Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 4797 – 4817

doi:10.3906/elk-1809-83

Adaptive iir filter design using self-adaptive search equation based artificial bee colony algorithm

Burhanettin DURMUŞ[1], **Gürcan YAVUZ**[2,*], **Doğan AYDIN**[2]
[1]Department of Electrical and Electronics Engineering, Faculty of Engineering, Dumlupınar University, Kütahya, Turkey
[2]Department of Computer Engineering, Faculty of Engineering, Dumlupınar University, Kütahya, Turkey

Received: 12.09.2018 • **Accepted/Published Online:** 27.05.2019 • **Final Version:** 26.11.2019

Abstract: Infinite impulse response (IIR) system identification problem is defined as an IIR filter modeling to represent an unknown system. During a modeling task, unknown system parameters are estimated by metaheuristic algorithms through the IIR filter. This work deals with the self-adaptive search-equation-based artificial bee colony (SSEABC) algorithm that is adapted to optimal IIR filter design. SSEABC algorithm is a recent and improved variant of artificial bee colony (ABC) algorithm in which appropriate search equation is determined with a self-adaptive strategy. Moreover, the success of the SSEABC algorithm enhanced with a competitive local search selection strategy was proved on benchmark functions in our previous studies. The SSEABC algorithm is utilized in filter modelings which have different cases. In order to demonstrate the performance of the SSEABC algorithm on IIR filter design, we have also used canonical ABC, modified ABC (MABC), best neighbor-guided ABC, and an ABC with an adaptive population size (APABC) algorithms as well as other algorithms in the literature for comparison. The obtained results and the analysis on performance evolution of compared algorithms on several filter design cases indicate that SSEABC outperforms all considered ABC variants and other algorithms in the literature.

Key words: Artificial bee colony, digital infinite impulse response filters, system identification, self-adaptive strategy

1. Introduction

Finite impulse response (FIR) and infinitive impulse response (IIR) filters, which are the most important types of linear digital filters, are widely used in fields such as signal processing, communication, and parameter estimation [1]. These filters have also become effective tools in system identification applications [1–7]. FIR filters are known as feed forward or nonrecursive because their inputs depend only on current and past inputs. Therefore, the performance levels of these filters in the system identification models are not effective. IIR filter outputs depend on the current inputs and the past inputs as well as the past outputs. Because of their recursive and feedback structure, the system identification efficiency of IIR filters is much better than that of FIR filters. The IIR-based system identification problem is defined as constructing an IIR filter to represent an unknown system. The construction process is modeled by applying the same inputs to the unknown system and the IIR filter to minimize the error in the outputs. This modeling eventually turns into an optimization problem by calculating the appropriate values of the IIR filter coefficients that minimize the error. There are many studies in the literature for designing an optimal IIR filter [1, 8–10]. In these studies, generally, filter network

*Correspondence: gurcanyavuz@gmail.com

applications were performed on a predetermined data set, and filter coefficients to produce the desired response were determined using gradient-based methods. However, it has been stated that gradient-based methods can easily be trapped at local minima in IIR filter design with multimodal error surface [11]. On the other hand, the instability of the adaptation process of IIR filter becomes a disadvantage for gradient-based methods. Because searching IIR filter coefficients in the inappropriate range of search space will distract the filter outputs from the desired outputs and thus will lead to an unstable search process [12]. In recent years, learning-based adaptive filter designs are preferred over traditional methods to tackle the aforementioned problems. More specifically, metaheuristics, which converge the global optimum more quickly and increase adaptability, are mostly used in IIR filter design applications [13–15]. Improved particle swarm optimization (IPSO) is proposed by Zou et al. and it is applied to the IIR system identification problem [5]. In another study [16], the bat algorithm was developed and compared with other IIR filter applications reported in the literature. Mean square error (MSE) method is taken as performance measure. Furthermore, the IIR filter-based system identification problem is defined as benchmark to test the performance of newly proposed heuristic methods [17]. The benchmarked system identification problem is solved by the modified-interior search algorithm (M-ISA) using the IIR filter model which is in the same order and the reduced order as the unknown system. Similar works have been done with metaheuristics such as cuckoo search algorithm, differential evolution (DE), and craziness-based PSO where adaptive IIR filter designs have been realized to determine the optimal parameters of an unknown system [18–20].

The artificial bee colony (ABC) algorithm, originally proposed in [21], is also a metaheuristic approach. In recent years, several new ABC variants and their application to real-world problems [22–26] are introduced. Some of these studies are based on ABC algorithms for IIR filter design problem as well [13, 27]. In many studies, the canonical ABC algorithm and its variants have shown that they are very competitive with many other algorithms for continuous optimization problems. However, the performance of ABC algorithms vary depending on the problem type and its size. The most crucial and sensitive component that influences algorithm performance is the search equation used in the steps of employed bees and onlooker bees. On the other hand, A selected search equation may yield good results in one problem, but it may yield bad results in another problem. In order to overcome this problem, a self-adaptive search equation generation method which can find the appropriate search equation related to the nature of the problem is needed. In our previous work, the self-adaptive search-equation-based artificial bee colony (SSEABC) algorithm was designed for this purpose and achieved successful results in several types of benchmark continuous optimization functions [28, 29]. In this study, the SSEABC algorithm is applied to IIR filter design problem.

The contribution of this study can be summarized as follows:

- With the SSEABC, three modifications are introduced to the canonical ABC algorithms. The first modification is ”self-adaptive search equation selection strategy” that adaptively determines the appropriate search equation for the tackling problem instance. The second is to use competitive local search selection strategy that controls the invocation of local search procedure which greatly helps the algorithm to escape local optima. The last modification is incremental population size strategy that leads algorithm population converges quickly to good solutions.

- The SSEABC algorithm was previously used to solve theoretical problems. In this paper, SSEABC algorithm is proposed for digital IIR filter design as the first case study on a real-world problem. As system identification, the unknown system parameters are estimated with the same- and reduced-order IIR filter models.

- The SSEABC algorithm estimates the system parameter better than several ABC variants and other metaheuristic methods in the literature.
- In addition to the simulation examples, experiments are also carried out on a practical application of IIR system identification problems.

The paper is structured as follows. In Section 2, we define the IIR filter design problem. In Section 3, we present the canonical ABC algorithm. Then in Section 4, we briefly describe the SSEABC algorithm. The simulation results on three different problem examples and the results of the practical applications of IIR system identification problems are provided in Section 5. Finally, the article is concluded in Section 6.

2. Problem definition

The main purpose of the filter-based system identification problem is to estimate the parameters of an unknown system over a filter. In other words, the transfer function of the unknown system is monitored by the transfer function of the filter to determine the most appropriate filter coefficients. This process turns into an optimization problem by minimizing output errors generated by applying the same input signal to both the unknown system and the filter. A filter-based system identification is shown in Figure 1.

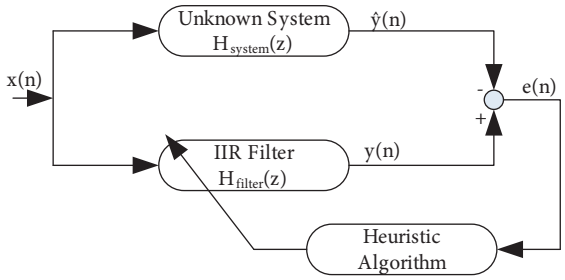


Figure 1. The schematic diagram of the IIR-filter-based system identification application

In general an IIR filter is represented by the following equation:

$$y(n) + \sum_{i=1}^{N} a_i y(n-i) = \sum_{i=0}^{M} b_i x(n-i), \tag{1}$$

where N is order of numerator, M is order of dominator, $x(n)$ and $y(n)$ are input and output value of the filter, a_i and b_i are the filter output and input coefficients at order i, respectively. The transfer function of the IIR filter is expressed as:

$$H_{filter}(Z) = \frac{Y(Z)}{X(Z)} = \frac{\sum_{i=0}^{M} b_i z^{-i}}{1 + \sum_{i=1}^{N} a_i z^{-i}}. \tag{2}$$

In the IIR-filter-based identification model, the goal is to approximate the transfer function of the unknown system, $H_{system}(z)$, with the filter transfer function, $H_{filter}(z)$. Therefore, the error difference between the output obtained from the transfer function of the unknown system and the IIR filter output is calculated. To minimize the error, the optimal solution vector is tried to be found. To do so, the objective

function is calculated from the following formula defined as the mean squared error (MSE) of the output values:

$$MSE = \frac{1}{L}\sum_{n=1}^{L} e^2(n) = \frac{1}{L}\sum_{n=1}^{L} [y(n) - \hat{y}(n)]^2 \tag{3}$$

where L is the total number of input samples, $y(n)$ and $\hat{y}(n)$ are outputs of IIR filter and unknown systems for sample input n, respectively.

3. Artificial bee colony algorithm

The artificial bee colony (ABC) [21] is one of the swarm-intelligence-based algorithms inspired by the foraging behavior of the bees in nature. This approach, which is used to solve continuous optimization problems at first, has been used effectively to solve real-world engineering problems [22, 23].

