EURASIP Journal on Applied Signal Processing 2005:8, 1269–1276

Digital IIR Filter Design Using Differential Evolution Algorithm

Nurhan Karaboga
Department of Electronic Engineering, Faculty of Engineering, Erciyes University, 38039 Melikgazi, Kayseri, Turkey
Email: nurhan_k@erciyes.edu.tr

Received 14 May 2004; Revised 3 December 2004; Recommended for Publication by Ulrich Heute

Any digital signal processing algorithm or processor can be reasonably described as a digital filter. The main advantage of an infinite impulse response (IIR) filter is that it can provide a much better performance than the finite impulse response (FIR) filter having the same number of coefficients. However, they might have a multimodal error surface. Differential evolution (DE) algorithm is a new heuristic approach mainly having three advantages; finding the true global minimum of a multimodal search space regardless of the initial parameter values, fast convergence, and using a few control parameters. In this work, DE algorithm has been applied to the design of digital IIR filters and its performance has been compared to that of a genetic algorithm.

Keywords and phrases: digital IIR filter, design, evolutionary algorithms, differential evolution, genetic algorithm.

1. INTRODUCTION

Anything that contains information can be considered as a signal. Therefore, signals arise in almost every field of science and engineering. Two general classes of signals can be identified, namely, continuous-time and discrete-time signals. A discrete-time signal is one that is defined at discrete instants of time. The numerical manipulation of signals and data in discrete-time signals is called digital signal processing (DSP). The extraordinary growth of microelectronics and computing has had a major impact on DSP. Therefore, DSP has already moved from being primarily a specialist research topic to a one with practical applications in many disciplines. Almost any DSP algorithm or processor can reasonably be described as a filter. Filtering is a process by which the frequency spectrum of a signal can be modified, reshaped, or manipulated according to some desired specifications. Digital filters can be broadly classified into two groups: recursive and nonrecursive. The response of nonrecursive (FIR) filters is dependent only on present and previous values of the input signal. However, the response of recursive (IIR) filters depends not only on the input data but also on one or more previous output values. The main advantage of an IIR filter is that it can provide a much better performance than the FIR filter having the same number of coefficients. Design of a digital filter is the process of synthesizing and implementing a filter network so that a set of prescribed excitations results in a set of desired responses [1, 2]. However, there are some problems with the design of IIR filters [3, 4, 5]. The fundamental problem is that they might have a multimodal error surface. A further problem is the possibility of the filter becoming unstable during the adaptation process. This second problem can be easily handled by limiting the parameter space. In order to avoid the first problem, a design method which can achieve the global minima in a multimodal error surface is required. However, the conventional methods based on gradient search can easily be stuck at local minima of error surface. Therefore, some researchers have attempted to develop the design methods based on modern global optimization algorithms such as the simulated annealing (SA) [6, 7, 8], genetic algorithm (GA) [9, 10, 11, 12, 13, 14], ant colony optimization (ACO) [15], and tabu search (TS) algorithm [16]. Among these algorithms, GA is the one which has been applied more times than others on the IIR filter design.

A simple GA has three main operators: crossover, mutation operators from genetic science, and a selection operator simulating natural selection phenomena. GA can efficiently search large solution spaces due to its parallel structure and the probabilistic transition rules employed in the operators. However, a standard GA has two drawbacks: lack of good local search ability and premature convergence. In order to overcome this disadvantage of GA in numerical optimization problems, the differential evolution (DE) algorithm has been introduced by Storn and Price [17]. It has shown a good performance in finding optimal solutions in many cases. DE algorithm is a population-based algorithm like genetic algorithms using the similar operators; crossover, mutation, and selection. The studies on the design of optimal digital filters by using DE algorithm are not as common as GA. In the literature, there are only a few studies related to the application of DE algorithm to the digital filter design [18].

```
Initialization
Evaluation
REPEAT
        Mutation
        Recombination
        Evaluation
        Selection
UNTIL (termination criteria are met)
```

Algorithm 1: Basic DE algorithm.

In this work, the performance comparison of the design methods based on DE and GA is presented for digital IIR filters since DE algorithm is very similar to, but much simpler than, GA. The paper is organized as follows. Section 2 presents a basic DE algorithm. Section 3 describes the problem. In Section 4, firstly the performance of DE is compared with that of GA on a set of well-known numeric test functions [19] and secondly, DE and GA are applied to the design of low- and high-order digital IIR filters and the results obtained are discussed.

2. DIFFERENTIAL EVOLUTION ALGORITHM

An optimization task consisting of D parameters can be represented by a D-dimensional vector. In DE, a population of NP solution vectors is randomly created at the start. This population is successfully improved by applying mutation, crossover, and selection operators.

The main steps of a basic DE algorithm is given in Algorithm 1.

2.1. Mutation

For each target vector $x_{i,G}$, a mutant vector is produced by

$$v_{i,G+1} = x_{i,G} + K(x_{r1,G} - x_{i,G}) + F(x_{r2,G} - x_{r3,G}), \quad (1)$$

where $i, r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$ that are randomly chosen and must be different from each other. In (1), F is the scaling factor belonging to $[0, 2]$ affecting difference vector $(x_{r2,G} - x_{r3,G})$, K is the combination factor.

