Applied Soft Computing 49 (2016) 423–436

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

# An adaptive and hybrid artificial bee colony algorithm (aABC) for ANFIS training

Dervis Karaboga [a], Ebubekir Kaya [b,*]

[a] Department of Computer Engineering, Engineering Faculty, Erciyes University, 38039 Kayseri, Turkey
[b] Department of Computer Technologies, Nevsehir Vocational College, Nevsehir Haci Bektas Veli University, 50300 Nevsehir, Turkey

ARTICLE INFO

*Article history:*
Received 15 November 2015
Received in revised form 24 June 2016
Accepted 22 July 2016
Available online 31 August 2016

*Keywords:*
ANFIS
Neuro-fuzzy
Artificial bee colony algorithm
Swarm intelligence
Nonlinear system identification

ABSTRACT

In this study, we propose an Adaptive and Hybrid Artificial Bee Colony (aABC) algorithm to train ANFIS. Unlike the standard ABC algorithm, two new parameters are utilized in the solution search equation. These are arithmetic crossover rate and adaptivity coefficient. aABC algorithm gains the rapid convergence feature with the usage of arithmetic crossover and it is applied on two different problem groups and its performance is measured. Firstly, it is performed over 10 numerical 'benchmark functions'. The results show that aABC algorithm is more efficient than standard ABC algorithm. Secondly, ANFIS is trained by using aABC algorithm to identify the nonlinear dynamic systems. Each application begins with the randomly selected initial population and then average RMSE is obtained. For four examples considered in ANFIS training, train error values are respectively computed as 0.0344, 0.0232, 0.0152 and 0.0205. Also, test error values for these examples are respectively found as 0.0255, 0.0202, 0.0146 and 0.0295. Although it varies according to the examples, performance increase between 4.51% and 33.33% occurs. Additionally, it is seen that aABC algorithm converges bettter than ABC algorithm in the all examples. The obtained results are compared with the neuro-fuzzy based approaches which are commonly used in the literature and it is seen that the proposed ABC variant can be efficiently used for ANFIS training.


## 1. Introduction

ANFIS is a hybrid artificial intelligence algorithm created by combining learning ability of neural network and inference feature of fuzzy logic [1]. Today, ANFIS is utilized for modelling and identification of many systems [2,3]. In its training stage, the parameters of ANFIS are optimized by using an optimization algorithm. When literature is reviewed, it is seen that derivative-based optimization algorithms and heuristic algorithms are used in training ANFIS. Since derivative based agorithm trip over local minimum in determination of parameters, the different heuristic algorithms such as genetic algorithm (GA), particle swarm optimization (PSO), artificial immune system (AIS), ant colony optimization (ACO), bacterial foraging optimization algorithm (BFOA), bee algorithm (BA), cuckoo search (CS), differential evolution (DE) and harmony search (HS) have been used in order to train ANFIS recently [4–23]. Sarkheyli et al. [4] optimized the parameters belonging to ANFIS by modified genetic algorithm (MGA). Wei [5] utilized a model based on GA and ANFIS to forecast stock price in Taiwan. Liu et al. [6] optimized the parameters of ANFIS by using an improved version of quantum-behaved particle swarm optimization (QPSO). Catalão et al. [7] proposed a novel hybrid approach based on wavelet transform, PSO and ANFIS for short-term electricity prices forecasting. Suja Priyadharsini et al. [8] used AIS to train ANFIS and suggested a model for the elimination of artefacts from EEG signals. Gunasekaran and Ramaswami [9] proposed ANFIS training by AIS to predict the future index value of National Stock Exchange (NSE) of India. Asadollahi-Baboli [10] utilized the modified ant colony optimization (MACO) combined with the shuffling cross-validation (SCV) technique and ANFIS. Teimouri and Baseri [11] proposed an ANFIS based on a continuous ant colony optimization (CACO) technique to correlate the process parameters. Varshini et al. [12] used BFOA and ANFIS together. Bhavani et al. [13] suggested a model based on ANFIS and BFOA for the load frequency control in a competitive electricity market. Zarei et al. [14] suggested to train ANFIS by using BA. Azarbad et al. [15] introduced hybrid intelligent methods based on ANFIS, radial basis function (RBF) and improved bee algorithm (IBA) for GPS (global positioning system). Rani and Sankaragomathi [16] designed an improved PID controller by using ANFIS and CS. Chen and Do [17] proposed a novel approach based on CS and ANFIS for the prediction of student academic performance.

* Corresponding author.
*E-mail addresses:* karaboga@erciyes.edu.tr (D. Karaboga), ebubekir@nevsehir.edu.tr, ebubekirkaya@yandex.com (E. Kaya).

http://dx.doi.org/10.1016/j.asoc.2016.07.039
1568-4946/

Khazraee et al. [18] used ANFIS and DE algorithm for modelling of a reactive batch distillation. Zangeneh et al. [19] employed DE and gradient descent algorithm (GDA) to train ANFIS for identification of the nonlinear dynamic system. Here, while the antecedent parameters are optimized with DE, consequent parameters are trained by with GDA. Wang et al. [20] performed training of ANFIS by using HS for the classification of the epileptic electroencephalogram (EEG) signals. Lutfy et al. [21] used global-best harmony search (GHS) to train a PID-like ANFIS controller. In addition to these, one of the global heuristic algorithms to be used in training ANFIS is ABC algorithm [22,23].

ABC algorithm was developed in 2005 by Karaboga who modeled the food-seeking behavior of honey bees [24–26]. Since then, ABC algorithm has been used to solve several different real world problems in different areas such as neural network training, engineering, software testing, image processing, data mining, sensor networks, protein structure optimization [27–42]. In neural network training; Karaboga and Ozturk [27] used ABC algorithm on training feed-forward neural networks for classifying different data sets which are widely used in the machine learning community. In engineering; Szeto et al. [28] proposed an artificial bee colony to solve the capacitated vehicle routing problem. Rao and Pawar [29] performed modelling and optimization of process parameters of wire electrical discharge machining by using ABC algorithm. Yousefi-Talouki et al. [30] presented a solution of optimal power flow incorporating unified power flow controller based on ABC algorithm. Karaboga [31] proposed a new metot for infinite impulse response filter design based on ABC algorithm. Karaboga and Akay [32] designed optimal Proportional—Integral—Derivative (PID) Controller with ABC algorithm. Mandal at al. [33] proposed an approach based on ABC algorithm and support vector machine (SVM) to overcome the problem of false leak detection of pipeline. In software testing; Mala et al. [34] suggested a novel approach based on ABC algorithm to achieve software test suite optimisation. In image processing; Horng [35] introduced a new multilevel maximum entropy thresholding method by using ABC algorithm. In data mining; Karaboga and Ozturk [36] used ABC algorithm for data clustering on benchmark problems. In sensor networks; Ozturk et al. [37] utilized the ABC algorithm to the dynamic deployment problem in wireless sensor networks with mobile sensors. In protein structure optimization; Bahamish et al. [38] realized protein tertiary structure prediction using ABC algorithm to find the lowest free energy conformation. In addition to these, Zhang and Dolg [39] performed the global optimization of atomic clusters to the one of clusters formed by rigid molecules by using the ABC algorithm. Santander-Jimenez and Vega-Rodriguez [40] described a Multiobjective ABC for inferring evolutionary histories according to parsimony and likelihood. Uehara et al. [41] proposed an approach based on artificial bee colony for the protein–ligand docking. Tang et al. [42] used the minimisation of metabolic adjustment (MOMA) and ABC to predict an optimal set of solutions in order to optimise the production rate of succinate and lactate

