ICSP2008 Proceedings

A Comparison between Genetic Algorithm and PSO for Linear Phase FIR Digital Filter Design

Meisam Najjarzadeh, Ahmad Ayatollahi
Iran University of Science & Technology
meisam_najjarzadeh@yahoo.com, Ayatollahi@iust.ac.ir

Abstract

A comparative study between Genetic Algorithm and Particle Swarm Optimization in FIR filter design is presented in this paper. FIR filter design involves multi-parameter optimization, on which the existing optimization algorithms don't work efficiently. Given the filter specification to be designed, both algorithms generate a set of filter coefficients and try to meet the ideal frequency characteristic. For the problem at hand, the simulation of designing FIR filters have been done and the simulation results demonstrate that PSO is better than GA, not only in the convergence speed but also in the performance of the filter.

1. Introduction

The digital filter is a basic building block for digital signal processing systems [1]. Finite Impulse Response (FIR) digital filters are preferred in the most wireless communication systems due to its distinguishing features such as: guaranteed stability, linear phase and low coefficient sensitivity [2]. Currently, there exist some approaches for FIR filter design. These methods, such as window based design [3], frequency sampling method and best uniform approximation, are all based on approximation to frequency characteristic of ideal filters.

Recently, many evolutionary computation techniques, such as GA and PSO have been emerged into optimum filter designs. As we know, GA is difficult to implementation because of the coding complexity and its convergence speed is low. This motivates to using a recently developed stochastic optimization algorithm called PSO to design optimum FIR filter.

The Particle Swarm Optimization (PSO), an evolutionary algorithm, is simple to implement and its convergence may be controlled via few parameters.

978-1-4244-2179-4/08/$25.00 ©2008 IEEE

The PSO has been used in wide range of optimization problems because of its simplicity. The PSO is similar to GA, in that the system is initialized with a population of random solution, firstly evaluated itself using a fitness function, do some random search according to fitness value [5]. However, it is unlike GA because PSO updates itself without any genetic operator such as crossover and mutation. In the PSO, each potential solution is assigned a randomized velocity, and the potential solutions, named as particles, with an important characteristic of memory are then flown through the problem space [6]. On the other hand, there are some differences between PSO and GA on the mechanism of information communion. Chromosomes share the information with each other so that the population moves to the best location smoothly [7]. But only the particle, which can find the potential solution in the population, can pass on the information to the others. So the whole process of searching and updating is to follow the current best solution. In that situation, all particles can converge to the best solution more quickly.

This paper will focus on using PSO in FIR digital filter design. Algorithm tries to find best coefficients that closely match the ideal frequency response. The rest of the paper is arranged as follows. In section 2, the general PSO is briefly described. In the section 3, the traditional GA is introduced. In section 4, the application of PSO to FIR filter design is presented and formulated. Section 5 is about the simulation results of designing two sample filters using PSO in comparison with GA. Some conclusion remark is given in section 6.

2. The particle swarm optimization

PSO was developed by Kennedy and Eberhart in 1995 [4], is a new global search technique. The underlying motivation for the development of PSO

algorithm was social behavior of animals such as bird flocking, fish schooling, and swarm theory. Like genetic algorithm, The PSO is a population-based random search technique and outperforms GA in many practical applications, particularly in nonlinear optimization problems. PSO has become an important method since it has less parameter, simplicity in software programming and the fast convergence rate.

