PSO where l_{best} is considered instead of g_{best}, which is the best position among the neighbors. This is shown in Fig. 1.8.

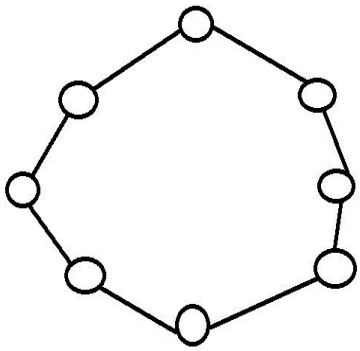


Figure 1.8: Ring Topology [10]

1.5.2.3 Wheel Topology

In this topology, one particle is connected with all of the other particles while the rest of the particles are not connected to each other directly. The information sharing takes place through the node, which is the center of the wheel. This is shown in Fig. 1.9.

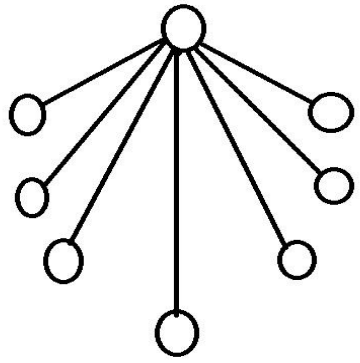


Figure 1.9: Wheel Topology [10]

1.6 REPORT ORGANIZATION

The report has been divided into five chapters as below:

Chapter 1 includes introduction of digital filter & its type with filter design technique. Basic of PSO is also described.

Chapter 2 provides original version of PSO & dynamic problem, new concept of PSO with diversity control.

Chapter 3 deals with description of PSO, Filter design methodology, adjustable & dynamic PSO, Attractive & repulsive PSO.

Chapter 4 concerned implementation of algorithm PSOP1 & PSOP2 with constraints as test function 1 (J1), test function 2 (J2), test function 3 (J3), test function 4 (J4).The simulation performed here using MATLAB 7.10.

Chapter 5 represents overall result and conclusion.

CHAPTER 2

LITERATURE SURVEY

Literature Survey

This chapter summarize literature survey of particle swarm paradigm, that was only a few years ago a curiosity, has now attracted the interest of researchers around the globe. This section is intended to give an overview of important work that gave direction and impetus to research in particle swarms as well as some interesting new directions and applications. Things change fast in this field as investigators discover new ways to do things, and new things to do with particle swarms. It is impossible to cover all aspects of this area within the strict page limits of this section. Thus this section should be seen as a snapshot of the view for particular authors, have at the time of writing.

Rao R. V. et al. (2019), [1] Digital filters are an important part of digital signal processing systems. Digital filters are divided into finite impulse response (FIR) digital filters and infinite impulse response (IIR) digital filters according to the length of their impulse responses. An FIR digital filter is easier to implement than an IIR digital filter because of its linear phase and stability properties. In terms of the stability of an IIR digital filter, the poles generated in the denominator are subject to stability constraints. In addition, a digital filter can be categorized as one-dimensional or multi-dimensional digital filters according to the dimensions of the signal to be processed. However, for the design of IIR digital filters, traditional design methods have the disadvantages of easy to fall into a local optimum and slow convergence. The Particle Swarm Optimization (PSO) and Teaching-Learning-Based Optimization algorithm has been proven beneficial in a wide range of engineering applications. To this end, this dissertation focuses on using DAPSO and its improved algorithms to design various types of digital filters, which include linear phase FIR digital filters, multi objective general FIR digital filters, multi objective IIR digital filters, two-dimensional (2-D) linear phase FIR digital filters, and 2-D nonlinear phase FIR digital filters. Among them, linear phase FIR digital filters, 2-D linear phase FIR digital filters, and 2-D nonlinear phase FIR digital filters use single-objective type of TLBO algorithms to optimize; multi objective general FIR digital filters use multi objective non dominated TLBO (MOTLBO) algorithm to optimize; and multi objective IIR digital filters use MOTLBO with Euclidean distance to optimize. The design results of the five types of filter designs are compared to those obtained by other state-of-the-

art design methods. In this dissertation, two major improvements are proposed to enhance the performance of the standard TLBO algorithm. The first improvement is to apply a gradient-based learning to replace the TLBO learner phase to reduce approximation error(s) and CPU time without sacrificing design accuracy for linear phase FIR digital filter design. The second improvement is to incorporate Manhattan distance to simplify the procedure of the multi objective non-dominated TLBO (MOTLBO) algorithm for general FIR digital filter design. The design results vs obtained by the two improvements have demonstrated their efficiency and effectiveness.

Sohl J. et al. (2015), [2] Finite-length impulse response (FIR) filters occupy a central place many signal processing applications which either alter the shape, frequency or the sampling frequency of the signal. FIR filters are used because of their stability and possibility to have linear-phase but require a high filter order to achieve the same magnitude specifications as compared to infinite impulse response (IIR) filters. Depending on the size of the required transition bandwidth the filter order can range from tens to hundreds to even thousands. Since the implementation of the filters in digital domain requires multipliers and adders, high filter orders translate to a large number of these arithmetic units for its implementation. Research towards reducing the complexity of FIR filters has been going on for decades and the techniques used can be roughly divided into two categories; reduction in the number of multipliers and simplification of the multiplier implementation. One technique to reduce the number of multipliers is to use cascaded sub filters with lower complexity to achieve the desired specification, known as frequency-response masking (FRM). One of the sub-filters is a upsampled model filter whose band edges are an integer multiple, termed as the period L, of the target filter's band edges. Other sub-filters may include complement and masking filters which filter different parts of the spectrum to achieve the desired response. From an implementation point-of-view, time-multiplexing is beneficial because generally the allowable maximum clock frequency supported by the current state-of-the-art semiconductor technology does not correspond to the application bound sample rate. A combination of these two techniques plays a significant role towards efficient implementation of FIR filters. Part of the work presented in this dissertation is architectures for time-multiplexed FRM filters that

benefit from the inherent sparsity of the periodic model filters. This time-multiplexed FRM filter not only reduce the number of multipliers but lowers the memory usage. Although the FRM technique requires higher number delay elements, it results in fewer memories and more energy efficient memory schemes when time-multiplexed. Different memory arrangements and memory access schemes have also been discussed and compared in terms of their efficiency when using both single and dual-port memories. An efficient v vi Abstract pipelining scheme has been proposed which reduces the number of pipelining registers while achieving similar clock frequencies.

Ababneh J.I. (2012), [3]The initial ideas on particle swarms were essentially aimed at producing computational intelligence by exploiting simple analogues of social interaction, rather than purely individual cognitive abilities. In PSO a number of simple entities—the particles—are placed in the search space of some problem or function, and each evaluates the objective function at its current location. Each particle then determines its movement through the search space by combining some aspect of the history of its own current and best (best-fitness) locations with those of one or more members of the swarm, with some random perturbations. The next iteration takes place after all particles have been moved. Eventually the swarm as a whole, like a flock of birds collectively foraging for food, is likely to move close to an optimum of the fitness function. The evolutionary computation (EC) community has shown a significant interest in optimization for many years. In particular, there has been a focus on global optimization of numerical, real-valued 'black-box' problems for which exact and analytical methods do not apply. Since the mid-sixties many general-purpose optimization algorithms have been proposed for finding near-optimal solutions to this class of problems; most notably: evolution strategies (ES), evolutionary programming (EP), and genetic algorithms (GA).Many efforts have also been devoted to compare these algorithms to each other. Typically, such comparisons have been based on artificial numerical benchmark problems. The goal of many studies was to verify that one algorithm outperformed another on a given set of problems. The single optimal point where the number of multiplications is minimum for non-time-multiplexed FRM filters is shown to become a function of both the period, L and time-multiplexing factor, M. This means that the minimum number of

multipliers does not always correspond to the minimum number of multiplications which also increases the flexibility of implementation. These filters are shown to achieve power reduction between 23% and 68% for the considered examples. To simplify the multiplier, alternate number systems like the logarithmic number system (LNS) have been used to implement FIR filters, which reduces the multiplications to additions. FIR filters are realized by directly designing them using integer linear programming (ILP) in the LNS domain in the minimax sense using finite word length constraints. The branch and bound algorithm, a typical algorithm to implement ILP problems, is implemented based on LNS integers and several branching strategies are proposed and evaluated. The filter coefficients thus obtained are compared with the traditional finite word length coefficients obtained in the linear domain. It is shown that LNS FIR filters provide a better approximation error compared to a standard FIR filter for a given coefficient word length. FIR filters also offer an opportunity in complexity reduction by implementing the multipliers using Booth or standard high-radix multiplication. Both of these multiplication schemes generate pre-computed multiples of the multiplicand which are then selected based on the encoded bits of the multiplier. In transposed direct form (TDF) FIR filters, one input data is multiplied with a number of coefficients and complexity can be reduced by sharing the pre computation of the multiplies of the input data for all multiplications. Part of this work includes a systematic and unified approach to the design of such computation sharing multipliers and a comparison of the two forms of multiplication. It also gives closed form expressions for the cost of different parts of multiplication and gives an overview of various ways to implement the select unit with respect to the design of multiplexers. Particle filters are used to solve problems that require estimation of a system. Improved resampling schemes for reducing the latency of the resampling stage is proposed which uses a pre-fetch technique to reduce the latency between 50% to 95% dependent on the number of pre-fetches. Generalized division-free architectures and compact memory structures are also proposed that map to different resampling algorithms and also help in reducing the complexity of the multinomial resampling algorithm and reduce the number of memories required by up to 50%.

Shaikh, F. et al., (2017), [4] Several investigators have attempted to adapt PSO parameters in response to information from the environment. Techniques from evolutionary computation and other methods have been borrowed by particle swarm researchers as well. One of the first intentionally hybridized particle swarms is produced. In this model, selection was applied to the particle population; "good" particles were reproduced with mutation, and "bad" particles were eliminated. Angeline's results showed that PSO could benefit from this modification. In that paradigm, points are perturbed by the addition of random values distributed around a mean of zero; the variance of the distribution is evolved along with function parameters. Those researchers used Gaussian random values to perturb $\varphi 1$, and $\varphi 2$, as well as the position of the neighborhood best-but not the individual best-using selection to adapt the variance. The evolutionary self-adapting particle swarm optimization method, a hybrid of PSO and evolutionary methods, has shown excellent performance in comparison to some standard particle swarm methods. Miranda has used it for the manufacture of optical filters as well as in the optimization of power systems. "Breeding", borrowed from genetic algorithms, in a recent particle swarm study. Some selected particles were paired at random, and both positions and velocities were calculated from weighted arithmetic averages of the selected particles' parameters. Those researchers also divided the particle swarm into subpopulations in order to increase diversity, with some probability that individuals would breed within their own subpopulation or with a member of another. Results were encouraging, though the model as reported was not clearly superior to standard PSO or GA. Cauchy mutation replaced with a version of PSO velocity in a fast evolutionary programming (FEP) algorithm, to give the FEP population direction. Their published results indicate that the approach is very successful on a range of functions; the new algorithm found global optima in tens of iterations, compared to thousands for the FEP versions tested. As a consequence of this observation, they hybridized the two algorithms by switching from one to the other after several hundred iterations. They found the best horn by going from PSO to GA (PSO-GA) and noted that the particle swarm by itself outperformed both the GA by itself and the GA-PSO hybrid, though the PSO-GA hybrid performed best of all. It appears from their result that PSO more effectively explores the search space for the best region, while GA is effective at finding the best point once the population has converged on a single region; this is consistent with other findings. Similarly alternated

among several methods, but allowed individuals in a population to choose whether to belong to a population of a genetic algorithm, a particle swarm, or to become solitary hill-climbers. In their self-adaptive search method, an individual changed its stage after 50 iterations with no improvement. The population was initialized as PSO particles; the "Lifecycle" algorithm outperformed all three of the methods that comprised it. A hybrid between a PSO and a hill-climber was proposed by considering swarms of particles sliding on a fitness landscape rather than flying over it. The method uses particles without memory and requires no book-keeping of personal best. Instead it uses the physics of masses and forces to guide the exploration of fitness landscapes. Forces include: gravity, springs, and friction. Gravity provides the ability to seek minima. Springs provide exploration. Friction slows down the search and focuses it.

Pardeep, K. et al., (2012), [5]In general, it has been possible to improve a given standard method within a restricted set of benchmark problems by making minor modifications to it. Recently, particle swarm optimization (PSO) and differential evolution (DE) have been introduced and particularly PSO has received increased interest from the EC community. Both techniques have shown great promise in several real-world applications. However, to our knowledge, a comparative study of DE, PSO, and GA on a large and diverse set of problems has never been made. In this study, we investigated the performance of DE, PSO, and an evolutionary algorithm (EA), 1 on a selection of 34 numerical benchmark problems. The main objective was to examine whether one of the tested algorithms would outperform all others on a majority of the problems. Additionally, since we used a rather large number of benchmark problems, the experiments would also reveal whether the algorithms would have any particular difficulties or preferences. Overall, the experimental results show that DE was far more efficient and robust (with respect to reproducing the results in several runs) compared to PSO and the EA. This suggests that more emphasis should be put on DE when solving numerical problems with real-valued parameters. However, on two noisy test problems, DE was outperformed by the other algorithms.

Zhong W., (2016), [6] Many tweaks and adjustments have been made to the basic algorithm over the past decade. Some have resulted in improved general performance, and some have improved performance on particular kinds of problems. Unfortunately, even focusing on the

PSO variants and specializations, from our point of view, have had the most impact and seem to hold the most promise for the future of the paradigm, these are too numerous for us to describe every one of them. So, we have been forced to leave out some streams in particle swarm research. A very simple alteration of the canonical algorithm is described that operates on bit-strings rather than real numbers. In their version, the velocity is used as a probability threshold to determine whether x_{id} the dth component of x_{ij} —should be evaluated as a zero or a one. They squashed x_{id} in a logistic function. Then generated a random number r for each bit-string site and compared it to $s(X_{id})$. If r was less that the threshold, then X_{id} was interpreted as 1, otherwise as 0. This binary particle swarm is compared to several kinds of GAs, using multimodal random problem generator. This paradigm allows the creation of random binary problems with some specified characteristics, e.g., number of local optima, dimension, etc. In that study, the binary particle swarm was the only algorithm that found the global optimum on every single trial, regardless of problem features. It also progressed faster than GAs with crossover, mutation, or both, on all problems except the very simplest ones, with low dimension and a small number of local optima; the mutation-only GA was slightly faster in those cases.

Harijan, B.L. (2018) [7] Several other researchers have proposed alterations to the particle swarm algorithm to allow it to operate on binary spaces. They used the locations of the particles as probabilities to select features in a pattern-matching task. Each feature was assigned a slice of a roulette wheel based on its floating-point value, which was then discredited to {0, 1}, indicating whether the feature was selected or not. One version, which he calls the "regulated discrete particle swarm," performed very well on a suite of test problems. Instead of directly encoding bit strings in the particles, each particle stored the small number of coefficients of a trigonometric model (angle modulation) which was then run to generate bit strings. Extending PSO to more complex combinatorial search spaces is also of great interest. The difficulty there is that notions of velocity and direction have no natural extensions for tours, permutations, schedules, etc.

Sadek, B. A. M., 2016, [8] Dynamic problems are challenging for PSO. These are typically modeled by fitness functions which change over time, rendering particle memory obsolete. A

slight adjustment is made to the inertia by randomizing the inertia weight between 0.5 and 1.0. The idea is that when tracking a single dynamic optimum, it can not be predicted whether exploration (a larger inertia weight) or exploitation (a smaller inertia weight) will be better at any given time. However, in many cases more specialized changes are required to handle dynamic problems. The problem can be split into two: how to detect change and how to respond. Change is indicated by a different function value upon re-evaluation. Two responses tried to change. In the first, the current position was set to be the previous best and in the second, and more successful response, previous bests are compared with current positions, and memory is updated accordingly. A similar scheme for change detection is used but, as a response, randomized the entire swarm in the search space.

Muhammad Imran, M., (2012), [9] As an alternative to these strategies, a level of diversity can be maintained throughout the run. Repulsion is used to keep particles away from detected optima. The charged particles is introduced into the swarm. These particles mutually repel and orbit a converging nucleus of 'neutral' particles, in analogy with a (picture of) an atom. Charged swarms can detect optimum shifts within their orbit and are therefore able to track quite severe changes. Diversity can also be maintained with a grid-like local neighborhood and using hierarchical structures. Dynamic multi-modal landscapes are especially challenging for PSO. The strategies mentioned above—restart and diversity enhancement—are less successful in situations where function peaks can vary in height as well as location because a small change in peak height might entail a large change in the position of the global optimum. In these cases, multi-population approaches have proven to be beneficial. The idea behind multi-swarm models is to position swarms on different peaks so that, should a suboptimal peak become optimal, a swarm will be ready to immediately begin re-optimizing. The size and number of swarms are changed dynamically by ordering the particles into 'species' in a technique called clearing. Borrowing from the atomic analogy referred to above, invoke an exclusion principle to prevent swarms from competing on the same peak and an anti-convergence measure to maintain diversity of the multi-swarm as a whole [15]. Recently, a self-adapting multi-swarm has been derived . The multi-swarm with exclusion has been favorably compared, on the moving peaks problem, to the hierarchical swarm, PSO re-

initialization and a state of the art dynamic-optimization evolutionary algorithm known as self-organizing scouts. Noisy fitness functions are important since they are often encountered in real-world problems. In these problems, unlike dynamic problems where the fitness function changes over time, the fitness function remains the same. However, its evaluation is noisy. Therefore, if a PSO explores the same position more than once, the fitness values associated to each evaluation may differ. The behavior of the PSO when Gaussian distributed random noise was added to the fitness function and random rotations of the search space were performed. Experimental results indicated that the PSO remained effective in the presence of noise, and, in some cases, noise even helped the PSO avoid being trapped in local optima. The PSO is compared to a noise-resistant variant where the main PSO loop was modified so that multiple evaluations of the same candidate solution are aggregated to better assess the actual fitness of this particular solution. The comparison considered several numerical problems with added noise, as well as unsupervised learning of obstacle avoidance using one or more robots. The noise-resistant PSO showed considerably better performance than the original.

Adhan S. et al., (2017), [10]It is shown that adaptation of the constriction factor, population size, and number of neighbors can produce improved results. His studies found that best performance were obtained when all three of these factors are adapted during the course of the run. Three rules are used: (a) Suicide and generation: a particle kills itself when it is the worst in its neighborhood and generates a new copy of itself when it is the best; (b) Modifying the coefficient: good local improvement caused an increase in the constriction coefficient, while poor improvement caused its decrease; (c) Change in neighborhood: the locally best particle could reduce the number of its neighbors, while poorly performing particles could increase theirs. Adaptive changes were not made on every iteration, but only occasionally. The idea of division borrowed of labor from research on insect swarm algorithms. In their hybrid particle swarm model, individuals in the swarm were assigned, after some number of iterations without improvement, to conduct local search. Local search was implemented by placing a particle at the population's global best position with a new random velocity vector. The division of labor modification was intended to improve performance on unmoral problems; this improvement was seen, though performance on multimodal functions was not significantly improved. A

hybridization of PSO with ant colony optimization is proposed but did not present any results. PSO is combined with differential evolution (DE) but with mixed results. While the hybridized algorithm did better than either PSO or DE on one multimodal problem. The particle swarm by itself tended to be faster and more robust than either DE or two versions of hybrids that were tested. However, more recently, others have obtained more positive results. A hybridization of PSO based on genetic programming (GP) was proposed showing some promise. GP is used to evolve new laws for the control of particles' movement for specific classes of problems. The method has consistently provided PSOs that performed better than some standard reference PSOs in the problem class used for training, and, in some cases, also generalized outside that class. A PSO that borrows from estimation of distribution algorithms has recently been proposed. In this approach, the swarm's collective memory is used to bias the particles' movement towards regions in the search space which are estimated to be promising and away from previously sampled low-quality regions. Experiments suggest that this PSO hybrid finds better solutions than the canonical PSO using fewer function evaluations.