2.2. Crossover

The parent vector is mixed with the mutated vector to produce trial vector

$$\mathcal{U}_{ji,G+1} = \begin{cases} v_{ji,G+1} & \text{if } (rnd_j \le CR) \text{ or } j = rn_i, \\ x_{ji,G} & \text{if } (rnd_j > CR) \text{ and } j \ne rn_i, \end{cases} \quad (2)$$

where $j = 1, 2, \ldots, D$; $r_j \in [0, 1]$ is the random number; CR stands for the crossover constant $\in [0, 1]$; and $rn_i \in (1, 2, \ldots, D)$ is the randomly chosen index.

2.3. Selection

Performance of the trial vector and its parent is compared and the better one is selected. This method is usually named greedy selection. All solutions have the same chance of being selected as parents without dependence on their fitness value. The better one of the trial solution and its parent wins the competition providing significant advantage of converging performance over genetic algorithms.

3. DEFINITION OF THE PROBLEM

Consider the IIR filter with the input-output relationship governed by

$$y(k) + \sum_{i=1}^{M} b_i y(k-i) = \sum_{i=0}^{L} a_i x(k-i), \quad (3)$$

where $x(k)$ and $y(k)$ are the filter's input and output, respectively, and M $(\ge L)$ is the filter order. The transfer function of this IIR filter can be written in the following general form:

$$H(z) = \frac{A(z)}{B(z)} = \frac{\sum_{i=0}^{L} a_i z^{-i}}{1 + \sum_{i=1}^{M} b_i z^{-i}}. \quad (4)$$

Hence, the design of this filter can be considered as an optimization problem of the cost function $J(\mathbf{w})$ stated as follows:

$$\min_{w \in W} J(\mathbf{w}), \quad (5)$$

where $w = [a_0 a_1 \cdots a_L b_1 \cdots b_M]^{\mathrm{T}}$ is the filter coefficient vector.

The aim is to minimize the cost function $J(\mathbf{w})$ by adjusting $\mathbf{w}$. The cost function is usually expressed as the time-averaged cost function defined by (4):

$$J(\mathbf{w}) = \frac{1}{N} \sum_{k=1}^{N} \big(d(k) - y(k)\big)^2, \quad (6)$$

where $d(k)$ and $y(k)$ are the filter's desired and actual responses of the filter, respectively, and N is the number of samples used for the calculation of the cost function.

4. SIMULATION RESULTS

In this section, firstly the performance of DE algorithm is compared to that of the well-known models of GA, Grefenstette [20], Eshelman et al. [21], and Mühlenbein et al. [22], on a set of numeric test functions defined by De Jong [19], and secondly, the algorithms are applied to the design of IIR filters for the purpose of system identification.

4.1. Numeric function optimization

Five functions (F1–F5) presented in Table 1 have been firstly proposed by De Jong [19] to measure the performance of GA. After Jong, they have been extensively used by the researchers in GA and other algorithm communities. The test environment includes functions which are nonconvex (F2), discontinuous (F3), stochastic (F4), and multimodal (F5).

Table 1: Test functions.

Function number	Function	Limits
F1	$\sum_{i=1}^{3} x_i^2$	$-5.12 \le x_i \le 5.12$
F2	$100\left(x_1^2 - x_2\right)^2 + \left(1 - x_1\right)^2$	$-2.048 \le x_i \le 2.048$
F3	$\sum_{i=1}^{5} \text{integer}\,(x_i)$	$-5.12 \le x_i \le 5.12$
F4	$\sum_{i=1}^{30} i x_i^4 + \text{Gauss}(0,1)$	$-1.28 \le x_i \le 1.28$
F5	$\left[0.002 + \sum_{j=1}^{25} \left(\frac{1}{j + \sum_{i=1}^{2} \left(x_i - a_{ij}\right)^6}\right)\right]^{-1}$	$-65.536 \le x_i \le 65.536$

Table 2: Average evaluation numbers.

Algorithms	Function 1 Domain: $[-5.12, 5.12]$ Dimension: 3	Function 2 Domain: $[-2.048, 2.048]$ Dimension: 2	Function 3 Domain: $[-5.12, 5.12]$ Dimension: 5	Function 4 Domain: $[-1.28, 1.28]$ Dimension: 30	Function 5 Domain: $[-65.536, 65.536]$ Dimension: 2
Grefenstette	2210	14229	2259	3070	4334
Eshelman et al.	1538	9477	1740	4137	3004
Mühlenbein et al. ($\lambda = 4$)	1170	1235	3481	3194	1256
Mühlenbein et al. ($\lambda = 8$)	1526	1671	3634	5243	2076
DE	320	1120	160	2800	1500

F1 (Sphere)

The first function is smooth, unimodal, strongly convex, and symmetric.

F2 (Banana)

Rosenbrock's valley is a classic optimization function, also known as Banana function. The global optimum is inside a long, narrow, parabolic-shaped flat valley. To find the valley is trivial; however, convergence to the global optimum is difficult and hence this problem has been repeatedly used to assess the performance of optimization algorithms.