In the standard PSO model, each individual is treated as volume-less particle in the D-dimensional space, with the position and velocity of *i*th particle represented as $X_i = (X_{i1}, X_{i2},\ldots, X_{iD})$ and $V_i = (V_{i1}, V_{i2},\ldots, V_{iD})$. The particles move according to the following equations:

$V_{id}=\omega*V_{id}+c_1*rand()*(P_{id}-X_{id})+c_2*rand()*(P_g-X_{id})$ (1)

$X_{id} = X_{id}+V_{id}$ (2)

Where c_1 and c_2 are positive constant and are typically, set to a value of 2.0. rand( ) and rand( ) are two random functions in the range of [0,1]. Vector $P_i = (P_{i1}, P_{i2},\ldots, P_{iD})$ is the best previous position (the position giving the best fitness value) of particle i called pbest, and vector $P_g = (P_{g1}, P_{g2},\ldots, P_{gD})$ is the position of the best particle in the population and called gbest. X_{id}, V_{id}, P_{id} are the *d*th dimension of vectors X_i, V_i, P_i. parameter ω is the inertia weight introduced to accelerate the convergence speed PSO. Usually parameters ω decreases linearly according to the following formula:

$\omega = \omega_{max} - iter*(\omega_{max-}\ \omega_{min})$ (3)

where ω_{max}, ω_{min} are the maximum and minimum value of ω respectively, which always take value 0.9 and 0.4, iteration is the number of evolutions, iter is currently iteration of the evolution.

A Typical PSO algorithm for filter designing is as follows:

1) Given the desired target of the filter
2) Set the parameters of the algorithm, including the number of the particles, dimension of the parameters
3) Initialize the position of the particles
4) Evaluate the fitness function value
5) Update the particles position
6) Recalculate the fitness function value
7) Compare updated gbest position of the particle and global best position (gbest) of the population to judge whether the gbest need to be updated
8) Loop from step4 to step7 until a stop criterion is met
9) Output the optimal solution that is the coefficient of the desired filter.

3. The genetic algorithm

Genetic algorithm is based on the concept of "survival of the fittest" [7]. Genetic algorithms are optimization methods that resemble the natural selection. A set of numbers which can be a solution to the problem at hand is called genome (chromosome). A set of genomes is called population. The GA creates new generations by applying some genetic operators to the individuals of a population. A typical GA can be summarized as follows:

1) Generate initial population and compute score of each individual (Initialization)
2) Select two individuals for mating (Selection)
3) Mate two selected individuals (Crossover)
4) Mutate the offspring (Mutation)
5) Calculate scores of offspring (Evaluation)
6) Repeat steps 2-5 until a predefined number of offspring is generated.
7) Insert new offspring into the population (Replacement)
8) Repeat steps 2-7 while termination criterion is not met.

The encoding of the genome and defining an evaluation function are the most important parts of GA design process. The structure of the genome must represent a solution to the problem. Evaluation function on the other hand, compares genomes to a goal and assigns a score to them. GA uses scores to rank the genomes in population.

4. Application of algorithms to FIR filter design

FIR digital filter has a finite number of nonzero entries of its impulse response such as h[n],n=0,1,…,N. Generally assume implicitly that $h[n] \neq 0$, $h[0] \neq 0$. The transfer function of the FIR filter is

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n} \quad (4)$$

And the frequency response of form is

$$H(e^{jw}) = \sum_{n=0}^{N-1} h(n) e^{-jwn} \quad (5)$$

Consider the ideal frequency response $H_d(e^{jw})$ with the samples divided into equal frequency interval, thus we can get

$$H_d(e^{jw})\Big|_{w=\frac{2\pi}{N}k} = H_d(k) \quad (6)$$

Where $H_d(k)$ is regarded as the frequency response of the filter to design. Equation (6) can be re-written as

$$H_d(k) = H_d(e^{jw})\Big|_{w=\frac{2\pi}{N}k}, k=0,1,\ldots,N\text{-}1 \quad (7)$$

To design a linear phase FIR filter, we must minimize the error between actual and ideal output. There exist some forms of error function for the filter

design. One of them is the least-squares method, which is applied in this paper.

