ISSN: 2229-6948(ONLINE)
DOI: 10.21917/ijct.2020.0316

DESIGN OF FINITE IMPULSE RESPONSE FILTERS USING EVOLUTIONARY TECHNIQUES - AN EFFICIENT COMPUTATION

Manoranjan Das
Department of Electronics and Communication Engineering, ICFAI University, Raipur, India

Abstract

In the recent times, Finite Impulse Response (FIR) filters are deemed to be the most suitable model for almost all the adaptive system applications because of its simplicity and assured stability. But, designing linear-phase Finite Impulse Response filters with fewer hardware resources is a challenging task. To address this issue, this paper presents an efficient and novel technique to design one-dimensional and two-dimensional linear-phase Finite Impulse Response filters. The primary focus of this paper is the implementation of crossover bacterial foraging and Cuckoo Search techniques for effectively designing one-dimensional and two-dimensional linear-phase Finite Impulse Response filters. The superiority of the design is affirmed by its ingenuity to obtain the solution through the convergence of a biased random search using crossover bacterial foraging optimization. Also, the solution is obtained in a quicker fashion through the convergence of a metaheuristic search technique called the Cuckoo Search technique. For better benchmarking and evaluation of the results, the outcomes of the proposed techniques are compared with the well-known genetic algorithm and bacterial foraging optimization techniques. The experimental analysis is carried out using three one-dimensional and two two-dimensional Finite Impulse Response filters.

Keywords:

1-D, 2-D FIR Filters, Genetic Algorithm, Cuckoo Search Algorithm, Crossover Bacterial Foraging Optimization Technique

1. INTRODUCTION

Recursive and non-recursive digital filters are widely used for image filtering [1] and in most of the adaptive systems such as system identification, equalization, pattern recognition. In the literature, various authors have focused on the design of both recursive and non-recursive filters. Two-dimensional (2D) filters have been widely used in most of the image processing [1], medical imaging [2], face recognition [3] tasks. Infinite impulse response (IIR) filters are recursive, whereas finite impulse response (FIR) filters are non-recursive. IIR filters exhibit maximally flat frequency response but have instability and non-linear phase characteristics. However, FIR filters ensure stability [4] and linear phase characteristics over a wide frequency range.

The stability of a digital filter is an important requirement for practical enactment. Most of the existing design techniques of digital filters were based on trial and error basis [5], hence resulting in instability conditions. 1-D recursive filters have been designed using least square error analysis by implementing a frequency sampling technique. However, this method is suitable at a lower frequency but exhibits a frequency warping problem at high frequencies. Some of the authors have focused on the design of 2-D recursive filters [6]-[11]. The existing design techniques of 2-D filters include the transformation of one-dimensional (1-D) filters [7]-[10] and various transformation techniques [11].

But, there is a requirement for a novel automatic design technique that will produce the optimum results with minimum computational effort. Since FIR filters have been implemented for most of the adaptive system applications, the value of filter coefficients must be precise to predict the optimum output.

In this work, the objective function is framed using filter coefficients, and is minimized through four different evolutionary algorithms. The main aim of using evolutionary optimization algorithms is to obtain the global optimum value of filter coefficients. The global optimum value of filter coefficients is obtained by using the global search technique of four different evolutionary techniques.

The aforementioned issue can be resolved by using soft computing techniques. Each of the soft computing techniques has certain strengths and weaknesses. But, to obtain an optimal solution, evolutionary algorithms are more suitable for soft computing techniques [7]. Some of the well-known evolutionary algorithms are Genetic Algorithm (GA), Bacteria Foraging Optimization (BFO), Cuckoo-Search (CS) and Crossover Bacteria Foraging Optimization (COBFO).

GA is executed using a purely random search technique, but the offspring never ends with a preferred location [12]. Similarly, BFO provides a random bias walk by consuming more time to optimize the parameters [13]. So, to have a proper tradeoff between the optimum results and the computational effort, both CS and COBFO have been implemented.

This paper presents a novel technique for the design of 1-D and 2-D linear phase FIR filters with low computational complexity and hardware cost. Three 1-D and two 2-D FIR filters with unknown coefficients have been designed using CS and COBFO algorithms. The results are compared with the well-known GA and BFO evolutionary algorithms.

The ensuing structure of this manuscript is as follows: section 2 explains the methodology of the proposed scheme, sections 3 and 4 disclose the results and discussions respectively, and section 5 concludes the paper.

2. METHODOLOGY

This work has been designed to perform the optimization of coefficients of five different FIR filters to get an optimum amplitude response. The cost function which is needed to be minimized has been designed by the difference of desired amplitude response and actual amplitude response. Four different evolutionary techniques namely GA, BFO, CS and CPBFO are implemented to optimize the filter coefficients. The initial population is considered as random, while different search paths are used to find the optimum solution by some optimization parameters.