In the studies related to ABC algorithm, either standard ABC algorithm is used or its new versions are developed. Search abilitiy of ABC algorithm has been continuously improved although being a strong global optimization algorithm. Therefore, many studies have been conducted and new versions have been released since the first development of ABC algorithm in order to improve its convergence property, success rate, accuracy and exploration capability [43–81]. Two different types come to the forefront in the studies which develop new versions of ABC algorithm. a) The first one is the development of the new version by making some performance enhancing modifications on the standard ABC algorithm [43–62]. b) The second one is the development of a new hybrid algorithm with the integration of other heuristic algorithms and ABC algorithm [63–81].

a) Banharnsakun et al. [43] used the best momentary solution in the solution search equation to increase the convergence speed of the ABC algorithm. Kang et al. [44] combined ABC algorithm and Rosenbrock's rotational direction method. They utilized standard ABC algorithm in the searching phase and the rotational direction method in the exploitation phase. In this manner, they developed an algorithm called Rosenbrock artificial bee colony algorithm. Dos Santos Coelho and Alotto [45] suggested the new version of ABC algorithm by using the Gaussian distribution. They showed that it was also efficient for electromagnetic optimization problems by applying to the Loney's solenoid problem. Rajasekhar et al. [46] developed a new version by using a mutation based on Levy Probability Distributions. Gao et al. [47] used opposition-based learning method and chaotic maps in the generation of initial population to increase the global convergence. Moreover, they updated the search strategy which is utilized in standard ABC algorithm. Babayigit and Ozdemir [48] suggested a different probability function and search mechanism to provide rapid convergence and a more qualified solution. Akay and Karaboga [49] presented a new version which controlled frequency of perturbation and magnitude of the perturbation to enhance the convergence speed of ABC algorithm. Accordingly, they used the control parameters; modification rate (MR) for frequency of perturbation and scaling factor (SF) for magnitude of the perturbation. El-Abd [50] updated to the standard ABC algorithm by using generalized opposition-based learning. Abro and Mohamad-Saleh [51] suggested a new method which exploited the best source among the possible solutions. Su et al. [52] developed a new version called Variable string length Artificial Bee Colony (VABC) algorithm in order to update or re-define some of the processes in standard ABC algorithm. In this version, they used the mutation process. Moreover, the fixed length strings were represented by using variable length strings. They were utilized in the application of automatic fuzzy partitioning approach. With this purpose, they suggested the VABC algorithm (VABC-FCM) based on Fuzzy C-Means clustering technique. Gao et al. [53] proposed a new equation to be used in the onlookers phase to increase the performance of standard ABC algorithm. Furthermore, they used the Powell's method as the local searching tool. Xiang and An [54] suggested an efficient and robust artificial bee colony (ERABC) algorithm. In ERABC, some modifications are made in selection process, solution search phase and determining the initial population. In their algorithm, they used 1) reverse selection based on roulette wheel to achieve the population diversity, 2) chaotic initialization to produce initial population and 3) chaotic search technique for scout bee phase. Moreover, a combinatorial solution search equation exists in ERABC. Sun et al. [55] introduced a non-linear factor for convergence control. In order to accelerate the convergence of ABC algorithm, Zhang and Yuen [56] suggested two conversions which they called one-position inheritance (OPI) mechanism and the opposite directional (OD) search. Shayeghi and Ghasemi [57] proposed a CIABC method based on ABC and chaos theory for the solution of multi-objective emission/economic dispatch (EED) problem. In CIABC, they used Chaotic Local Search (CLS) in order to increase the self-search ability of standard ABC algorithm. Karaboga and Gorkemli [58] proposed a new version called quick artificial bee colony which enhanced the local searching ability of standard ABC algorithm and which modeled the behaviors of onlooker bees accurately. In their study, they reported that the convergence of the standard ABC algorithm is increased in case that the radius of neighborhood was set up correctly. Additionally, adaptif solution generation mechanisms are proposed in some studies. Alam and Islam [59] presented the Artificial Bee Colony with Self-Adaptive Mutation (ABC-SAM) method which con-

ducted a search by adapting the mutation step size dynamically. In a separate study, Hasan and Ahmed [60] used the mutation step size to provide an adaptive solution generation mechanism. Gu et al. [61] presented a method called self-adaptive ABC. By using their method, they set the value of $\Phi_{ij}$ parameter existing in standard ABC algorithm according to the variable iteration number. Liao et al. [62] updated the previous study of Akay and Karaboga [49]; and suggested a new method varying modification rate (MR) control parameter dynamically. They tested the performance of their method by using the data from hydropower systems of Three Gorges in China.