We define the error function as the error between the desired magnitude and the actual amplitude at a certain frequency, that is

$$E(e^{jw}) = H_d(e^{jw}) - H(e^{jw}) \quad (8)$$

Thus we can adopt the objective function for the minimization as total squared error across frequency domains as follows

$$E(e^{jw}) = \sum_{i=1}^{M} \left[\left| H_d(e^{jw_i}) \right| - \left| H(e^{jw_i}) \right| \right]^2 \quad (9)$$

Where M is the number of frequency interval. From (5) we can write the above equation as

$$E(e^{jw}) = \sum_{i=1}^{M} \left[\left| H_d(e^{jw_i}) \right| - \left| \sum_{n=0}^{N-1} h(n) e^{-jw_i n} \right| \right]^2 \quad (10)$$

The problem is reduced to find out *h(n)* by minimizing the squared error *E*.

4.1. Coefficient encoding

The filter impulse response coefficients, h(0) to h(N), are sufficient to represent a digital FIR filter. Thus, N+1 coefficients of the filter form the genome and the particle position in the GA and the PSO, respectively. Each coefficient is represented by a floating number in the range [-1,1], inclusive.

4.2. Fitness function

we use the total squared error as the fitness function of FIR digital filter, that is

$$E(e^{jw}) = \sum_{i=1}^{M} \left[\left| H_d(e^{jw_i}) \right| - \left| \sum_{n=0}^{N-1} h(n) e^{-jw_i n} \right| \right]^2 \quad (11)$$

5. Simulation results

The sample filters are a low pass and a band pass filter. As we know, their ideal frequency response is as follows:

$$H_d(e^{jw}) = \begin{cases} 1, & 0 \le w \le w_c \\ 0, & w_c \le w \le \pi \end{cases} \quad (12)$$

where $w_c = 0.5\pi$.

$$H_d(e^{jw}) = \begin{cases} 1, & w_{cl} \le w \le w_{ch} \\ 0, & 0 \le w \le w_{cl}, w_{ch} \le w \le \pi \end{cases} \quad (13)$$

where $w_{cl} = 0.3\pi, w_{ch} = 0.7\pi$.

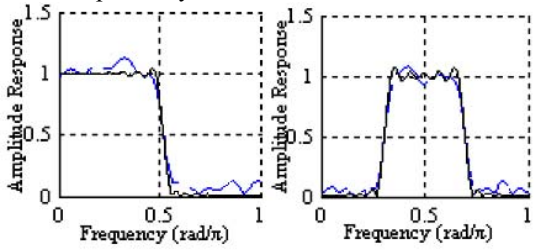
Each experiment is executed for 100 trial runs. As mentioned in previous section, the search scope of the coefficients of the filter is set as [-1,1]. The length of the filter is N=33.

Experiment 1: we applied PSO and GA to design the filter for the performance comparison. In each running the maximum iteration number is 1500, with population size 100. Other experiment setting is as follows:

GA: Tournament selection, uniform crossover, flip mutation (changes the value of the coefficient randomly in the range [-1,1]). Crossover rate=0.5, mutation rate=0.08.

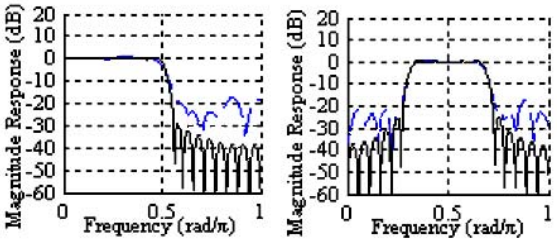
PSO: Maximum velocity=1, w_{max}=0.9, w_{min}=0.4, c1=c2=2.

Figure 1, 2, 3 demonstrate the comparison of amplitude response, magnitude response and convergence behavior of different two algorithms in FIR filter design. From the results generated for the two sample filters in figure 1, 2, 3 it can be exploited that PSO result in better frequency responses than GA with more rapid convergence rate. In figure 1 to 3, blue broken line and black real line denote the experiments by GA and PSO respectively.