Xiaohul, H. et al., (2004), [11] Some researchers have noted a tendency for the swarm to converge prematurely on local optima. Several approaches have been implemented in order to correct for the decline of diversity as the swarm concentrates on a single optimum. Self-organized criticality to help the PSO attain more diversity, making it less vulnerable to local optima. When two particles are too close to one another, a variable called the "critical value" is incremented. When it reaches the criticality threshold, the particle disperses its criticality to other particles that are near it and relocates itself. Other researchers have attempted to diversify the particle swarm by preventing the particles' clustering too tightly in one region of the search space. Collision-avoiding swarms achieve this by reducing the attraction of the swarm center. "Spatially extended" particles, developed where each particle is conceptualized as being surrounded by a sphere of some radius. When the spatially extended particle collides with another, it bounces off. Negative entropy added to the particle swarm in order to discourage premature convergence (excessively rapid convergence towards a poor quality local optimum). In some conditions, they weighted the velocity and, in some conditions, the particle's location, by some random value, thereby obtaining a sort of "dissipative particle

swarm." In bare-bones formulations of PSO, move particles according to a probability distribution rather than through the addition of velocity-a "velocity-free" PSO. Bare-bones seek to throw light on the relative importance of particle motion and the neighborhood topology of the $\overrightarrow{pi}$ which inform this motion. In a first study of bare-bones PSO, the particle update rule was replaced with a Gaussian distribution of mean $(\overrightarrow{\text{pi}} + \overrightarrow{\text{pg}})/2$ standard deviation$|\overrightarrow{\text{pi}} + \overrightarrow{\text{pg}}|$. This idea was inspired by a plot of the distribution of positions attained by a single canonical PSO particle moving in one dimension under the influence of fixed $\overrightarrow{\text{pi}}$ and $\overrightarrow{\text{ps}}$. This empirical distribution resembled a bell curve centered at $(\overrightarrow{\text{pi}} + \overrightarrow{\text{pg}})/2$. Theoretical studies also support this finding. Gaussian bare-bones works quite well, imitating the performance of PSO on some problems, but proving less effective on others. On closer examination, what appears to be a bell curve actually has a kurtosis which increases with iteration, and the distribution has fatter tails than Gaussian. It has been suggested that the origin of this lies in the production of "bursts of outliers". The trigger for these bursts is unknown; however Kennedy discovered that if burst events are added by hand to Gaussian bare-bones, performance is improved.

CHAPTER 3

METHODOLOGY

3.1 FILTER DESIGN METHODOLOGY

This chapter presents a general algorithm for the design of a large class of finite impulse response (FIR) linear phase digital filters. Emphasis is placed on a description of how the algorithm works, and several examples are included which illustrate specific applications [18].

The algorithm uses the Remez exchange method, to design filters with minimum weighted Chebyshev error in approximating a desired ideal frequency response D (f) [9]. Several authors have studied the FIR design problem for special filter types using several different algorithms [2] - [14]. The advantage of the present approach is that it combines the speed of the Remez procedure with a capability for designing a large class of general filter types. While the algorithm to be described has a special section for the more common filter types (e.g., bandpass filters with multiple bands, Hilbert transform filters, and differentiators), an arbitrary frequency response can also be approximated.

3.2 FORMULATION OF APPROXIMATION PROBLEM

The frequency response of an FIR digital filter with an N-point impulse response {h (k)} is the z-transform of the sequence evaluated on the unit circle is given below as

$$H(f)^1 = H(z)|_{z=e^{j2\pi f}} = \sum_{k=0}^{N-1} h(k)e^{-j2\pi kf} \tag{3.1}$$

The frequency response of a linear phase filter can be written as

$$H(f) = G(f)e^{f}(\frac{L\pi}{2} - \left(\frac{N-1}{2}\right))2\pi f \tag{3.2}$$

Where G (f) is a real valued function and L = 0 or 1. It is possible to show that there are exactly four cases of linear phase FIR filters [1]. These four cases differ in the length of the impulse response (even or odd) and the symmetry of the impulse response [positive (L = 0) or negative (L = l)]. By positive symmetry we mean h(k) = h(N - 1 - h), and by negative symmetry h(h) = - h(N - 1 - h).

In all cases, the real function G(f) will be used to approximate the desired ideal magnitude specifications since the linear phase has no effect on the magnitude response of the filter[13]. The form of G(f) depends on which of the four cases is being used. Using the appropriate symmetry relations, G(f) can be expressed as follows.

Case 1: Positive symmetry, odd length:

$$G(f) = \sum_{k=0}^{n} a(k)\cos(2\pi k f) \tag{3.3}$$

where n = (N - 1)/2, a(0) = h(n), and a(h) = 2h(n - k) for h = 1, 2,. . . , n.

Case 2: Positive symmetry, even length:

$$G(f) = \sum_{k=1}^{n} b(k)\cos[2\pi(k - \tfrac{1}{2})f] \tag{3.4}$$

where n = N/2 and b(k) = 2h(n - h) for h = 1, . , n.

Case 3: Negative symmetry, odd length:

$$G(f) = \sum_{k=1}^{n} c(k)\sin(2\pi k f) \tag{3.5}$$

where n = (N - 1)/2 and c(h) = 2h(n - h) for 12 = 1, 2, . . . , n and h(n) = 0.

Case 4: Negative symmetry, even length:

$$G(f) = \sum_{k=1}^{n} d(k)\sin[2\pi(k - \tfrac{1}{2})f] \tag{3.6}$$

where n = N/2 and d (h) = 2h(n - h) for 12 = 1, - . , n.

Earlier efforts at designing FIR filters concentrated on Case 1 designs, but it is now possible to combine all four cases into one algorithm. This is accomplished by noting that G(f) can be rewritten as G(f) = Q(f)P(f) where P(f) is a linear combination of cosine functions.

The above four cases in a common form are used for single central computation routine to calculate the best approximation in each of the four cases [17]. This is accomplished by

modifying both the desired magnitude function and the weighting function to formulate a new equivalent approximation problem.

The original approximation problem can be stated as follows: given a desired magnitude response D(f) and a positive weight function W(f) , both continuous on a compact subset F C [0, 1/2] (note that the sampling rate is 1.0) and one of the four cases of linear phase filters [i.e., the forms of G(f)] , then one wishes to minimize the maximum absolute weighted error, defined as

$$\|E(f)\| = \max_{f \in F} W(f)|D(f) - G(f)| \tag{3.7}$$

over the set of coefficients of G(f).

The error function E (f) can be rewritten in the form

$$E(f) = w(f)[D(f) - G(f) = W(f)Q(f)|\frac{D(f)}{Q(f)} - P(f)\,| \tag{3.8}$$

If one is careful to omit those endpoint(s) where Q(f) = 0. Letting $\hat{D_f} = {D_f}/{Q_f}$ and

$\hat{W_f} = {W_f}/{Q_f}$, then an equivalent approximation problem would be to minimize the quantity

$$\|E(f)\| = \max_{f \in F} \widehat{W}(f)\left|\widehat{D}(f) - P(f)\right| \tag{3.9}$$

by choice of the coefficients of P(f) . The set F is replaced by

F' = F - {endpoints where Q(f) = 0}.

The net effect of this reformulation of the problem is a unification of the four cases of linear phase FIR filters from the point of view of the approximation problem. Furthermore a simplified viewpoint provided from which it is easy to see the necessary and sufficient conditions which are satisfied by the best approximation [18]. The set of necessary and sufficient conditions for this best approximation is given in the following alternation theorem [4].

Alternation theorem: If P(f) is a linear combination of *r* cosine functions i.e.,

$$P(f) = \sum_{k=0}^{r=1} \alpha(k) cos 2\pi k f \tag{3.10}$$

then a necessary and sufficient condition that P(f) be the unique best weighted Chebyshev approximation to a continuous function D(f) on F' is that the weighted error function E(f) = $\widehat{W}(f)\left[\widehat{D}(f) - P(f)\right]$ exhibit at least r + 1 external frequencies in F'.

3.3 DESCRIPTION OF THE DESIGN ALGORITHM

As seen in Fig. 3.1, the design algorithm consists of an input section, formulation of the appropriate equivalent approximation problem, solution of the approximation problem using the Remez exchange method, and calculation of the filter impulse response.

A. Particle Swarm Optimization

PSO was introduced by Kennedy and Eberhart. It was inspired by the swarming behavior as is displayed by a flock of birds, a school of fish, or even human social behavior being influenced by other individuals [17]. PSO consists of a swarm of particles moving in an n dimensional, real-valued search space of possible problem solutions. Every particle has a position vector $\vec{x}$encoding a candidate solution to the problem (similar to the genotype in EAs) and a velocity vector.$\vec{v}$. Moreover, each particle contains a small memory that stores its own best position seen so far $\vec{p}$and a global best position $\vec{g}$obtained through communication with its neighbor particles. In this study we used the fully connected network topology for passing on information [10] .

Intuitively, the information about good solutions spreads through the swarm, and thus the particles tend to move to good areas in the search space. At each time step t, the velocity is updated and the particle is moved to a new position. This new position is calculated as the sum of the previous position and the new velocity:

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t+1) \tag{3.11}$$

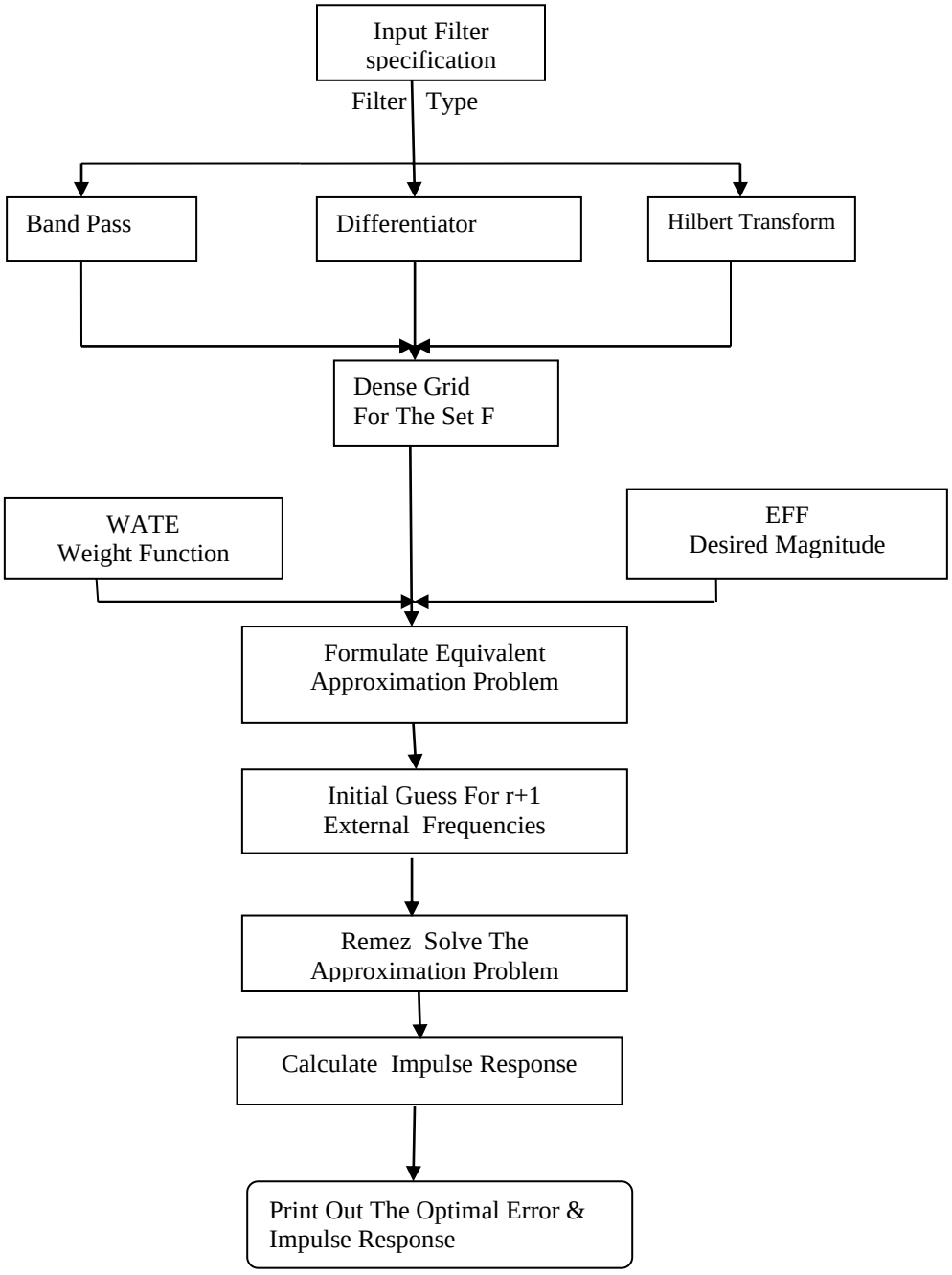


Figure 3.1: Overall flowchart of filter design algorithm

The update of the velocity from the previous velocity to the new velocity is determined by the following equation:

$$\vec{v}(t+1) = w.\vec{v}(t) + U(0,\emptyset 1)\big(\vec{p}(t) - \vec{x}(t)\big) + U(0,\emptyset 2)(\vec{g}(t) - \vec{x}(t) \qquad (3.12)$$

Where U(a, b) is a uniformly distributed number between a and b. The parameter$w$ is called the inertia weight and controls the magnitude of the old velocity $\vec{v}(t)$ in the calculation of the new velocity, whereas $\emptyset 1$ and $\emptyset 2$ determine the significance of $\vec{p}(t)$ and $\vec{g}(t)$, respectively[10]. Furthermore, at any time step of the algorithm vi is constrained by the parameter vmax.

The swarm in PSO is initialized by assigning each particle to a uniformly and randomly chosen position in the search space. Velocities are initialized randomly in the range [−vmax, vmax].

B. Attractive and Repulsive PSO

The attractive and repulsive PSO was introduced by Vesterstrom and Riget to overcome the problems of premature convergence [2]. The modification of the basic PSO scheme is to modify the velocity update formula when the swarm diversity becomes less than a value d_{low}. This modification corresponds to repulsion of the particles instead of the usual attraction scheme. Thus, the velocity is updated according to:

$$\vec{v}(t+1) = w.\vec{v}(t) - U(0,\emptyset 1)\big(\vec{p}(t) - \vec{x}(t)\big) - U(0,\emptyset 2)(\vec{g}(t) - \vec{x}(t) \qquad (3.13)$$

This will increase the diversity over some iteration, and eventually when another value d_{high} is reached, the commonly used velocity update formula will be used again. Thus, PSO is able to zoom out when an optimum has been reached, followed by zooming in on another hot spot, possibly discovering a new optimum in the vicinity of the old one. Previously, PSO was shown to be more robust than the basic PSO on problems with many optima [16].

The PSO algorithm was included in this study as a representative for the large number of algorithmic extensions to PSO that try to avoid the problem of premature convergence. Other PSO extensions could have been chosen but we selected this particular one, since the performance of PSO was as good (or better) as many other extensions [20].

3.4 PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is an evolutionary computation technique, which is inspired by flocks of birds and shoals of fish [10]. In PSO, a number of simple entities (the particles) are placed in the space of some problem and each evaluates its fitness as its current location. Each particle determines its movement through the space by considering the particle which had the best fitness and the history of its own, then it moves with a velocity. Finally, the swarm is likely to move close to the best location. The velocity and position of each particle is adjusted by the following formulas:

$$V_{id} = w * V_{id+} c_1 * rand() * (P_{id} - X_{id}) + c_2 * rand * (P_{gd} - X_{id}) \quad (3.14)$$

$$X_{id} = X_{id} + V_{id} \quad (3.15)$$

where c1 and c2 are termed the cognitive and social learning rates. These two parameters control the relative importance of the memory of the particle itself to the memory of the neighborhood. The variable rand() and rand() are two random functions that is uniformly distributed in the range [0,1].X_i = (X_{i1}, X_{i2},.....,X_{iD}) represents the ith particle. P_i = (P_{i1}, P_{i2},.....,P_{iD}) represents the best previous position of the ith particle. The symbol g represents the index of the best particle among all the particles.V_i = (V_{i1}, V_{i2},.....,V_{iD}) represents the velocity of the ith particle. Variable is the inertia weight.

The general process of PSO is as follows.

- Calculate fitness of particle
- Update pbest if the current fitness is better than pbest
- Determine nbest for each particle: choose the particle with the best fitness value of all the neighbors as the nbest
- For each particle Calculate particle velocity according to eq. 3.14

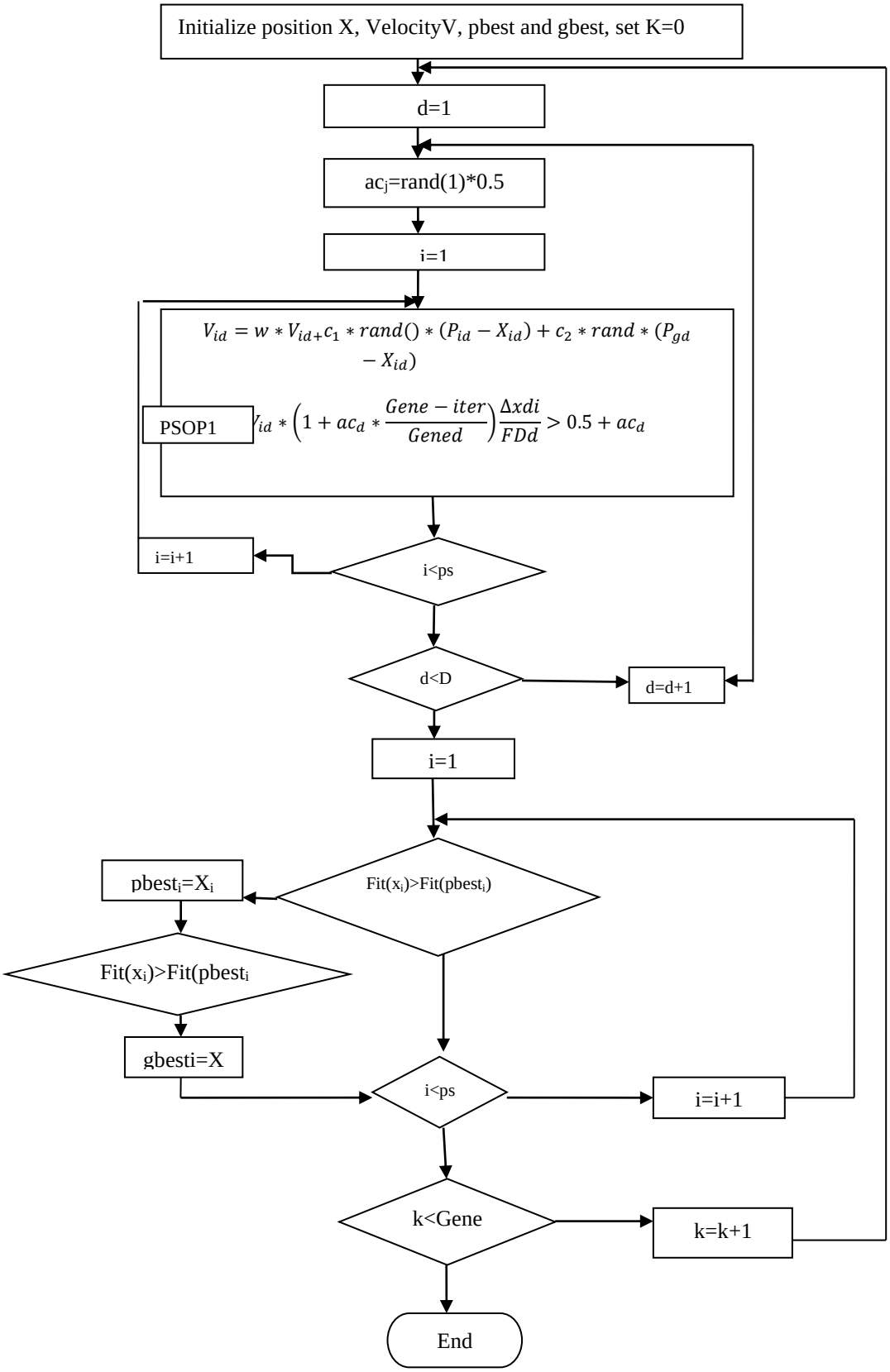


Figure 3.2: Flow chart for PSOP1

- Update particle position according to eq. 3.15 while maximum iterations or minimum criteria is not attained

Since the introduction of the PSO algorithm, several improvements have been suggested. In 1998, inertia weight was first proposed by Shi and Eberhart. The function of inertia weight is to balance global exploration and local exploitation. In the following year, Eberhart and Shi compared inertia weight with constriction factors and found that the constriction factors was better convergence than inertia weight [6].