b) Inspired by PSO, Zhu and Kwong [63] developed an algorithm called as the gbest-guided ABC (GABC) used best global solution information in solution search equation. Mezura-Montes and Velez-Koeppel [64] utilized the best solution found in the local searching process. Zhang et al. [65] added a linear weight to the GABC algorithm which was developed by Zhu and Kwong [63] and proposed a method called as WGABC. They suggested a new solution search equation which is used in the scout bee phase. Nagur et al. [66] developed the solution search equation in which the best global solution information is used in order to solve the economic dispatch (ED) problem. Luo et al. [67] introduced a new method named converge onlookers ABC (COABC). In their method, all onlookers directly converged into the best solution. This fact made the calculation of roulette wheel selection more affordable. Also in their approach, there was no need to calculate the fitness value of the food sources. Inspired by DE, Gao and Liu [68] developed improved artificial bee colony (IABC) algorithm based on standard ABC for the global optimization problems. In the algorithm, they used two advanced solution search equations; "ABC/best/1" and "ABC/rand/1". In addition, they utilized the chaotic system and opposition-based learning method in order to accelerate the global convergence while producing the initial population. Gao and Liu [69] presented the solution search mechanism called as DE/best/1 in a different study. By enhancing this method, Gao et al. suggested the different solution search mechanisms which they called ABC/best (DE/best/1 and DE/best/2) [70]. In these mechanisms, they used the best solution provided by the previous iteration. While generating the initial population, they again utilized the chaotic system and opposition-based learning method. Inspired by DE, Wang and Wang [71] suggested a new version called as pbest-guided ABC (PABC) involving the best local solution information in the solution search equation. Moreover, they modified the frequency of perturbation for each iteration. Rubio-Largo et al. [72] developed a hybrid approach named Multi-objective Artificial Bee Colony with Differential Evolution (MO-ABCIDE) by combining the features of DE and ABC algorithms. Li and Yin [73] suggested a different hybrid method based on DE and ABC algorithms for the reconfigurable antenna array with quantized phase shifter. Hati et al. [74] developed a new version of ABC algorithm called DE-PM-ABC by using DE and Polynomial Mutation (PM). They applied the mutation strategy called 'DE/rand/1' along with the modification rate (MR) in order to improve the utilization mechanism of employed and scout bees. Wang et al. [75] was inspired by JADE (adaptive differential evolution with optional external archive) algorithm [76] and suggested the improved ABC (IABC) algorithm. In IABC, the best solution found for each generation was included to the records and a new candidate solution was obtained by searching the neighborhoods of the solution which was randomly selected among these records. Gao et al. [77] suggested a new algorithm called as EABC. In EABC, they introduced two updated equation to generate a candidate solution in employed bee and scout bee phases respectively. Tsai et al. [78] proposed Interactive Artificial Bee Colony algorithm which explained the movements of onlooker bees with universal gravitation. Fister et al. [79] developed a memetic ABC (MABC) algorithm which was hybridized with the two local search heuristics: the Nelder-Mead algorithm (NMA) and the random walk with direction exploitation (RWDE) methods. Chen et al. [80] suggested a new algorithm in which Simulated annealing algorithm was used to find the best solution in the searching process of the employed bees. Zhang et al. [81] proposed the hybrid artificial bee colony (HABC) algorithm inspired by the food searching behavior of the bacteria and bees for the purpose of estimating parameters of the proton exchange membrane fuel cell (PEMFC) model. HABC algorithm used the solution search equation which mimicked the chemotactic effect of the bacteria to improve the local searching ability. Moreover, Adaptive Boltzmann Selection Scheme was adapted into HABC algorithm in order to prevent the early convergence.

As specified in the paragraphs above, various versions are recommended in order to increase the convergence speed of standard ABC. Yet, improving the local convergence speed of ABC without affecting global research ability still continues as an area to be studied. Thus, using arithmetic crossover operation of genetic algorithms for solution generation mechanism and then the adaptation of solution generation mechanism depending on the abandonment criteria value being one of the control parameters of ABC seem a good strategy. Crossover operator and abandonment criterion (limit) have been used in this way in the solution generating mechanism for the first time, and stand as an innovative solution for improving convergence speed of ABC.

Under the light of what is stated in this section, it can be seen that ABC algorithm is used to find optimal solutions for several difficult optimization problems and it is successful. Training ANFIS is also considered as one of the difficult optimization problems. Therefore in the earlier studies, ANFIS is trained to identify the non-linear static and dynamic systems by using the standard ABC algorithm [22,23]. In this study, an Adaptive and Hybrid Artificial Bee Colony (aABC) algorithm based on arithmetic crossover operation of GA and the adaptation of solution generation mechanism depending on the abandonment criteria value is suggested to increase the performance of training ANFIS by using ABC algorithm. The remainder of the paper is organized as follows: In Section 2, The ANFIS is described. In Section 3, The standard ABC and proposed aABC is presented. In Section 4, the experimental results are given and evaluated. Finally, conclusions are provided in Section 5.

## 2. Adaptive network fuzzy inference system (ANFIS)

As stated before, ANFIS is a hybrid artificial intelligence algorithm created by combining learning ability of neural networks and inference feature of fuzzy logic [1]. ANFIS structure contains input-output data pairs in fuzzy inference system and IF-THEN rules. ANFIS is trained by using the existing input-output data pairs for the solution of available problems. Thus, IF-THEN rules in ANFIS are obtained. To create rule in ANFIS means that it is benefited from expert opinions. Thus, ANFIS is used since it enables artificial neural networks to be benefited from expert opinions in many estimation problems.

Network structure of ANFIS consists of two parts. First part is called as antecedent part and second part as conclusion part. These parts are interconnected by fuzzy rules within network structure of ANFIS. ANFIS is trained and updated by using parameters found in these parts. Structure of ANFIS consists of five layers.These layers are explained below:

### 2.1. Layer 1

This part is called as fuzzification layer. Fuzzification layer uses membership functions in order to obtain fuzzy clusters from input values. In this part, different membership functions such as generalized bell function (gbell) and the triangular function (trimf) may be used. Parameters like {a, b, c} in membership functions determine the form of membership function and these parameters are called as antecedent parameter. Membership degrees of each membership function are determined by using these parameters, as given in Eqs. (1) and (2). The membership degrees obtained from this layer are shown with $\mu_x$ and $\mu_y$

$$\mu_{A_i}(x) = gbellmf(x; a, b, c) = \frac{1}{1 + |\frac{x-c}{a}|^{2b}} \quad (1)$$

$$O_i^1 = \mu_{A_i}(x) \quad (2)$$

### 2.2. Layer 2

This layer is called as rule layer. Firing strengths ($w_i$) for the rules are generated by using membership values computed in fuzzification layer. $w_i$ values are found by multiplying the membership values as the following (3).

$$O_i^2 = w_i = \mu_{A_i}(x).\mu_{B_i}(y) \quad i = 1,\ 2. \quad (3)$$

### 2.3. Layer 3

This layer is called as normalization layer. It calculates normalized firing strengths belonging to each rule. The normalized value is the ratio of the firing strength of the $i^{th}$ rule to the total of all firing strengths as given in (4).

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2} \quad i = 1,\ 2. \quad (4)$$

### 2.4. Layer 4

This layer is called as defuzzification layer. Weighted values of rules are calculated in each node of this layer as given in (5). This value is calculated by using first order polynomial.

$$O_i^4 = \overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i) \quad (5)$$

$\overline{\omega_i}$ is the output of the normalization layer and $\{p_i, q_i, r_i\}$ is the parameter set. These are called the conclusion parameters. The number of conclusion parameters of each rule is one more than the number of input. For example; in the structure of the ANFIS with four inputs, the number of conclusion parameters of each rule is five.

### 2.5. Layer 5

It is called as the summation layer. The actual output of ANFIS is obtained by summing the outputs obtained for each rule in defuzzification layer.

$$O_i^5 = overalloutput = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (6)$$

## 3. Artificial bee colony (ABC) algorithms

### 3.1. Standard ABC algorithm

In the artificial bee colony algorithm, an artificial colony contains three groups of bees: employed, onlooker and scout bees. First half of the colony consists of the employed artificial bees and the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality of the associated solution. The number of the employed bees or the onlooker bees is taken to the equal to the colony size.

The basic ABC algorithm has three fundamental control parameters: colony size, limit value and the value of maximum number of cycles (MCN). Main steps of ABC algorithm are given below.