In the ABC, three types of bees visit food sources and produce solutions by tracing new food sources. Each food source stands for a candidate solution, and the quality of a food source demonstrates the quality of the objective function of the related solution. The algorithm has a simple structure consisting of four steps: initialization, employed bees, onlooker bees, and scout bees [30]. In the first step, food sources (candidate solutions) are randomly generated in the environment and the population of artificial bee colony is assumed to be twice as much as food sources. Half of this population is employed bees and the other half is onlooker bees. Each of the employed bees is responsible for a food source and seeks out new food sources around it. If an employed bee abandons a food source, it becomes a scout bee and starts randomly searching for a new food source. The onlooker bees, on the other hand, visit food sources according to their quality, unlike the employed bees, and search in the vicinity of the visited food source. These three different bee searching activities continue in a loop until the algorithm ends. When the algorithm is terminated, the best food source found so far is considered a solution to the problem.

The implementation details of the algorithm steps are given below:

- Initialization step: N numbers of food sources are placed in the D-dimensional search space randomly as the following equation:

$$x_{i,j} = x_j^{min} + \zeta_i^j (x_j^{max} - x_j^{min}), \tag{4}$$

  where $x_{i,j}$ is the value of food source x_i ($i \in \{1, 2, 3, \ldots, N\}$) at dimension j ($j \in \{1, 2, 3, \ldots, D\}$), x_j^{min} and x_j^{max} are the lower and the upper bound values of dimension j, ζ_i^j is a uniform random number in $[0, 1]$, respectively. Furthermore, the *limit* parameter for food sources is initialized. This parameter refers to the trial limit for each food source. Another parameter, $trial_i$ which saves the current number of trials is initialized to zero for each food source x_i. If a food source is visited and a new good solution (a better food source) is not found around it, the trial value ($trial_i$) is increased. In the scout bees step, employed bee abandons the food source when the number of trials of the food source reaches the trial limit (*limit*).

- Employed bees step: At this stage, each employed bee i searches around for a food source x_i that it is responsible for. When performing the search, it utilizes the position of another randomly selected food source, X_r. Every time only one dimension, j, is selected randomly and new food source, V_i, is generated based on the following search equation:

$$v_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{r,j}), \tag{5}$$

where $\phi_{i,j}$ is a random number generated from a uniform distribution in $[-1,1]$. If the quality of v_i is better than x_i, then x_i is replaced by v_i and the $trial_i$ value is reset. Otherwise, $trial_i$ value is increased.

- Onlooker bees step: At this stage, onlooker bees do not visit the food sources they are responsible for, unlike the employed bees. Instead, they decide on the food sources they will visit according to the attractiveness of each food source. The quality of each food source increases the likelihood of being attracted and therefore being selected. For the minimization problems, p_i (the selection probability of each food source x_i) is calculated as:

$$p_i = \frac{\frac{1}{1+f(x_i)}}{\sum_i^N \frac{1}{1+f(x_n)}}, \tag{6}$$

where $f(x_i)$ is the objective function of the food source x_i. Then onlooker bees select their food sources according to selection probabilities and they search around the selected food source the same as employed bees do.

- Scout bees step: When a food source x_i is abandoned (when $trail_i$ is equal to $limit$), the responsible employed bee becomes a scout bee and finds a new food source in search space according to the Equation 4. Then, the new food source is replaced with the abandoned food source.

4. Self-adaptive search-equation-based artificial bee colony algorithm

The SSEABC algorithm [28] introduces three strategies to the basic ABC algorithm to improve performance quality. The first strategy is a self-adaptive search equation determination, the second is incrementing the size of population during the execution, and the third one is the competitive local search selection strategy. The pseudo-code of the proposed SSEABC algorithm is presented in [28] and the supplementary document (http://194.27.229.73/sirlab/wp-content/uploads/2019/05/supplementary.pdf). Also, the flowchart of the SSEABC is given in Figure 2. In the following subsections, we give a detailed description of these three performance improvement strategies.

4.1. The self-adaptive search equation determination strategy

Employed and onlooker bees use a randomly selected food source as reference when discovering new better food sources. This leads to an improvement in the diversification behavior of the algorithm, while weakening the intensification behavior. Therefore, many ABC variants in the literature have proposed several search equations in order to establish a good balance between intensification and diversification behaviors of the algorithm. Because some types of problems require intensification, some others can only be solved by diversification. However, it is not possible to predict which one should be preferred because the surface of the problem search space is not known in advance. Therefore, there is no single search equation that can give good results for all kinds of problems, and a suitable search equation should be proposed for each problem instance. However, in the canonical ABC algorithm and several ABC variants, a single search equation that is not changed throughout the execution is used. As a result, to overcome this issue, we proposed the "self-adaptive search equation determination strategy". Instead of using one unchanged search equation, a pool of equations filled with randomly generated search equations is used in this study. Search equations in the pool are filled according to a search equation template which is called "generalized search equation" and shown in Algorithm 1.

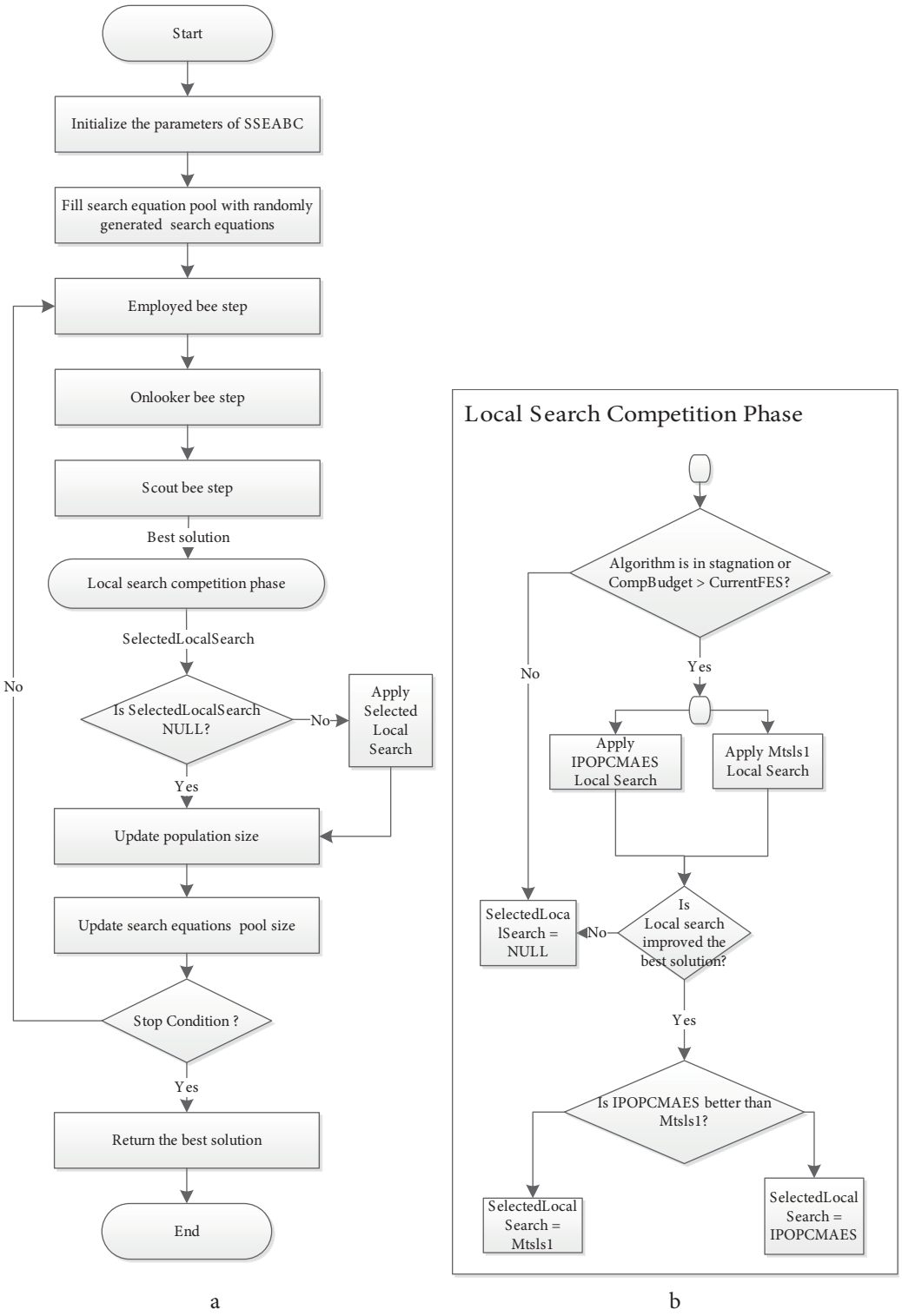


Figure 2. a) The flowchart of the SSEABC algorithm, b) the local search competition phase.