In DAPSOs, we define the "long distance" as the distance from the particle to the global best beyond$(0.5 + \mathrm{ac}) * \mathrm{FD_d}$and the "short distance" as the distance from the particle to the global best is smaller than $(0.5 - \mathrm{ac}) * \mathrm{FD_d}$

In PSOP1, the particles far away from the global best should be given larger value of velocity so it may explore an unknown region, whereas those close to the global best should be given smaller value of velocity so that it may exploit the neighbor hood of the global best.

3.5 DYNAMIC AND ADJUSTABLE PSO:

In this section, we propose two improved algorithms called Dynamic and Adjustable Particle Swarm Optimization (DAPSO), PSOP1 and PSOP2. In DAPSOs,(PSOP1 and PSOP2) in order to adjust the velocity of each particle, all particles are calculated the distance from itself to the global best position by the following function.

$$\Delta xd_i = |(xd_i - x_{gbest})| \tag{3.16}$$

$$FD_d = \max(\Delta xd_i) \tag{3.17}$$

where x_{di} is the position of the i^{th} particle, x_{gbest} is the position of gbest. FD_d is the furthest distance from the particle to gbest. In DAPSO1, the velocity and position of each particle is adjusted by the following formulas:

$$V_{id} = w * V_{id+} c_1 * rand() * (P_{id} - X_{id}) + c_2 * rand * (P_{gd} - X_{id}) \tag{3.18}$$

$$ac = rand() * 0.5 \tag{3.19}$$

$$V_{new} = V_{id} * \left(1 + ac_d * \frac{Gene-iter}{Gened}\right)\frac{\Delta xdi}{FDd} > 0.5 + ac_d \quad (3.20)$$

$$V_{id} * \left(1 + ac_d * \frac{Gene-iter}{Gened}\right)\frac{\Delta xdi}{FDd} < 0.5 - ac_d \quad (3.21)$$

$$0.5 - ac <= \frac{\Delta xdi}{FDd} < 0.5 + ac_d \quad (3.22)$$

$$X_{id} = X_{id} + V_{new} \quad (3.23)$$

where X_{id} is updated by the velocity which is adjusted by the distance from particle to the global best and is the adjustment coefficient.PSOP1 and PSOP2 differ from the adjusting method. In PSOP2, the velocity and position of each particle is adjusted by the following formulas:

$$V_{id} = w * V_{id} + c_1 * rand() * (P_{id} - X_{id}) + c_2 * Rand * (P_{gd} - X_{id}) \quad (3.24)$$

where $ac = rand() * 0.5$

$$V_{new} = V_{id}\left(1 + \frac{rand()}{4} * \frac{Gene-iter}{Gened}\right)\frac{\Delta xdi}{FDd} > 0.5 + ac \quad (3.25)$$

$$V_{id}\left(1 + \frac{rand()}{4} * \frac{Gene-iter}{Gened}\right)\frac{\Delta xdi}{FDd} < 0.5 - ac \quad (3.26)$$

V_{id} $0.5 - ac <= \frac{\Delta xdi}{FDd} <= 0.5 + ac$ (3.27)

$$X_{id} = X_{id} + V_{new} \quad (3.28)$$

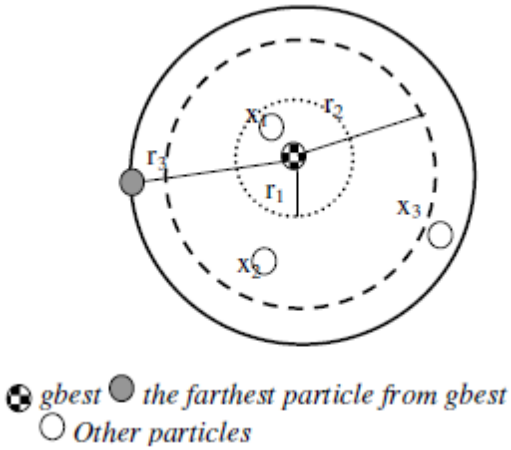


Figure3.3: Three radius of$(0.5 - ac) * FD_d$, $(0.5 + ac) * FD_d$, and FD_d .[11]

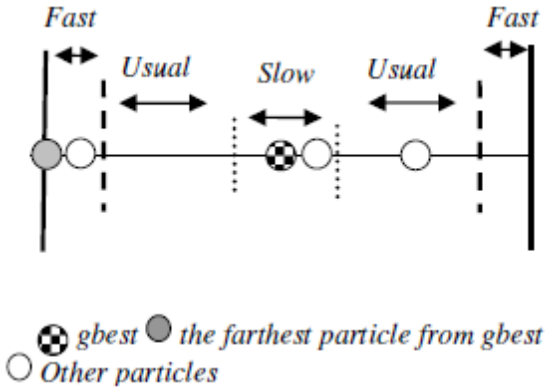


Figure 3.4: Adjust the velocity according to the distance from the particle to gbest[11]

In Fig.3.3 x_1, x_2 and x_3 all have difference distance from itself to global best position. X_1 drops within the radius of$(0.5 - ac) * FD_d$. The distance from x_2 to global best position is between $(0.5 - ac) * FD_d$and$(0.5 + ac) * FD_d$. x_3 drops beyond the radius of $(0.5 + ac) * FD_d$ in PSOP1.

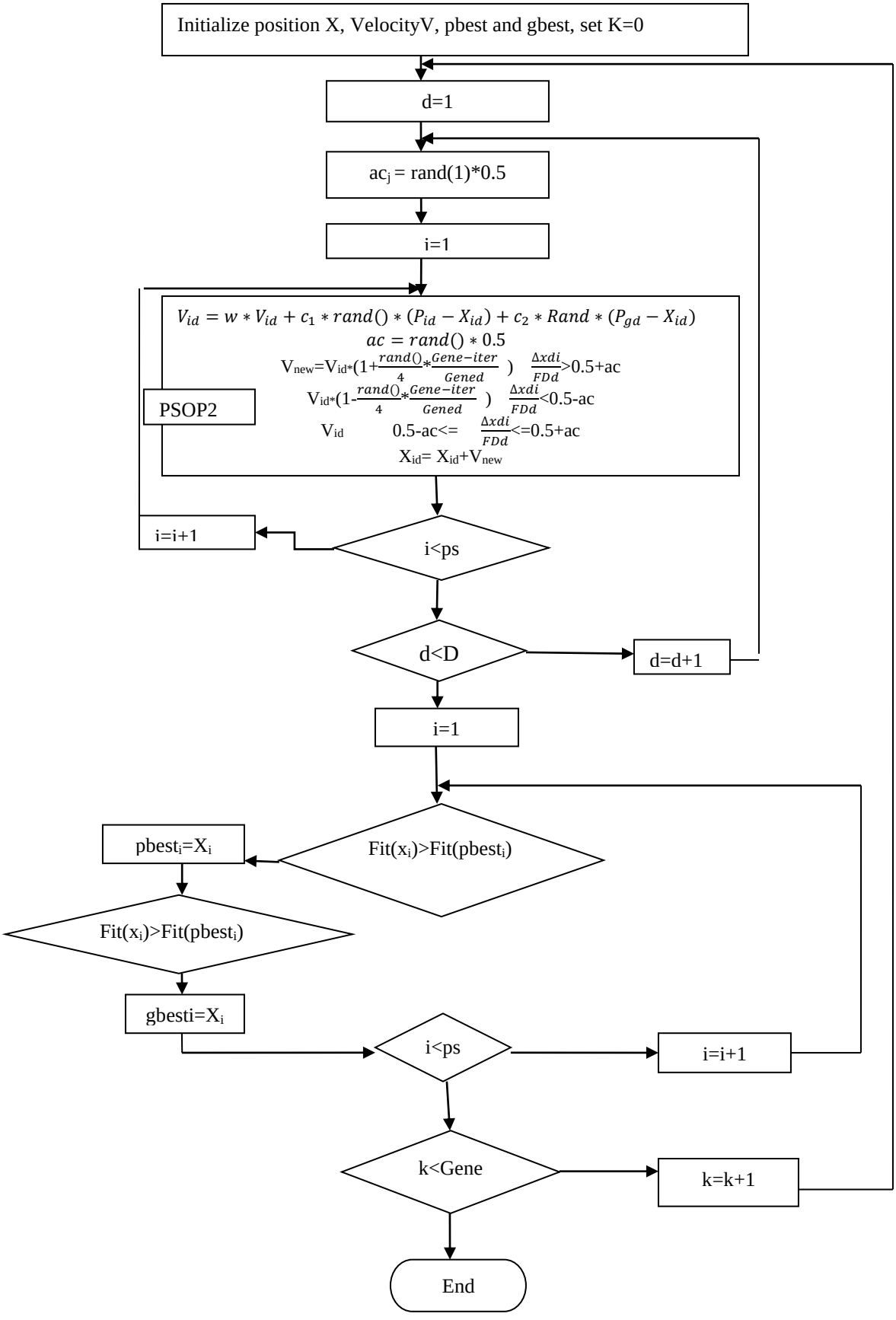


Figure 3.5: Flow chart for PSOP2

In PSOP2, if there were many particles far away from the global best position, then the velocities should be given a larger value. If there were many particles near from the global best position, then the velocities should be given a smaller value. PSOP1 only adjusts the velocity of the certain particle, but in PSOP2, the velocities of all particles are adjusted together.

The flowchart of PSOP2 is shown as follows.

- Initialization of a population of particles with random positions and velocities
- Evaluation of particles.
- Calculate the distance from each particle to the global best position and save the farthest distance in the memory.
- Adjust particle's velocity according to its distance from itself to the global best position.
- Update particle's position by the adjusted velocity.
- Repeat Step.2~Step.5 until termination criteria are met.

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

$$H\left(e^{jwk}\right) = \sum_{n=0}^{N} h(n)e - jwkn \tag{3.29}$$

Where $\omega_k = \frac{2\pi k}{N}$; $H\left(e^{jwk}\right)$ is the Fourier transform complex vector. This is the FIR filter's frequency response. The frequency is sampled in [0, π] with N points. Different kinds of

error fitness functions have been used in different literatures. An error function given by eq. 3.30 is the approximate error used in PM algorithm for filter design [16].

$$E(w) = G(w)[H_d(e^{jw}) - H_i(e^{jw})] \tag{3.30}$$

where $H_d(e^{jw})$ is the frequency response of the designed approximate filter.$H_i(e^{jw})$ is the frequency response of the ideal filter; G(ω) is the weighting function used to provide different weights for the approximate errors in different frequency bands. For ideal LP filter,$H_i(e^{jw})$ is given as, $H_i(e^{jw})$= 1 for $0 \leq \omega \leq \omega_c$; 0 otherwise (3.31)

Where ω_c is the cut-off frequency. The test function J1 is defined as

$$J1 = (abs(|E(\omega)|)) \quad (3.32)$$

For test function J2 the error function is

$$E(w) = G(w)[H_d(e^{jw}) - H_{pm}(e^{jw})] \quad (3.33)$$

For test function J2 equation 3.33 is used as error function

$$J2 = \max(abs(|E(\omega)|)) \quad (3.34)$$
$$0 < \omega \leq \omega p \ \& \ \omega s < \omega \leq 0.5$$

The major drawback of PM algorithm is that the ratio of $\frac{\delta p}{\delta s}$ is fixed. To improve the flexibility in the error function to be minimized, so that the desired level of δ_P and δ_S may be specified, the error function given in 3.33 has been considered as fitness function in many literatures [13][1]. The error fitness to be minimized using the evolutionary algorithms, is defined as:

$$J_3=\max(|E(\omega)|-\delta_P)+ \max(|E(\omega)|-\delta_S) \quad (3.35)$$
$$\omega\leq\omega_P \qquad \omega\geq\omega_S$$

where δ_P and δ_S are the ripples in the pass band and stop band, respectively, and p ω and s ω are pass band and stop band normalized cut-off frequencies, respectively. Since the coefficients of the linear phase positive symmetric even order filter are matched, the dimension of the problem is halved. This greatly reduces the computational burdens of the algorithms. In this chapter, a novel error fitness function given by eq. 3.35 has been adopted in order to achieve higher stop band attenuation and to have better control on the transition width. By using eq. 3.35, it is found that the proposed filter deign approach results in considerable improvement over the PM and other optimization techniques.

$$J_4=\sum abs[abs(|H_d(w)|-1)-\delta p]+\sum[abs(|H_d(w)-\delta s] \quad (3.36)$$

For the first term of eq.3.36, ω ∈pass band including a portion of the transition band and for the second term of eq. 3.37, ω ∈stop band including the rest portion of the transition band.

CHAPTER 4

IMPLEMENTATION OF ALGORITHM

4.1 IMPLEMENTATION OF ALGORITHM

In this chapter we have implemented PSOP1 (Particle swarm optimization 1) and PSOP2 (Particle swarm optimization 2) technique and the simulated results performed in MATLAB 7.10 for the design of FIR LP filter. The filter order (N) is taken as 20, which results in the number of coefficients as 21. The sampling frequency is taken to be f_s = 1Hz. The number of frequency samples is 128. Each algorithm is run for several times to obtain its best results.

In previous chapter we describe two methods of PSOP1 (Particle swarm optimization 1) and PSOP2 (Particle swarm optimization 2).

- PSOP1 is classical method.
- PSOP2 is modified diversity control, dynamic& adjustable PSO algorithm.

These algorithms are applied to get optimized solution of filter design. We have used four test function named as J1, J2, J3, J4. Their details are given in previous chapter in equation 3.32, 3.34, 3.35, 3.36.

The parameters of the filter to be designed using the DAPSO2 are: (pass band ripple) δ_P = 0.1, (stopband ripple) δ_s = 0.01. For the LP filter, pass band (normalized) edge frequency ω_p= 0.45, stop band (normalized) edge frequency ω_s = 0.5, transition width = 0.1. The filter has order 20. The parameters that are selected for PSO algorithm are given

- Population size=75
- Iteration cycle =500
- C_1=2.05, $C_{2=}$2.05
- Vimin=0.01,Vimax=1.0
- Wmax=1.0,Wmin=0.4
- Z=100

We have taken both PSO (Particle swarm optimization) algorithm PSOP1 (Particle swarm optimization1) &PSOP2 (Particle swarm optimization2) one by one for above mentioned filter parameters. The response is represented in term of four kinds of plots. These plots are

Convergence plot of PSO (Particle swarm optimization), Impulse response of filter, normalized frequency response & magnitude response of filter.

(a) Convergence plot

Convergence plot is drawn in terms of error values (cost) versus no. of iterations. For a good optimization method, this plot monotonically decreases & after a particular iteration, it becomes saturated. Lower the no. of iteration indicates, higher the speed of convergence of optimization technique. We have taken 500 iteration in above PSO algorithm.

(b) Impulse Response

Impulse response is defined by filter coefficient with respect to their weight in filter design differential equation. Since we have used a filter order of 20, there are 21 samples of filter coefficient. This response is in between h (n) (filter coefficient)) versus no. of samples n.

(c) Normalized frequency Response

This response is defined the 20log of the Fourier transform of h(n)(filer coefficient) and frequency. It indicates normalized frequency. It is in between 20log|H(jω)|db versus f Hz (frequency). In this plot we have also shown pass band & stop band ripples additionally.

(d) Magnitude Response

This response is defined the pass band & stop band magnitude & ripples. It is in between H(jω) & frequency. In both plot (c) & (d) X axis is frequency, since we have taken f_s as 1, that is why the frequency on X axis from 0 to 1.

4.1.1 Case-1(A)

TECHNIQUE	TEST FUNCTION
PSOP1 Particle swarm optimization 1	J1 $J1 = (abs(\lvert E(\omega)\rvert)$

The PSO approach is PSOP1 and the test functions are J1, J2, J3, J4 taken one by one in this case & their results are discussed below.

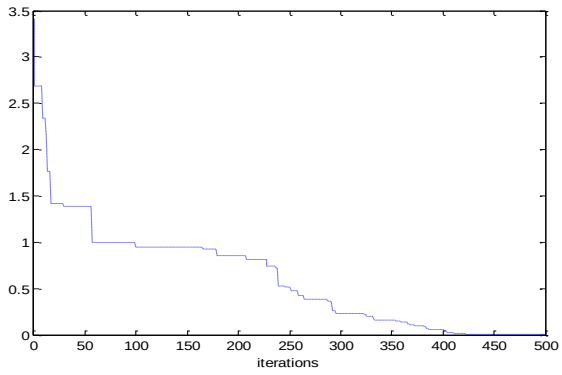


Fig4.1(a) Converge plot for PSOP1 and test function J1

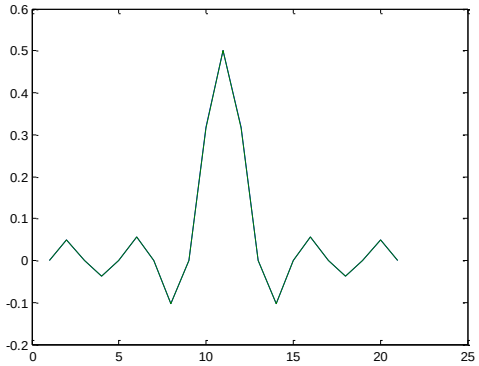


Fig4.1(b) Impulse response for PSOP1 and test function J1

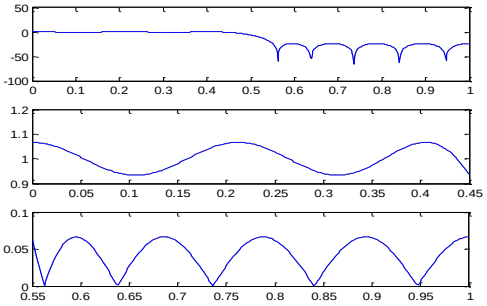


Fig4.1(c) Normalized frequency response(dB)for PSOP1 and test Function J1
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

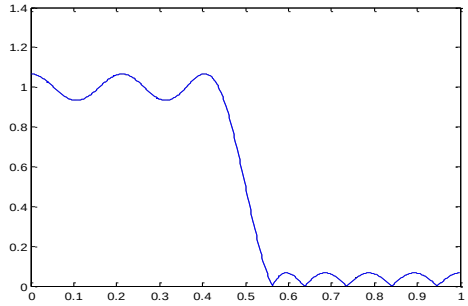


Fig4.1(d) Magnitude response for PSOP1 and test function J1
Dashed-PM
Solid- PSOP1

Figure 4.1: PSOP1 with test function J1

The Fig. 4.1 shown below represents the result better obtain after running simulation for several times but here shown only best 1.

Table 4.1: Filter coefficient for PSOP1 and test function J1

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test function J1
1	h(1)	0.0000
2	h(2)	0.0481
3	h(3)	-0.0000
4	h(4)	-0.0369
5	h(5)	-0.0000
6	h(6)	0.0573
7	h(7)	-0.0000
8	h(8)	-0.1022
9	h(9)	-0.0000
10	h(10)	0.3170
11	h(11)	0.5001

Fig. 4.1(a) it is convergence plot. In this figure the error or cost function is shown. Y-axis represents error in desired response. This error reduces as the no. of iteration increases. Initially the error is 3.5, after 400 iteration it is almost converges to zero. In this case we obtained error in between response obtain by PM (Parks McClellan) algorithm & our desired response. This case is only to check the performance of our PSO technique. In Fig.4.1 (b), there are impulse response of both h_{PM}(Parks McClellan impulse response) & h_{psop1},(PSOP1 impulse response)where h_{PM} is desired response obtained & $h_{psop1}$is optimized response obtained by PSOP1.However there are two results of $h_{PM}[n]$ & $h_{psop1}[n]$ but since PSO(Particle swarm optimization) has completely superimposed on desired response, so we are getting only single result.