Initialize and evaluate the population of solutions $X_i$, i = 1...SN

REPEAT

Produce new solutions $V_i$ for the employed bees by using (8), evaluate them and apply the greedy selection process

Calculate $P_i$ values for food caves depending on their quality

Produce the new solutions $V_i$ for the onlookers from $X_i$, selected based on $P_i$, evaluate them and apply the greedy selection process

Determine the abandoned solution for the scout and replace it with a new solution generated by $X_i$ by (7)

Memorize the best solution achieved so far

cycle = cycle + 1

UNTIL (cycle = MCN)

In ABC algorithm, the process startes with the determination of initial food sources randomly. Random position production process is conducted by generating a value between lower and upper limits of each parameter by using (7).

$$X_{ij} = X_j^{\min} + rand(0,\ 1)(X_j^{\max} - X_j^{\min}) \quad (7)$$

where i = 1...SN, j = 1...PN. SN is number of food sources and PN is the number of parameter to be optimized. $X_j^{\min}$ describes the lower limit of j. parameter and $X_j^{\max}$ describes the upper limit of j. parameter.

Employed bees determine a new neighbour food source by using the equality given in (8) in the neighborhood of her food source and evaluate its quality. Here, j is an integer produced randomly in interval of [1,D]. $\Phi_{ij}$ is a random value taken in interval of [−1,1], k is the index of the solution chosen randomly. In (8), $V_i$ source is found by changing only one parameter of $X_i$.

$$V_{ij} = X_{ij} + \Phi_{ij}(X_{ij} - X_{kj}) \quad (8)$$

If $V_i$ is better than $X_i$, employed bee deletes its address in her memory. Otherwise, it continues to go to the former address. Probabilistic selection process by onlooker bees is done by using the quality value corresponding to nectar amount. Selection process based on quality value was done by using roulette wheel method in standard ABC algorithm. According to roulette wheel method, the selection probability of sources is computed by (9). Here, fit-

ness; shows the quality of i. source and SN represents the number of employed bees.

$$P_i = \frac{fitness_i}{\sum_{j=1}^{SN} fitness_j} \quad (9)$$

After onlooker bees choose a food source, each of them finds a new solution in the neighborhood of the food source chosen by using (8) and a greedy selection is applied between the new solution and the food source chosen in order to improve the solution at hand. If a food source can not be improved through a predetermined cycles, called "limit", it is removed from the population and the employed bee belonging to food source becomes scout. The scout bee finds a new random food source position by (7).

### 3.2. Adaptive and hybrid artificial bee colony algorithm

In order to increase the convergence speed and performance of standard ABC algorithm, the solution generating mechanism which models the onlookers' behaviors is updated and two new control parameters are included to its structure. These are arithmetic crossover rate and adaptivity parameter which is generated basing on failure counter.

Crossover is the new individual generation process by taking some genes of the elected parents. The new individual has the characteristics of the parents. The purpose here is to generate more qualified individuals. At the same time, crossover increases the speed of convergence. Therefore, crossover rate is one of the most important control parameters which are used in several evolutionary algorithms, GA and DE being in the first place. Similarly, the arithmetic crossover is utilized to increase the convergence speed of ABC algorithm. By including the arithmetic crossover to the standard ABC algorithm, the solution generating mechanism given in (10) is obtained. In (10), $\gamma$ is the crossover rate and it has a value within the range of [0.5, 1]. Also, $x_g$ is the best global solution information.

The failure counter in the ABC algorithm gives the number of the successive failures in generating new solutions. When the value of failure counter reaches the limit value—which is one of the control parameters of ABC algorithms—the new candidate solution is randomly generated with the usage of (7). With this process, the failure counter is set to zero. This fact prevents the generation of new qualified solutions and continues during long iterations. Therefore, in order to eliminate this prevention, an adaptive mechanism depending on failure counter is developed. The new solution generating mechanism which is developed by using the failure counter is given in (11).

$$v_{ij} = x_{ij}\gamma + x_{g_j}(1-\gamma) + \Phi_{ij}(x_{ij} - x_{kj}) \quad (10)$$

$$v_{ij} = x_{ij} + B_i\Phi_{ij}(x_{ij} - x_{kj}) \quad (11)$$

$$B_i = \left(\frac{1}{1+trial_i}\right)^{\alpha} \quad (12)$$

$$v_{ij} = x_{ij}\gamma + x_{g_j}(1-\gamma) + B_i\Phi_{ij}(x_{ij} - x_{kj}) \quad (13)$$

B, which is displayed in (11), is the adaptivity parameter and indicates the parameter which is produced with the usage of failure counter. $B_i$ is the adaptivity parameter which is obtained with the usage of failure counter corresponding to i. solution. $B_i$ is calculated by using (12). $trial_i$ given here is the value of failure counter corresponding to i. solution. "$\alpha$" parameter is the adaptivity coefficient and it has a value within a range of [0,n]. Here, "n" value is a positive real number. As can be seen from (12), when the values of $\alpha$ and $trial_i$ parameters increase, the value of $B_i$ decreases and gets closer to zero. Similarly, When the values of these parameters decrease; the value of $B_i$ increases and gets closer to one. This fact shows that the value of B parameter is arranged adaptively according to the values of $\alpha$ and $trial_i$. Therefore, $\Phi_{ij}\left(x_{ij} - x_{kj}\right)$ changes in accordance with the $B_i$ parameter adaptively. Consequently, the best step magnitude is set according to the situations and conditions.

With the integrated arithmetic crossover and adaptivity structure displayed in (10) and (11), the solution generating mechanism of the aABC algorithm given in (13) is developed. With this developed mechanism, aABC algorithm gains with the feature of rapid convergence and its performance to reach the best solution increases. Additionally, as different from standard ABC algorithm, two new control parameters are included; $\gamma$ crossover rate and $\alpha$ adaptivity coefficient.

Adaptive structures presented in [59,60] were also provided with arithmetic crossover operation. However, they don't employe this operation to generate new solution. In the proposed method, arithmetic crossover is used in acquisition of a new solution from two solutions. In such way, it is aimed to obtain more qualified new solutions quickly. Also, in the proposed method, adaptive structure is realized by using the failure counter (limit) that is one of control parameters of ABC algorithm. Thus, the proposed method is different from other methods [59,60], it has original and innovative structure.

## 4. Simulation studies and discussions

Benchmark problems are primarily used to test the performance of aABC algorithm. Then, the algorithm is performed by training ANFIS to identify the nonlinear dynamic systems.