Algorithm 1 The proposed generalized search equation

1: **for** $t = 1$ to m **do**
2: randomly select dimension j
3: $x_{i,j} = term_1 + term_2 + term_3 + term_4$

According to this template, each search equation may have four terms and m parameter. Each of these terms is randomly selected from the components listed in Table 1, independently of each other. In Table 1, x_i is the selected reference solution, x_G is the best solution found so far, x_{r1} and x_{r2} $(r1 \neq r2 \neq i)$ are two randomly selected solutions, x_{GD} is the global-best distance solution [31],x_{WO} is the worst solution in population, x_{MD} is the median solution in population, x_{SC} is the second best solution in population, and finally $x_{AVE,j}$ is the average value of all solutions in population at dimension j. In each iteration of the algorithm, a search equation is taken sequentially from the pool in order to be applied in employed bees and onlooker bees steps. At the end of each iteration, the number of food sources improved using the search equation is calculated and it is recorded as the success ratio of the selected search equation. After all the search equations in the pool are used, the search equations in this pool are sorted in descending order of success ratio. The pool size is then reduced by using Equation 7 to eliminate inappropriate search equations for tackling problem instance.

$$ps = \frac{ps^2}{itr_{MAX}}. \tag{7}$$

Here itr_{MAX} is the approximate maximum iteration number calculated by the equation 8

$$itr_{MAX} = \frac{MAXFES}{2 \times SN} \tag{8}$$

4.2. The competitive local search selection strategy

When ABC algorithms are hybridized with the appropriate local search algorithm, performance of any ABC algorithm can be increased. However, there is no efficient local search procedure for each type of problem. In this article, a competition-based selection procedure is proposed which can find the appropriate local search algorithm for an IIR filter problem.

It consists of two successive steps in the name of competition and deployment. In the competition step (given as "Local Search Competition Phase" in Figure 1), the SSEABC algorithm executed with the fix budget of function evaluations (called as $CompBudget$). Then, first local search of multiple trajectory search (Mtsls1) [32] and evolution strategy with covariance matrix adaptation (IPOPCMAES) algorithms are run as local search procedures for the same amount of budget, $CompBudget$. The solution value obtained by the algorithm and the results of the local search procedures are compared. If one of the local searches is improved x_G then the better local search is selected as local search procedure for the deployment step. In the deployment step; if any local search procedure is selected, then x_G is used as the initial solution the selected search procedure is called from. The final solution found through local search becomes the calling best so far solution if it is better than the initial solution. In the SSEABC, the local search procedure is not called at every iteration for every food source. This is done in an effort to save as many function evaluations as possible. The local search procedure is called only when it is expected that its invocation will result in an improvement of the best so far solution. However, since we are dealing with black-box optimization, it is not possible to be completely certain that a

Table 1. Alternative options for each component in the generalized search equation of Algorithm 1; x_i: the selected reference solution, x_G: the best solution found so far, x_{r1},x_{r2} $(r1 \neq r2 \neq i)$: two randomly selected solutions, x_{GD}: the global-best distance solution, x_{WO}: the worst solution in population, x_{MD}: the median solution in population, x_{SC}: the second best solution in population, $x_{AVE,j}$: the average value of all solutions in population at dimension j.

m	$term1$	$term2$	$term3$	$term4$
1	$x_{i,j}$	$\phi1(x_{i,j} - x_{G,j})$	$\phi2(x_{i,j} - x_{G,j})$	$\phi3(x_{i,j} - x_{G,j})$
k $(1 \leq k \leq D)$	$x_{G,j}$	$\phi1(x_{i,j} - x_{r1,j})$	$\phi2(x_{i,j} - x_{r1,j})$	$\phi3(x_{i,j} - x_{r1,j})$
$[t,k]$ $(1 \leq t < k \leq D)$	$x_{r1,j}$	$\phi1(x_{G,j} - x_{r1,j})$	$\phi2(x_{G,j} - x_{r1,j})$	$\phi3(x_{G,j} - x_{r1,j})$
		$\phi1(x_{r1,j} - x_{r2,j})$	$\phi2(x_{r1,j} - x_{r2,j})$	$\phi3(x_{r1,j} - x_{r2,j})$
		$\phi1(x_{i,j} - x_{GD,j})$	$\phi2(x_{i,j} - x_{GD,j})$	$\phi3(x_{i,j} - x_{GD,j})$
		$\phi1(x_{i,j} - x_{SC,j})$	$\phi2(x_{i,j} - x_{SC,j})$	$\phi3(x_{i,j} - x_{SC,j})$
		$\phi1(x_{i,j} - x_{MD,j})$	$\phi2(x_{i,j} - x_{MD,j})$	$\phi3(x_{i,j} - x_{MD,j})$
		$\phi1(x_{i,j} - x_{WO,j})$	$\phi2(x_{i,j} - x_{WO,j})$	$\phi3(x_{i,j} - x_{WO,j})$
		$\phi1(x_{SC,j} - x_{MD,j})$	$\phi2(x_{SC,j} - x_{MD,j})$	$\phi3(x_{SC,j} - x_{MD,j})$
		$\phi1(x_{MD,j} - x_{WO,j})$	$\phi2(x_{MD,j} - x_{WO,j})$	$\phi3(x_{MD,j} - x_{WO,j})$
		$\phi1(x_{G,j} - x_{WO,j})$	$\phi2(x_{G,j} - x_{WO,j})$	$\phi3(x_{G,j} - x_{WO,j})$
		$\phi1(x_{r1,j} - x_{MD,j})$	$\phi2(x_{r1,j} - x_{MD,j})$	$\phi3(x_{r1,j} - x_{MD,j})$
		$\phi1(x_{G,j} - x_{MD,j})$	$\phi2(x_{G,j} - x_{MD,j})$	$\phi3(x_{G,j} - x_{MD,j})$
		$\phi1(x_{r1,j} - x_{WO,j})$	$\phi2(x_{r1,j} - x_{WO,j})$	$\phi3(x_{r1,j} - x_{WO,j})$
		$\phi1(x_{SC,j} - x_{r1,j})$	$\phi2(x_{SC,j} - x_{r1,j})$	$\phi3(x_{SC,j} - x_{r1,j})$
		$\phi1(x_{i,j} - x_{AVE,j})$	$\phi2(x_{i,j} - x_{AVE,j})$	$\phi3(x_{i,j} - x_{AVE,j})$
		$\phi1(x_{r1,j} - x_{AVE,j})$	$\phi2(x_{r1,j} - x_{AVE,j})$	$\phi3(x_{r1,j} - x_{AVE,j})$
		$\phi1(x_{G,j} - x_{AVE,j})$	$\phi2(x_{G,j} - x_{AVE,j})$	$\phi3(x_{G,j} - x_{AVE,j})$
		do not use	do not use	do not use

solution is already in a local optimum so it is impossible to improve it with a local search. We use, therefore, a heuristic approach to decide whether to call the local search procedure from the best so far solution or not. The approach performs the local search procedure to obtain a value to identify its exit condition. If the procedure is improved to initial solution, the local search procedure is called again at the following iteration of the algorithm. Otherwise, the SSEABC turns back to the competition step to identify whether another local search procedure is needed or not.

4.3. The incremental population size strategy

The performance of population-based algorithms is influenced by the population size. Therefore, determining the most appropriate size of the population is important for the performance of the algorithm. De Oca et al. [33] expresses that individuals learn faster when the algorithm has a small population size in the limited function evaluations. It is also seen that when the population size becomes larger, the algorithm obtains better-quality results. It is desirable to keep these two situations in balance. For this, the strategy of "Incremental social framework (ISL)" [33] has been proposed. According to the ISL, the population starts working with a small number of individuals. After the algorithm has been running for a certain period gp, a new solution is added to the population. While this new individual is being produced, it is aimed to benefit from the experienced individuals who are included in the population. Thus, better learning is achieved.

In this study, incremental population strategy is used as follows. Firstly, the algorithm starts with few solutions. Then, for every gp iteration of a given growth period, the new solution is added to the population. This insertion process progresses until the population reaches its maximum size, which is defined at the beginning of the algorithm. The new solution position ($x_{j,new}$) inserted into the population is initialized using Equation 9 which uses the best solution found so far(X_{gbest}):

$$\acute{x}_{new,j} = x_{new,j} + \varphi_{i,j}(x_{gbest,j} - x_{new,j}). \tag{9}$$

Here x_{new} is the new solution created by the ABC algorithm's random solution-generating equation. However, $x_{j,new}$, produced using x_{gbest} with x_{new}, represents the new solution desired to be included in the population. Taking advantage of the position of expert solutions when producing a new solution helps the algorithm to move towards better solutions.

5. Results

In order to assess the performance of the SSEABC on adaptive IIR filter design, three benchmark systems extensively used in many studies are selected [13, 17–20, 34]. In the first two examples, the unknown system and filter model are of the same order. In the following examples, two cases have been considered; identification is utilized with the same-order and reduced-order IIR filter models. Both simulation and application were carried out for experimental studies.

5.1. Simulation results

Simulation studies are done on a computer with C++ and i7 8 GB RAM hardware. For SSEABC, ABC [21], MABC [35], NABC [36], and APABC [37], the results are obtained with 100 independent runs with 7500 function evaluations (FEs) for each sample. A Gaussian white noise signal with 100 samples is applied as input signal to both the unknown system and the IIR filter for each algorithm run.

Although the problem-specific search equations are determined in a self-adaptive search equation determination strategy in the SSEABC algorithm, there are other parameters of the SSEABC algorithm, which affect the performance significantly as well. In this study, the parameter values of the SSEABC algorithm are determined by irace [38], the offline parameter configuration tool. irace tool is the iterated version of F-race procedure [39] which is based on racing and Friedman's nonparametric two-way analysis of variance by ranks. irace has also some parameters; however, we have run irace with default parameter values defined in the literature. Moreover, it is important to note that problem instances used in the parameter tuning task with irace should differ from those used in the experiments. For this, we have used some other examples of IIR system identification problem, and synthetic problem instances created by us. We have used irace for the parameter configuration of SSEABC and other ABC-based algorithms used in comparison. The best values of parameters for algorithms are determined over five independent runs of the irace tool. The obtained parameter values of the basic ABC, MABC, NABC, APABC, and SSEABC are given in Table 2.