This has occurred because in J1 test function we did not take any constraints on pass band & stop band. This indicates that PSO algorithm is capable of giving similar results as given by PM(Parks McClellan) algorithm. Similar results are also found for Fig.4.1(c) & 4.1(d).

We can see that Gain is high near about 0db or 1 for frequency less than 0.45 and that is pass band as shown in Fig.4.1(c) &Fig.4.1 (d).

Stop band is obtained after 0.55Hz in both figures. The middle plot of Fig. 4.1(c) represents stops band ripples. These stop band ripples range is 0.05.We want to get the stop band in range of 0.01.

Since the PM algorithm cannot minimize the result but in further cases we will minimize the result using constraints over the stop band for getting the optimized solution closer to ideal filter response. Table 4.1 shows value of filter coefficients obtain by this algorithm.

Case-1(B):

TECHNIQUE	TEST FUNCTION
PSOP1 Particle swarm optimization 1	J2 $J2 = \max\ (abs(\lvert E(\omega)\rvert)$ where $E(w) = G(w)[H_d(e^{jw}) - H_{pm}(e^{jw})]$

Fig.4.2A is representing the optimized result for test function J2, here we have applied constraints on stop bond. So the test function has included limits on δs =0.01 for the optimized solution for a desired response of an ideal filter hi[n].

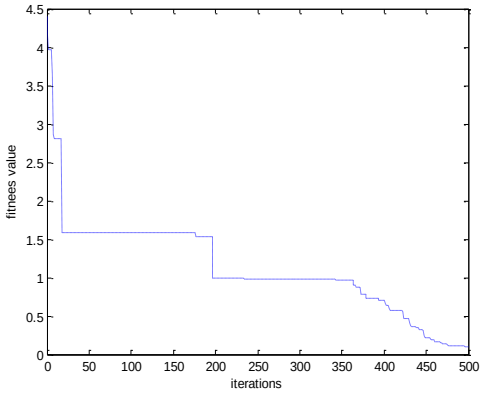


Fig4.2A(a) Converge plot for PSOP1 and test function J2

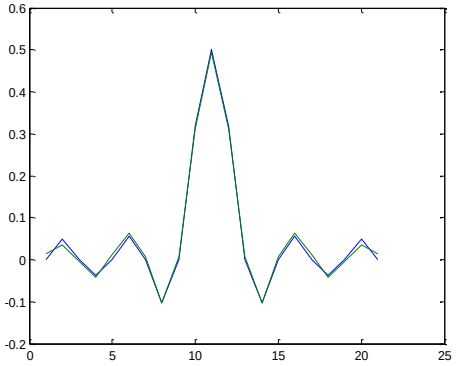


Fig4.2A(b) Impulse response for PSOP1 and test function J2

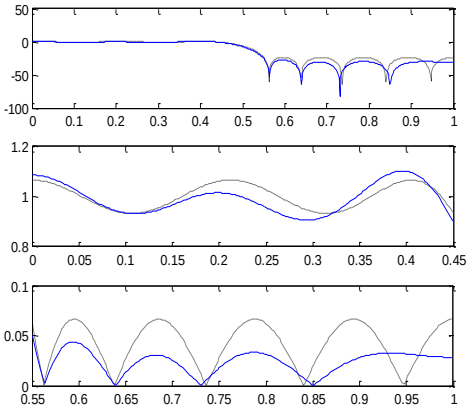


Fig4.2A(c) Normalized frequency response(dB) for PSOP1 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

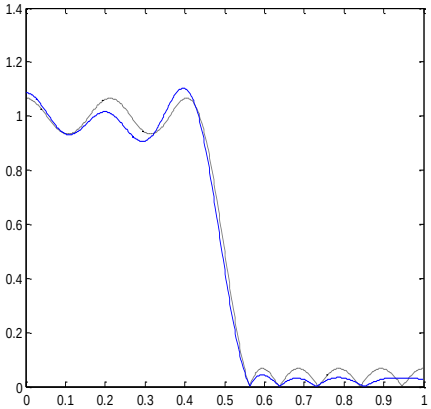


Fig4.2A(d) Magnitude response for PSOP1 and test function J2
Dashed-PM
Solid- PSOP1

Figure 4.2A: PSOP1 with test function J2

Table 4.2: Filter coefficient for PSOP1 and test function J2

S.NO	h(n) (Filter coefficient)	PSOP1(Particle swarm optimization with test function J2
1	h(1)	0.0136
2	h(2)	0.0348
3	h(3)	-0.0052
4	h(4)	-0.0413
5	h(5)	0.0112
6	h(6)	0.0626
7	h(7)	0.0065
8	h(8)	-0.1026
9	h(9)	0.0062
10	h(10)	0.3111
11	h(11)	0.4931

Fig4.2A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and convergence after 450 iteration but it is slightly higher than 0 and less than 0.1. The initial cost is 4.5. It indicates that PSO algorithm is gradually minimizing cost after some iteration.

Fig4.2A (b) shows the impulse response of hPM[n](Parks McClellan)& hpsop1, solid line is for PSOP1 and dashed line for PM. Both are showing small difference in filter coefficient values, it means that we are getting our response equivalent to PM (Parks McClellan).For getting the idea of stop band ripples we have shown frequency response in Fig.4.2A(c) (top), which shows the normalized frequency response. For frequency less than 0.45 Hz performance of both algorithms is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP1(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.2A(c)(middle),for pass band, both response are almost similar. In the stop band response in Fig.4.2A(c)(bottom), stop band ripples due to PSOP1 is almost half of the stop band

ripples(δs) solid of PM(dashed).Similar perception can also be drawn from Fig4.2A (d) using frequency response. In these figure δs in PSOP1 (solid) is less than δs of PM (dashed) algorithm.Fig.4.1(c) and Fig.4.1 (d) are also the best optimized result for case 1(A).For both figures we can see that obtained ripples are less than the ripples as obtained in PM (Parks McClellan) algorithm. In this way, we run our algorithm several times but here we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM (Parks McClellan) algorithm, for stop band constraints of LPF filter design.

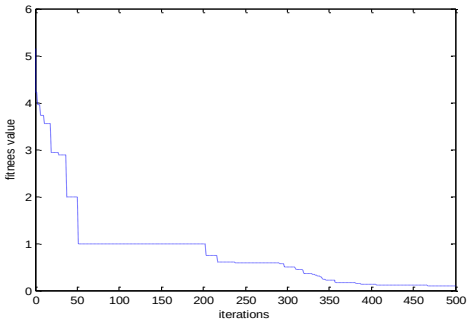


Fig.4.2B(a)Converge plot for PSOP1 and test function J2

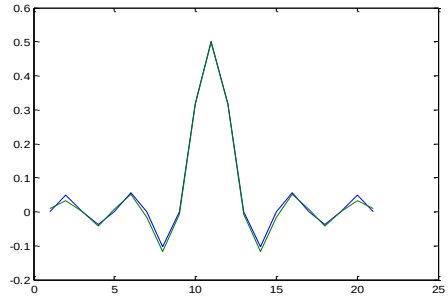


Fig4.2B(b) Impulse response for PSOP1 and test function J2

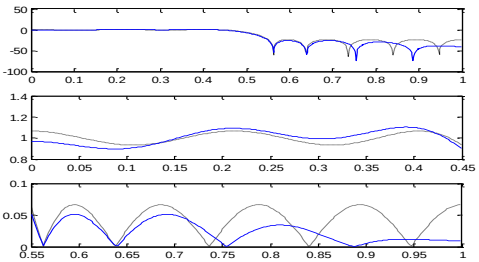


Fig4.2B(c) Normalized frequency response(dB)for PSOP1 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

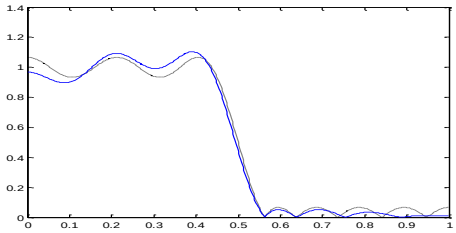


Fig4.2B(d) Magnitude response for PSOP1 and test function J2
Dashed-PM
Solid- PSOP1

Figure: 4.2B: PSOP1 with test function J2

Table 4.2C: Filter coefficient for PSOP1 and test function J2

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test functionJ2
1	h(1)	0.0102
2	h(2)	0.0336
3	h(3)	-0.0004
4	h(4)	-0.0428
5	h(5)	0.0081
6	h(6)	0.0508
7	h(7)	-0.0162
8	h(8)	-0.1168
9	h(9)	-0.0064
10	h(10)	0.3148
11	h(11)	0.4983

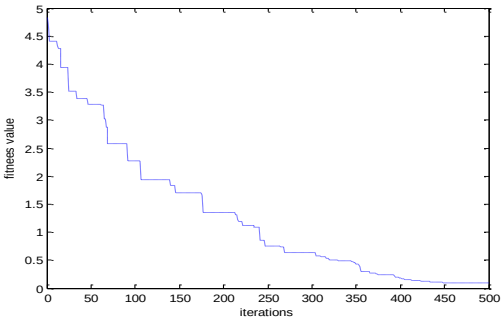


Fig4.2C(a)Converge plot for PSOP1 and test function J2

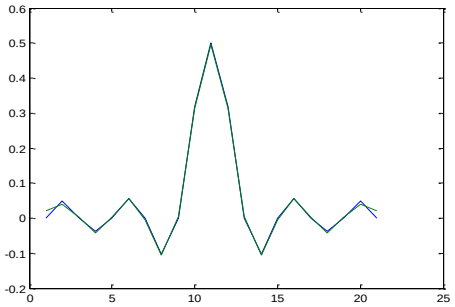


Fig4.2C(b) Impulse response for PSOP1 and test function J2

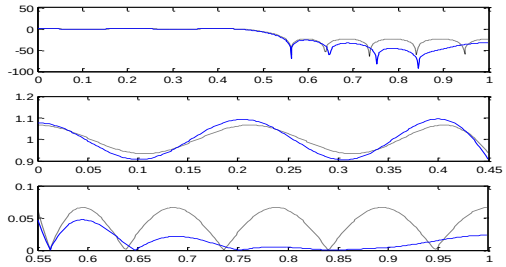


Fig4.2C(c) Normalized frequency response(dB)for PSOP1 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

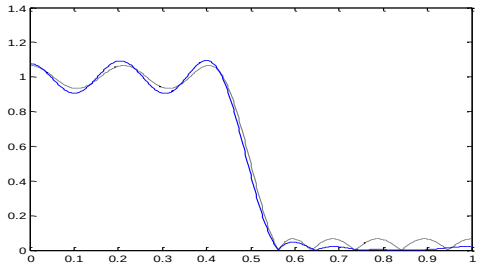


Fig4.2C(d) Magnitude response for PSOP1 and test function J2
Dashed-PM
Solid- PSOP1

Figure: 4.2C: PSOP1 with test function J2

Table 4.2(c): Filter coefficient for PSOP1 and test function J2

S.NO	h(n) (Filter coefficient)	PSOP1(Particle swarm optimization1) with test function J2
1	h(1)	0.0207
2	h(2)	0.0387
3	h(3)	0.0021
4	h(4)	-0.0423
5	h(5)	0.0029
6	h(6)	0.0572
7	h(7)	-0.0040
8	h(8)	-0.1048
9	h(9)	0.0055
10	h(10)	0.3146
11	h(11)	0.4958

Case-1(C):

TECHNIQUE	TESTFUNCTION
PSOP1 Particle swarm optimization1	J3 $J_3=\max(\lvert E(\omega)\rvert-\delta_P)+\max(\lvert E(\omega)\rvert-\delta_S)$ $\omega\leq\omega_P \qquad \omega\geq\omega_S$

Fig.4.3A represents the optimized result for test function J3, here we subtract the δ_p &δ_s from the test function J2 for the optimized solution for a desired response of an ideal filter hi[n].

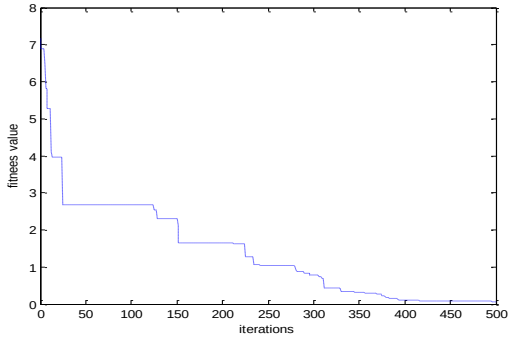


Fig4.3A(a)Converge plot for PSOP1 and test function J3

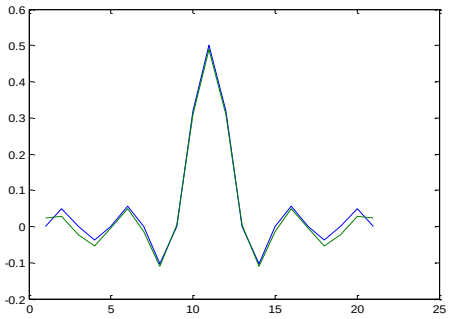


Fig.4.3A(b) Impulse response for PSOP1 and test function J3

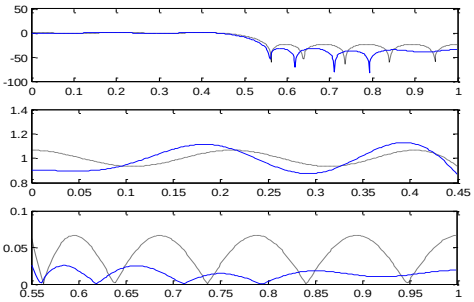


Fig.4.3A(c) Normalized frequency response(dB)for PSOP1 and test Function J3
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

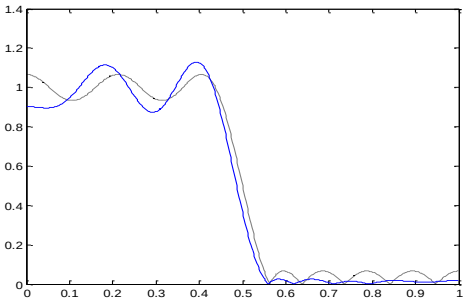


Fig.4.3A(d) Magnitude response for PSOP1 and test function J3
Dashed-PM
Solid- PSOP1

Figure 4.3A: PSOP1 with test function J3

Table 4.3(a) Filter coefficient for PSOP1 and test function J3

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test function J3
1	h(1)	0.0232
2	h(2)	0.0282
3	h(3)	-0.0231
4	h(4)	-0.0545
5	h(5)	-0.0035
6	h(6)	0.0489
7	h(7)	-0.0145
8	h(8)	-0.1100
9	h(9)	0.0043
10	h(10)	0.3084
11	h(11)	0.4880

Fig.4.3A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and convergence after 450 iteration but it is slightly higher than 0 and less than 0.1. The initial cost is7.0 it indicates that PSO algorithm is gradually minimizing cost after some iterations.

Fig.4.3A (b) shows the impulse response of $h_{PM}[n]$& $hpsop_{1,}$ solid line is for PSOP1 (Particle swarm optimization1) and dashed line for PM (Parks McClellan).Both are showing small difference in filter coefficient values. It means that we are getting response equivalent to PM.For getting the idea of stop band ripples we have shown frequency response in Fig.4.3A(c)(top) shows the normalized frequency response for frequency less than 0.45 Hz, in which performance of both algorithm is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP1(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.3A(c)(middle),for pass band, both response are almost similar. We can see the stop band

response in Fig.4.3A(c)(bottom),in which stop band ripples due to PSOP1 is almost half of the stop band ripples(δs)(solid) of PM(dashed).

Similar perception can also be drawn from Fig.4.3A (d) using frequency response. In this figure δs in PSOP1 (solid)is less than δs of PM(dashed)algorithm.

Fig.4.2A(c) and Fig.4.2A(d)are also the best optimized result for case 1(B).For both figures we can see that ripples obtained is less than the ripples as obtained in PM algorithm. In this way, we run our algorithm several times but here we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM algorithm, for our stop band constraints of LPF filter design.

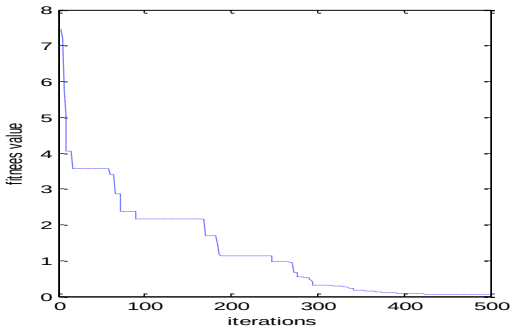


Fig.4.3B(a)Converge plot for PSOP1 and test function J3

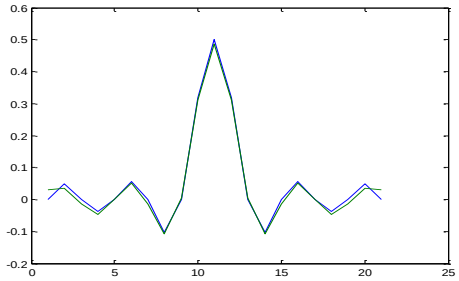


Fig4.3B(b) Impulse response for PSOP1 and test function J3

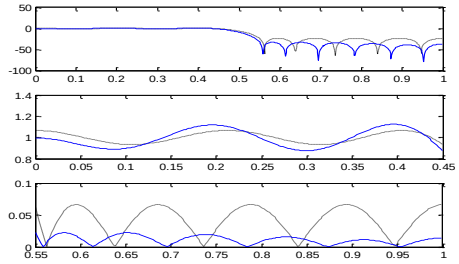


Fig.4.3B(c) Normalized frequency response(dB)for PSOP1 and test Function J3
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

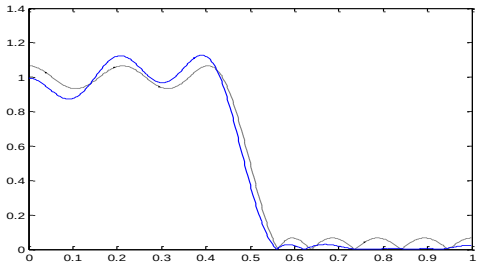


Fig.4.3B(d) Magnitude response for PSOP1 and test function J3
Dashed-PM
Solid- PSOP1

Figure 4.3B:PSOP1 with test function J3

Table 4.3(b) Filter coefficient for PSOP1and test function J3

S.NO	h(n) (Filter coefficient)	PSOP1(Particle swarm ptimization1) with test function J3
1	h(1)	0.0310
2	h(2)	0.0358
3	h(3)	-0.0133
4	h(4)	-0.0468
5	h(5)	0.0008
6	h(6)	0.0519
7	h(7)	-0.0134
8	h(8)	-0.1074
9	h(9)	0.0043
10	h(10)	0.3118
11	h(11)	0.4860

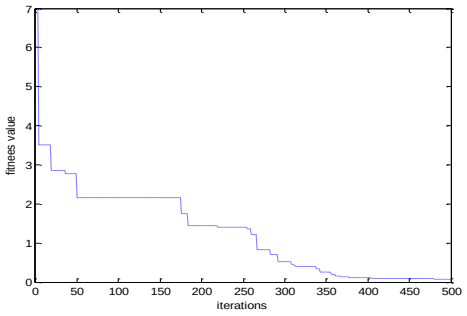


Fig4.3C(a)Converge plot for PSOP1 and test function J3

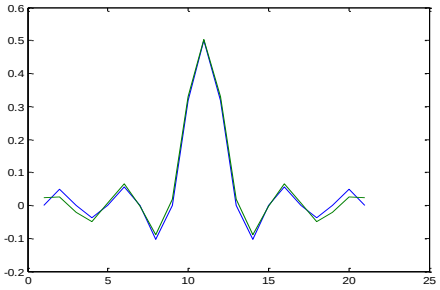


Fig.4.3C(b) Impulse response for PSOP1 and test function J3

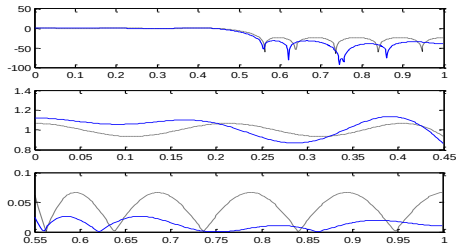


Fig4.3C(c) Normalized frequency response(dB)for PSOP1 and test Function J3
(above) passband
(below) stopband
Dashed-PM
Soliod- PSOP1

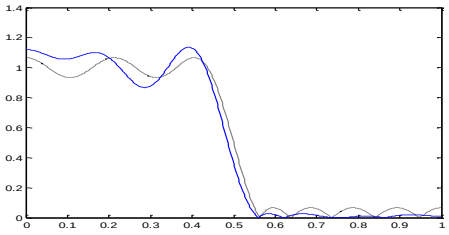


Fig4.3C(d) Magnitude response for PSOP1 and test function J3
Dashed-PM
Solid- PSOP1

Figure 4.3 C: PSOP1 with test function J3

Table 4.3(c) Filter coefficient for PSOP1 and test function J3

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test function J3
1	h(1)	0.0223
2	h(2)	0.0250
3	h(3)	-0.0217
4	h(4)	-0.0503
5	h(5)	0.0068
6	h(6)	0.0657
7	h(7)	-0.0012
8	h(8)	-0.0880
9	h(9)	0.0197
10	h(10)	0.3303
11	h(11)	0.5035

Case-1(D):

TECHNIQUE	TESTFUNCTION
PSOP1 Particle swarm optimization1	J4 $J_4=\sum abs[abs(\lvert H_d(w) \rvert - 1)\text{-}\delta p)]+\sum[abs(\lvert H_d(w) \rvert\text{-}\delta s)]$

Fig.4.4A represents the optimized result for test function J4, here we sum absolute value of test function J3 for the optimized solution for a desired response of an ideal filter hi[n].