### 4.1. Simulation studies on benchmark functions

In this section, 10 benchmark problems are utilized to test the performance of proposed ABC algorithm. The information related to the test problems such as characteristic of the problems (C), dimensions of the problems (D), bounds of the search spaces (Interval) and the global optimum values of the problems are presented in Table 1. Here, C denotes one of characteristics such as the unimodal-seperable (US), unimodal-nonseperable (UN), multimodal-seperable (MS) or multimodal-nonseperable (MN). In the applications, the different values of $\gamma$ ($\gamma = 0.5$, $\gamma = 0.6$, $\gamma = 0.7$, $\gamma = 0.8$, $\gamma = 0.9$, $\gamma = 1$) and $\alpha$ ($\alpha = 0$, $\alpha = 0.25$, $\alpha = 0.5$, $\alpha = 1$, $\alpha = 2$, $\alpha = 3$) are used in (12) and (13). With the combination of the $\gamma$ and $\alpha$ values; 36 different ($\gamma$, $\alpha$) couples are obtained. Accordingly, 36 different applications are performed for each of the test problems. The results are compared with the results of the GA, PSO, DE, standard ABC and qABC algorithms taken from [58]. In order to make a fair comparison, the control parameters which are used in [58] are taken into consideration. Accordingly, the colony size is 50 and the maximum evaluation number is 500.000. If the limit control parameter (1) is calculated with (14) by using the colony size (CS) and dimension of the problem (D).

$$l = \frac{CS \times D}{2} \quad (14)$$

Each application is run 30 times by using the randomly selected initial populations. Then, the average value (Mean), standard deviation (SD), the best and the worst value are calculated by analysing the results.The performance is measured by using 36 different ($\gamma$, $\alpha$) combination for each problem. In case that all the findings are stated in the paper, it is seen that it will take a very big place. Therefore, this study does not involve all the findings.Within the scope of the paper, the ($\gamma$, $\alpha$) parameters which provide the best results for each problem are presented. The best results achieved for testing problems are given in Table 2. The values smaller than E-15 are accepted as "zero" in the table. Here, the ($\gamma$, $\alpha$) parameter value

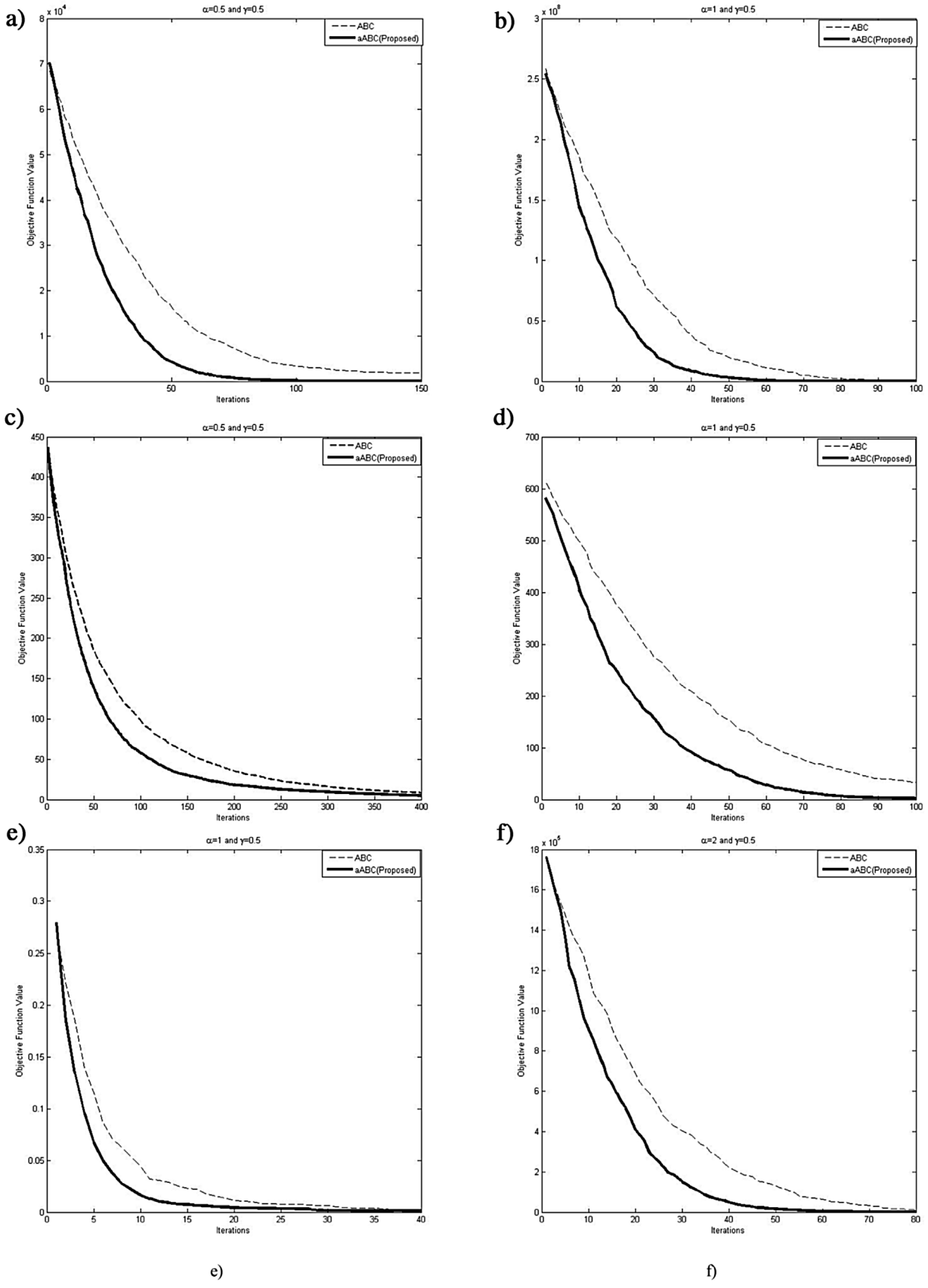


**Fig. 1.** Comparison of ABC and aABC algorithms' convergence on a) Sphere b) Rosenbrock c) Rastrigin d) Griewank e) Schaffer f) Dixon-Price function.