2.1 PROBLEM FORMULATION

The classification of digital filters is based on the value of impulse response of the linear time-invariant system. In the case of the FIR filter, the impulse response is defined for a finite range [4] and can be given by using Eq.(1).

$$y(n) = \sum_{k=0}^{N-1} h(k)\, x(n-k) \tag{1}$$

where N is the number of finite points

The transfer function of Eq.(1) can be obtained by using Z-transform and is given by Eq.(2).

$$H(z) = \sum_{k=0}^{N-1} h(k) z^{-k} \tag{2}$$

Let us consider, M_d as a desirable amplitude response of the 2-D filter as a function of the frequencies ω_1 and ω_2, (ω_1,ω_2[-π, π]).

$$M_d(\omega) = \begin{cases} 0 & for -\pi \le \sqrt{(\omega_1^2 + \omega_2^2)} \le -0.5\pi \\ 1 & for -0.5\pi \le \sqrt{(\omega_1^2 + \omega_2^2)} \le 0.5\pi \\ 0 & for\ 0.5\pi \le \sqrt{(\omega_1^2 + \omega_2^2)} \le \pi \end{cases} \tag{3}$$

To minimize the least square error between the desirable and actual amplitude responses, I have framed an objective function as:

$$J = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \left[\left| M(\omega_1, \omega_2) \right| - M_d(\omega_1, \omega_2) \right]^p \tag{4}$$

where, $M(\omega_1, \omega_2)$ is the Fourier transform of $H(z_1,z_2)$

(5)

$\omega_1 = \left(\pi / N_1 \right) n_1$, $\omega_2 = \left(\pi / N_2 \right) n_2$ and p is an even positive integer (usually p = 2 or 4).

The main objective of the proposed technique is to minimize the difference between the actual and desired amplitude response (see Eq.(6)) of the filter at (N_1, N_2) points.

$$J = \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \left[\begin{array}{l} \left| M\left(\left(\frac{\pi}{N_1} \right) n_1, \left(\frac{\pi}{N_2} \right) n_2 \right) \right| \\ -M_d\left(\left(\frac{\pi}{N_1} \right) n_1, \left(\frac{\pi}{N_2} \right) n_2 \right) \end{array} \right]^p \tag{6}$$

To implement the proposed technique, two 2-D FIR filters have been formulated (Problem-1 and Problem-2)

2.1.1 Problem-1:

$$H(z_1, z_2) = \sum_{i=0}^{N} \sum_{j=0}^{N} a_{ij} z_1^i z_2^j, a_{00} = 1 \tag{7}$$

For N=2

$$\begin{aligned} H(z_1, z_2) = 1 &+ a_{01} z_2 + a_{02} z_2^2 + a_{10} z_1 + a_{11} z_1 z_2 \\ &+ a_{12} z_1 z_2^2 + a_{20} z_1^2 + a_{21} z_1^2 z_2 + a_{22} z_1^2 z_2^2 \end{aligned} \tag{8}$$

In the present problem under consideration, the aim is to minimize the vector (X).

where,

$$X = \left[a_{01}, a_{02}, a_{10}, a_{11}, a_{12}, a_{20}, a_{21}, a_{22} \right] \tag{9}$$

For the successful implementation, the study needs to calculate

$$M(\omega_1, \omega_2) = A_R - jA_I \tag{10}$$

where,

$$\begin{aligned} A_R = a_{00} &+ a_{01}c_{01} + a_{02}c_{02} + a_{10}c_{10} + a_{20}c_{20} \\ &+ a_{11}c_{11} + a_{12}c_{12} + a_{21}c_{21} + a_{22}c_{22} \end{aligned} \tag{11}$$

$$c_{ij} = c_{ij}(\omega_1, \omega_2) = \cos(i\omega_1 + j\omega_2) \text{ for } i, j = 0,1,2,... \tag{12}$$

$$A_I = \left[a_{01}s_{01}, a_{02}s_{02}, a_{10}a_{10}, a_{11}s_{11}, a_{12}s_{12}, a_{21}s_{21}, a_{22}s_{22} \right] \tag{13}$$

$$s_{ij} = s_{ij}(\omega_1, \omega_2) = \sin(i\omega_1 + j\omega_2) \text{ for } i, j = 0,1,2. \tag{14}$$

Therefore, the study obtains an equation for

$$\left| M(\omega_1, \omega_2) \right| = \sqrt{A_R^2 + A_I^2} \tag{15}$$

2.1.2 Problem-2

$$H(z_1, z_2) = \prod_{k=0}^{N} \left(1 + b_k z_1 + c_k z_2 + d_k z_1 z_2 \right) \tag{16}$$

For N=2

$$\begin{aligned} H(z_1, z_2) = &(1 + b_0 + c_0 + d_0) \\ &(1 + b_1 z_1 + c_1 z_2 + d_1 z_1 z_2) \\ &(1 + b_2 z_1^2 + c_2 z_2^2 + d_2 z_1 z_2) \end{aligned} \tag{17}$$

In the aforementioned problem under consideration, the aim is to minimize the vector (X).

where,

$$\left| M(\omega_1, \omega_2) \right| = \sqrt{\left(A_{1R}^2 + A_{1I}^2 \right) \left(A_{2R}^2 + A_{2I}^2 \right)} \tag{18}$$

$$X = \left[b_0, c_0, d_0, b_1, c_1, d_1, b_2, c_2, d_2 \right] \tag{19}$$

Similarly, to generalize the proposed technique, the proposed technique has been implemented with three different 1-D FIR filters. Suppose, M_d is the desirable amplitude response of the 1-D filter as a function of the frequencies ω(ω[-π, π]).

$$M_d(\omega) = \begin{cases} 0 & \text{for} -\pi \le \omega \le -0.5\pi \\ 1 & \text{for} -0.5\pi \le \omega \le 0.5\pi \\ 0 & \text{for}\ 0.5\pi \le \omega \le \pi \end{cases} \tag{20}$$

The design task at hand amounts to finding a transfer function $H(Z)$, which approximates the desired amplitude response. This approximation can be achieved by minimizing

$$J = \sum_{n=0}^{N} \left[\left| M(\omega) \right| - M_d(\omega) \right]^p \tag{21}$$

where, $M(\omega)$ is the Fourier transform of $H(z)$

$$M(\omega) = H(z) \Big|_{z = e^{-j\omega}} \tag{22}$$

$\omega = \left(\pi / N \right) n$, and p is an even positive integer (usually p=2 or 4).