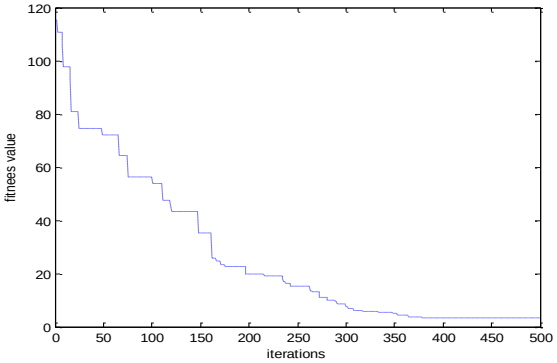


Fig.4.4A(a)Converge plot for PSOP1 and test function J4

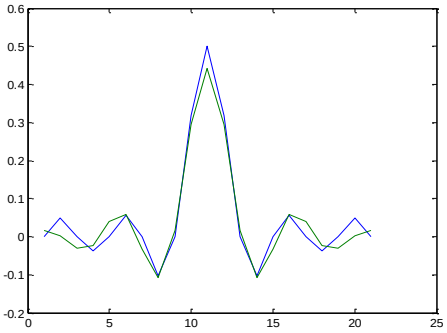


Fig.4.4A(b) Impulse response for PSOP1 and test function J4

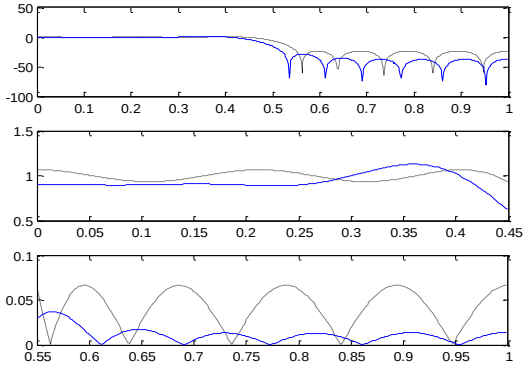


Fig.4.4A(c) Normalized frequency response(dB)for PSOP1 and test Function J4
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

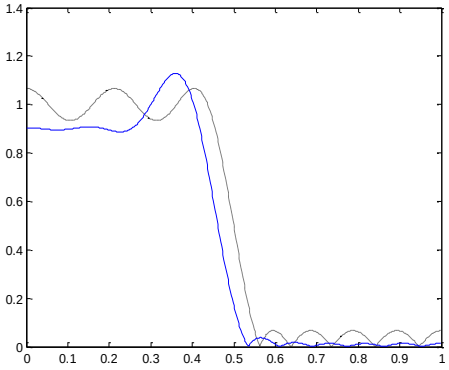


Fig.4.4A(d) Magnitude response for PSOP1 and test function J4
Dashed-PM
Solid- PSOP1

Figure 4.4A:PSOP1 with test function J4

Table 4.4(a) Filter coefficient for PSOP1 and test function J4

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test function J4
1	h(1)	0.0163
2	h(2)	0.0022
3	h(3)	-0.0316
4	h(4)	-0.0243
5	h(5)	0.0399
6	h(6)	0.0593
7	h(7)	-0.0323
8	h(8)	-0.1083
9	h(9)	0.0162
10	h(10)	0.2936
11	h(11)	0.4421

Fig.4.4A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and does not convergence after greater than500 iteration because it is sum of absolute value. It is slightly higher than 0 and less than 4. The initial cost was 100 it indicates that PSO algorithm is gradually minimizing.

Fig.4.4A (b) shows the impulse response of $h_{PM}[n]$ (Parks McClellan) & h_{psop1}(Particle swarm optimization). Solid line is for PSOP1 and dashed line for PM. Both are showing small difference in filter coefficient values. It means that we are getting our response approximately equivalent to PM.For getting the idea of stop band ripples we have shown frequency response in Fig.4.4A(c)(top) ,shows the normalized frequency response for frequency less than 0.45 Hz. Performance of both algorithm is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP1(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.4A(c)(middle),for pass band, both response are almost similar. We can

see the stop band response in Fig.4.4A(c) (bottom), in which response of stop band ripples due to PSOP1 is almost half of the stop band ripples (δs) (solid) of PM (dashed).

Similar perception can also be drawn from Fig4.4A (d) using frequency response. In this figure δs in PSOP1 (solid)is less than δs of PM(dashed)algorithm.

Fig4.2A(c) and Fig.4.2A (d) are also the best optimized result for case 1(c).For both figures we can see that is obtained ripples are less than the ripples as obtained in PM algorithm. In this way, we run our algorithm several times but here we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM algorithm, for our stop band constraints of LPF filter design

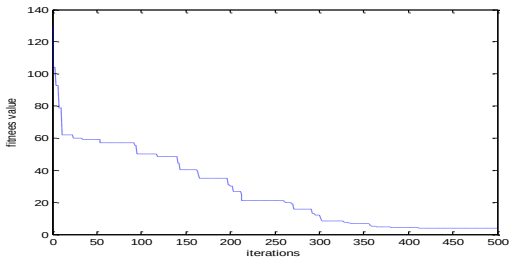


Fig4.4B(a)Converge plot for PSOP1 and test function J4

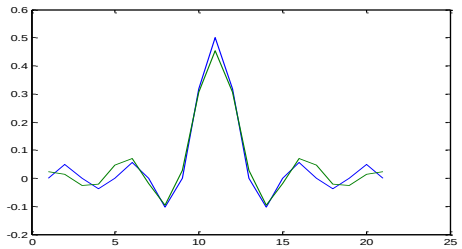


Fig4.4B(b) Impulse response for PSOP1 and test function J4

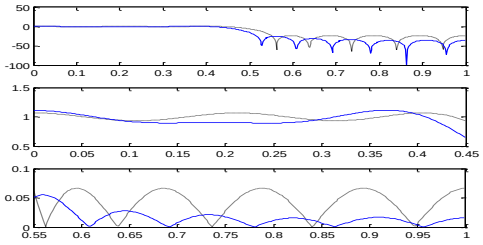


Fig.4.4B(c) Normalized frequency response(dB)for PSOP1 and test Function J4
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1

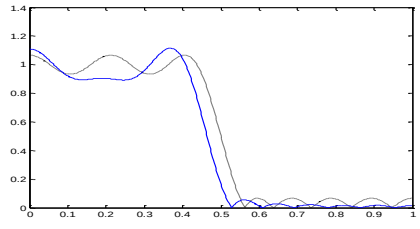


Fig.4.4B(d) Magnitude response for PSOP1 and test function J4
Dashed-PM
Solid- PSOP1

Figure 4.4B:PSOP1 with test function J4

Table 4.4(b): Filter coefficient for PSOP1 and test function J4

S.NO	h(n) (Filter coefficient)	PSOP1 (Particle swarm optimization1) with test function J4
1	h(1)	0.0243
2	h(2)	0.0134
3	h(3)	-0.0263
4	h(4)	-0.0213
5	h(5)	0.0469
6	h(6)	0.0709
7	h(7)	-0.0187
8	h(8)	-0.0951
9	h(9)	0.0288
10	h(10)	0.3055
11	h(11)	0.4527

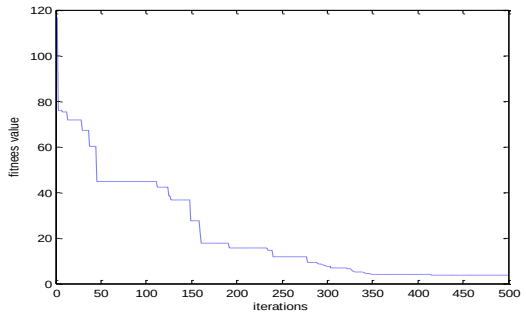
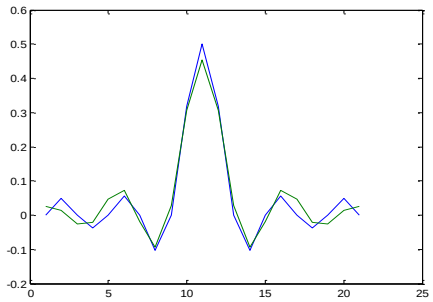
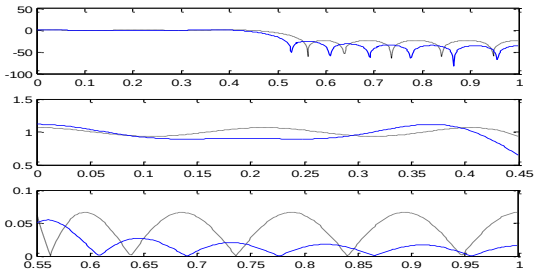
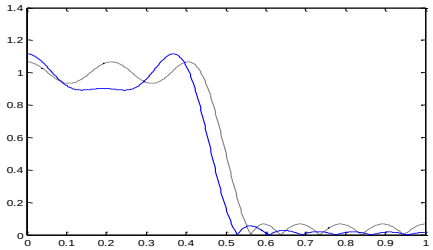
<table>
<tr><td> 
</td><td>  </td></tr>
<tr><td>Fig4.4C(a)Converge plot for PSOP1 and test function J4</td><td>Fig4.4C(b) Impulse response for PSOP1 and test function J4</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>Fig4.4C(c) Normalized frequency response(dB)for PSOP1 and test Function J4
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP1</td><td>Fig4.4C(d) Magnitude response for PSOP1 and test function J4
Dashed-PM
Solid- PSOP1</td></tr>
</table>

Figure 4.4C: PSOP1 with test function J4

Table 4.4(c) Filter coefficient for PSOP1 and test function J4

S.NO	h(n) (Filter coefficient)	PSOP1(Particle swarm optimization1) with test function J4
1	h(1)	0.0247
2	h(2)	0.0136
3	h(3)	-0.0259
4	h(4)	-0.0209
5	h(5)	0.0470
6	h(6)	0.0716
7	h(7)	-0.0184
8	h(8)	-0.0947
9	h(9)	0.0290
10	h(10)	0.3055
11	h(11)	0.4533

4.1.2Case-2(A):

The PSO approach is PSOP2 and the test functions are J1, J2, J3, J4 taken one by one in these case & their results are discussed below.

TECHNIQUE	TEST FUNCTION
PSOP2 Particle swarm optimization2	J1 $J1 = (abs(\lvert E(\omega)\rvert)$

The Fig. 4.5 shown below represents the result better obtain after running simulation for several times but here shown only best 1.

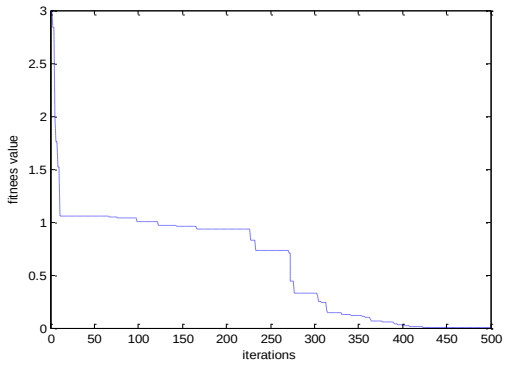


Fig.4.5(a)Converge plot for PSOP2 and test function J1

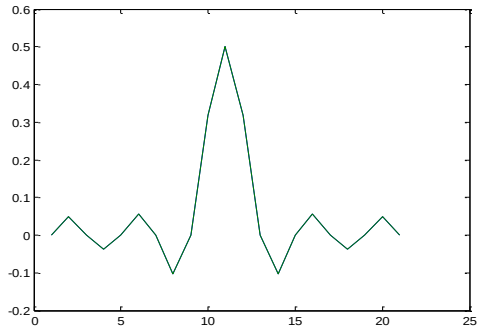


Fig.4.5(b) Impulse response for PSOP2 and test function J1

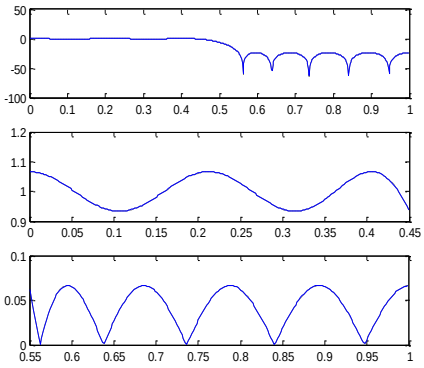


Fig4.5(c) Normalized frequency response(dB)for PSOP2 and test Function J1
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

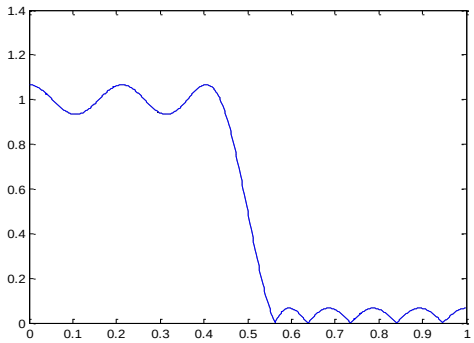


Fig4.5(d) Magnitude response for PSOP2 and test function J1
Dashed-PM
Solid- PSOP2

Figure 4.5 :PSOP2 with test function J1

Table 4.5 Filter coefficient for PSOP2 and test function J1

S.NO	h(n) (Filter coefficient)	PSOP2(Particle swarm optimization2) with test function J1
1	h(1)	-0.0000
2	h(2)	0.0480
3	h(3)	-0.0000
4	h(4)	-0.0369
5	h(5)	-0.0000
6	h(6)	0.0572
7	h(7)	-0.0000
8	h(8)	-0.1022
9	h(9)	0.0000
10	h(10)	0.3170
11	h(11)	0 .5000

Fig.4.5 (a) represents convergence plot.In this figure we can see that error or cost function. X-axis represents error in desired response. This error reduces as the no. of iteration increases. Initially the error is 3.0.After 400 iteration it is almost converges to zero. In this case we found error in between response obtain by PM (Parks McClellan) algorithm & our desired response. This case is only to check the performance of our PSO technique. In Fig4.5 (b),there are impulse response of both h_{PM}(Parks McClellan)& h_{psop2}(Particle swarm optimization 2),where $h_{PM}$is desired response obtained & $h_{psop2}$is optimized response obtained by PSOP2.However there are two result of $h_{PM}[n]$ & $h_{psop2}[n]$ but since PSOP2 has completely superimposed on desired response, so we are getting only single result.

This has occurred because in J1 test function we did not take any constraints on pass band & stop band. This indicates that PSO algorithm is capable of giving similar results as given by PM algorithm. Similar results are also found for Fig.4.5(c) & 4.5(d).

Stop band is obtained after 0.55Hz in both figures. The middle plot of Fig. 4.5(c) represents stops band ripples. These stop band ripples range is 0.05.We want to get the stop band in range of 0.01.

Since the PM algorithm cannot minimize the result but in further cases we will minimize the result using constraints over the stop band for getting the optimized solution closer to ideal filter response.

Case-2(B):

TECHNIQUE	TEST FUNCTION
PSOP2 Particle swarm optimization 2	J2 $J2 = \max\,(abs(\lvert E(\omega)\rvert)$ where $E(w) = G(w)[H_d(e^{jw}) - H_{pm}(e^{jw})]$

Fig.4.6A represents the optimized result for test function J2; here we have applied constraints on stop bond. So the test function has included limits on δs =.010 for the optimized solution for a desired response of an ideal filter hi[n].

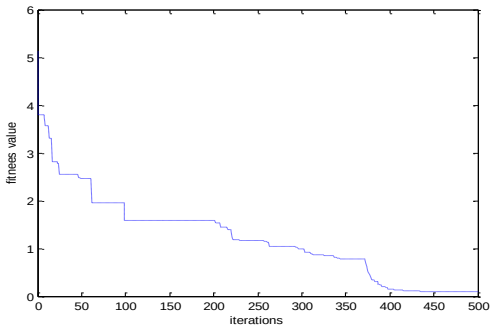


Fig4.6A(a)Converge plot for PSOP2 and test function J2

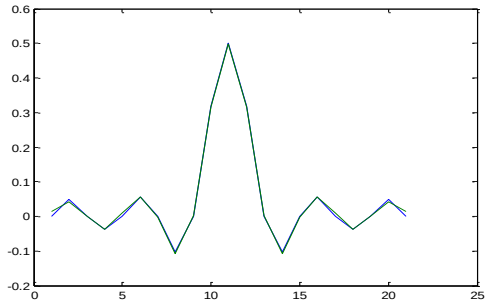


Fig4.6A(b) Impulse response for PSOP2 and test function J2

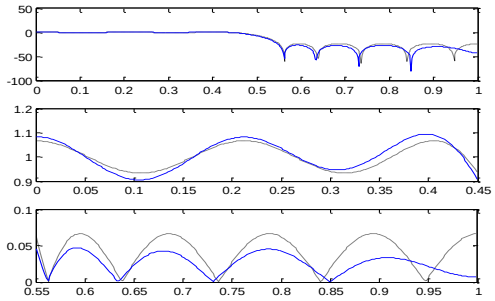


Fig4.6A(c) Normalized frequency response(dB)for PSOP2 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

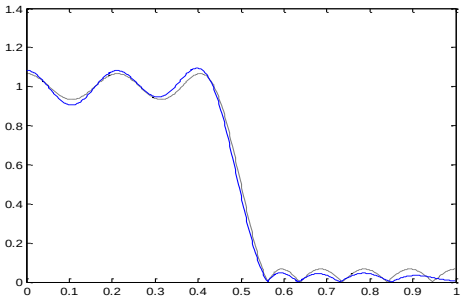


Fig4.6A(d) Magnitude response for PSOP2 and test function J2
Dashed-PM
Solid- PSOP2

Fig.4.6A: PSOP2 with test function J2

Table 4.6(a) Filter coefficient for PSOP2 and test function J2

S.NO	h(n) (Filter coefficient)	PSOP2 (Particle swarm optimization2) with test function J2
1	h(1)	0.0133
2	h(2)	0.0427
3	h(3)	-0.0000
4	h(4)	-0.0373
5	h(5)	0.0097
6	h(6)	0.0557
7	h(7)	-0.0032
8	h(8)	-0.1084
9	h(9)	0.0032
10	h(10)	0.3164
11	h(11)	0.4992

Fig.4.6A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and convergence after 450 iteration but it is slightly higher than 0 and less than 0.1. The initial cost is4.0. It indicates that PSO algorithm is gradually minimizing cost after some iteration.

Fig.4.6A (b) shows the impulse response of hPM[n] & hpsop2 solid line is for PSOP2 and dashed line for PM. Both are showing small difference in filter coefficient values. It means that we are getting our response equivalent to PM. For getting the idea of stop band ripples we have shown frequency response in Fig4.6A(c) (i), which shows the normalized frequency response for frequency less than 0.45 Hz, performance of both algorithms is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP2(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.6A(c)(middle),for pass band, both response are almost similar. We can see the stop band response in Fig.4.6A(c) (bottom), in which

the response of stop band ripples due to PSOP2 is almost half of the stop band ripples (δs) solid of PM (dashed).