F3 (Step)

This function represents the problem of flat surfaces. Flat surfaces are obstacles for optimization algorithms because they do not give any information about which direction is favourable. Unless an algorithm has variable step sizes, it can get stuck on one of the flat plateaus.

F4 (Stochastic)

This is a simple unimodal function padded with noise. In this type function, the algorithm never gets the same value on the same point. Algorithms not doing well on this test function will do poorly on noisy data.

F5 (Foxholes)

Function F5 is an example of a function with many local optima. Many standard optimization algorithms get stuck in the first peak they find.

The DE algorithm has a few control parameters: number of population NP, scale vector F, combination coefficient K, and crossover rate CR. The problem-specific parameters of DE algorithm are maximum generation number $G_{\max}$ and number of parameters defining problem dimension D. The values of these two parameters depend on the problem to be optimized. The average evaluation numbers providing the minimum values of test functions in three-digit accuracy have been presented in Table 2. In this work, first the performance of DE algorithm was compared to that of GAs described by Grefenstette [20], Eshelman et al. [21], and Mühlenbein et al. [22]. DE algorithm was run 50 times for each function. For every run, the initial population was randomly created by means of using different seed numbers.

The results belonging to DE algorithm in Table 2 were achieved using the following parameter values: population size = 50, crossover rate = 0.8, scaling factor = 0.8, and combination factor = 0.8. From the results obtained by using GAs and DE, it is easy to see that the DE algorithm is usually able to find similar solutions for the test functions with less number of evaluations.

4.2. Digital IIR filter design

Application of the IIR filter in system identification has been widely studied since many problems encountered in signal processing can be characterized as a system identification problem (Figure 1). Therefore, in the simulation study, IIR filters are designed for the system identification purpose. In this case, the parameters of the filters are successively

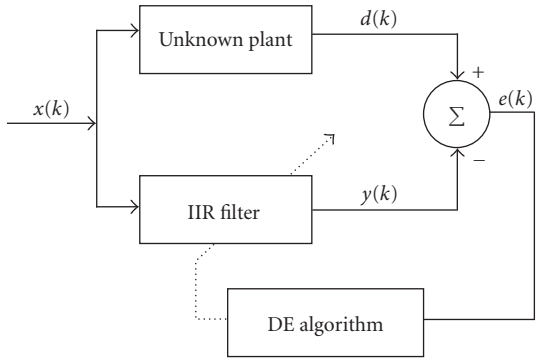


Figure 1: Block diagram of the system identification process using IIR filter designed by the DE algorithm.

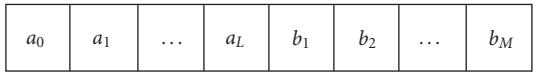


Figure 2: Representation of the parameters in the string form.

adjusted by the DE algorithm until the error between the output of the filter and the unknown system is minimized. By constraining the range of the filter coefficients, the stability is guaranteed. The filter coefficients are encoded in the string form as shown in Figure 2.

The fitness value of a solution i in the population is determined by using the following formula:

$$\text{fit}(i) = \frac{1}{1 + J(\mathbf{w})_i}, \tag{7}$$

where $J(\mathbf{w})_i$ is the cost function value computed for the solution i. The first two examples (low-order IIR filters) used in the simulation studies were taken from [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23].

Example 1. In the first example, the unknown plant and the filter had the following transfer functions:

$$\begin{aligned} H[z^{-1}] &= \frac{1}{1 - 1.2z^{-1} + 0.6z^{-2}}, \\ H_M[z^{-1}] &= \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2}}. \end{aligned} \tag{8}$$

The input, $x(k)$, to the system and the filter was a white sequence. Since the filter order is equal to the system order, a local-minima problem does not occur. Figure 3 presents the error surface for this filter. Figure 4 shows the evolution of the mean square error (MSE) averaged over 50 different runs of the DE. Each run had a randomly chosen initial $\mathbf{w}$. Figure 5 also demonstrates the evolution of parameters for a run.

The effect of the control parameters on the DE algorithm's performance was studied by Price [24]. The parameter values used in this study were selected according to the recommended values in that work. Also, in order to carry out the comparison of the algorithms in similar conditions, the values of similar control parameters of the algorithms were chosen to be equal to each other; for example, population

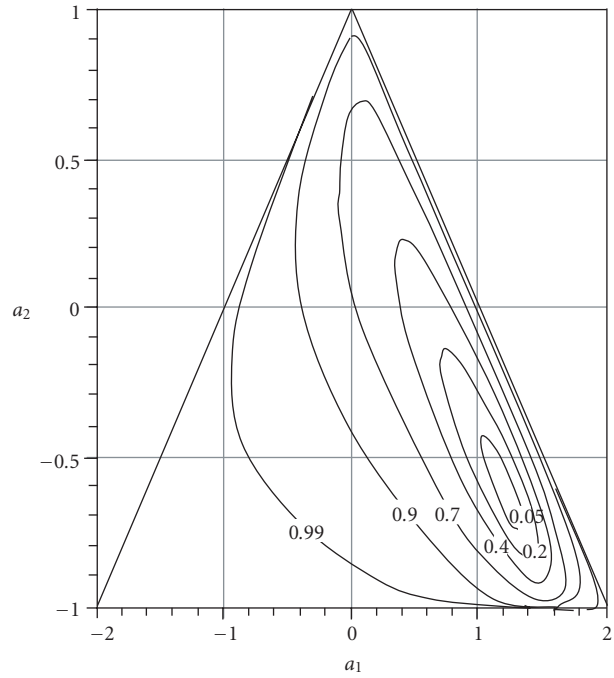


Figure 3: Error surface for the first filter.