Hence, the aim is to minimize the difference between the actual and desired amplitude response of the filter at (N) points. The three different 1-D FIR filter transfer functions considered for design purpose are given in section 2.1.3, 2.1.4, and 2.1.5.

2.1.3 Problem-3:

$$H(z)=\sum_{i=0}^{N} a_i z^i, a_0=1 \tag{23}$$

For N=8

$$H(z)=1+a_1z+a_2z^2+a_3z^3+a_4z^4 + a_5z^5+a_6z^6+a_7z^7+a_8z^8 \tag{24}$$

In the present problem under consideration, the aim is to minimize the vector (X).

where,

$$X=\left[a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8\right] \tag{25}$$

For the successful implementation, the study needs to calculate

$$M(\omega)=A_R-jA_I$$

where

$$A_R=a_0+a_1c_1+a_2c_2+a_3c_3+a_4c_4 + a_5c_5+a_6c_6+a_7c_7+a_8c_8 \tag{26}$$

$$c_i=c_i(\omega)=\cos(i\omega) \text{ for } i=0,1,2,3,...,8.$$

$$A_I=a_1s_1+a_2s_2+a_3s_3+a_4s_4 + a_5s_5+a_6s_6+a_7s_7+a_8s_8$$

$$s_i=s_i(\omega)=\sin(i\omega) \text{ for } i=0,1,2,...,8.$$

Therefore, the study obtains an equation for $\left|M(\omega)\right|$

$$\left|M(\omega)\right|=\sqrt{A_R^2+A_I^2} \tag{27}$$

2.1.4 Problem-4:

$$H(z)=\prod_{i=1}^{N}\left(1+a_iz^{-i}\right) \tag{28}$$

For N=8

$$H(z)=(1+a_1z^{-1})(1+a_2z^{-2})(1+a_3z^{-3})(1+a_4z^{-4})(1+a_5z^{-5}) (1+a_6z^{-6})(1+a_7z^{-7})(1+a_8z^{-8})$$

In the problem-4 under consideration, the aim is to minimize the vector (X).

where,

$$X=\left[a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8\right] \tag{29}$$

2.1.5 Problem-5:

$$H(z)=\left[\sum_{i=1}^{N}(1+a_iz^{-i})\right]\left[\sum_{j=1}^{M}(1+b_jz^{-j})\right] \tag{30}$$

For N=M=4

$$H(z)=\left(1+a_1z^{-1}+a_2z^{-2}+a_3z^{-3}+a_4z^{-4}\right) \left(1+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}+b_4z^{-4}\right) \tag{31}$$

In the aforementioned problem under consideration, the aim is to minimize the vector (X).

where,

$$X=\left[a_1,a_2,a_3,a_4,b_1,b_2,b_3,b_4\right] \tag{32}$$

2.2 EVOLUTIONARY ALGORITHMS

This section describes the four well-known evolutionary-based soft computing techniques.

2.2.1 Genetic Algorithm (GA):

GA uses some genetics inspired operations namely crossover, mutation, and inversion for the natural selection of chromosomes from one population to another [6]. GA provides an optimal combination of things by navigating large search spaces [14]. The problem of GA is that the offspring never ends with a preferred location. The pseudo-code for GA can be given as:

Step 1: Produce a random population of N chromosomes.

Step 2: Calculate the fitness $f(x)$ of each chromosome x in the population.

Step 3: Create a new population until the new population is complete.

Step 4: Use the newly generated population for a further run of the algorithm.

Step 5: If the end condition is satisfied, stop and return the best solution in the current population.

2.2.2 Bacterial Foraging Optimization (BFO) Algorithm:

BFO is a foraging behavior of E Coli bacteria and nature-inspired optimization algorithm [15]. It has been widely adopted because of its simplicity and ease of implementation [16]. For complex optimization problems, BFO possesses a poor convergence behavior as compared to other nature-inspired optimization techniques [17]. Another problem with BFO is that it consumes more time to optimize the parameters

2.2.3 Crossover Bacterial Foraging Optimization (COBFO) Algorithm:

The disadvantages of both GA and BFO can be tackled by the use of COBFO. The COBFO works on the principle of crossover mechanism, which is used to search the nearby locations by deploying 50 percent of bacteria randomly at different locations [15]. This process shall enable us to get some more missing nutrients as compared to the BFO algorithm. Based on the problem formulation and the area of optimization in this context, some characteristics of chemotaxis and swarming behavior of bacteria have been ignored to make the simulation process simple.

2.2.4 Cuckoo-Search (CS) Algorithm:

Although COBFO provides optimum results as compared to GA and BFO, it consumes more time leading to computational burden. This can be overcome by implementing the CS algorithm. CS is based on the breeding behavior of animals which can effectively optimize the problems with minimal computational effort [18]-[20]. Another advantage of the CS algorithm is that it

has lesser number of optimization parameters as compared to GA, BFO and COBFO.

The entire process of methodology can be summarized as follows:

Step 1: 1-D and 2-D FIR filters are considered as input.

Step 2: The initial random population is generated according to filter coefficients.

Step 3: Objective function value is evaluated and minimized by using Evolutionary techniques.

Step 4: Different performance metrics are computed to analyze the best performance.