In the following subsections, the comparison results of the SSEABC, ABC, MABC, NABC, and APABC algorithms are shown and criticized for each case. In addition, the comparisons with other algorithms in the literature are summarized. The results of the compared algorithms except ABC variants were taken directly from the reference articles. In order to make a fair comparison, the algorithms were tested in the same or better experimental conditions (such as research which uses the same number of or more function evaluations in the experimental study) are included in the comparisons. All the comparisons are performed over the MSE and MSE in dB defined in Equation 3.

Table 2. Tuned parameters and their values for considered ABC algorithms.

Parameters	ABC	MABC	NABC	APABC	SSEABC
Initial population size (SN)	14	8	73	72	38
The limit factor (limitF)	2.0456	2637	0.7791	0.5764	1.9074
Modification Rate (MR)	-	0.8512	-	-	-
adaptiveSF	-	1	-	-	-
Scaling Factor (SF)	-	0.7277	-	-	-
Number of neighbors	-	-	13	-	-
Maximum population size (SN_max)	-	-	-	50	55
Growth period of pop. size (g)	-	-	-	22	5
a parameter of IPOP-CMA-ES (a)	-	-	-	-	1.1127
a parameter of IPOP-CMA-ES (b)	-	-	-	-	2.4823
a parameter of IPOP-CMA-ES (c)	-	-	-	-	0.5671
a parameter of IPOP-CMA-ES (d)	-	-	-	-	3.3964
a parameter of IPOP-CMA-ES (e)	-	-	-	-	-17.8882
a parameter of IPOP-CMA-ES (f)	-	-	-	-	-17.9765
a parameter of IPOP-CMA-ES (g)	-	-	-	-	-19.2638
Mtsls1 iterations (MTSLS_itr)	-	-	-	-	22
FES for IPOP-CMA-ES (IPOP-CMA-ES_FES)	-	-	-	-	0.3
Maximum FES budget for the competition phase (CompBudget)	-	-	-	-	0.15
The search equations pool size (ps)	-	-	-	-	2000

5.1.1. Example I

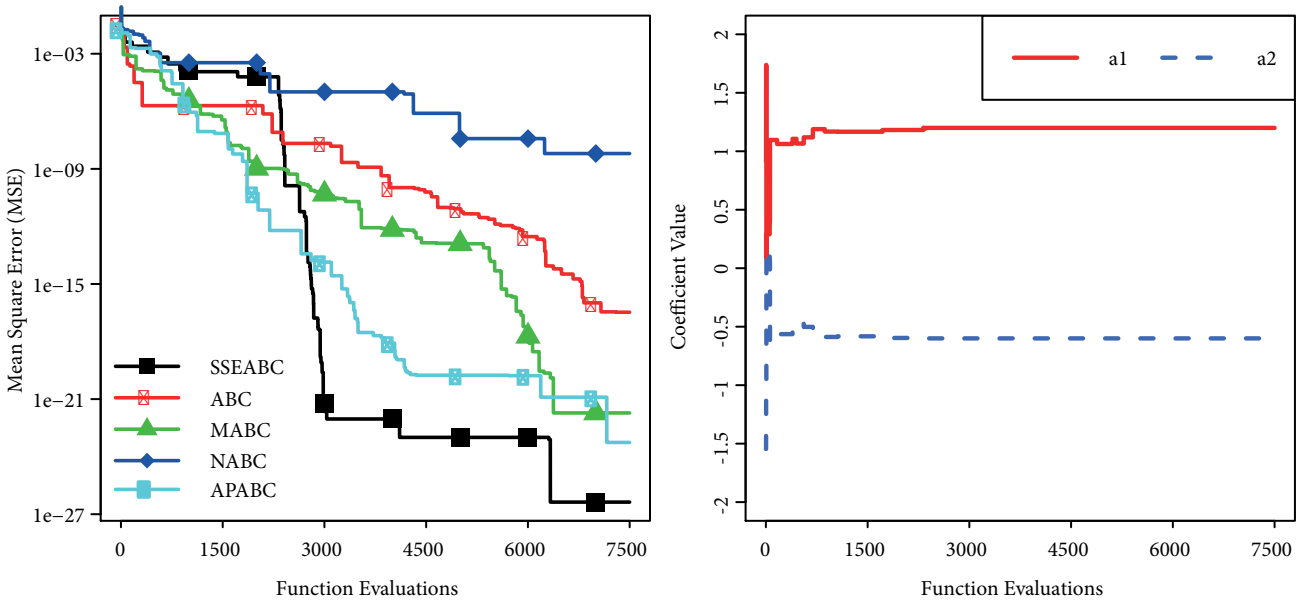
Transfer functions of the unknown system and the filter model taken from [20], which is also used in different studies [13, 17, 19, 34], are given in Equations 10 and 11, respectively:

$$H_{system}(Z) = \frac{1}{1 - 1.2z^{-1} + 0.6z^{-2}} \tag{10}$$

$$H_{filter}(Z) = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2}}. \tag{11}$$

The global optimum of this problem is at $a_1 = 1.2$ and $a_2 = -0.6$. The convergence curves of the best solutions obtained by SSEABC and other ABC variants are shown in Figure 3a. The variation of the coefficient values along the evolutionary process of SSEABC algorithm is shown in Figure 3b. In Table 3, best, average results, and standard deviation (Std) of the results obtained by ABC algorithms over 100 runs and other techniques in literature are presented. The other algorithms used in comparison are Harmony Search (HS) [40], Genetic Algorithm (GA) [40], Real coded GA (RGA) [40], DE [40], PSO [40], and craziness-based PSO (CRPSO) [40].

As seen in Figure 3, the SSEABC algorithm converges to the global optimum more quickly than other ABC variants. While the SSEABC algorithm reaches $4.277E-27$ MSE approximately after 6300 FEs, canonical



(a) Convergence graph

(b) Variations of coefficient values

Figure 3. (a) Convergence characteristic of SSEABC, MABC, ABC, NABC, and APABC and (b) Coefficient changes over time while using SSEABC on for Example I.

Table 3. Statistical results of MSE for Example I.

Type		Algorithm		MSE		MSE(dB)	
	Name	Year	Best	Mean	Std	Best	Mean
	SSEABC		4.2770E-27	3.3485E-24	4.5905E-24	−263.689	−234.752
ABC variants	ABC	2007	3.3757E−17	8.0739E-08	2.9699E-07	−164.716	−70.929
	MABC	2012	1.8986E-22	2.0717E-12	9.5637E-12	−217.216	−116.837
	NABC	2018	6.2913E-09	9.9649E-07	1.2536E-06	−82.013	−60.015
	APABC	2017	5.5153E-24	4.1098E-21	2.9462E-21	−232.584	−203.862
Other techniques	HS [40]	2014	1.5626E-07	NR	NR	−168.062	−160.782
	GA [41]	2011	NR	8.8600E-01	NR	NR	−0.526
	RGA [19]	2014	3.1700E-02	5.1986E-02	1.5231E+00	−14.989	−12.841
	PSO [19]	2014	2.3000E-03	2.9846E-03	1.2528E+00	−26.383	−25.251
	DE [19]	2014	3.3328E-05	6.1105E-05	1.4234E+00	−44.772	−42.139
	CRPSO [19]	2014	1.4876E-20	2.9275E-20	1.4086E+00	−198.275	−195.335

ABC and MABC algorithms are trapped at $3.3757E-17$ MSE and $1.8986E-22$ MSE after 7000 and 6300 FEs, respectively. However, the NABC is the worst ABC algorithm among the ABCs with $6.2913e-09$ MSE, while the APABC is the second best algorithm following the SSEABC with a value of $5.5153E-24$ MSE. In Figure 5, it has been clearly seen that estimated parameter values obtained by the SSEABC are actual values reached in early iterations. When we compare the results in Table 3, it is seen that the algorithm having the smallest MSE value among the compared algorithms is SSEABC.