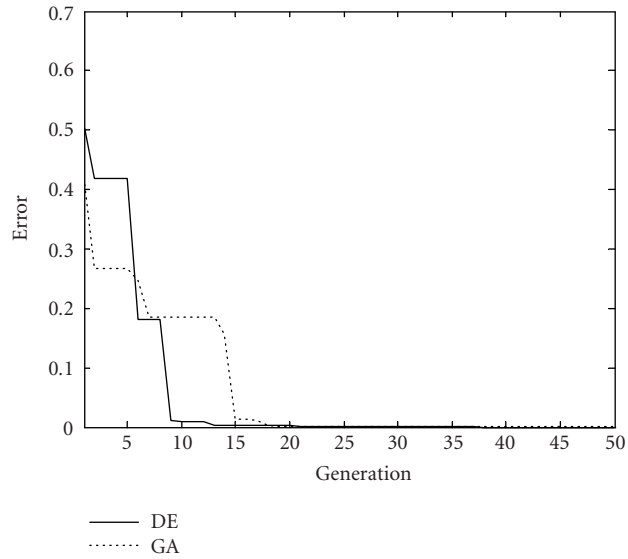
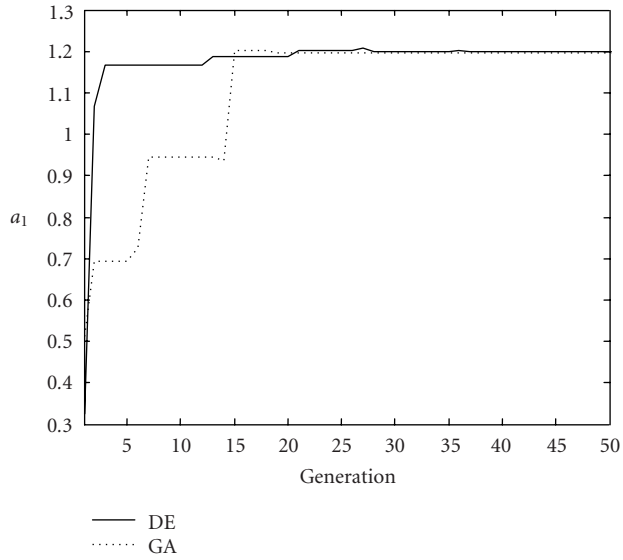


Figure 4: Cost function value versus number of evaluations averaged over 50 random runs for DE and GA.

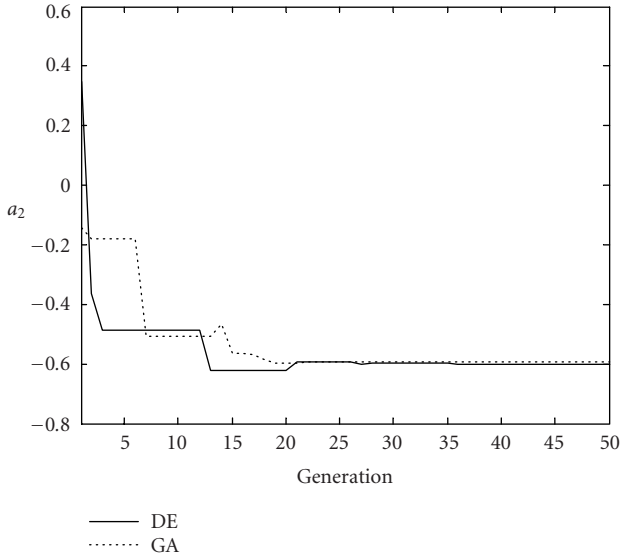
size, generation number, and crossover rate. Table 3 shows the control parameter values used for both algorithms in IIR filter design.

Example 2. In the second example, the plant was a second-order system and the filter was a first-order IIR filter with the following transfer functions:

$$\begin{aligned} H[z^{-1}] &= \frac{0.05 - 0.4z^{-1}}{1.0 - 1.1314z^{-1} + 0.25z^{-2}}, \\ H_M[z^{-1}] &= \frac{b}{1 - az^{-1}}. \end{aligned} \tag{9}$$



(a)



(b)

Figure 5: Evolution of the parameters of the first filter for both algorithms.

Table 3: Control parameter values used for the first two examples.

Differential evolution algorithm	Genetic algorithm
Population size = 20	Population size = 20
Crossover rate = 0.8	Crossover rate = 0.8
Scaling factor (F) = 0.8	Mutation rate = 0.2
Combination factor (K) = 0.8	
Generation number = 50	Generation number = 50

The system input was a uniform white sequence. The data length used in calculating the MSE was $N = 100$. Since the reduced-order filter is employed, the MSE is multimodal.