Step 5: Convergence of each evolutionary technique is tested.

Step 6: The best optimum output values are noted.

3. RESULTS

3.1 COMPUTATIONAL REQUIREMENTS

The maximum dimension of search space for optimization in this research work is 15. The two cost functions (Eq.(6) and Eq.(21)) have been optimized using four evolutionary algorithms specifically GA, BFO, CS, and COBFO. The entire experiment is done using a system with Windows 8 operating system, i3 core processor, and a 2GB RAM.

The algorithms are written using MATLAB release in 2018. To optimize the cost functions, different optimization parameters are chosen empirically by hit and trial basis. The optimization parameters for the four different algorithms are tabulated in Table.1.

Table.1. Optimization Parameters setting for GA, BFO, CS, and COBFO

Parameters setting
GA
Number of chromosomes:25 Number of iterations:25 Crossover Probability:0.75 Mutation probability:0.25
BFO
Number of bacteria:100 Number of chemotactic steps:10 Crossover Probability:0.75 Mutation probability:0.25 The length of a swim:5 Probability of elimination-dispersal:0.05
CS
Number of nests:25 Mutation Probability:0.25 Number of iterations:25 Scale factor:1.25
COBFO
Number of bacteria:100 Number of chemotactic steps:10 Crossover Probability:0.75 Mutation probability:0.25 The length of a swim:5 Probability of elimination-dispersal:0.05

3.2 EXPERIMENTAL RESULTS

The experimental analysis has been conducted to evaluate the best fitness function value of the cost functions by obtaining the optimum coefficient values using GA, BFO, CS, and COBFO algorithm for the five different problems (as mentioned in the problem formulation section of methodology). The corresponding amplitude response of each problem are disclosed by the ensuing sections 3.2.1, 3.2.2, 3.2.3, 3.2.4, and 3.2.5.

3.2.1 Results of Problem-1:

The filter coefficients for problem-1 have been optimized using four optimization algorithms and the corresponding predictable values are presented in Table.2. The estimated amplitude responses for problem-1 using CS and COBFO algorithms are shown in Fig.1 and Fig.2 respectively. From the table, it can be observed that the COBFO algorithm has the best results with minimum error and more closure values.

Table.2. Estimated Parameters for Problem-1

Optimization algorithm	**COBFO**		**CS**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
a_{01}	**0.2596**	**0.0503**	0.1577	-0.9593	0.1482	-0.3351	0.1591	0.4818
a_{02}	**-0.2105**	**-0.4986**	-0.187	-0.3794	-0.2351	-0.7059	-0.2389	0.6804
a_{10}	**0.1325**	**-0.3901**	0.2111	0.6052	0.1482	-0.0455	0.1283	0.6236
a_{11}	**0.7893**	**0.6754**	0.7274	0.6269	0.814	-0.1917	0.8007	-0.5087
a_{12}	**0.1071**	**0.6427**	0.0321	-0.0416	0.0615	-0.1168	0.089	0.7539
a_{20}	**-0.1971**	**-0.4978**	-0.1551	0.2988	-0.2351	0.1207	-0.24	-0.47
a_{21}	**0.0479**	**0.6201**	0.072	0.5394	0.0615	0.4046	0.0486	0.333
a_{22}	**0.2223**	**-0.6392**	0.2287	-0.0122	0.2696	-0.211	0.2478	-0.4111

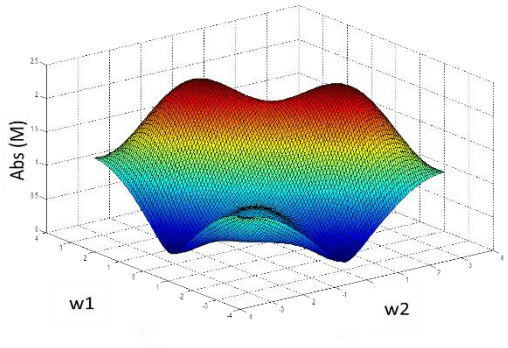

Fig.1. Amplitude Response of problem-1 using CS

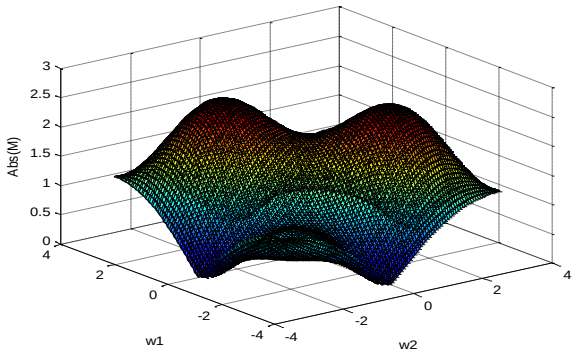

Fig.2. Amplitude Response of problem-1 using COBFO

3.2.2 Results of Problem-2:

The Table.3 shows the optimized coefficient values for problem-2 using four different algorithms and also signifies that COBFO has better results as compared to GA, BFO and CS. The amplitude responses using CS and COBFO are shown in Fig.3 and Fig.4 respectively.