**Table 1**
Test problems.

| | Test function | C | D | Interval | Min | Formulation |
|---|---|---|---|---|---|---|
| 1 | Sphere | US | 30 | [−100,100] | $F_{min}=0$ | $f(x)=\sum_{i=1}^{D} x_i^2$ |
| 2 | Rosenbrock | UN | 30 | [−30,30] | $F_{min}=0$ | $f(x)=\sum_{i=1}^{D-1} 100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2$ |
| 3 | Rastrigin | MS | 30 | [−5.12, 5.12] | $F_{min}=0$ | $f(x)=\sum_{i=1}^{D}\left(x_i^2-10\cos(2\pi x_i)+10\right)$ |
| 4 | Griewank | MN | 30 | [−600,600] | $F_{min}=0$ | $f(x)=\frac{1}{4000}\left(\sum_{i=1}^{D} x_i^2\right)-\left(\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)\right)+1$ |
| 5 | Schaffer | MN | 2 | [−100,100] | $F_{min}=0$ | $f(x)=0.5+\frac{\sin\left(\sqrt{\sum_{i=1}^{D} x_i^2}\right)^2-0.5}{\left(1+0.001\left(\sum_{i=1}^{D} x_i^2\right)\right)^2}$ |
| 6 | Dixon-Price | UN | 30 | [−10,10] | $F_{min}=0$ | $f(x)=(x_1-1)^2+\sum_{i=2}^{D} i\left(2x_i^2-x_{i-1}\right)^2$ |
| 7 | Ackley | MN | 30 | [−32,32] | $F_{min}=0$ | $f(x)=20+e-20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)-exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$ |
| 8 | Schwefel | MS | 30 | [−500,500] | $F_{min}=-12{,}569.5$ | $f(x)=\sum_{i=1}^{D} -x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$ |
| 9 | SixHumpCameBack | MN | 2 | [−5,5] | $F_{min}=-1{,}03163$ | $f(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$ |
| 10 | Branin | MS | 2 | [−5,10] x [0,15] | $F_{min}=0{,}398$ | $f(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos(x_1)+10$ |

**Table 2**
The best results obtained by using aABC algorithm on Benchmark test functions.

| | Test Functions | Alfa (α) | Gama (γ) | Mean | SD | Best | Worst |
|---|---|---|---|---|---|---|---|
| 1 | Sphere | All | All | 0 | 0 | 0 | 0 |
| 2 | Rosenbrock | 0.5 | 0.7 | 0.0246333 | 0.0481561 | 2.1913e-005 | 0.217807 |
| 3 | Rastrigin | α <= 0.5 | γ <= 0.8 | 0 | 0 | 0 | 0 |
| 4 | Griewank | α <= 0.5 | All | 0 | 0 | 0 | 0 |
| 5 | Schaffer | All | γ <= 0.7 | 0 | 0 | 0 | 0 |
| 6 | Dixon-Price | 0.25 | 1 | 1.053E − 014 | 3.4014E − 014 | 2.2822e − 015 | 1.9328e − 013 |
| 7 | Ackley | 0 | 0.5 | 2.990E − 014 | 3.053E − 015 | 2.2204e − 014 | 3.2862e − 014 |
| 8 | Schwefel | α <= 0.25 | γ <= 0.7 | −12569.487 | 0 | −12569.487 | −12569.487 |
| 9 | SixHumpCamelBack | All | All | −1.0316284 | 0 | −1.0316284 | −1.0316284 |
| 10 | Branin | All | All | 0.3978874 | 0 | 0.3978874 | 0.3978874 |

shows different behaviors according to the problems. In general terms, the best results are achieved when γ <= 0.8 and α <= 0.5. In Fig. 1 and Fig. 2, the convergence graphics which are provided with the aABC algorithm over the test functions are displayed and it is compared with the standard ABC algorithm. In these graphics, different (γ, α) values are used for aABC algorithm. Moreover, it is seen that the convergence speed of aABC algorithm is better than the standard ABC algorithm over all the test functions.

The performance of aABC and ABC algorithms over the Benchmark test functions is compared by using Wilcoxon signed rank test. Wilcoxon signed rank test is non-parametric statistical test which is used for modelling the behaviors of evolutionary algorithms [82]. The results of the comparison which include the average difference and p-Value are given in Table 3. The average difference is found as "0" for Sphere, Rastrigin, Griewank, Dixon-Price, Ackley, Schwefel, SixHumpCameBack and Branin functions. For the other two functions, an average which is different from "0" is achieved. As for Rosenbrock function, p-Value is obtained as 0.002. Therefore, there is a significant difference for being 0.002 < 0.05. It is seen that the performance of aABC algorithm is better than the performance of ABC algorithm over the Rosenbrock function. Along with it, for the other function the p-Value is found as "0". Accordingly, it is concluded that the performance of aABC algorithm is better than ABC algorithm for Schaffer.

**Table 3**
Wilcoxon signed rank test results for Benchmark functions.

| Function | Mean difference | p-Value |
|---|---|---|
| Sphere | 0 | – |
| Rosenbrock | 0.1520624 | 0.002 |
| Rastrigin | 0 | – |
| Griewank | 0 | – |
| Schaffer | 1.0367E − 010 | 0 |
| Dixon-Price | 0 | – |
| Ackley | 0 | – |
| Schwefel | 0 | – |
| SixHumpCameBack | 0 | – |
| Branin | 0 | – |

In Table 4, the performance of aABC algorithm is compared with the GA, PSO, DE, ABC and qABC algorithms. Also, the average and standard deviation obtained as a result of the 30 runs related to the algorithms are given. In the comparison table, the values under E-12 are taken as "0", as it is accepted in [58]. Over the Sphere function,