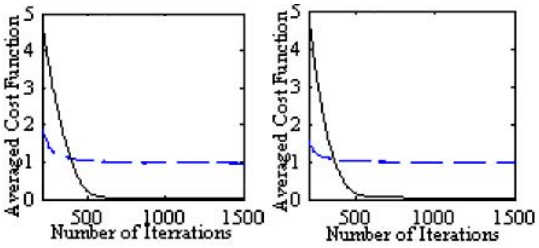
(a) **(b)**

Figure 1. Amplitude response of low pass (a) and band pass (b) FIR filters designed by algorithms



(a) **(b)**

Figure 2. Magnitude response of low pass (a) and band pass (b) FIR filters designed by algorithms

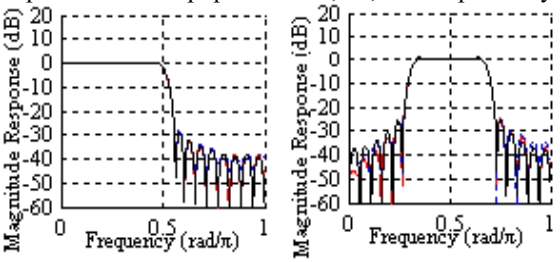


(a) (b)

Figure 3. Convergence behavior of algorithms in design low pass (a) and band pass (b) FIR filter

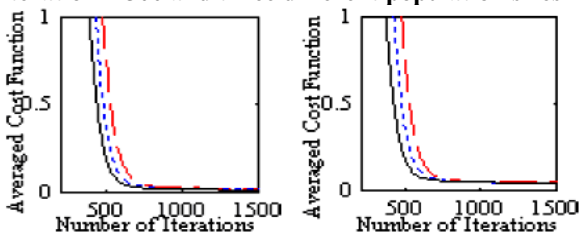
Experiment 2: In this experiment, we apply only PSO to design filters with different iteration number 1500; with different population sizes 40, 80, 100.

Figures 4, 5 demonstrate the comparison of magnitude response and convergence behavior of FIR filter designed by PSO and with iteration number 1500 and three different population sizes. From the results in figures 4, 5, it can be concluded that with the same iterations, the PSO has similar performance with different population sizes and the convergence speeds are almost the same. In figures 4, 5, red broken line, blue short dash line and black real line denote the experiments with population 40, 80, 100 respectively.



(a) (b)

Figure 4. Magnitude response of low pass (a) and band pass (b) FIR filter designed by PSO with iteration=1500 and three different population sizes



(a) (b)

Figure 5. Convergence behavior of PSO in design of low pass (a) and band pass (b) filter with iteration=1500 and three different population sizes

6. Conclusions

The application of the PSO to design FIR filter has been carried out. Comparison with GA has been made. The PSO, a recently developed stochastic efficient optimization algorithm, shows excellent ability to design such filters. According to the simulation results, we can conclude that PSO is able to converge to the global optima with less time consumption and seems to outperform Genetic Algorithm.

7. References

[1] A. Antoniou, Digital Filters: Analysis, Design, and Applications, McGraw-Hill, Singapore, 1993.

[2] S.K. Mitra, Digital Signal Processing: A Computer-based Approach, McGraw-Hill, 2001.

[3] A.V Oppenheim, R.W. Schafer, Digital Signal Processing, Prentice-Hall, 1975.

[4] J. Kennedy, R. Eberhart, "Particle Swarm Optimization", in Proc. IEEE int. Conf. On Neural Network,1995.

[5] R. Eberhart, Shi. Y., "Comparison Between Genetic Algorithms and Particle Swarm Optimization", Proc. 7th Ann. Conf. on Evolutionary Computation, San Diego, 2000.

[6] R. Eberhart, Shi. Y., "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization". Proc. Congress on Evolutionary Computation, San Diego, 2000.

[7]. D.E. Goldberg, "Genetic Algorithms in search Optimization and Machine learning", Addision-Wesely, 1989.