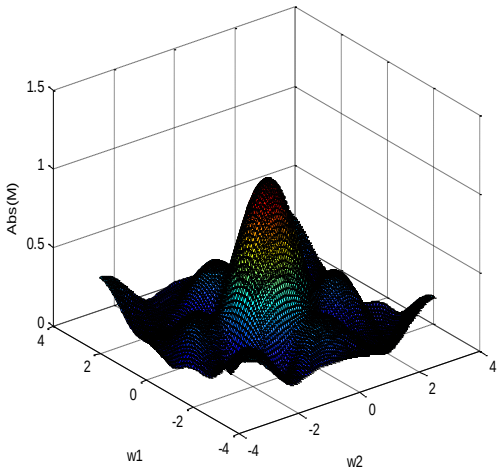

Fig.3. Amplitude Response of problem-2 using CS

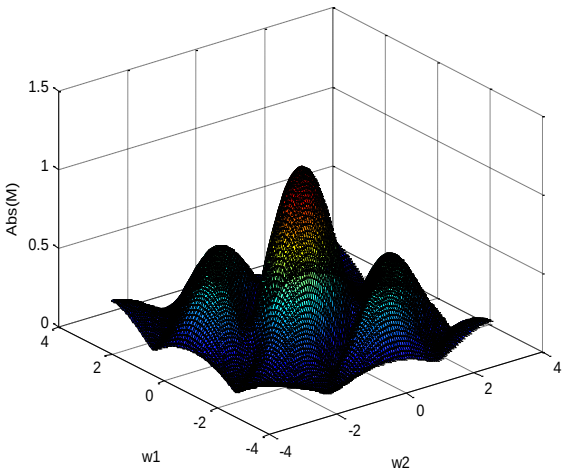

Fig.4. Amplitude Response of problem-2 using COBFO

Table.3. Estimated Parameters for Problem-2

Optimization algorithm	**COBFO**		**CS**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
b_0	**1.3238**	**-0.1589**	-6.6086	-1.6075	-1.8269	-2.2155	3.6872	3.4723
c_0	**-0.0731**	**0.6896**	0.8525	-2.1519	2.9236	-0.5384	0.606	0.923
d_0	**-2.1992**	**-1.5147**	4.7736	2.7727	4.94	1.7552	-1.2457	-2.4745
b_1	**0.6409**	**-0.5654**	1.4429	1.5887	0.8682	1.869	2.856	1.907
c_1	**0.8778**	**1.315**	0.8389	0.7004	0.8789	-0.5394	2.3993	1.3661
d_1	**0.8869**	**0.8154**	1.8176	1.9922	1.4535	1.0431	1.6813	0.8286
b_2	**1.7112**	**0.3854**	3.9213	-1.0874	3.4271	-0.3828	0.7166	0.7886
c_2	**3.3517**	**-0.3661**	7.6015	0.1009	2.7991	-0.3233	0.8444	-1.3044
d_2	**0.5174**	**0.6407**	-1.3271	-1.0475	-0.1248	-0.8873	0.1183	0.1742

3.2.3 Results of Problem-3:

There are 8 different filter coefficients in problem-3. The optimized values of these coefficients have been computed using four algorithms and the corresponding outcomes are tabulated in Table.4. From the table, it can be found that COBFO has superior performance as compared to the other three evolutionary techniques. The magnitude of amplitude response for problem-3 using CS and COBFO is shown in Fig.5 and Fig.6 respectively.

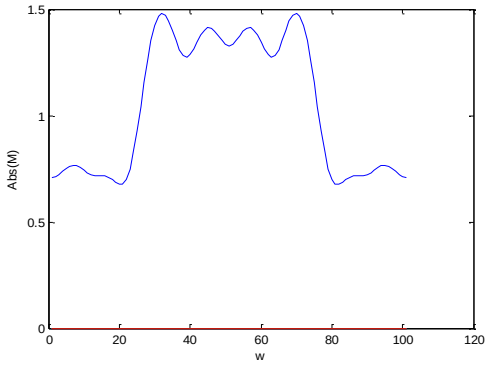

Fig.5. Amplitude Response of problem-3 using CS

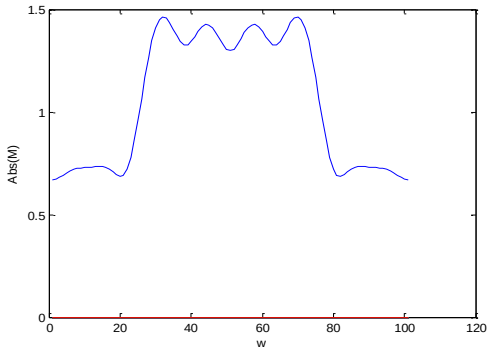

Fig.6. Amplitude Response of problem-3 using COBFO

Table.4. Estimated Parameters for Problem-3

Optimization algorithm	**COBFO**		**CS**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
a_1	**0.4166**	**0.9552**	0.4009	-0.2403	0.4166	-0.4066	0.3969	-0.8803
a_2	**0.0606**	**0.6675**	0.0774	0.9712	0.0606	0.8302	0.0772	1.1877
a_3	**-1292**	**0.4477**	-0.123	1.1965	-1292	-0.5538	-0.1365	2.1729
a_4	**-0.0557**	**-0.6084**	-0.0922	0.3111	-0.0557	-0.4611	-0.0394	0.8227
a_5	**0.0789**	**2.6964**	0.0949	0.9235	0.0789	0.0637	0.0957	-0.2176
a_6	**0.0213**	**0.1999**	0.0643	1.2303	0.0213	0.8138	0.0349	0.3898
a_7	**-0.0523**	**-0.9995**	-0.0053	0.2401	-0.0523	-2.3558	-0.0463	1.943
a_8	**-0.0419**	**-0.6299**	-0.0919	-0.2864	-0.0419	-0.7056	-0.055	0.3972