Similar perception can also be drawn from Fig.4.6A (d) using frequency response. In this figure, δs, in PSOP2 (solid) is less than δs of PM (dashed) algorithm.

Fig4.6A(c) and Fig.4.6A (d)are also the best optimized result for case 2(B).For both figures we can see that is obtained ripples are less than the ripples as obtained in PM algorithm. In this way, we run algorithm several times but here we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM algorithm, for our stop band constraints of LPF filter design.

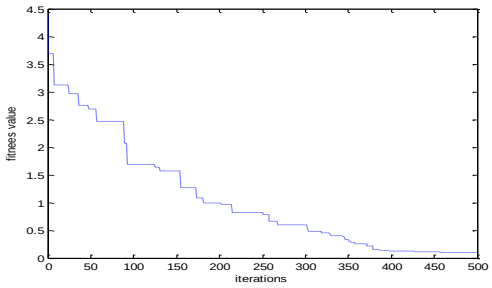


Fig.4.6B(a)Converge plot for PSOP2 and test function J2

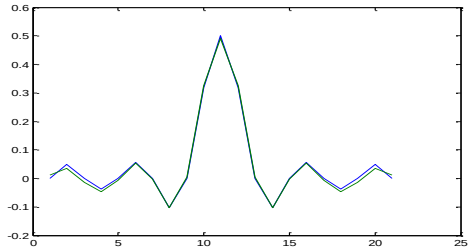


Fig.4.6B(b) Impulse response for PSOP2 and test function J2

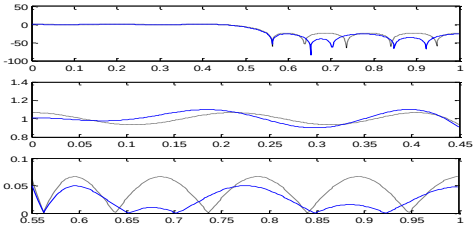


Fig.4.6B(c) Normalized frequency response(dB)for PSOP2 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

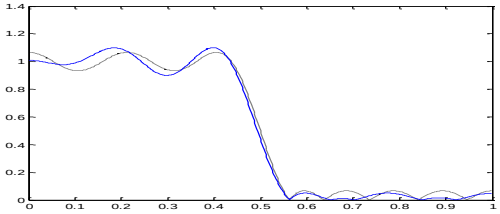


Fig4.6B(d) Magnitude response for PSOP1 and test function J2
Dashed-PM
Solid- PSOP2

Fig.4.6B: PSOP2 with test function J2

Table 4.6(b) Filter coefficient for PSOP2and test function J2

S.No.	h(n) (Filter coefficient)	PSOP2(Particle swarm optimization2) with test function J2
1	h(1)	0.0119
2	h(2)	0.0343
3	h(3)	-0.0138
4	h(4)	-0.0470
5	h(5)	-0.0063
6	h(6)	0.0545
7	h(7)	-0.0025
8	h(8)	-0.1031
9	h(9)	0.0048
10	h(10)	0.3256
11	h(11)	0.4923

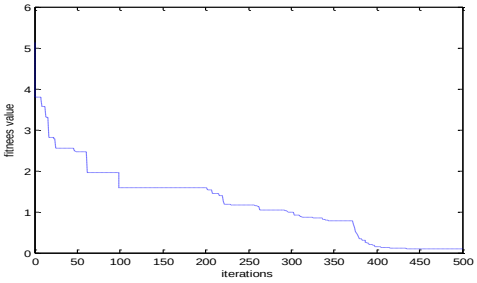


Fig4.6C(a)Converge plot for PSOP2 and test function J2

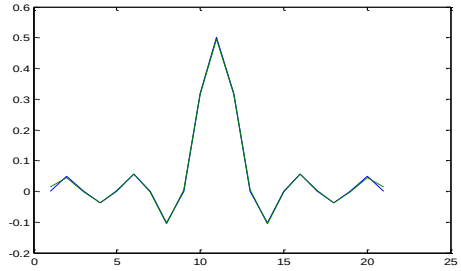


Fig4.6C(b) Impulse response for PSOP2 and test function J2

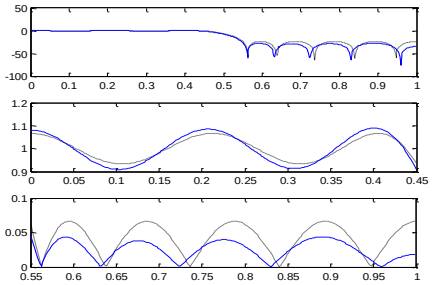


Fig4.6C(c) Normalized frequency response(dB)for PSOP2 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

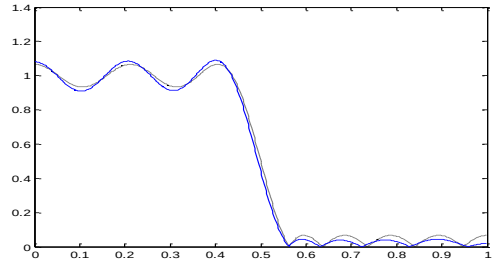


Fig4.6C(d) Magnitude response for PSOP2 and test function J2
Dashed-PM
Solid- PSOP2

Fig.4.6C PSOP2 with test function J2

Table 4.6(c) Filter coefficient for PSOP2 and test function J2

S.NO	h(n) (Filter coefficient)	PSOP 2 (Particle swarm optimization2) with test function J2
1	h(1)	0.0150
2	h(2)	0.0452
3	h(3)	-0.0032
4	h(4)	-0.0372
5	h(5)	0.0032
6	h(6)	0.0567
7	h(7)	-0.0025
8	h(8)	-0.1044
9	h(9)	0.0048
10	h(10)	0.3147
11	h(11)	0.4968

Case-2(C):

TECHNIQUE	TESTFUNCTION
PSOP2 Particle swarm optimization2	J3 $J_3=\max(\lvert E(\omega)\rvert-\delta_P)+\max(\lvert E(\omega)\rvert-\delta_S)$ $\omega\leq\omega_P \qquad \omega\geq\omega_S$

Fig.4.7A represents the optimized result for test function J3, here we subtract the δ_p &δ_s from the test function J2 for the optimized solution for a desired response of an ideal filter hi[n].

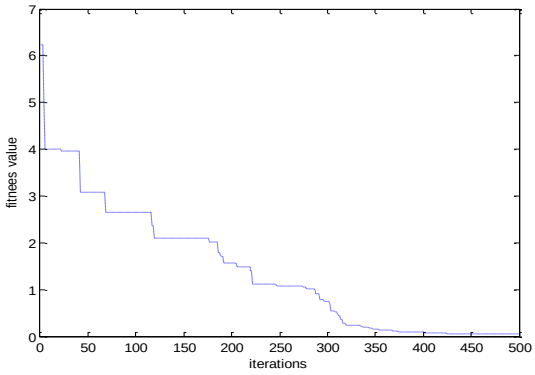


Fig4.7A(a)Converge plot for PSOP2 and test function J3

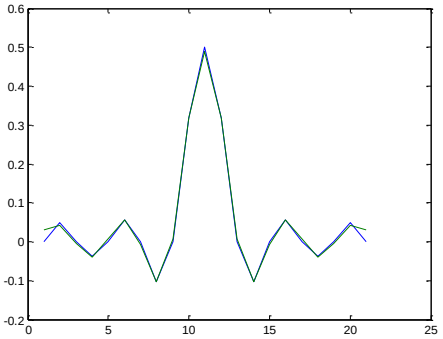


Fig4.7A(b) Impulse response for PSOP2 and test function J3

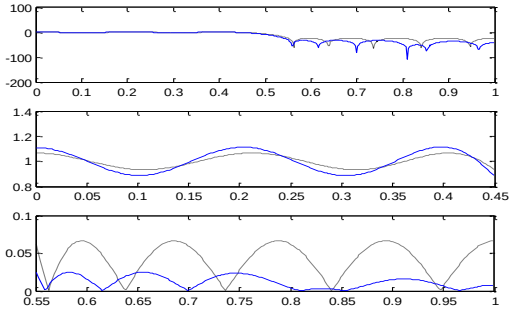


Fig4.7A(c) Normalized frequency response(dB)for PSOP2 and test Function J3
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

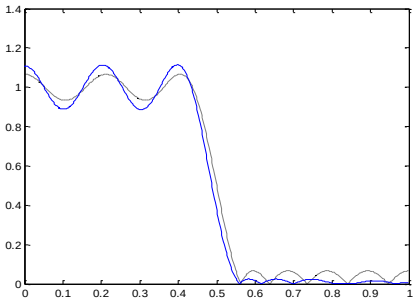


Fig4.7A(d) Magnitude response for PSOP2 and test function J3
Dashed-PM
Solid- PSOP2

Fig.4.7A PSOP2 with test function J3

Table 4.7(a) Filter coefficient for PSOP2 and test function J3

S.NO	h(n) (Filter coefficient)	PSOP2(Particle swarm optimization1) with test function J3
1	h(1)	0.0315
2	h(2)	0.0422
3	h(3)	-0.0045
4	h(4)	-0.0395
5	h(5)	0.0073
6	h(6)	0.0564
7	h(7)	-0.0072
8	h(8)	-0.1021
9	h(9)	0.0071
10	h(10)	0.3185
11	h(11)	0.4900

Fig4.7A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and convergence after 450 iteration but it is slightly higher than 0 and less than 0.1. The initial cost was 6.5 to 7.0 It indicates that PSO algorithm is gradually minimizing cost after some iterations.

Fig.4.7A (b) shows the impulse response of $h_{PM}[n]$ (Parks McClellan) & h_{psop2} (Particle swarm optimization2). Solid line is for PSOP2 and dashed line for PM. Both are showing small difference in filter coefficient values. It means that we are getting our response equivalent to PM. For getting the idea of stop band ripples we have shown frequency response in Fig.4.7A(c) (i) shows the normalized frequency response, for frequency less than 0.45 Hz, performance of both algorithm is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP2(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.7A(c)(middle)for pass band, here both response are almost similar. We can see the stop

band response in Fig.4.7A(c) (bottom), in which the response of stop band ripples due to PSOP2 is almost half of the stop band ripples (δs) (solid) of PM (dashed).

Similar perception can also be drawn from Fig4.7A (d) using frequency response. In these figure δs in PSOP2 (solid)is less than δs of PM (dashed) algorithm.

Fig4.7A(c) and Fig.4.7A (d) are also the best optimized result for case 2(C).For both figures we can see that is ripples obtained is less than the ripples as obtained in PM algorithm. In this way, we run our algorithm several times but we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM algorithm, for our stop band constraints of LPF filter design.

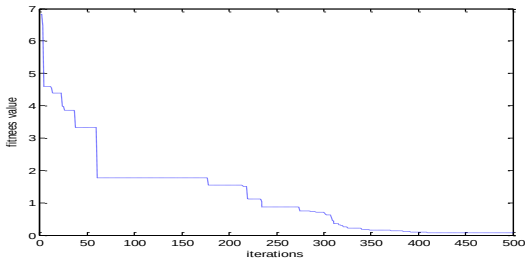
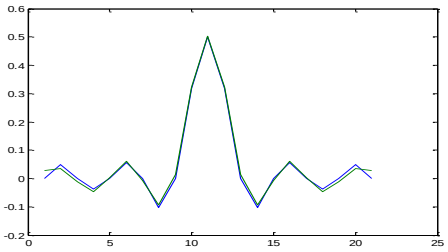
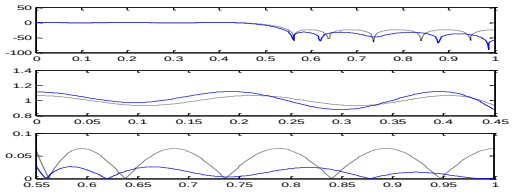
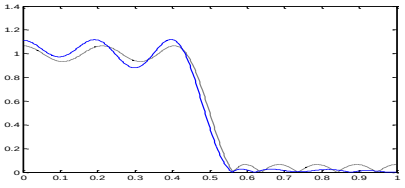
<table>
<tr><td> 
</td><td>  </td></tr>
<tr><td>Fig4.7B(a)Converge plot for PSOP2and test function J3</td><td>Fig4.7B(b) Impulse response for PSOP2 and test function J3</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>Fig.4.7B(c) Normalized frequency response(dB)for PSOP2 and test Function J3
(above) passband
(below) stopband
Dahed-PM
Solid- PSOP2</td><td>Fig.4.7B(d) Magnitude response for PSOP2 and test function J3
Dashed-PM
Solid- PSOP2</td></tr>
</table>

Fig4.7B PSOP2 with test function J3

Table 4.7(b) Filter coefficient for PSOP2 and test function J3

S.NO	h(n) (Filter coefficient)	PSOP2 (Particle swarm optimization2) with test function J3
1	h(1)	0.0280
2	h(2)	0.0346
3	h(3)	-0.0109
4	h(4)	0.0458
5	h(5)	0.0030
6	h(6)	0.0602
7	h(7)	-0.0076
8	h(8)	-0.0947
9	h(9)	0.0148
10	h(10)	0.3236
11	h(11)	0.5026

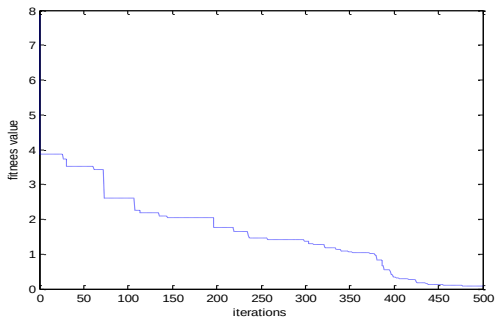


Fig4.7C(a)Converge plot for PSOP2 and test function J3

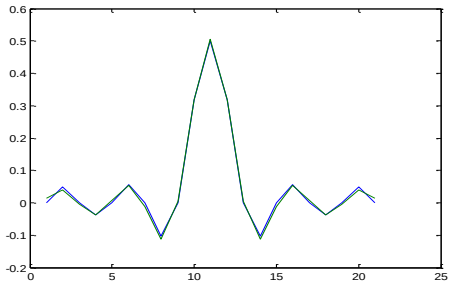


Fig4.7C(b) Impulse response for PSOP2 and test function J3

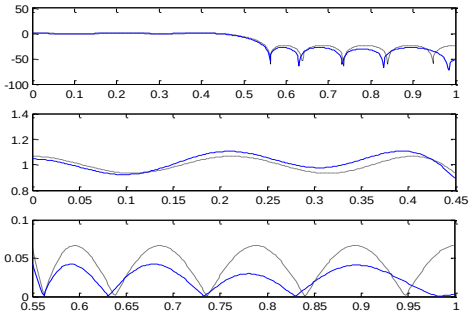


Fig4.7C(c) Normalized frequency response(dB)for PSOP2 and test Function J3
(above) passband
(below) stopband
Dahed-PM
Solid- PSOP2

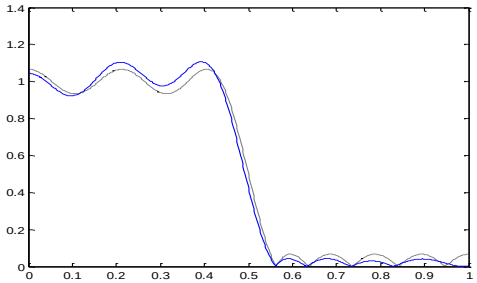


Fig4.7C(d) Magnitude response for PSOP2 and test function J3
Dashed-PM
Solid- PSOP2

Fig.4.7C PSOP2 with test function J3

Table 4.7(c) Filter coefficient for PSOP2 and test function J3

S.NO	h(n) (Filter coefficient)	PSOP 2 (Particle swarm optimization2) with test function J3
1	h(1)	0.0132
2	h(2)	0.0393
3	h(3)	-0.0051
4	h(4)	-0.0370
5	h(5)	0.0075
6	h(6)	0.0541
7	h(7)	-0.0119
8	h(8)	-0.1128
9	h(9)	0.0036
10	h(10)	0.3181
11	h(11)	0.5062

Case-2(D):

TECHNIQUE	TEST FUNCTION
PSOP2 Particle swarm optimization 2	J4 $J_4=\sum abs[abs(\lvert H_d(w)\rvert - 1)-\delta p)]+\sum[abs(\lvert H_d(w)\rvert - \delta s)]$

Fig.4.8A represents the optimized result for test function J4, here we sum absolute value of test function J3 for the optimized solution for a desired response of an ideal filter hi[n].

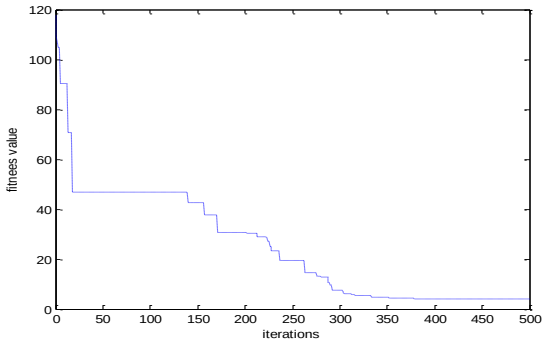


Fig4.8A(a)Converge plot for PSOP2 and test function J4

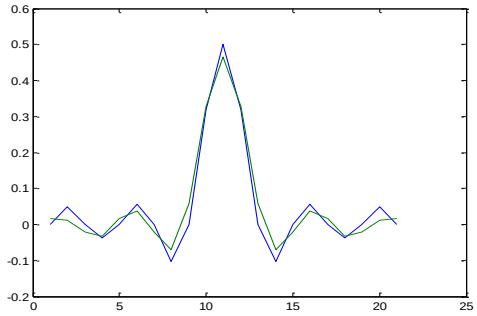


Fig4.8A(b) Impulse response for PSOP2 and test function J4

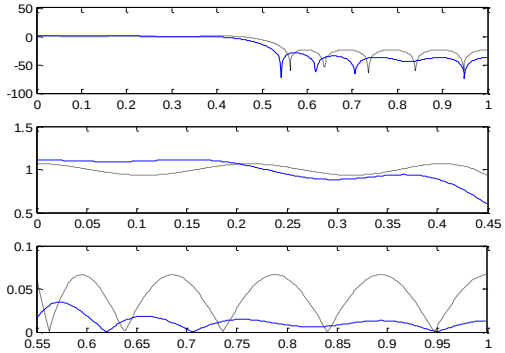


Fig4.8A(c) Normalized frequency response(dB)for PSOP2 and test Function J4
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

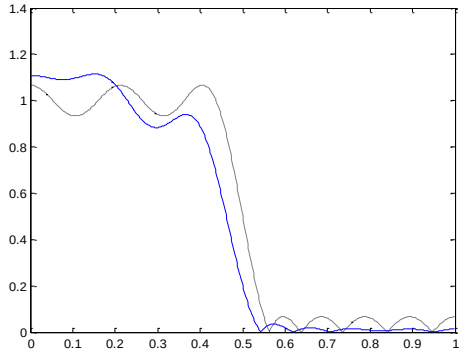


Fig4.8A(d) Magnitude response for PSOP2 and test function J4
Dashed-PM
Solid- PSOP2

Fig.4.8A PSOP2 with test function J4

Table 4.8(a) Filter coefficient for PSOP2and test function J4

S.NO	h(n) (Filter coefficient)	PSOP2(Particle swarm optimization1) with test function J4
1	h(1)	0.0160
2	h(2)	0.0128
3	h(3)	-0.0215
4	h(4)	-0.0333
5	h(5)	0.0154
6	h(6)	0.0379
7	h(7)	-0.0210
8	h(8)	-0.0705
9	h(9)	0.0585
10	h(10)	0.3270
11	h(11)	0.4661

Fig.4.8A (a) represents the convergence plot. Here we can see that cost function is reducing monotonically and does not convergence after greater than500 iteration because it is sum of absolute value. It is slightly higher than 0 and less than 4. The initial cost was 100 it indicates that PSO algorithm is gradually minimizing.