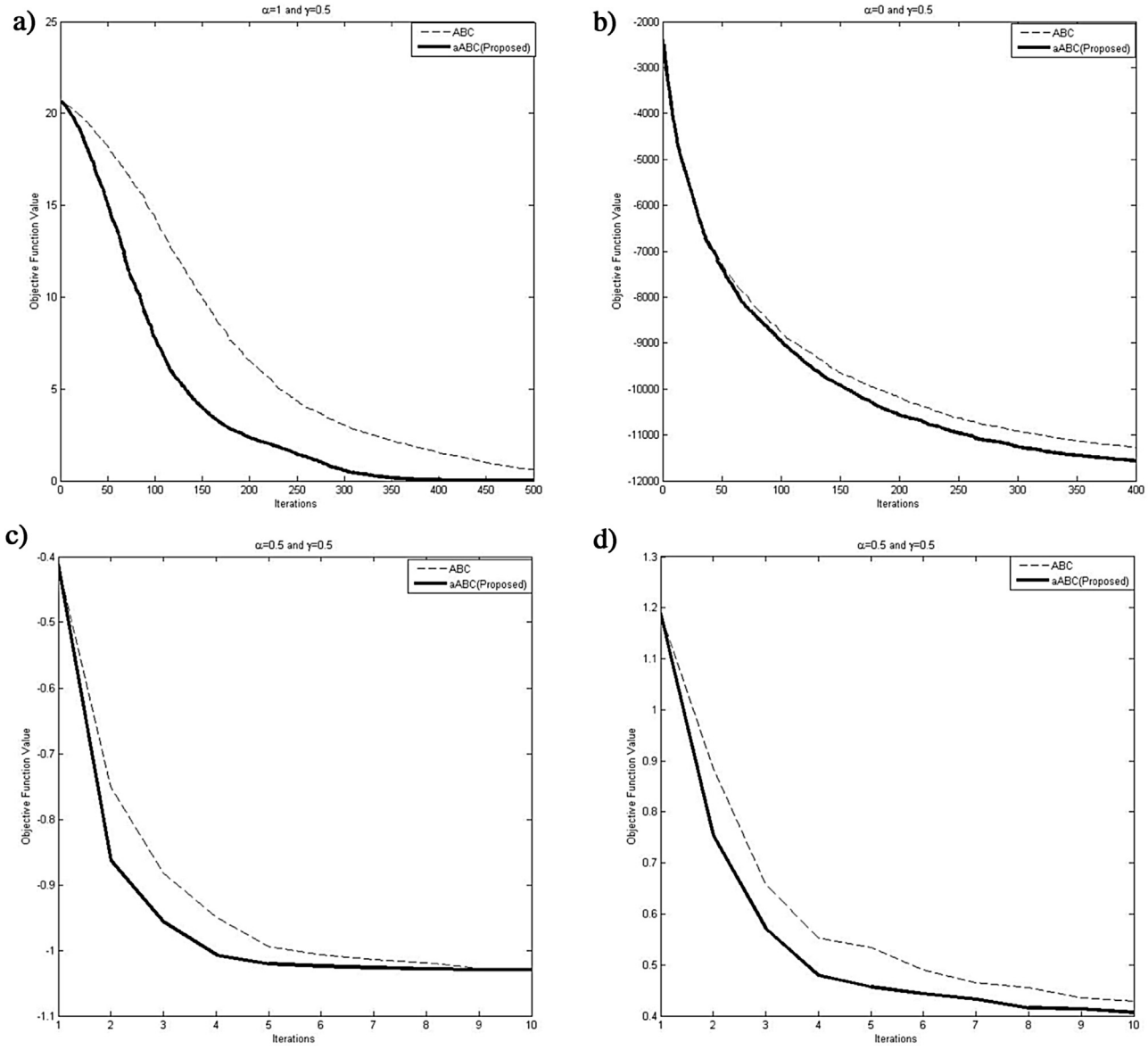


**Fig. 2.** Comparison of ABC and aABC algorithms'convergence on a)Ackley b) Schwefel c) SixHumpCameBack d) Branin function.

for all the algorithms, except for GA, the optimum solution is found for average and standard deviation. The best average and standard deviation values for Rosenbrock function are obtained with aABC algorithm. Accordingly, the average is found as 0.0246333 and standard deviation is obtained as 0.0481561. The best results over Rastrigin and Griewank functions are achieved with the ABC, qABC and aABC algorithm. For the Ackley function, the optimum solution, which is "0", is found with DE, ABC, qABC and aABC algorithms. Efficient results are achieved over the Schaffer function by using of PSO, DE and aABC algorithms. −12569.487 for the average and 0 for the standard deviation over the Schwefel function are obtained with the usage of aABC algorithm. The most efficient result over the Dixon-Price function is found with the usage of ABC and aABC algorithms and the optimum solution is reached. As for the SixHumpCameBack and Branin functions, the optimum solution can only be obtained with the usage of GA. When all the results are noticed as given in Table 4, it is seen that the performance of aABC in terms of reaching the optimum solution is better than the performances of GA, PSO, DE, ABC and qABC.

The results obtained by using aABC algorithm in 10000 iterations are compared with different methods and it is observed that results are effective. At the same time, graphics of convergence are given in Fig. 1 and 2 for 10 examples. The error values in the low iterations should be analyzed to determine the significance of graphics of convergence. Therefore, the results obtained by aABC and ABC algorithm for 50 iterations are presented in Table 5. In addition, p-Value is determined by using Wilcoxon signed rank test and is added to Table 5. When Table 5 is examined, it can be seen that better results are obtained by using the aABC algorithm for 9 examples. The same result is found only with the ABC algorithm for SixHumpCamelBack. Furthermore, p – Values are lower than 0.05 for 9 examples. This situation shows that the results for 50 iterations are significant. Namely, it indicates that convergence speed of aABC algorithm is much better than convergence speed of ABC algorithm. All tables and figures given for this section describe that better results are obtained in low and high iterations in aABC algorithm than in ABC algorithm. This performance increase is the effect

**Table 4**
Comparison of proposed aABC algorithm with state of art algorithms [58].

| Function | GA | | PSO | | DE | | ABC | | qABC | | aABC (Proposed) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Mean | SD | Mean | SD | Mean | SD | Mean | SD | Mean | SD | Mean | SD |
| Sphere | 1.11E+03 | 74.214474 | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Rosenbrock | 1.96E+05 | 3.85E+04 | 15.088617 | 24.170196 | 18.203938 | 5.036187 | 0.1766957 | 0.2661797 | 0.1329198 | 0.1799131 | **0.0246333** | **0.0481561** |
| Rastrigin | 52.92259 | 4.56486 | 43.977137 | 11.728676 | 11.716728 | 2.538172 | **0** | **0** | **0** | **0** | **0** | **0** |
| Griewank | 10.63346 | 1.161455 | 0.0173912 | 0.020808 | 0.0014792 | 0.002958 | **0** | **0** | **0** | **0** | **0** | **0** |
| Ackley | 14.67178 | 0.178141 | 0.1646224 | 0.493867 | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| Schaffer | 0.004239 | 0.004763 | **0** | **0** | **0** | **0** | 1.04E−10 | 4.83E−10 | 8.66E−06 | 7.83E−06 | **0** | **0** |
| Schwefel | −11593.40 | 93.25424 | −6909.1359 | 457.95778 | −10.266 | 521.84929 | **−12569.49** | 2.07E−12 | **−12569.49** | 2.51E−12 | **−12569.49** | **0** |
| Dixon-Price | 1.22E+03 | 2.66E+02 | 0.6666667 | E8 | 0.6666667 | E9 | 0 | 0 | 1.15E−12 | 3.36E−12 | **0** | **0** |
| SixHumpCameBack | **−1.03163** | **0** | −1.0316285 | 0 | −1.031628 | 0 | −1.0316284 | 0 | −1.0316284 | 0 | −1.0316284 | 0 |
| Branin | **0.397887** | **0** | 0.3978874 | 0 | 0.3978874 | 0 | 0.3978874 | 0 | 0.3978874 | 9.70E−10 | 0.3978874 | 0 |

The best results is given bold.

**Table 5**
Comparison of the obtained results aABC and ABC algorithm on Benchmark test functions (the number of iterations = 50).

| Example | ABC | aABC (Proposed) | p Value |
|---|---|---|---|
| Sphere | 16332 | **4236.97** | 0.000 |
| Rosenbrock | 1.97971e+07 | **3.01371e+06** | 0.000 |
| Rastrigin | 184.694 | **138.408** | 0.000 |
| Griewank | 154.152 | **56.2696** | 0.000 |
| Ackley | 18.2421 | **15.0231** | 0.000 |
| Schaffer | 0.000795534 | **0.000209608** | 0.021 |
| Schwefel | −7041.44 | **−7378.81** | 0.041 |
| Dixon-Price | 128614 | **16344.8** | 0.000 |
| SixHumpCameBack | −1.03163 | −1.03163 | – |
| Branin | 0.397929 | **0.397887** | 0.000 |

The best results is given bold.

of solution generation mechanism that has adaptive and hybrid properties.