Table.5. Estimated Parameters for Problem-4

Optimization algorithm	**COBFO**		**CS**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
a_1	**0.417**	**0.8142**	0.3466	-0.3489	0.417	1.2971	0.4293	-0.3534
a_2	**0.0578**	**0.1028**	-0.0147	-0.1656	0.0578	-0.1759	0.0388	0.7967
a_3	**-0.1548**	**-0.2008**	-0.1513	-0.5909	-0.1548	-0.9972	-0.1032	0.8351
a_4	**0.0136**	**0.0187**	-0.0003	0.8194	0.0136	-0.4534	-0.039	-0.6195
a_5	**0.0872**	**-0.8507**	0.1183	0.2934	0.0872	1.0086	0.0719	1.0565
a_6	**-0.0218**	**-0.1997**	-0.063	0.193	-0.0218	0.8267	-0.0144	0.2212
a_7	**-0.0528**	**-0.2866**	-0.0245	-0.0311	-0.0528	0.6832	-0.0442	0.056
a_8	**0.0198**	**0.1959**	0.0725	0.3202	0.0198	0.5761	-0.0052	-0.1246

3.2.4 Results of Problem-4:

The Table.5 presents the optimized coefficient values for problem-4 using four different algorithms and also signifies that COBFO has better results as compared to GA, BFO and CS. The amplitude responses of 1-D FIR filter (problem-4) using CS and COBFO are shown in Fig.7 and Fig.8 respectively.

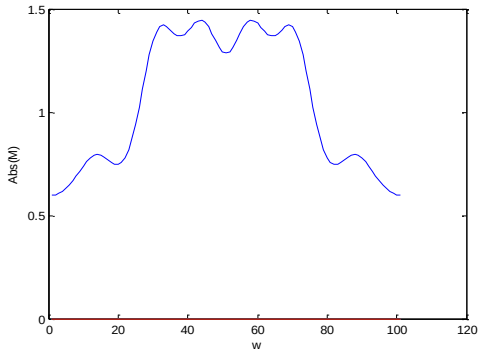

Fig.7. Amplitude Response of problem-4 using CS

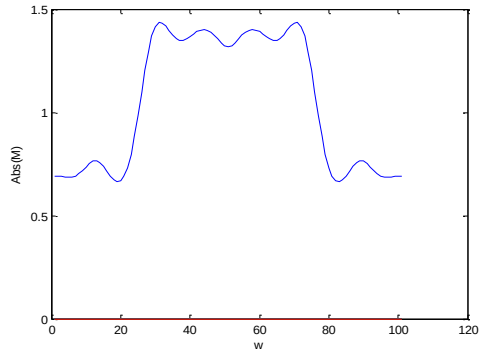

Fig.8. Amplitude Response of problem-4 using COBFO

3.2.5 Results of Problem-5:

The filter coefficients for problem-5 are optimized using four optimization algorithms and the corresponding estimated values are presented in Table.6. The estimated amplitude responses for problem-5 using CS and COBFO algorithms are shown in Fig.9 and Fig.10 respectively. The best results with minimum error and more closure values are shown in boldface letters. It is seen that, for problem-5, the best results are obtained by using the COBFO algorithm.

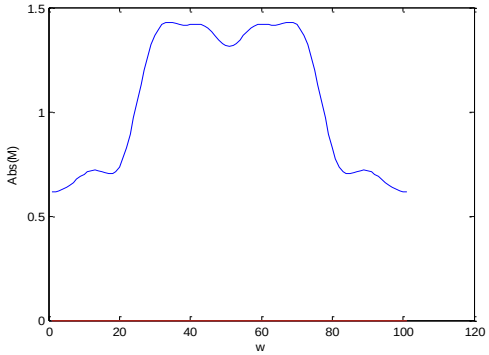

Fig.9. Amplitude Response of problem-5 using CS

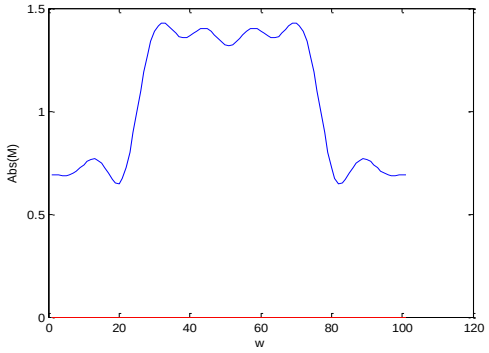

Fig.10. Amplitude Response of problem-5 using COBFO

All these magnitude responses are shown using the results obtained by minimizing *J* for the evaluation of 9000 functions. In each of the magnitude responses, it can be observed that the best magnitude response is provided by COBFO as compared to CS. The percentage of minimization of objective function value is more in the case of COBFO as compared to the other three evolutionary techniques.

Table.6. Estimated Parameters for Problem-5

Optimization algorithm	**COBFO**		**CS**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
a_1	**0.3811**	**-0.7231**	0.1292	-0.2621	0.1073	0.3334	-0.0329	0.2871
a_2	**0.1073**	**0.637**	-0.14	0.3406	0.2312	-0.3282	0.2005	-0.5549
a_3	**-0.1833**	**0.133**	0.1211	-0.0359	-0.3036	-0.5221	-0.3366	-0.1647
a_4	**-0.254**	**0.1843**	0.0678	0.1072	-0.0754	0.1481	-0.0825	-0.2184
b_1	**0.0349**	**0.7966**	0.3213	0.2258	0.3083	-0.9874	0.4527	-0.2866
b_2	**-0.0612**	**-0.9106**	0.1243	0.8133	-0.2028	-0.1511	-0.0652	0.5151
b_3	**0.0729**	**1.8359**	-0.2174	-0.1164	0.1233	-0.8379	0.1539	0.9424
b_4	**0.1828**	**0.4559**	-0.1135	-0.4685	0.1461	0.4773	0.1565	-0.7384

4. DISCUSSIONS

In the previous section, the experimental results for different problems of FIR filter using four different evolutionary techniques is discussed. This proposed work aims to provide optimum results with minimal computational effort. The computational effort of any optimization algorithm is assessed in terms of CPU elapsed time and cost function. So, this section discusses the computational burdens of each algorithm.