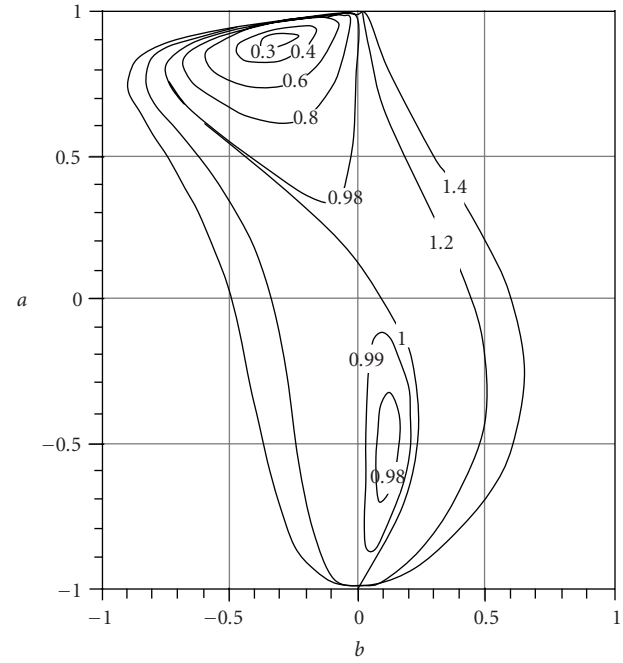


Figure 6: Error surface for the second filter.

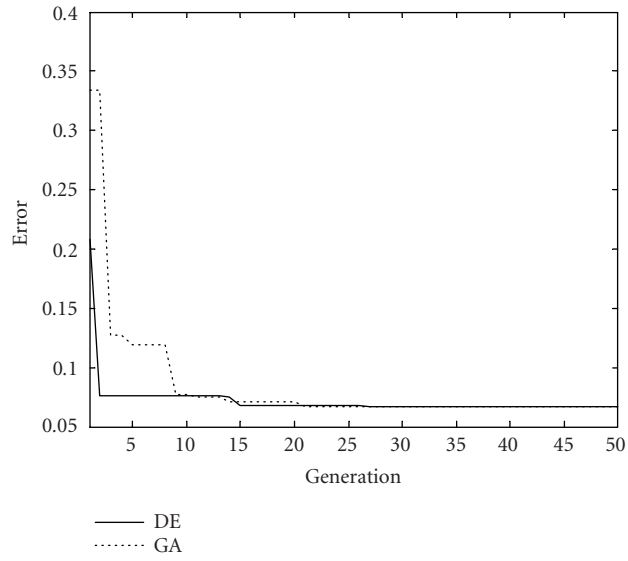
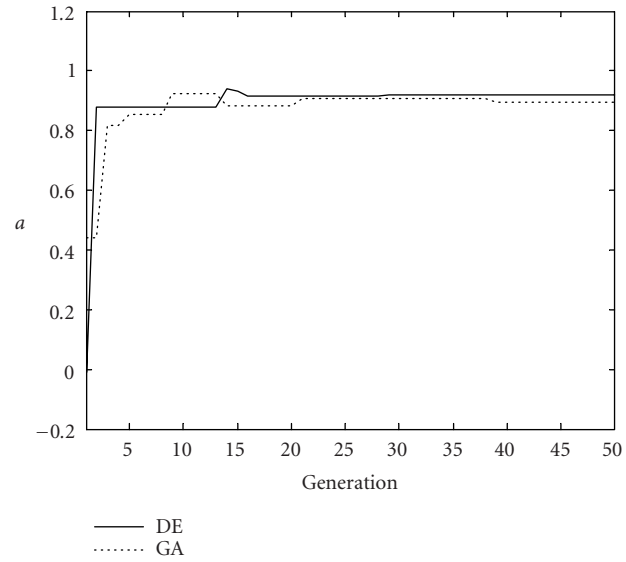


Figure 7: Cost function value versus number of evaluations averaged over 50 random runs for DE and GA.

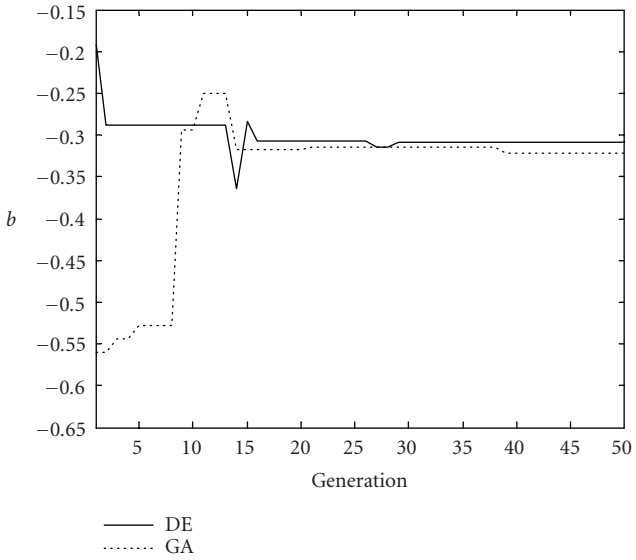
The error surface is given in Figure 6. Figure 7 presents the cost function value versus number of cost function evaluations averaged over 50 random runs. Each run had a randomly chosen initial $\mathbf{w}$ as in the first example. Figure 8 presents the evolution of both parameters for a run.