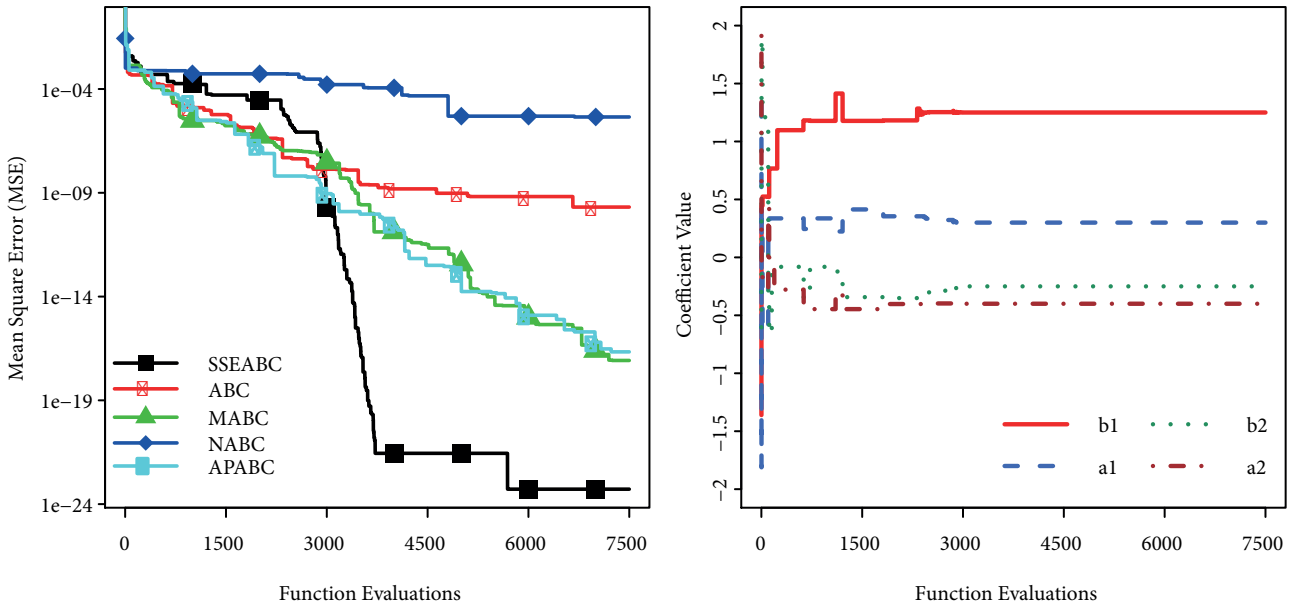
5.1.2. Example II

In this example, an unknown system in second-order filter is identified by a second-order IIR filter [13, 15]. The transfer functions of the unknown system and IIR filter are given in Equations 12 and 13, respectively.

$$H_{system}(z) = \frac{1.25z^{-1} - 0.25z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2}}, \tag{12}$$

$$H_{filter}(z) = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}}. \tag{13}$$

The convergence curves of the best individual obtained by ABC algorithms and the variation of the coefficient values along the evolutionary process of the SSEABC are shown in Figure 4a and 4b, respectively. Comparison of the SSEABC with ABC variants and other algorithms in the literature is presented in Table 4. The algorithms that we use for comparison are again GA, DE, and PSO algorithms. However, contrary to the reference in the first example, the reference results are taken from the most recent implementations of these algorithms [42]. In addition to these algorithms, we used HS [40], RGA [40], DE with hybrid mutation operator with self-adapting control parameters (HSDE) [42], opposition-based hybrid coral reefs optimization algorithm (OHCRO) [42], multistrategy immune cooperative evolutionary PSO (ICPSO-MS) [42], PSO with quantum infusion (PSO-QI) [14], and differential evolution PSO (DEPSO) [14] algorithms in the comparisons.



(a) Convergence graph

(b) Variations of coefficient values

Figure 4. (a) Convergence characteristic of SSEABC, MABC, ABC, NABC, and APABC and (b) Coefficient changes over time while using SSEABC on for Example II.

As seen in Figure 6, while the SSEABC algorithm reaches $5.2536E-24$ MSE approximately after 5700 FEs, canonical ABC, MABC, and APABC algorithms are trapped at $2.0501E-10$ MSE, $8.3578E-18$ MSE, and $2.1696E-17$ after 7000 and 7400 FEs, respectively. In addition, NABC obtains $4.4979E-06$ MSE after 4500 FEs. In Figure 6, estimated parameter values obtained by the SSEABC are the actual values reached at early iterations. It obviously indicates that convergence rate of the proposed SSEABC algorithm is higher

Table 4. Statistical results of MSE for Example II.

Type		Algorithm		MSE		MSE(dB)	
	Name	Year	Best	Mean	Std	Best	Mean
	SSEABC		5.2536E-24	4.7700E-22	4.6105E-22	–232.795	–213.215
ABC variants	ABC	2007	2.0501E-10	1.4552E-06	2.3507E-06	–96.882	–58.371
	MABC	2012	8.3578E-18	9.4358E-11	8.8071E-10	–170.779	–100.252
	NABC	2018	4.4979E-06	3.0485E-05	1.7774E-05	–53.470	–45.159
	APABC	2017	2.1696E-17	3.1451E-15	6.1357E-15	–166.636	–145.024
Other techniques	HS [40]	2014	7.3562E-10	2.3180E-09	2.0855E+00	–91.333	–86.349
	RGA [40]	2014	4.5000E-02	6.4653E-02	1.2821E+00	–13.468	–11.894
	GA [42]	2017	3.7500E-08	2.3800E-05	2.5800E-05	–74.260	–46.234
	PSO [42]	2017	1.4600E-10	5.6400E-05	1.6600E-04	–98.356	–42.487
	ICPSO-MS [42]	2017	9.2000E-17	1.9000E-14	2.7500E-14	–160.362	–137.212
	DE [42]	2017	5.2300E-11	1.1200E-10	2.8700E-11	–102.815	–99.508
	HSDE [42]	2017	1.2000E-17	2.5300E-16	3.2300E-16	–169.208	–155.969
	OHCRO [42]	2017	6.5900E-19	1.5400E-15	6.1600E-15	–181.811	–148.125
	PSO–QI [14]	2010	7.1020E-04	7.1020E-04	1.1480E-07	–31.486	–31.486
	DEPSO [14]	2010	7.1020E-04	7.2780E-04	4.3910E-05	–31.486	–31.380

than those of the competitor algorithms. When we compare the results in Table 4, it is seen that the SSEABC outperforms all the algorithms compared over best and average results.

5.1.3. Example III

In this example, the transfer function of a fifth-order system given in Equation 14 is identified by a same-order IIR filter (Case 1) and a reduced-order IIR filter (Case 2) [17–19].

$$H_{system}(z) = \frac{0.1084 + 0.5419z^{-1} + 1.0837z^{-2} + 1.0837z^{-3} + 0.5419z^{-4} + 0.1084z^{-5}}{1 + 0.9853z^{-1} + 0.9738z^{-2} + 0.3864z^{-3} + 0.1112z^{-4} + 0.0133z^{-5}}. \tag{14}$$

Case I In this case, the fifth-order unknown system is modeled by using a fifth-order IIR filter. The transfer function of the filter is defined as follows:

$$H_{filter}(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2} + b_3z^{-3} + b_4z^{-4} + b_5z^{-5}}{1 - a_1z^{-1} - a_2z^{-2} - a_3z^{-3} - a_4z^{-4} - a_5z^{-5}} \tag{15}$$

The convergence behaviors of different ABC algorithms are presented in Figure 5. Evolution of coefficient values of the IIR filter during the execution of SSEABC are presented in Figure 6. Moreover, the comparison results with ABC variants and other algorithms are relisted in Table 5.

As seen in Figure 7, while ABC and MABC algorithms cannot obtain better MSE than $1E-5$ for 7500 FEs, SSEABC reaches $9.9531E-09$ MSE. In addition, APABC and NABC are the worst algorithms in terms of MSE. This shows the better convergence behavior of the algorithm as it is expected. In this example, unlike the Examples 1 and 2, the number of coefficients to be estimated is large, which causes the MSE value to increase. Therefore, the MSE value between the system and the filter model are expected to be larger. Table 5

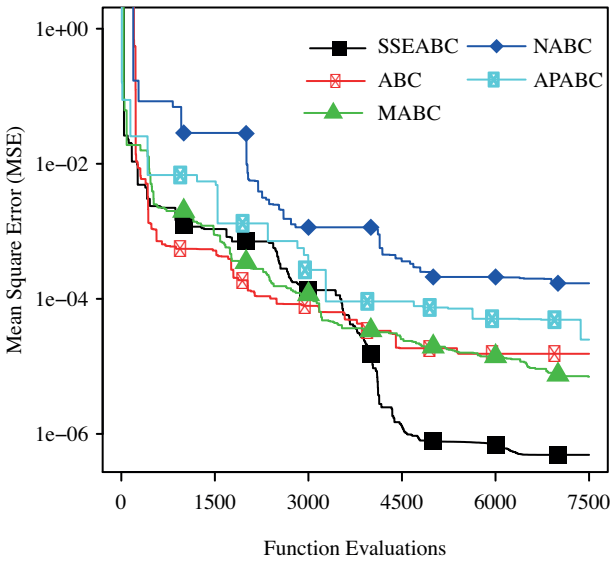


Figure 5. Convergence characteristic of SSEABC, MABC, ABC, NABC, and APABC on Case I of Example III.