Fig.4.8A (b) shows the impulse response of $h_{PM}[n]$ (Parks McClellan) & h_{psop2}(Particle swarm optimization).Solid line is for PSOP2 and dashed line for PM. Both are showing small difference in filter coefficient values. It means that we are getting response approximately equivalent to PM. For getting the idea of stop band ripples we have shown frequency response in Fig .4.8A(c)(top) which shows the normalized frequency response for frequency less than 0.45 Hz. Performance of both algorithm is almost similar in pass band. In stop band of frequency greater than 0.55 Hz, response due to PSOP2(solid)is lower than the response of PM(dashed).This can be easily seen in Fig.4.8A(c)(middle),for pass band, both response are almost similar. The stop

band response in Fig.4.8A(c)(bottom),the stop band ripples due to PSOP2 is almost half of the stop band ripples(δs) (solid) of PM(dashed).

Similar perception can also be drawn from Fig.4.8A (d) using frequency response. In these figure δs in PSOP2 (solid) is less than δs of PM (dashed) algorithm.

Fig.4.8A(c) and Fig.4.8A (d)are also the best optimized result for case 2(D).For both figures we can see that is obtained less than the ripples as obtained in PM algorithm. In this way, we run our algorithm several times but here we are showing best three results for this case. It indicates that optimization algorithm, always gives different possible optimum solution but all the solutions are better than PM algorithm, for our stop band constraints of LPF filter design

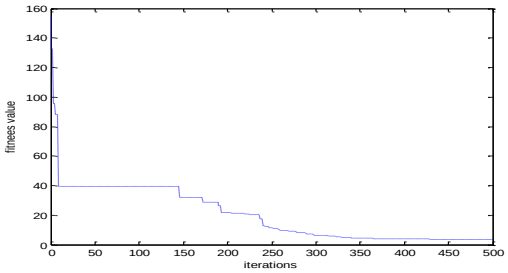


Fig4.8B(a)Converge plot for PSOP2 and test function J4

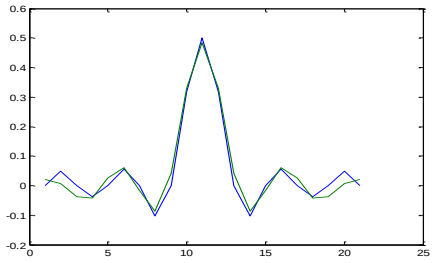


Fig4.8B(b) Impulse response for PSOP2 and test function J4

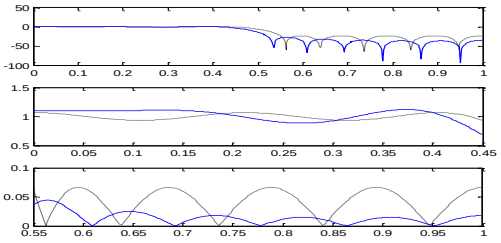


Fig4.8B(c) Normalized frequency response(dB)for PSOP2 and test Function J2
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

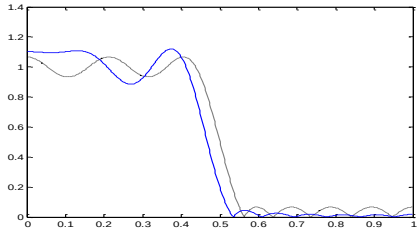


Fig4.8B(d) Magnitude response for PSOP2 and test function J4
Dashed-PM
Solid- PSOP2

Fig.4.8B PSOP2 with test function J4

Table 4.8(b) Filter coefficient for PSOP2 and test function J4

S.NO	h(n) (Filter coefficient)	PSOP2 (Particle swarm optimization1) with test function J4
1	h(1)	0.0222
2	h(2)	0.0081
3	h(3)	-0.0375
4	h(4)	-0.0410
5	h(5)	0.0266
6	h(6)	0.0611
7	h(7)	-0.0164
8	h(8)	-0.0872
9	h(9)	0.0427
10	h(10)	0.3300
11	h(11)	0.4847

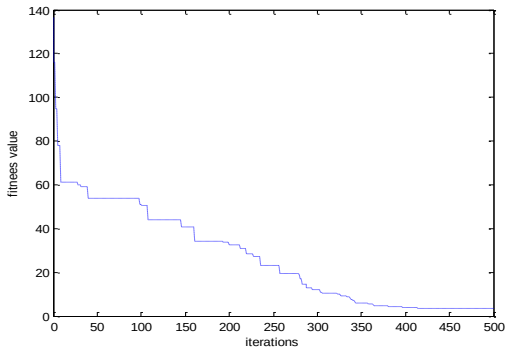


Fig4.8C(a)Converge plot for PSOP2 and test function J4

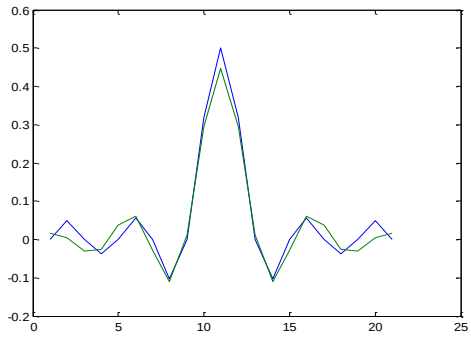


Fig4.8C(b) Impulse response for PSOP2 and test function J4

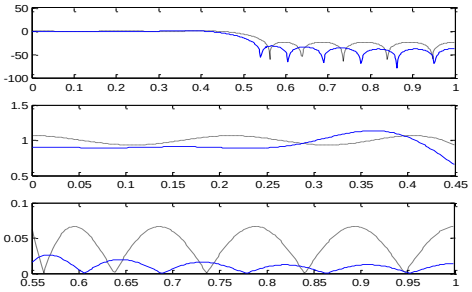


Fig.4.8C(c) Normalized frequency response(dB)for PSOP2 and test Function J4
(above) passband
(below) stopband
Dashed-PM
Solid- PSOP2

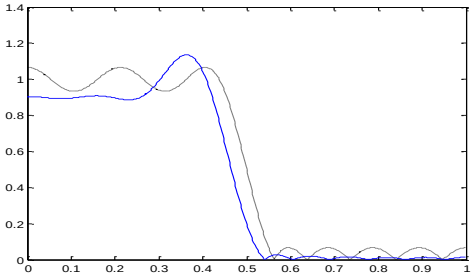


Fig.4.8C(d) Magnitude response for PSOP2 and test function J4
Dashed-PM
Solid- PSOP2

Fig.4.8C PSOP2 with test function J4

Table 4.8(c) Filter coefficient for PSOP2 and test function J4

S.No.	h(n) (Filter coefficient)	PSOP 2 (Particle swarm optimization1) with test function J4
1	h(1)	0.0169
2	h(2)	0.0038
3	h(3)	-0.0314
4	h(4)	-0.0261
5	h(5)	0.0375
6	h(6)	0.0607
7	h(7)	-0.0281
8	h(8)	-0.1094
9	h(9)	0.0114
10	h(10)	0.2937
11	h(11)	140.4467

Table 4.9: Result obtained for PSOP1 and PSOP2 for Various Test Functions

S. No.	**Test Function**	**PSOP 1**			**PSOP 2**		
		δ_P	δ_S	**Error**	δ_P	δ_S	**Error**
1	**Test function (J1)**	0.812	0.0275	0.1036	0.575	0.0075	0.1002
2	**Test function (J2)**	0.603	0.01	0.0697	0.6	0.0075	0.0536
3	**Test function (J3)**	0.8325	0.0125	0.0630	0.81	0.00525	0.0588
4	**Test function (J4)**	0.818	0.01	0.07	0.8325	0.035	0.0817

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION AND FUTURE SCOPE

In this work we have presented an optimal design of linear phase digital low pass finite impulse response (FIR) filter using Dynamic and Adjustable Particle Swarm Optimization (DAPSO). DAPSO 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 distance from each particle to the global best position is calculated in order to adjust the velocity suitably of each particle. The inertia weight has been modified in this PSO to enhance its search capability that leads to a higher probability of obtaining the global optimal solution. The key feature of the modified inertia weight mechanism is to monitor the weights of particles, which linearly decrease in general applications. In the design process, the filter length, pass band and stop band frequencies, feasible pass band and stop band ripple sizes are specified. FIR filter design is a multi-modal optimization problem. Classical particle swarm optimization (PSO), and the Dynamic and Adjustable particle swarm optimization (DAPSO) have been used in this work for the design of linear phase FIR low pass (LP) filter. A comparison of simulation results reveals the optimization efficacy of the algorithm over the prevailing optimization techniques for the solution of the multimodal, non-differentiable, highly non-linear, and constrained FIR filter design problems. Three different kinds of error fitness functions have been named as J1, J2 and J3 for the comparison of DAPSO with Standard PSO. The experimental results prove that the DAPSO can achieve good results by using function J3.

It is revealed that DAPSO has the ability to converge to the best quality near optimal solution and possesses the best convergence characteristics in much less execution times among the algorithms. The simulation results clearly indicate that DAPSO demonstrates the best performance in terms of magnitude response, minimum stop band ripple and maximum stop band attenuation with the narrowest transition width. It can be used as a good optimizer for obtaining the optimal filter coefficients in any practical digital filter design problem of digital signal processing systems.

The quality of the obtained FIR frequency responses is better than the Parks-McClellan method can give far a given number of coefficients (M).

In future we can check the performance of suboptimal results for varying the order of filter. By just changing the modeling, this structure can be used for very different scenarios like IIR filter design, employee scheduling, and best route between two points.

We can also check the effect of quantization of filter coefficient on the results of our optimum value of solutions. Because in the system design problems many times the filter coefficients are obtained for very large decimal values in such case they are stored in a quantized value this may sometimes results a drastic variation in filter performance. So this task is also opens a scenario where we can pick the application where high accuracy are needed in filter design and there after we can find the best quantized optimum value of coefficients that can minimum deviation from desired response.

REFERENCES

[1]. R.V. Rao, V.J. Savsani, and J. Balic, "Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems," Engineering Optimization, vol. 44, no. 12, pp. 1447–1462, Mar. 2019.

[2]. J. Sohl, Efficient Compilation for Application Specific Instruction set DSP Processors with Multi-bank Memories, Linköping Studies in Science and Technology, Diss., No. 1641, Feb. 2015.

[3]. Ababneh J.I., Bataineh M.H., Linear phase FIR filter design using particle swarm optimization and genetic algorithms, Digital Signal Processing, 18, 657–668, 2012.

[4]. F. Shaikh, T.D. Memon and I. H. Kalwar, "Design and Analysis of Linear Phase FIR Filter in FPGA using PSO Algorithm," in 6th Mediterranean Conf. on Embd. Comput., Montenegro, 2017.

[5]. Kaur Pardeep, Simarpreet Kaur, "Optimization of FIR Filters Design using Genetic Algorithm, ISSN 2278- 6856, Volume 1, Issue 3, September – October 2012.

[6]. Wei Zhong, "Linear phase FIR Digital filter design using Differential Evolution Algorithms." M.S. thesis, Dept. of Elect. & Computer. Eng., University of Windsor, Ontario, Canada,2016.

[7]. Bharat Lal Harijan, Farrukh Shaikh, Burhan Aslam Arain, Tayab Din Memon and Imtiaz Hussain Kalwar, "FPGA based Synthesize of PSO Algorithm and its Area-Performance Analysis" International Journal of Advanced Computer Science and Applications (IJACSA), 9(6), 2018. http://dx.doi.org/10.14569/IJACSA.2018.090639.

[8]. A. Praneeth and P. K. Shah, "Design of FIR Filter using Particle Swarm Optimization," Int. Adv. Research. J. in Sci, Eng. and Techno, vol. 3, no. 5, 2016.

[9]. B. A. Mohamed sadek and SAKLY Anis, "FPGA Implementation of Parallel Particle Swarm Optimization Algorithm and Compared with Genetic Algorithm," Int. J. of Adv. Comput. Sci. and Appl., vol. 7, no. 8, 2016.

[10]. Muhammad Imran, Rathiah Hashim and Noor Elaiza Abd Khalid, "An Overview of Particle Swarm Optimization Variants" , Malaysian Technical Universities Conference on Engineering & Technology 2012,MUCET 2012.

[11]. Hu Xiaohul,Shi Yuhui,Eberhart Russ, Recent Advance in Particle swarm optimization,0-7803-8515-2/04©2004 IEEE.

[12]. Shivani Adhan and Puneet Bansal, "Applications and Variants of Particle Swarm Optimization : A review," International Journal of Electronics, Electrical and Computational System IJEECS, Vol. 6, no. 6, June 2017.

[13]. P. M. Palangpour, "Fpga Implementation of PSO Algorithm and Neural Networks," M.S. thesis, Missouri University of Science and Technology, 2010.

[14]. P. Fodisch, A. Bryksa, B. Lange, W. Enghardt and P. Kaever, "Implementing high order FIR filters in FPGAs," 2016: Available arXiv:1610.03360v2.

[15]. L.Tan and J. Jiang, "Finite impulse response filter design," Digital Signal Processing Fundamentals and Applications, 2nd ed. pp 217-290, ELSVIER.

[16]. Angeline P.H., Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In Porto V.W.,Saravanan N., Waagen D., and Eiben A.E., editors, Evolutionary Programming VII, pp. 601–610. Springer, 1998.

[17]. Cheney E.W., Introduction to Approximation Theory. New York: McGraw-Hill, 1966, pp. 72-100.

[18]. Eberhart, R. C. and Shi, Y., A Modified Particle Swarm Optimization, Proceedings of IEEE International Conference on Evolutionary Computation, 1998, pp.69-73.

[19]. Eberhart, R. C. and Shi, Y., Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization, Proceedings of the 2000 Congress on Evolutionary Computation, Vol. 1, 2000, pp.84

[20]. Gold B., and Jordan K.L., "A direct search procedure for designing finite duration impulse response filters," IEEE Trans. Audio Electroacoust, vol. AU-17, pp. 33- 36, Mar. 1969.

[21]. Haupt L. Randy, Haupt Ellen Sue, Published by John wiley and sons Inc. Hoboken, New Jersey©2004.

[22]. Kennedy J. and Eberhart R.C., Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Vol. 4, pp. 1942–1948. IEEE Press, 1995.

[23]. McClellan J.H., “On the design of one-dimensional and two-dimensional FIR digital filters,” Ph.D. dissertation, Rice Univ., Houston, Tex., Apr. 1973.

[24]. McClellan J.H., and Parks T.W., “A unified approach to the design of optimum FIR linear phase digital filters,’ IEEE Trans. Circuit Theory, vol. CT-20, pp. 697-701, Nov. 1973.88.

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

[26]. Poli Riccardo, Kennedy James, Blackwell Tim, Particle swarm optimization An Overview©springer science and Buisiness media,LLC 2007.

[27]. Riget J. and Vesterstrøm. J.S. A diversity-guided particle swarm optimizer – the arPSO. Technical report, EVA Life, Dept. of Computer Science, University of Aarhus, Denmark, 2002.

[28]. Remez E. Ya., “General computational methods of Tchebycheff approximation,” Kiev, 1957 (Atomic Energy Translation 4491, pp. 1-85).

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

[30]. Shi Y. and Eberhart. R.C. A modified particle swarm optimizer. In Proceedings of the IEEE Conference on Evolutionary Computation, pp. 69–73. IEEE Press, 1998

[31]. Vesterstrøm J.S. and Riget J. Particle swarms: Extensions for improved local, multi-modal, and dynamic search in numerical optimization. Master’s thesis, EVA Life, Dept. of Computer Science, University of Aarhus, Denmark, 2002.

[32]. Zhang Liping,Yu Huanjun and Hu shangxu, A new approach to improve particle swarm optimization. Paz et al(Eds.)134-139,2003©springer-verlag Berlin Heidelberg(2003).

ANNEXURE

2021 10th International Conference on System Modeling & Advancement in Research Trends (SMART) | 978-1-6654-3970-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/SMART52563.2021.9676284

Performance Analysis of Low Pass FIR Filter Design using Dynamic and Adjustable Particle Swarm Optimization Techniques

Kaushal Kishor Tripathi[1], Mohd. Suhaib Kidwai[2], Imran Ullah Khan[3*],
[1,2,3]Deptt. of ECE, Integral University, Lucknow, India.
E-mail: [1]erkkt@student.iul.ac.in, [2]mskidwai@iul.ac.in, [3]iukhan@iul.ac.in

***Abstract*—Digital filters have wide extension in control system, sound system, communication, video processing, and clinical/biomedical applications. Exact linear phase and multirate activity can be accomplished with digital executions. In this paper digital filters design problem that involves multiple, often conflicting, design criteria and specifications are discussed. Tracking down an optimum filter configuration is the fundamental goal in this work. If simple/analytic iterative methods is used it will lead to sub-optimal designs. So, an optimization-based method is needed. A low pass finite impulse response (LPFIR) filter optimal design can be achieved by Particle Swarm Optimization and/or dynamic adjustable PSO (DAPSO). In this paper DAPSO algorithms are discussed for different error function. The DAPSO is an improved version of PSO. It proposes swarm updating and velocity vector and hence the solution quality is improved.**

Keywords: FIR Filter, PSO, DAPSO, Low Pass Filter and Filter Coefficient.

I. Introduction

In FIR filters output only depends on the present inputs. It can be represented as:

$$y_n = b_0x_n+b_1x_{n-1}+b_2x_{n-2+....+}b_Nx_{n-M} \quad (1)$$

The transfer function of FIR filter for a unit delay element z^{-1}, for that $z^{-1}x_n=x_{(n-1)}$, can be represented as:

$$\frac{Y(Z)}{X(Z)} = b_0+ b_1Z^{-1}+b_2Z^{-2}{}_{+....+}b_NZ^{-M} \quad (2)$$

Equation 1 and 2 represents time and Z domain difference equation and figure 1 below represents basic parameters of a low pass filter [1-3].

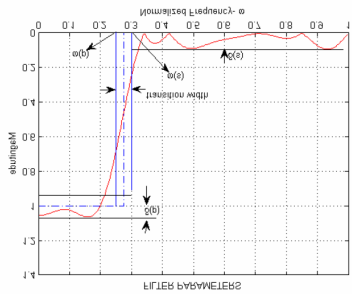


Fig.1: Representation of basic Filter Parameters [2]

II. Dynamic and Adjustable PSO

Two improved algorithms called $DAPSO_1$ and $DAPSO_2$ are elaborated in details. In DAPSOs, in order to adjust the velocity of each particle, all particles are calculated the distance from itself to the global best position by the following function [4-6]

The general flow of DAPSOs and the flowchart of DAPSO are shown below:

Step 1. Initialization

Step 2. Evaluation

Step 3. Calculate the distance from each particle to the global best position and save the farthest distance in the memory.

Step 4. Particle's velocity is adjusted according to distance from itself to the global best position.

Step 5. Update position by the adjusted velocity.

Step 6. Repeat Step-2 to 5 till termination criteria meet.

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

$$H(e^{jwk}) = \sum_{n=0}^{n} h(n)e^{-j\omega kn} \quad (3)$$

Where $\omega_k = \frac{2\pi k}{N}$; $H(e^{jwk})$ is the Fourier transform complex vector. This is the FIR filter's frequency response. The frequency is sampled in [0, π] with N points. Different kinds of error fitness functions have been used in different literatures.

$$E(\omega)=G(\omega)[H_d(e^{j\omega})- H_i(e^{j\omega})] \quad (4)$$

Where

- $H_de^{(j\omega)}$ - Frequency response of the designed approximate filter;
- $H_ie^{(j\omega)}$ - Frequency response of the ideal filter;
- $G(\omega)$ - Weighting function to provide different weights for the approximate errors in different frequency bands.