Example 3. In the third example, the plant was a sixth-order system and had the transfer function [25]

$$H[z^{-1}] = \frac{1 - 0.4z^{-2} - 0.65z^{-4} + 0.26z^{-6}}{1 - 0.77z^{-2} - 0.8498z^{-4} + 0.6486z^{-6}}. \quad (10)$$



(a)



(b)

Figure 8: Evolution of the parameters of the second filter for both algorithms.

The IIR filter was the fifth order and had the following transfer function:

$$H_M[z^{-1}] = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4} + b_5 z^{-5}}{1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + a_4 z^{-4} + a_5 z^{-5}}. \tag{11}$$

Since the system was a sixth-order system and the filter fifth order, the error surface is bimodal as in the second example. The system input was a uniform white sequence and the data length used in calculating the MSE was $N = 200$ in this example. Figure 9 presents the cost function value versus the number of cost function evaluations averaged over 50 random runs. For each run, a randomly chosen initial $\mathbf{w}$ was used as in the first two examples. Figures 10 and 11 present the evolution of the denominator and the nominator parameters for a run, respectively. The control parameter values employed in this example were the same as in the first two examples except the generation number. In this example, the algorithms were run for 1000 generations.

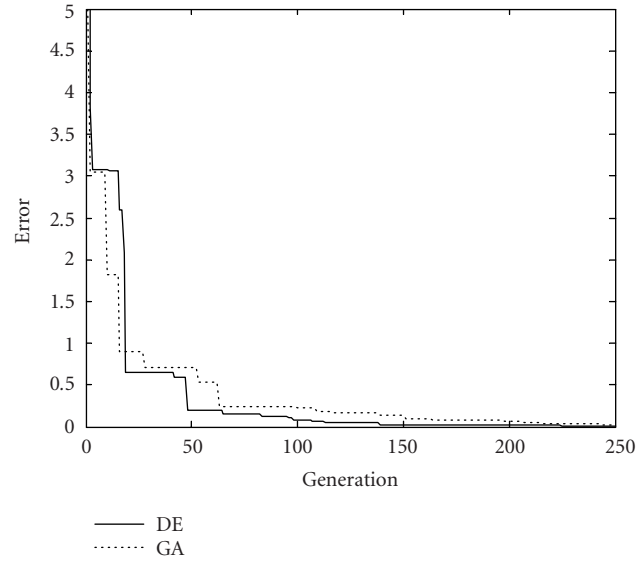


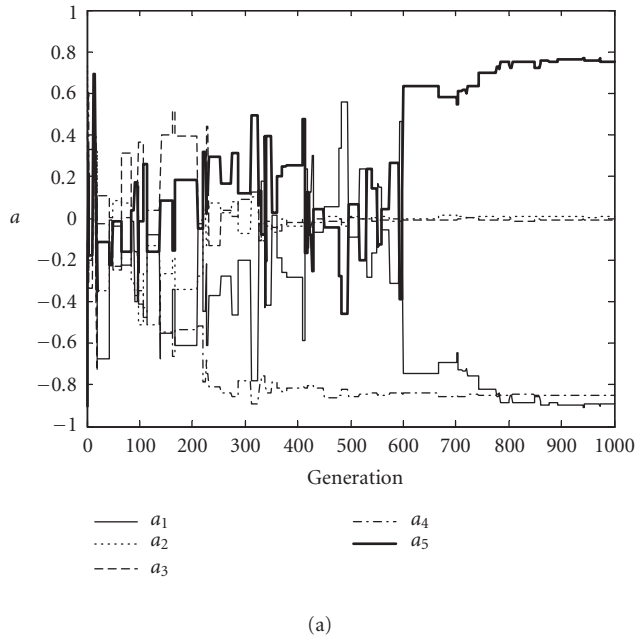
Figure 9: Cost function value versus number of evaluations averaged over 50 random runs for DE and GA.

In the simulations, three IIR filters were designed for the system identification purpose. As seen from Figure 4, DE usually designs an acceptable filter at around 10 generations although GA needs 15 generations for similar designs. For the second example, as seen from Figure 7, the convergence speed of DE is much better than GA, too. For the high-order filter example, as expected, the algorithms require more generations to design an acceptable filter. The DE algorithm needs about 140 generations although GA has 200 generation for designing an optimal filter design.