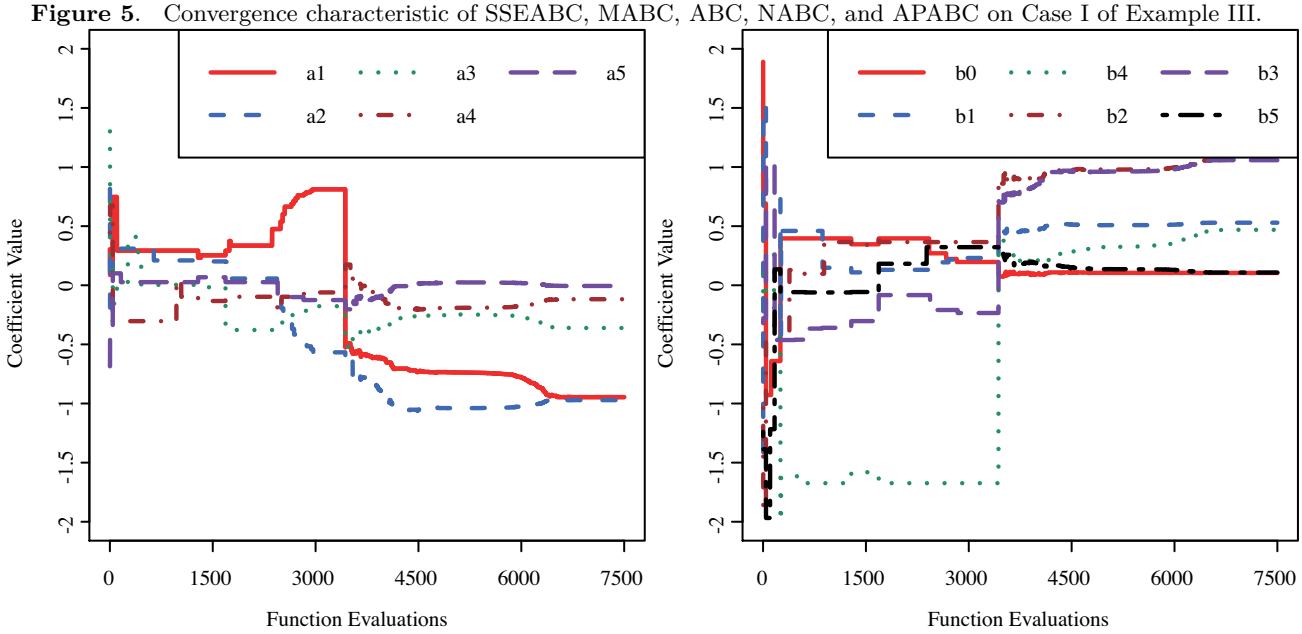


Figure 6. Coefficient "a" (left) and "b" (right) values changes over time while using SSEABC on Case I of Example III.

also clearly shows this. Despite the difficulty of the problem, it is seen that the SSEABC algorithm is able to model the IIR filter much better than the other algorithms. As a result, SSEABC produces much smaller MSE values which can be clearly seen in Table 5.

Case II The transfer function of the reduced-order IIR filter is as follows:

$$H_{filter}(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4}}{1 - a_1 z^{-1} - a_2 z^{-2} - a_3 z^{-3} - a_4 z^{-4}}. \tag{16}$$

As in the second case of the previous example, the optimal identification is also a more difficult task than

Table 5. Statistical results of MSE for Case I of Example III

Type	Algorithm		MSE			MSE(dB)	
	Name	Year	Best	Mean	Std	Best	Mean
	SSEABC		9.9531E-09	2.5157E-08	7.6117E-08	–80.020	–75.993
ABC variants	ABC	2007	4.6349E-05	2.2581E-04	1.5281E-04	–43.340	–36.463
	MABC	2012	1.5567E-05	1.2420E-04	2.1923E-04	–48.078	–39.059
	NABC	2018	1.7013E-04	9.6887E-04	5.9861E-04	–37.692	–30.137
	APABC	2017	2.4907E-05	8.6520E-05	3.9231E-05	–46.037	–40.629
Other techniques	RGA [40]	2014	3.0700E-02	4.9768E-02	1.5039E+00	–15.129	–13.031
	PSO [40]	2014	3.5000E-03	1.0375E-02	2.0593E+00	–24.559	–19.840
	DE [40]	2014	6.8819E-04	1.2694E-03	1.4962E+00	–31.623	–28.964
	HS [40]	2014	7.1407E-06	1.9247E-05	1.9145E+00	–51.463	–47.156

the one with same-order. The convergence curves of the ABC-based algorithms and coefficient changes at the run-time of the best solution of SSEABC are given in Figures 7 and 8, respectively. In addition, the best, the average MSE and the dB values of the algorithms in the literature are listed in Table 6.

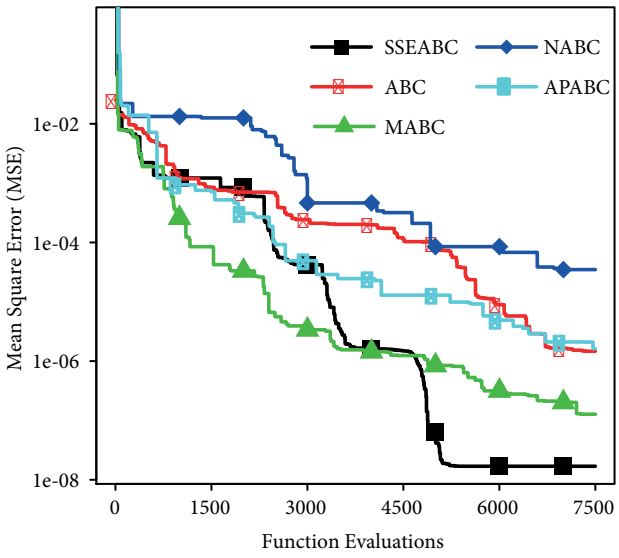


Figure 7. Convergence characteristic of SSEABC, MABC, ABC, NABC, and APABC on Case II of Example III.

When the results listed in Table 6 are examined, it is seen that SSEABC ranks first with $1.6902E-08$ MSE. MABC is the second best, and ABC is the third best performer. In addition, as in Case I, APABC and NABC are again the algorithm with the worst MSE value. As shown in Figure 9 and Table 6, SSEABC is observed to converge better than other ABC algorithms and to outperform compared algorithms in the literature in terms of MSE.

5.2. Experimental results

This section presents the results of the practical applications of IIR system identification problems. An experimental platform was constructed on ARM Cortex-M4 microcontroller (μC) named as STM32F407VGT6

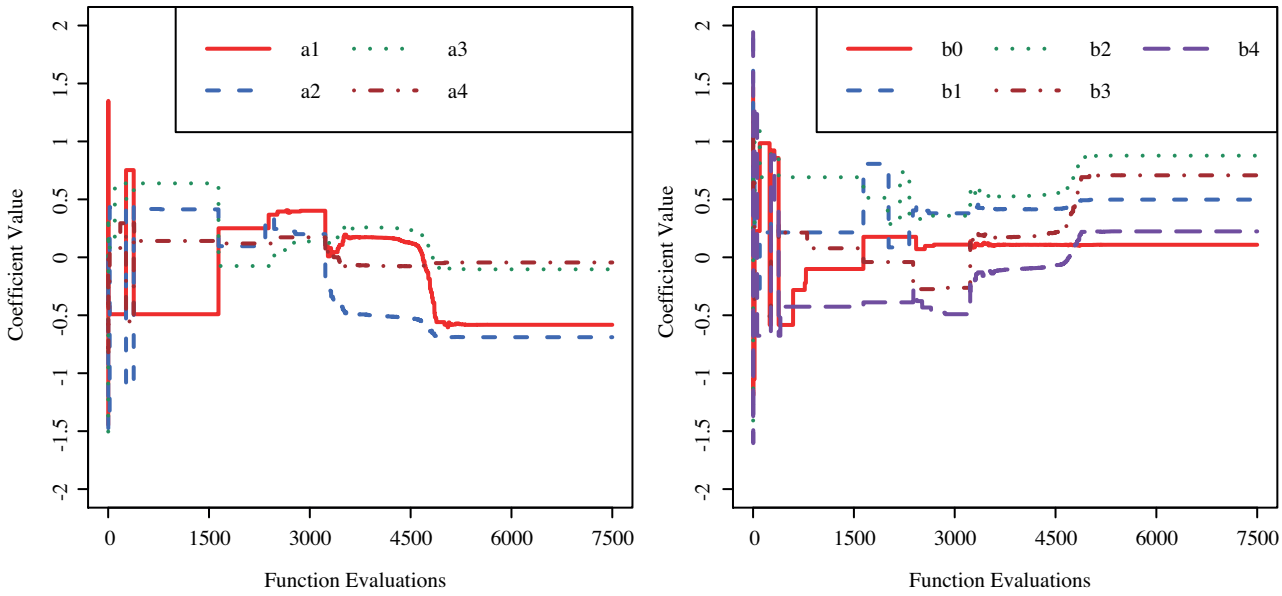


Figure 8. Coefficient "a" (left) and "b" (right) values changes over time while using SSEABC on Case II of Example III.

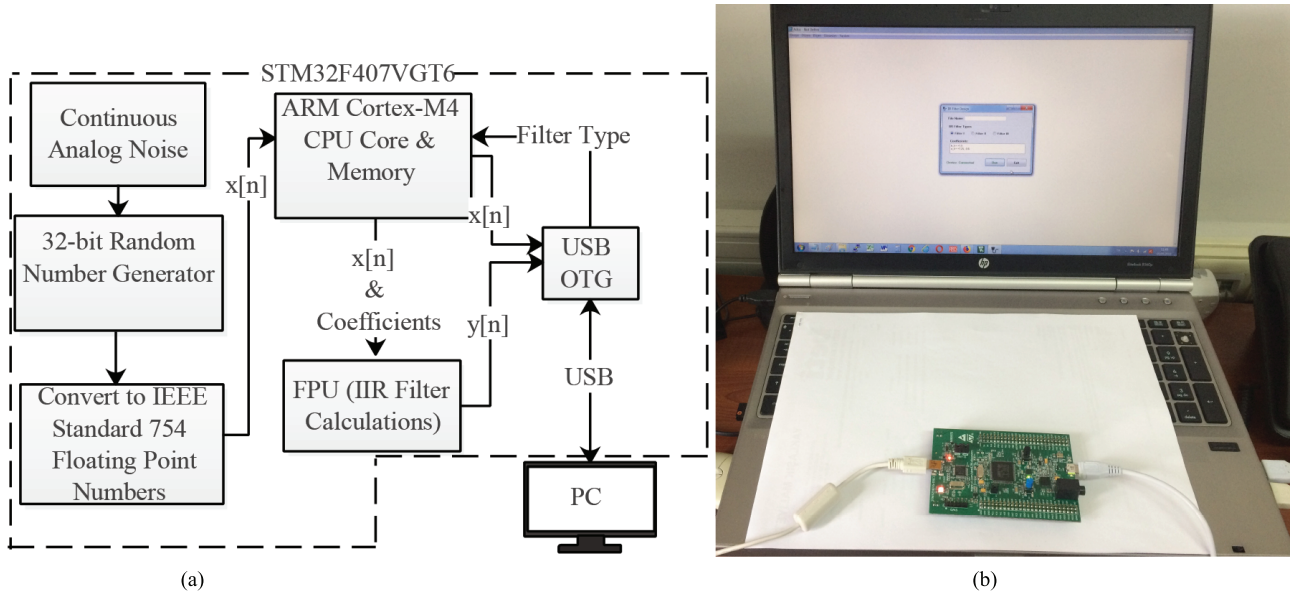


Figure 9. a) The general block diagram of implementation system b) The photograph of the implementation system.

on the STM32F4Discovery board [43]. The general block diagram and the photograph of the system are shown in Figures 9a and 9b, respectively.