For ideal LP filter, $H_ie^{(j\omega)}$ is given as,

$$H_i(e^{j\omega}) = \begin{cases} 1 \text{ for } 0 \le \omega \le \omega_c ; \\ 0 \text{ otherwise} \end{cases} \quad (5)$$

where ω*c* is the cut-off frequency. The test function J_1 is defined as

J_1=max(abs(|E(ω)|)

For test function J_1 the error function is

$$E(\omega)=G(w)[H_d(e^{j\omega})-H_{pm}(e^{j\omega})] \quad (6)$$

For test function J_2 equation is used as error function

$$J_2=\max(\text{abs}(|E(\omega)|) \quad (7)$$

By using algo the error fitness can be minimized, and is defined as

$$J_3=\max(|E(\omega)|-\delta_p)+\max(|E(\omega)|-\delta_S) \quad (8)$$

$\omega \le \omega_P \qquad \omega \ge \omega_S$

where δ_p and δ_S are the ripples in the pass band and stop band, respectively, and ω_p and ω_s are pass band and stop band normalized cut-off frequencies, respectively. Figure 2 and 3 shows flow chart of DAPSO1 and DAPSO2 [7, 8].

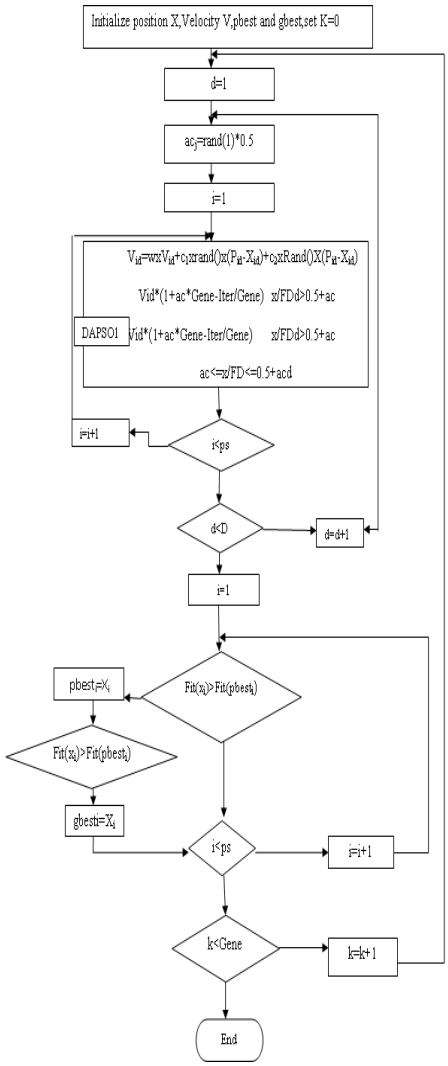


Fig. 2: Flowchart of $DAPSO_1$

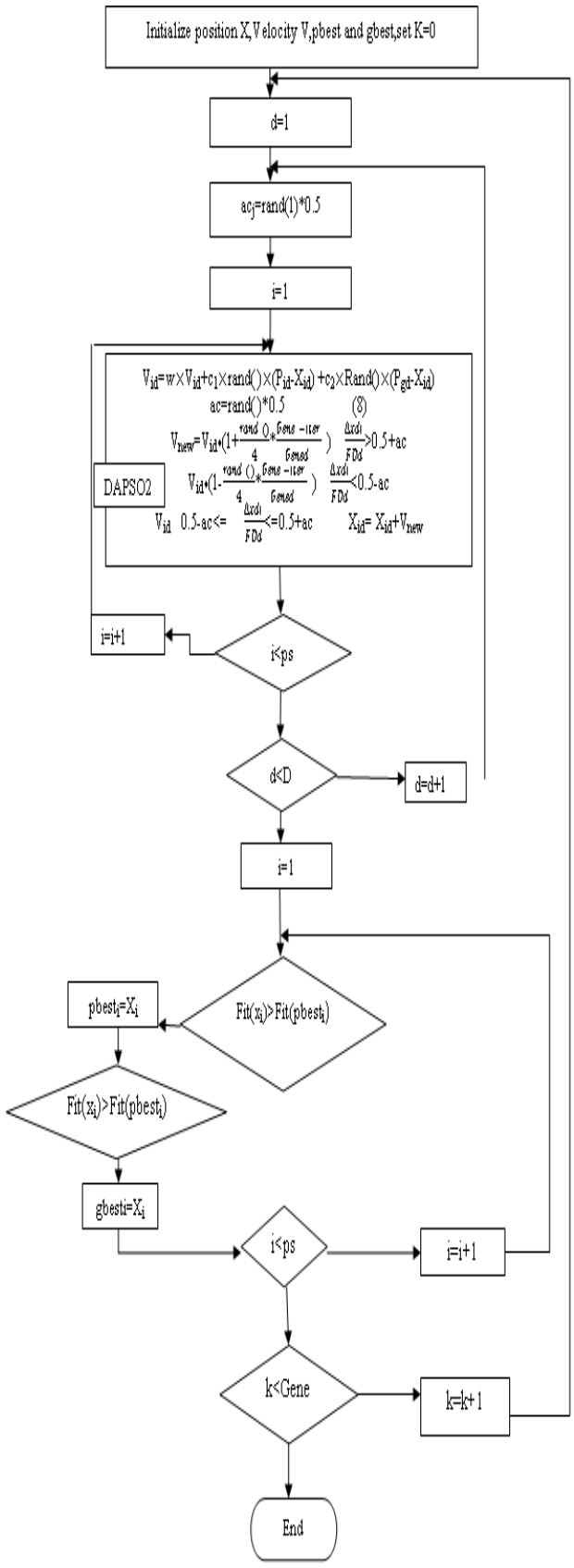


Fig. 3: Flowchart of $DAPSO_2$

III. Results

The sampling frequency is taken to be f_s = 1Hz. The number of frequency samples is 128.

TABLE 1: Filter Coefficient for DAPSO1 with test Function J_1

S.No.	h(n)	$DAPSOP_1$ with test function J_1
1	h(1)	0.0000
2	h(2)	0.0481
3	h(3)	-0.0000
4	h(4)	-0.0369
5	h(5)	-0.0000
6	h(6)	0.0573
7	h(7)	-0.0000
8	h(8)	-0.1022
9	h(9)	-0.0000
10	h(10)	0.3170
11	h(11)	0.5001

TABLE. 2: Filter Coefficient for DAPSO1 with Test Function J_2

S.No.	h(n)	$DAPSOP_1$ with test Function J_2
1	h(1)	0.0136
2	h(2)	0.0348
3	h(3)	-0.0052
4	h(4)	-0.0413
5	h(5)	0.0112
6	h(6)	0.0626
7	h(7)	0.0065
8	h(8)	-0.1026
9	h(9)	0.0062
10	h(10)	0.3111
11	h(11)	0.4931

Table 1 to 3 represents Filter coefficient for $DAPSO_1$ with test function J_1 to J_3 and table 4 shows values for $DAPSO_2$ with test function J_1.Figure 4-6 shows different characteristics implemented on MatLab for $DAPSO_1$ with test function J_1 to J_3 and figure 7 for $DAPSO_2$ with test function J_1.

TABLE. 3: Filter Coefficient for $DAPSO_1$ with Test Function J_3

S.No.	h(n)	$DAPSO_1$ with test function J_3
1	h(1)	0.0223
2	h(2)	0.0250
3	h(3)	-0.0217
4	h(4)	-0.0503
5	h(5)	0.0068
6	h(6)	0.0657
7	h(7)	-0.0012
8	h(8)	-0.0880
9	h(9)	0.0197
10	h(10)	0.3303
11	h(11)	0.5035

TABLE. 4: Filter Coefficient for $DAPSO_2$ with Test Function J_1

S.NO	h(n)	$DAPSO_2$ with test Function J_1
1	h(1)	-0.0000
2	h(2)	0.0480
3	h(3)	-0.0000
4	h(4)	-0.0369
5	h(5)	-0.0000
6	h(6)	0.0572
7	h(7)	-0.0000
8	h(8)	-0.1022
9	h(9)	0.0000
10	h(10)	0.3170
11	h(11)	0 .5000

TABLE. 5: Result Obtained for $DAPSOP_1$ and $DAPSOP_2$ for Various Test Functions

S. No.	Test Function	$DAPSOP_1$			$DAPSOP_2$		
		δ_P	δ_S	Error	δ_P	δ_S	Error
1	Test function(J_1)	0.52	0.0125	0.1025	0.827	0.015	0.0942
2	Test function(J_1)	0.812	0.0275	0.1036	0.575	0.0075	0.1002
3	Test function(J_2)	1.001	0.0139	0.0949	0.6	0.02	0.0905
4	Test function(J_2)	0.603	0.01	0.0697	0.6	0.0075	0.0536
5	Test function(J_3)	0.81	0.00525	0.0588	0.8325	0.0125	0.0630
6	Test unction(J_3)	0.818	0.01	0.07	0.8325	0.035	0.0817

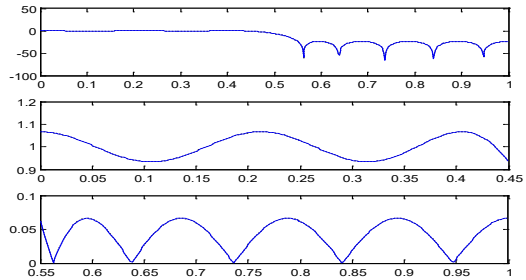


Fig.4(a). Converge Plot for $DAPSO_1$ and test Function J_1

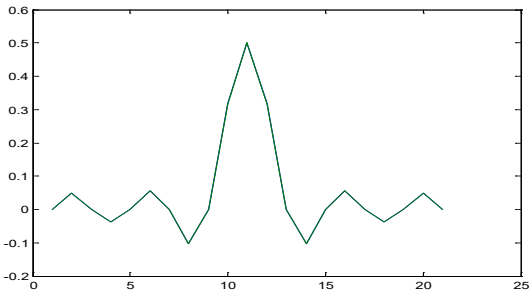


Fig. 4 (b). Impulse Response for DAPSO1 and Test Function J_1

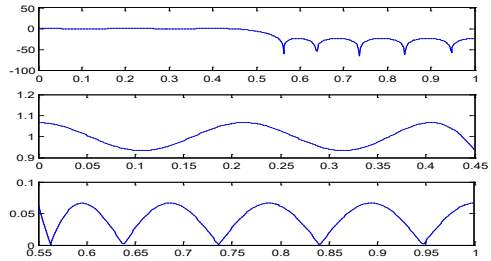


Fig.4(c). Normalized Frequency Response(dB)for DAPSO1 and test Function J1 (above) Passband (below) Stopband Dashed-PM Solid- $PSOP_1$

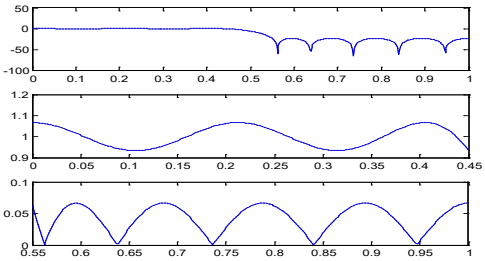


Fig. 5(c). Normalized Frequency Response(dB)for $DAPSOP_1$ and test Function J_2 (Above) Passband (Below) Stopband Dashed-PM Solid- $PSOP_1$

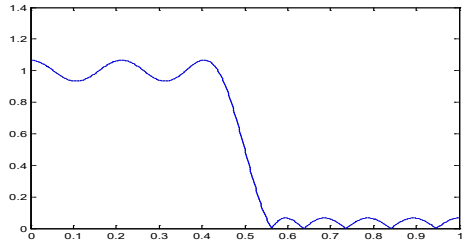


Fig.4(d). Magnitude Response for DAPSO1 and test Function J_1 Dashed-PM Solid- $PSOP_1$

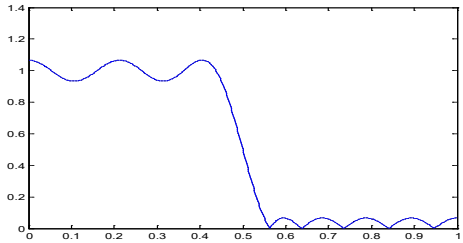


Fig.5(d). Magnitude Response for $DAPSO_1$ and test Function J_2 Dashed-PM Solid- $PSOP_1$

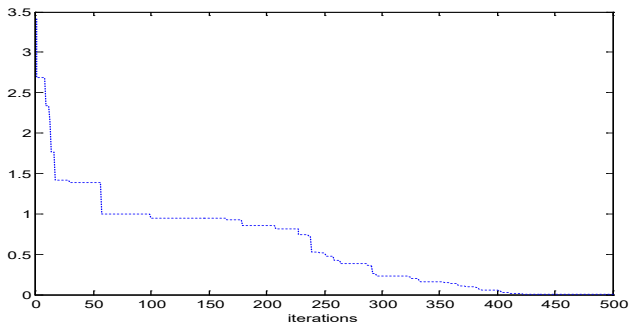


Fig. 5(a). Converge Plot for $DAPSO_1$ and test Function J_2

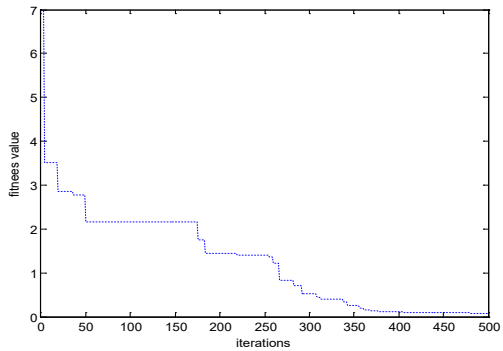


Fig.6(a). Converge Plot for $PSOP_1$ and test Function J_3

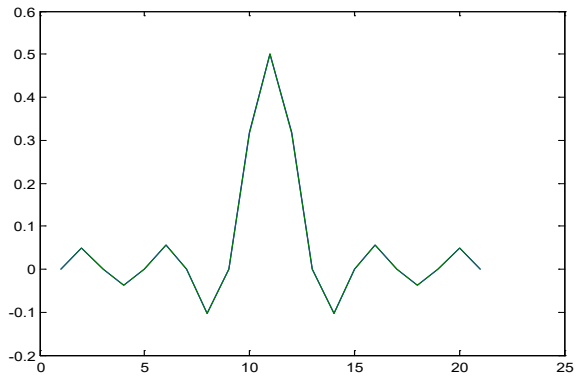


Fig. 5 (b). Impulse Response for DAPSO1 and test Function J_2

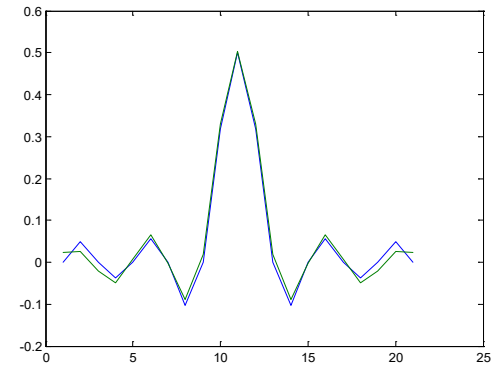


Fig.6(b). Impulse Response for $PSOP_1$ and test Function J_3

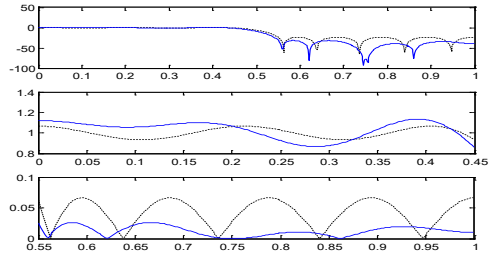


Fig.6(c). Normalized Frequency Response(dB)for $DAPSO_1$ and test Function J_3 (Above) Passband (Below) Stopband Dashed-PM Solid- $DAPSO_1$

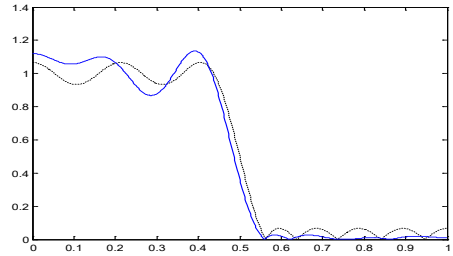


Fig.6(d). Magnitude Response for $DAPSO_1$ and test Function J_3 Dashed-PM Solid- $PSOP_1$

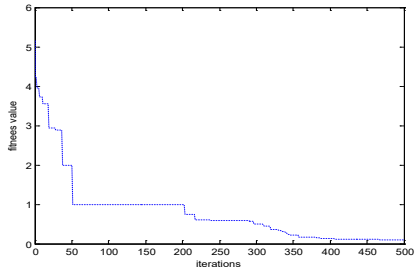


Fig.7(a). Converge Plot for $DAPSO_2$ and Test Function J_1

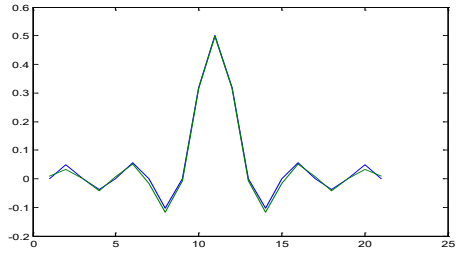


Fig.7(b). Impulse Response for $DAPSO_2$ and test Function J_1

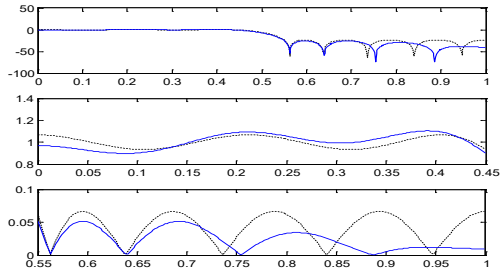


Fig.7(c). Normalized Frequency Response(dB)for $DAPSOP_2$ and test Function J_1 (Above) Passband, (Below) Stopband, Dashed-PM Solid $DAPSOP_1$

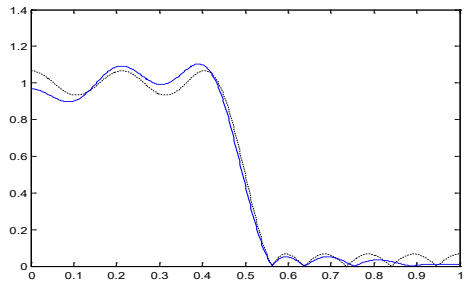


Fig.7(d). Magnitude Response for $DAPSOP_2$ and test Function J_1 Dashed-PM, Solid-$DAPSOP_1$

IV. Conclusion

LPFIR filter is implemented using $DAPSO_1$ and 2 techniques for J_1 to J_3 test function. MatLab implementation shows normalized frequency response, converge plot, magnitude response and impulse response. On closely observing it is find that test function J_3 give best performance in terms of ripples, pass band, stop band frequencies and error. Values for pass band, stop band ripples and error function are .5-.8, 0.005-0.01 and 0.05-0.1 respectively.

References

[1] C. Zhang, Z. Yue, F. W. Chengdu, and D. Yan "A Novel Framework for FIR Digital Filter Design Based on P system with PSO" International Conference on Information Systems and Computer Aided Education (ICISCAE), 2018 : 297-307.

[2] Xi Yuanhai, "An Improved Particle Swarm Optimization Algorithm for FIR filter design" in 20th international conference on Electronics, circuits and system volume(ICECS), Abu Dhabi, 2013, pp.261-264.

[3] G. Lui, Y.X. Li and G. He, "Design of digital FIR filters Using Differential Evolution Algorithm Based on reserved on evolutionary computation", in IEEE international conference of information science and management Engineering, July 2010, pp.177-180.

[4] Chen S, "model identification using batch-recursive adaptive simulated annealing algorithm" at processing of 6th annual Chinese Automation and Computer science Conference in UK, 2000, pp.151- 155.

[5] Karaboga, D.H. Horrocks, N. Karaboga and A. Kalinli, "Designing digital FIR filters using Tabu search algorithm", IEEE International Symposium on Circuits and Systems, vol.4, pp.2236-2239, 1997.

[6] H.C. Lu and S.T. Tzeng, "Design of arbitrary FIR log filters by genetic algorithm approach", in IEEE international symposium on circuit and system emerging technologies for the 21st century. Proceedings (IEEE Cat No.00CH36353), Geneva, 2000, pp. 333-336 vol.2.

[7] M. Fikret Ercan and Xiang Li, "particle swarm optimization and its hybrids", in international journal of computer and communication engineering, vol.2, n.1 , January 2013.

[8] Jehad I. Ababneh , H. Bataineh, "Linear Phase FIR filters design using particle swarm optimization and Genetic Algorithms", vol. 18.