4.1 CPU ELAPSED TIME

Using 9000 and 15000 number of function evaluations for the problems 1 to 5 in section 2.1 of this paper, the performance of CS, GA, BFO, and COBFO algorithms are compared in terms of the elapsed time (in sec) and *J* (min). The Table.7 tabulates the estimated time lapse for 9000 number of function evaluations, while Table.8 tabulates the same for 15000 number of function evaluations. The Table.7 shows that the Cuckoo Search requires approximately 20% less time to compute when compared to COBFO, BFO, and GA. Thus, it is inferred that CS is the least time consuming process.

The estimated elapsed time (in sec) for the problems 1 to 5 using CS, GA, BFO and COBFO algorithms for 9000 and 15000 number of function evaluations are shown in Fig.11 and Fig.12 respectively. Both the figures reveal that the time required for CS to execute two sets of function evaluations is the minimum.

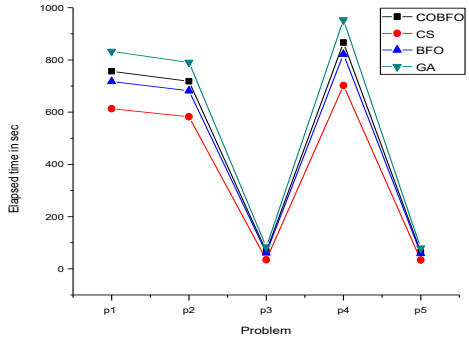

Fig.11. Elapsed Time for 9000 number of function evaluations

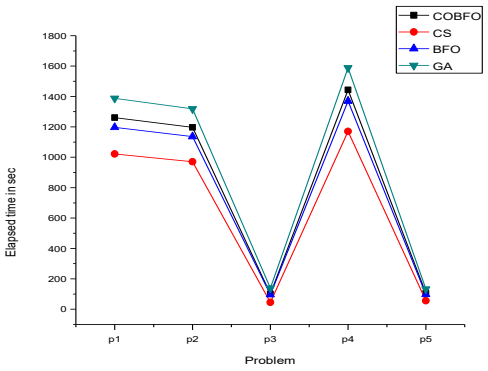

Fig.12. Elapsed Time for 15000 number of function evaluations

Table.7. Estimated Elapsed Time (in sec)

Optimization algorithm	**COBFO**		**Cuckoo search**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
P_1	756.6	1261	**613.28**	**1022.13**	717.84	1196.4	832.66	1387.77
P_2	718.38	1197.3	**582.3**	**970.5**	681.58	1135.97	790.6	1318.4
P_3	66.09	100.15	**34.9**	**44.84**	61.55	95.92	83.37	138.95
P_4	866.25	1443.75	**702.16**	**1170.27**	821.87	1369.78	953.32	1588.87
P_5	63.02	105.04	**33.28**	**55.47**	58.6	97.67	79.38	132.3

Table.8. Estimated J (min)

Optimization algorithm	**COBFO**		**Cuckoo search**		**BFO**		**GA**	
No. of function evaluations	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**	**9000**	**15000**
P_1	**11.48**	**10.62**	11.75	10.86	11.5	10.63	11.76	10.87
P_2	**15.3**	**14.15**	18.77	17.35	15.34	14.18	18.91	17.48
P_3	**16.91**	**15.64**	16.98	15.7	16.92	15.64	17.26	15.96
P_4	**8.63**	**7.98**	8.74	8.08	8.63	8	8.94	8.27
P_5	**16.92**	**15.65**	17.14	15.85	16.96	15.68	17.39	16.08

4.2 COST FUNCTION EVALUATION

The estimated J (min) for problem-1 to problem-5 using CS, GA, BFO and COBFO algorithms for 9000 and 15000 number of function evaluations are shown in Fig.13 and Fig.14 respectively. Though CS proves to be the time-efficient process as per Table.8, we see that the best valued optimizations with minimum error are obtained using COBFO as compared to CS, BFO, and GA.