To easily set μC configurations, a graphical software configuration tool, STM32CubeMX was used. The embedded program which runs on the μC was written in Keil IDE by using C programming language. The computer software whose GUI is shown in Figure 10 was developed to select the filter type. Using this software, the user could select the filter type and receive the input and output parameters of the specific filter design.

After receiving the filter type information and corresponding coefficient values, the μC created noise

Table 6. Statistical results of MSE for Case II of Example III

Type	Algorithm		MSE			MSE(dB)	
	Name	Year	Best	Mean	Std	Best	Mean
	SSEABC		1.6902E-08	1.2626E-07	6.0743E-07	–77.721	–68.988
ABC variants	ABC	2007	1.4686E-06	3.0501E-05	3.2520E-05	–58.331	–45.157
	MABC	2012	1.2746E-07	3.3653E-05	1.7205E-04	–68.946	–44.730
	NABC	2018	3.4885E-05	2.7122E-04	1.7542E-04	–44.574	–35.667
	APABC	2017	1.6229E-06	6.0430E-06	3.4857E-06	–57.897	–52.187
Other techniques	RGA [40]	2014	1.0870E-01	1.5875E-01	1.3638E+00	–9.638	–7.993
	PSO [40]	2014	1.2700E-02	2.7924E-02	1.9161E+00	–18.962	–15.540
	DE [40]	2014	2.7000E-03	3.1404E-03	1.1698E+00	–25.686	–25.030
	HS [40]	2014	6.1214E-06	6.9624E-06	1.1408E+00	–52.132	–51.572

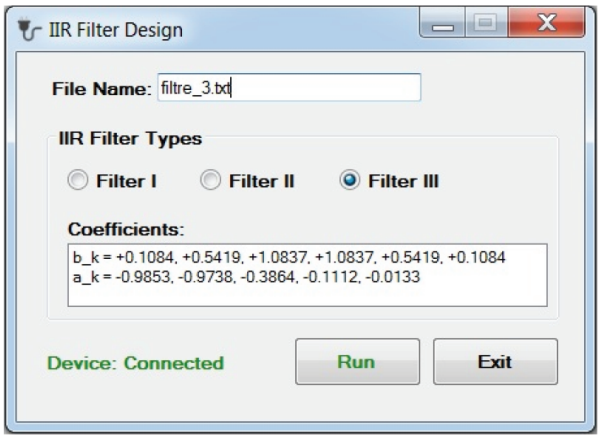


Figure 10. The graphical user interface (GUI) for filter design.

signal for the inputs of the IIR filter system. To obtain the noise signal whose level is between –1.0 and +1.0, the random number generator, which is based on a continuous analog noise and provides random 32-bit values to the CPU core, was used. These 32-bit integer numbers were converted to the IEEE standard 754 floating point number, and they were saved on μC memory. Then, the necessary filter calculations were performed by using the floating point unit (FPU) of the μC. All the inputs and outputs of the filter system were transferred to the PC. Thus, coefficients of IIR filter model based on SSEABC are optimized according to the inputs and outputs of the IIR filter system.

Table 11 shows the results obtained from the μC-based IIR filter implementation. As the experimental results of the example problems were not found in the literature, only ABC variants were compared.

As seen from Table 11, SSEABC has reached the least MSE value in all cases. Figure **??** shows the outputs of the IIR filter system based on the μC and IIR filter model using SSEABC. According to Figure **??**, the IIR filter model is closely tracking the outputs of the actual IIR filter.

6. Conclusion

In IIR-based system identification problems, metaheuristic-based design models may be trapped at the local minima due to the multimodal error surface of the IIR filters. For this reason, the search behavior of the selected

Table 7. Experimental results of IIR filter implementation.

		Best	Mean	Std
Example I	SSEABC	1.6142E-22	4.4109E-21	3.0273E-21
	ABC	1.7521E-12	8.2948E-07	2.7182E-06
	MABC	1.7306E-21	2.3817E-12	8.5526E-12
	NABC	3.5401E-07	1.6270E-05	2.0882E-05
	APABC	3.6180E-22	4.9650E-21	3.0433E-21
Example II	SSEABC	3.1534E-21	7.0601E-21	1.6441E-21
	ABC	1.1487E-06	1.4849E-04	2.0462E-04
	MABC	1.2150E-18	3.8058E-10	1.6394E-09
	NABC	6.8693E-05	1.0199E-03	8.1328E-04
	APABC	3.3350E-16	7.4285E-13	3.0355E-12
Example III Case I	SSEABC	8.4123E-07	8.4564E-07	1.2272E-08
	ABC	5.1406E-04	4.0407E-03	4.3779E-03
	MABC	3.4560E-04	1.3839E-02	3.9467E-02
	NABC	4.5676E-03	2.9423E-02	2.1697E-02
	APABC	2.6030E-04	2.0541E-03	1.2770E-03
Example III Case II	SSEABC	1.3278E-06	1.2750E-05	5.5958E-05
	ABC	4.0510E-04	2.7155E-03	2.4347E-03
	MABC	2.3454E-05	2.4707E-03	8.6036E-03
	NABC	4.3692E-03	8.9457E-03	3.6553E-03
	APABC	1.3631E-04	5.5561E-04	2.9774E-04

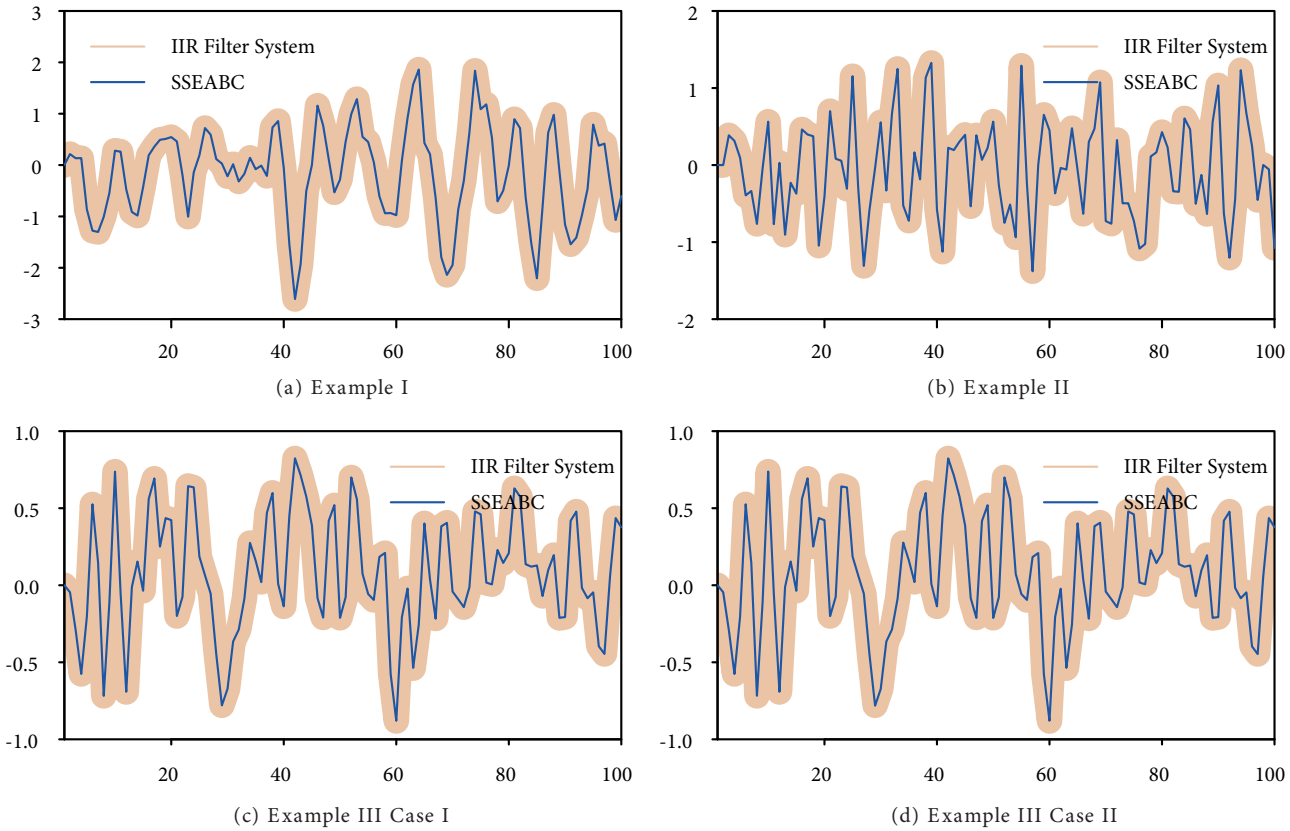


Figure 11. The outputs of the IIR filter system based on the C and IIR filter model using SSEABC.

metaheuristic algorithm must be very strong. In this study, the SSEABC algorithm, which has proven to have a powerful search capability, has been used to deal with the system identification problem. The performance of the proposed algorithm was evaluated over a filter design benchmark set which is widely used in the literature. The high convergence rate of SSEABC algorithm and MSE-based comparisons show that the proposed method is an effective tool for both same- and reduced-order system identification problems. It is also observed that ABC variants can provide more suitable solutions to solve system identification problems with reduced order models.