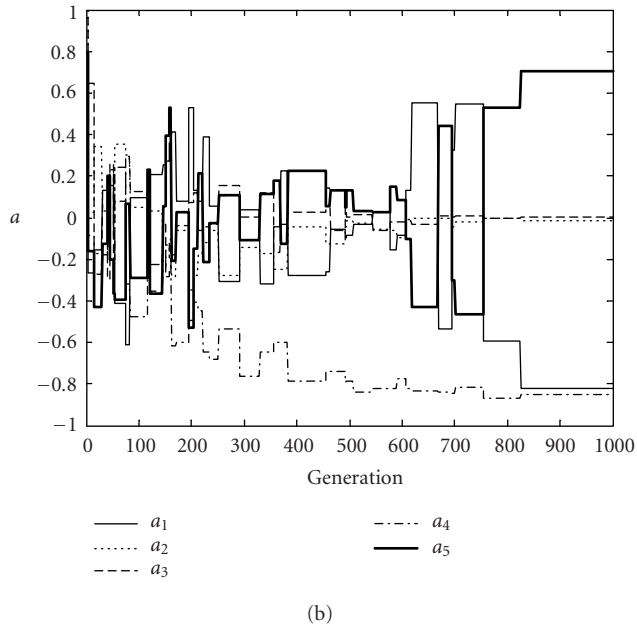
Consequently, DE algorithm produced good solutions to both the unimodal and multimodal filter cases. The performance of DE and GA can be compared in terms of the computation time, too. In the simulations, it was seen that the DE algorithm requires about 2–3 seconds although the GA algorithm needs approximately 50 seconds to design an optimal IIR filter for 50 generations. In terms of the final solution, the performance of DE is comparable to that of GA since the local search ability of DE is better than that of GA.

5. CONCLUSION

DE algorithm is a new heuristic approach mainly having three advantages; finding the true global minimum of a

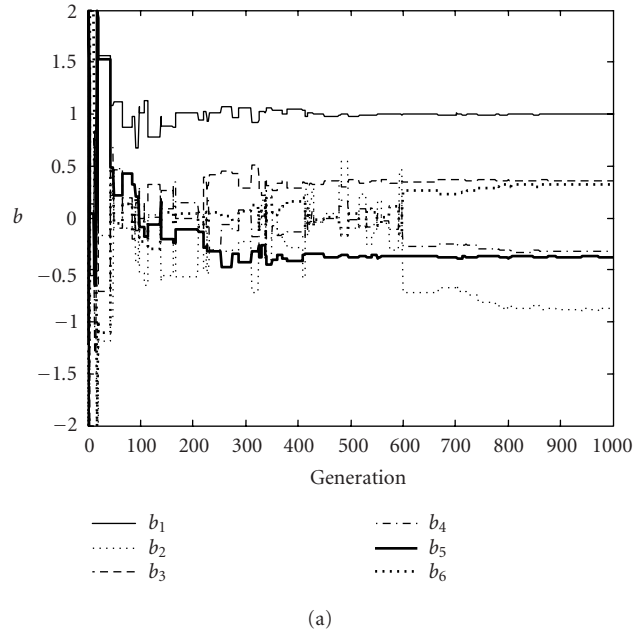


(a)

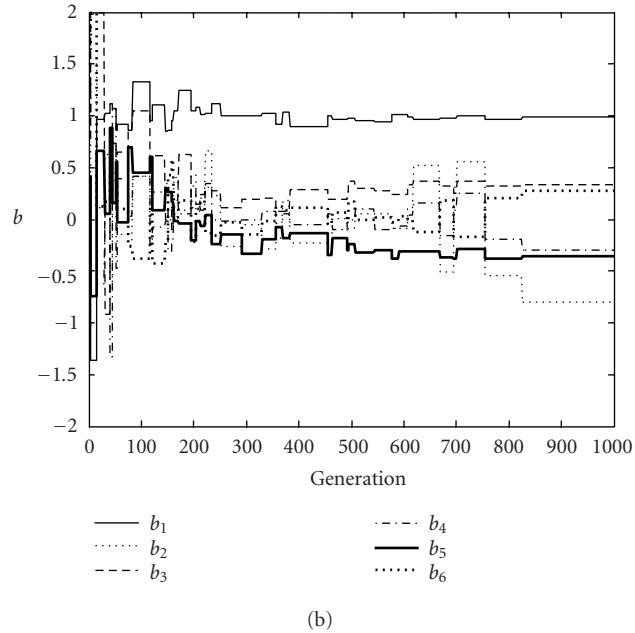


(b)

Figure 10: Evolution of the denominator parameters of the high-order filter for both algorithms; (a) DE and (b) GA.



(a)



(b)

Figure 11: Evolution of the nominator parameters of the high-order filter for both algorithms; (a) DE and (b) GA.

multimodal search space regardless of the initial parameter values, fast convergence, and using a few control parameters. In this work, the DE algorithm was applied to the IIR filter design. In the simulations, three digital IIR filters were designed for the purpose of system identification. From the simulation results, it was observed that the performance of the standard DE algorithm in terms of convergence speed and computation time required is better than that of the standard GA.

REFERENCES

[1] T. Gulzow, T. Ludwig, and U. Heute, "Spectral-subtraction speech enhancement in multirate systems with and without non-uniform and adaptive bandwidths," *Signal Processing*, vol. 83, no. 8, pp. 1613–1631, 2003.

[2] J. Kliewer, T. Karp, and A. Mertins, "Processing arbitrary-length signals with linear-phase cosine-modulated filter banks," *Signal Processing*, vol. 80, no. 8, pp. 1515–1533, 2000.

[3] J. J. Shynk, "Adaptive IIR filtering," *IEEE ASSP Magazine*, vol. 6, no. 2, pp. 4–21, 1989.