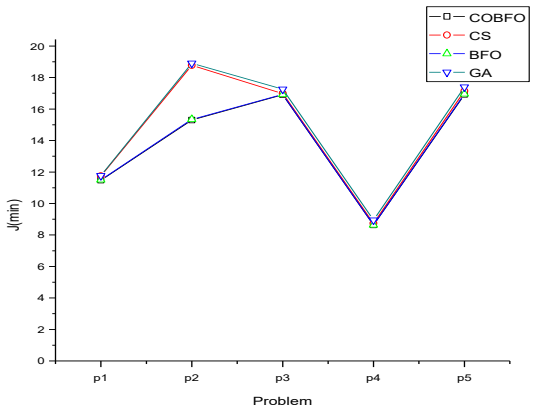

Fig.13. Estimated J (min) for 9000 number of function evaluations

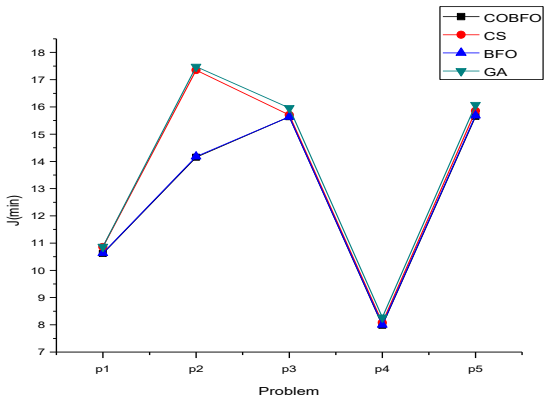

Fig.14. Estimated J (min) for 15000 number of function evaluations

5. CONCLUSION

The stability of the FIR filter is not a major concern but the optimal design with less computational burden is an important requirement. To address this requirement, the efficient computation of 2-D and 1-D FIR filters using COBFO, CS, BFO, and GA algorithms has been analyzed with the help of five different examples (two 2-D and three 1-D FIR filters).

From the experimental results and discussions, it can be concluded that COBFO is a better optimization approach than CS, BFO, and GA for FIR filter design. Though, it should be noted that the CS algorithm is an efficient computational technique when it comes to utilizing the least time to predict the filter coefficients. Yet, for a better evaluation of cost function and minimum error *J* (min) value, COBFO is the best suitable optimized evolutionary algorithm as compared to GA, BFO and CS.

REFERENCES

[1] R. Gonzalex and R. Woods, "*Digital Image Processing*", 2nd Edition, Prentice Hall, 2001.

[2] R. Kreuger, "Vertex-EM FIR Filter for Video Applications", Available at: http://direct.xilinx.com/bvdocs/appnotes/xapp241.pdf, Accessed on 2014.

[3] S.M.S. Gong and A. Psarrou, "*Dynamic Vision: From Images to Face Recognition*", 1st Edition, Imperial College Press, 2000.

[4] J.G. Poakis and D.G. Manolakis, "*Digital Signal Processing (Principals, Algorithms and Application)*", 3rd Edition, Pearson Education, 2007.

[5] S.G. Tzafestas, "*Multidimensional Systems, Techniques and Applications*", Pearson Education, 1986.

[6] N. Mastorakis, I.F. Gonos and M.N.S. Swamy, "Design of Two-Dimensional Recursive Filters using Genetic Algorithms", *IEEE Transactions on Circuits and Systems*, Vol. 50, No. 2, pp. 634-639, 2003.

[7] S.K. Sarangi, R. Panda and M. Dash, "Design of 1-D and 2-D Recursive Filters using Crossover Bacterial Foraging and Cuckoo Search Techniques", *Engineering Applications of Artificial Intelligence*, Vol. 34, No. 3, pp. 109-121, 2014.

[8] T. Kaczorek, "*Two-Dimensional Linear Systems*", Springer, 1985.

[9] S.T. Tzeng, "Design of 1-D FIR Digital Filters with Symmetric Properties of Magnitude and Phase Responses by Genetic Algorithm Approach", *Proceedings of National Symposium on Telecommunication*, pp. 43-48, 2002.

[10] S.T. Tzeng, "A Unified Approach to the Design of 2-D FIR Digital Filters by Genetic Algorithm Approach", *Proceedings of International Conference on Electronic Communication and Applications*, pp. 409-414, 2004.

[11] W.P. Zhu, M.O. Ahmad and M.N.S. Swamy, "A Closed-Form Solution to the Least-Square Design Problem of 2-D Linear-Phase FIR Filters", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, Vol. 44, No. 12, pp. 1032-1039, 1997.

[12] M. Oner, "A Genetic Algorithm for Optimization of Linear Phase FIR Filter Coefficients", *Proceedings of the Asilomar Conference on Signals. Systems and Computers*, pp. 1397-1400, 1998.

[13] M.K. Naik and R. Panda, "Adaptive Nonlinear Signal Approximation using Bacterial Foraging Strategy", *Proceedings of International Conference on Swarm, Evolutionary and Memetic Computing*, pp. 362-369, 2010.

[14] S. Mangano, "*Genetic Algorithms, Computer Design*", Springer, 1995.

[15] R. Panda and M.K. Naik, "A Crossover Bacterial Foraging Optimization Algorithm", *Applied Computational Intelligence and Soft Computing*, Vol. 23, No. 2, pp. 1-7, 2012.

[16] R. Panda and M.K. Naik, "Fusion of Infrared and Visual Images using Bacterial Foraging Strategy", *WSEAS Transactions on Signal Processing*, Vol. 8, No. 4, pp. 145-156, 2012.

[17] R. Panda, M.K. Naik and B.K. Panigrahi, "Face Recognition using Bacterial Foraging Strategy", *Swarm and Evolutionary Computation*, Vol. 1, No. 1, pp. 138-146, 2011.

[18] X.S. Yang and S. Deb, "Cuckoo Search Via Levy Flights", *Proceedings of World Congress on Nature and Biologically Inspired Computing*, pp. 210-214, 2009.

[19] X.S. Yang and S. Deb, "Engineering Optimization by Cuckoo Search", *International Journal of Mathematical Modelling and Numerical Optimization*, Vol. 1, No. 4, pp. 330-343, 2010.

[20] R. Panda, S. Agrawal and S. Bhuyan, "Edge Magnitude based Multilevel Thresholding using Cuckoo Search Algorithm", *Expert Systems with Applications*, Vol. 40, No. 18, pp. 7617-7628, 2013.