For the future work, we are planning to extend the work presented in this article in two ways. Firstly, we will try to improve the performance of the SSEABC algorithm with adding new self-adaptive strategies and components to generalized search equation, and adding new local search methods to competitive local search selection strategy. A second direction is to apply the SSEABC algorithm to solve more complex system identification problems.

References

[1] Mitra SK, Kuo Y. Digital Signal Processing: A Computer-based Approach; vol. 2. New York, NY, USA: McGraw-Hill, 2006.

[2] Lin J, Chen C. Parameter estimation of chaotic systems by an oppositional seeker optimization algorithm. Nonlinear Dynamics 2014; 76 (1): 509-517.

[3] Regalia P. Adaptive IIR Filtering in Signal Processing and Control; vol. 90. Boca Raton, FL, USA: CRC, 1994.

[4] Sarangi A, Kumar Sarangi S, Panigrahi SP. An approach to identification of unknown iir systems using crossover cat swarm optimization. Perspectives in Science 2016; 8: 301-303.

[5] Zou DX, Deb S, Wang GG. Solving IIR system identification by a variant of particle swarm optimization. Neural Computing and Applications 2018; 30(3): 685-698.

[6] Singh S, Ashok A, Kumar M, Rawat TK. Adaptive infinite impulse response system identification using teacher learner based optimization algorithm. Applied Intelligence 2018; 1-18.

[7] Sharifi MA, Mojallali H. A modified imperialist competitive algorithm for digital IIR filter design. Optik 2015; 126 (21): 2979-2984.

[8] Radenkovic M, Bose T. Adaptive IIR filtering of nonstationary signals. Signal Processing 2001; 81(1): 183-195.

[9] Shynk J J. Adaptive IIR filtering. IEEE Assp Magazine 1989; 6(2): 4-21.

[10] Shengkui Z, Zhihong M, Suiyang K. A fast variable step-size LMS algorithm with system identification. In: 2007 2nd IEEE Conference on Industrial Electronics and Applications. IEEE; 2007. pp. 2340-2345.

[11] Krusienski D, Jenkins W. Design and performance of adaptive systems based on structured stochastic optimization strategies. IEEE Circuits and Systems Magazine 2005; 5(1): 8-20.

[12] Karaboga N, Cetinkaya B. Design of digital fir filters using differential evolution algorithm. Circuits, Systems, and Signal Processing 2006; 25(5): 649-660.

[13] Karaboga N. A new design method based on artificial bee colony algorithm for digital IIR filters. Journal of the Franklin Institute 2009; 346(4): 328-348.

[14] Luitel B, Venayagamoorthy GK. Particle swarm optimization with quantum infusion for system identification. Engineering Applications of Artificial Intelligence 2010; 23 (5): 635-649.

[15] Dai C, Chen W, Zhu Y. Seeker optimization algorithm for digital IIR filter design. IEEE Transactions on Industrial Electronics 2010; 57 (5): 1710-1718.

[16] Kumar M, Aggarwal A, Rawat TK. Bat algorithm: application to adaptive infinite impulse response system identification. Arabian Journal for Science and Engineering 2016; 41(9): 3587-3604.

[17] Kumar M, Rawat TK, Aggarwal A. Adaptive infinite impulse response system identification using modified-interior search algorithm with lèvy flight. ISA Transactions 2017; 67: 266-279.

[18] Kumar M, Rawat TK. Optimal fractional delay-IIR filter design using cuckoo search algorithm. ISA Transactions 2015; 59: 39-54.

[19] Upadhyay P, Kar R, Mandal D, Ghoshal S. Craziness based particle swarm optimization algorithm for IIR system identification problem. AEU-International Journal of Electronics and Communications 2014; 68(5): 369-378.

[20] Karaboga N. Digital IIR filter design using differential evolution algorithm. EURASIP Journal on Applied Signal Processing 2005; 2005: 1269-1276.

[21] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. Journal of global optimization 2007; 39(3): 459-471.

[22] Karaboga D, Gorkemli B, Ozturk C, Karaboga N. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review 2014; 42(1): 21-57.

[23] Afsar B, Aydın D. Comparison of artificial bee colony algorithms on engineering problems. Application of Mathematics 2016; 10(2): 495-505.

[24] Erçin Ö, Çoban R. Identification of linear dynamic systems using the artificial bee colony algorithm. Turkish Journal of Electrical Engineering & Computer Sciences 2012; 20(Sup. 1):1175-1188.

[25] Ata B, Çoban R. Artificial bee colony algorithm based linear quadratic optimal controller design for a nonlinear inverted pendulum. International Journal of Intelligent Systems and Applications in Engineering 2015; 3(1): 1-6.

[26] Çoban R, Erçin Z. Multi-objective bees algorithm to optimal tuning of pid controller. Çukurova Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi 2012; 27(2): 13-26.

[27] Agrawal N, Kumar A, Bajaj V. Optimized design of digital IIR filter using artificial bee colony algorithm. In: 2015 International Conference on Signal Processing, Computing and Control (ISPCC); Solan, India; 2015. pp. 316-321.

[28] Yavuz G, Aydın D, Stützle T. Self-adaptive search equation-based artificial bee colony algorithm on the CEC 2014 benchmark functions. In: 2016 IEEE Congress on Evolutionary Computation (CEC); Vancouver, Canada;2016. pp. 1173-1180.

[29] Aydın D, Yavuz G. Self-adaptive search equation-based artificial bee colony algorithm with CMA-ES on the noiseless BBOB testbed. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion; Berlin, Germany; 2017. pp. 1742-1749.

[30] Aydın D. Composite artificial bee colony algorithms: From component-based analysis to high-performing algorithms. Applied Soft Computing 2015; 32: 266-285.

[31] Gao W, Liu S. Improved artificial bee colony algorithm for global optimization. Information Processing Letters 2011; 111(17): 871-882.

[32] Tseng L Y, Chen C. Multiple trajectory search for large scale global optimization. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence); Hong Kong, China; 2008. pp. 3052-3059.

[33] De Oca MAM, Stützle T, Van den Enden K, Dorigo M. Incremental social learning in particle swarms. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 2011; 41(2): 368-384.

[34] Durmuş B, Gün A. Parameter identification using particle swarm optimization. In: International Advanced Technologies Symposium (IATS 11); Elazığ, Turkey; 2011. pp. 188-192.

[35] Akay B, Karaboga D. A modified artificial bee colony algorithm for real-parameter optimization. Information Sciences 2012; 192: 120-142.

[36] Peng H, Deng C, Wu Z. Best neighbor-guided artificial bee colony algorithm for continuous optimization problems. Soft Computing 2018; 1-18.

[37] Cui L, Li G, Zhu Z, Lin Q, Wen Z et al. A novel artificial bee colony algorithm with an adaptive population size for numerical function optimization. Information Sciences 2017; 414: 53-67.

[38] López-Ibáñez M, Dubois-Lacoste J, Cáceres L P , Birattari M, Stützle T. The irace package: Iterated racing for automatic algorithm configuration. Operations Research Perspectives 2016; 3: 43-58.

[39] Birattari M, Yuan Z, Balaprakash P, Stützle T. F-race and iterated f-race: An overview. In: Bartz-Beielstein Th, Chiarandini M., Paquete L, Preuss M. In: Beielstein B, Chiarandini T, Paquete M, Preuss L M (editors). Experimental methods for the analysis of optimization algorithms. Springer; 2010, pp. 311-336.

[40] Saha SK, Kar R, Mandal D, Ghoshal SP. Harmony search algorithm for infinite impulse response system identification. Computers & Electrical Engineering 2014; 40(4): 1265-1285.

[41] Rashedi E, Nezamabadi-Pour H, Saryazdi S. Filter modeling using gravitational search algorithm. Engineering Applications of Artificial Intelligence 2011; 24(1): 117-122.

[42] Yang Y, Yang B, Niu M. Adaptive infinite impulse response system identification using opposition based hybrid coral reefs optimization algorithm. Applied Intelligence 2017;48(7)1-18.

[43] ARM Cortex-M4 Processor Technical Reference Manual; revision r0p1 ed. ARM Limited Company; 2015.