[4] M. Radenkovic and T. Bose, "Adaptive IIR filtering of nonstationary signals," *Signal Processing*, vol. 81, no. 1, pp. 183–195, 2001.
[5] S. D. Stearns, "Error surface of recursive adaptive filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 29, no. 3, pp. 763–766, 1981.
[6] R. Nambiar and P. Mars, "Genetic and annealing approaches to adaptive digital filtering," in *IEEE 26th Asilomar Conference on Signals, Systems & Computers*, vol. 2, pp. 871–875, Pacific Grove, Calif, USA, October 1992.
[7] S. Chen, R. H. Istepanian, and B. L. Luk, "Digital IIR filter design using adaptive simulated annealing," *Digital Signal Processing*, vol. 11, no. 3, pp. 241–251, 2001.
[8] J. Radecki, J. Konrad, and E. Dubois, "Design of multidimensional finite-wordlength FIR and IIR filters by simulated annealing," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 6, pp. 424–431, 1995.
[9] D. M. Etter, M. J. Hicks, and K. H. Cho, "Recursive adaptive filter design using an adaptive genetic algorithm," in *IEEE International Conference Acoustics, Speech, Signal Processing (ICASSP '82)*, vol. 7, pp. 635–638, Albuquerque, NM, USA, May 1982.
[10] N. E. Mastorakis, I. F. Gonos, and M. N. S. Swamy, "Design of two-dimensional recursive filters using genetic algorithms," *IEEE Trans. on Circuits and Systems I-Fundamental Theory and Applications*, vol. 50, no. 5, pp. 634–639, 2003.
[11] K. S. Tang, K. F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Processing Mag.*, vol. 13, no. 6, pp. 22–37, 1996.
[12] S. C. Ng, S. H. Leung, C. Y. Chung, A. Luk, and W. H. Lau, "The genetic search approach: a new learning algorithm for IIR filtering," *IEEE Signal Processing Mag.*, vol. 13, no. 6, pp. 38–46, 1996.
[13] R. Thamvichai, T. Bose, and R. L. Haupt, "Design of 2-D multiplierless IIR filters using the genetic algorithm," *IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 49, no. 6, pp. 878–882, 2002.
[14] A. Lee, M. Ahmadi, G. A. Jullien, R. S. Lashkari, and W. C. Miller, "Design of 1-D FIR filters with genetic algorithm," in *Proc. IEEE International Symposium on Circuits and Systems*, vol. 3, pp. 295–298, Orlando, Fla, USA, May/June 1999.
[15] N. Karaboga, A. Kalinli, and D. Karaboga, "Designing IIR filters using ant colony optimisation algorithm," *Journal of Engineering Applications of Artificia Intelligence*, vol. 17, no. 3, pp. 301–309, 2004.
[16] A. Kalinli and N. Karaboga, "New method for adaptive IIR filter design based on tabu search algorithm," to appear in *AEÜ International Journal of Electronics and Communications*, 2004.
[17] R. Storn and K. Price, "Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces," Tech. Rep. TR-95-012, International Computer Science Institute (ICSI), Berkeley, Calif, USA, March 1995.
[18] R. Storn, "Differential evolution design of an IIR-filter with requirements for magnitude and group delay," in *Proc. IEEE International Conference on Evolutionary Computation (ICEC '96)*, pp. 268–273, Nagoya, Japan, May 1996.
[19] K. A. De Jong, *An analysis of the behaviour of a class of genetic adaptive systems*, Ph.D. thesis, University of Michigan, Ann Arbour, Mich, USA, Dissertation Abstracts International vol. 36, no. 10, 5140B. (University Microfilms No. 76-9381), 1975.
[20] J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems Man and Cybernetics*, vol. 16, no. 1, pp. 122–128, 1986.
[21] L. J. Eshelman, R. A. Caruana, and J. D. Schaffer, "Biases in the crossover landscape," in *Proc. 3rd International Conference on Genetic Algorithms*, pp. 10–19, San Mateo, Calif, USA, 1989.
[22] H. Mühlenbein, M. Schomisch, and J. Born, "The parallel genetic algorithm as function optimizer," in *Proc. 4th International Conference on Genetic Algorithms*, pp. 271–278, Morgan Kaufmann Publishers, San Diego, Calif, USA, 1991.
[23] S. Chen and B. L. Luk, "Adaptive simulated annealing for optimisation in signal processing applications," *Signal Processing*, vol. 79, no. 1, pp. 117–128, 1999.
[24] K. V. Price, "An introduction to differential evolution," in *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds., chapter 6, McGraw Hill, London, UK, 1999.
[25] M. S. White and S. J. Flockton, "Adaptive recursive filtering using evolutionary algorithms," in *Evolutionary Algorithms in Engineering Applications*, D. Dasgupta and Z. Michalewicz, Eds., pp. 361–376, Springer, Berlin, Germany, 1997.

Nurhan Karaboga received the B.S., M.S., and Ph.D. degrees in electronic engineering from Erciyes University, Turkey, in 1987, 1990, and 1995, respectively. From 1987 to 1995, she was a Research Assistant in the Department of Electronic Engineering, Erciyes University. Currently, she is an Assistant Professor at the same department. From 1992 to 1994, she also worked as an Academic Visitor in the University of Wales College of Cardiff, UK. Her current research interests include genetic algorithms, ant colony algorithms, simulated annealing algorithm, differential evolution algorithm, immune algorithm, digital filter design, and data communication.