It is witnessed that the crossover and adaptivity operation which exist in the structure of aABC algorithm are both efficient in order to increase the speed of convergence over benchmark functions. The best convergence is achieved when $\gamma <= 0.8$ for all $\alpha$ values. On the contrary, it is detected that the convergence rate decreases when the value of $\gamma$ gets closer to 1. A clear conclusion can not be achieved in terms of reaching the optimum solution and it is observed that different parameter values are efficient in accordance with the problems. Moreover, when analyzed the results of Wilcoxon signed rank test and the comparison results stated in Table 4, it is seen that aABC algorithm is more efficient than the standard ABC algorithm in terms of reaching the optimum solution.

### 4.2. Simulation studies on training ANFIS to identify dynamic systems

In this application, ANFIS is trained by using aABC algorithm in order to identify the nonlinear dynamic systems. The results are compared with the neuro-fuzzy based methods such as RSONFIN [83], RFNN [84], TRFN-S [85], DFNN [86], HO-RNFS [87], RSEFNN-LF [88], WRFNN [89], RBF-AFS [90], OLS [91], GDFNN [92], FAOS-PFNN [93] and GOSFNN [94] which are commonly used in the literature.

In the applications, 4 different nonlinear dynamic systems—each comprises of one input and one output—are utilized. The systems used are presented in Table 6. Here, u(t) points out the previous input and y(t) indicates the previous output. In the Example 1, (15) is used in order to provide the previous input. (16) is utilized for Example 2 and 3. In order to conduct the ANFIS training, sample data are provided by using the equations belonging to the nonlinear systems. For the Example 1–4 respectively 1000, 1000, 250 and 200 data is produced. Some part of the data which is produced is used for ANFIS training and some part of it is utilized for ANFIS testing. Accordingly, 900 data group is used for Example-1, 900 data group is used for Example-2, 200 data group is used for Example-3 and 100 data group is used for Example-4 in ANFIS training. Testing is performed with the other data groups. For Example 1, 2 and 4, ANFIS structure which includes two inputs and one output is utilized. In these ANFIS structures; u(t) and y(t) are determined as input and y(t+1) is defined as output. In the Example-3, the ANFIS structure which contains of 3 inputs and 1 output is preferred. Therefore, u(t), y(t) and y(t-1) are selected as ANFIS inputs and y(t+1) is choosed as ANFIS output. In the applications, generalized bell function is used as membership function. In the Example 1 and 2, 4 rules are composed by using 2 membership functions for each input. 24 parameters in total are optimized. In Example 3, 2 membership functions are used for each input and 8 rules are composed. 50

**Table 6**
Non-linear dynamic systems and parameter values used in training ANFIS.

| Example | Number of Rules | Number of MFs | Number of Parameters | Inputs of ANFIS | Number of Train/Test Data | System |
|---|---|---|---|---|---|---|
| 1 | 4 | 4 | 24 | $u(k), y_p(k)$ | 900/100 | $y_p(k+1)=\frac{y_p(k)y_p(k-1)y_p(k-2)u(k-1)(y_p(k-2)-1)+u(k)}{1+y_p(k-1)^2+y_p(k-2)^2}$ |
| 2 | 4 | 4 | 24 | $u(k), y_p(k)$ | 900/100 | $y_p(k+1)=0.72y_p(k)+0.025y_p(k-1)u(k-1)+0.01u^2(k-2)+0.2u(k-3)$ |
| 3 | 8 | 6 | 50 | $u(k), y_p(k), y_p(k-1)$ | 200/50 | $y_p(k+1)=\frac{y_p(k)y_p(k-1)[y_p(k)+2.5]}{1+y_p(k)^2+y_p(k-1)^2}+u(k)$ |
| 4 | 16 | 8 | 72 | $y_p(k), y_p(k-1)$ | 100/100 | $y_p(k+1)=-1.4y_p(k)^2+0.3y_p(k-1)+1.0$ |

**Table 7**
The best results obtained by using aABC algorithm on Example 1–4.

| Example | TRAIN | | | | | | TEST | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Alfa ($\alpha$) | Gama ($\gamma$) | Mean | SD | Best | Worst | Mean | SD | Best | Worst |
| 1 | 1.5 | 0.9 | 0.0343990 | 0.00163188 | 0.0315564 | 0.0378710 | 0.0254703 | 0.00255909 | 0.0190953 | 0.0319061 |
| 2 | 1.5 | 0.8 | 0.0232973 | 0.00029730 | 0.0229809 | 0.0243227 | 0.0201836 | 0.00147256 | 0.0178608 | 0.0227911 |
| 3 | 1.5 | 1 | 0.0152358 | 0.00409767 | 0.0102189 | 0.0304819 | 0.0145597 | 0.00419437 | 0.0089779 | 0.0304033 |
| 4 | 1.0 | 0.7 | 0.0204562 | 0.00469554 | 0.0140201 | 0.0356166 | 0.0294599 | 0.00737983 | 0.0182248 | 0.0436709 |

parameters in total are optimized. In the Example 4, 4 membership functions are utilized for each input and 16 rules are obtained. Thus, 72 parameters are optimized.

$$u(k)=\begin{cases}\sin(\pi k/25) & k<250\\ 1 & 250\le k<500\\ -1 & 500\le k<750\\ 0.3\sin(\pi k/25)+0.1\sin(\pi k/32)+0.6\sin(\pi k/10) & 750\le k<1000\end{cases} \quad (15)$$

$$u(k)=\sin(2\pi k/25) \quad (16)$$

In the previous studies, the system identification is performed with ANFIS by using standard ABC algorithm [22,23]. In order to make a fair comparison among the obtained results, the same control parameter values are used. Therefore, the size of the colony, limit and maximum cycle number are selected successively as 20, 250 and 5000. Besides, two new control parameters are included into the aABC algorithm. In the applications, the different values of $\gamma$ ($\gamma=0.5$, $\gamma=0.6$, $\gamma=0.7$, $\gamma=0.8$, $\gamma=0.9$, $\gamma=1$) and $\alpha$ ($\alpha=0$, $\alpha=0.5$, $\alpha=1$, $\alpha=1.5$, $\alpha=2$, $\alpha=2.5$, $\alpha=3$) are utilized in (12) and (13). In this manner, 42 different ($\gamma$, $\alpha$) couples are obtained and 42 applications are made for each example.

Each application begins with the randomly selected initial population and it is run 30 times. This condition is repeated for 42 applications which exist in each example. The findings achieved are analyzed and the average error value (Mean), standard deviation (SD), the best and the worst error are found. Due to the fact that the number of application is high, the paper does not include all the findings. Only the ($\gamma$, $\alpha$) values belonging to the best results are stated. The findings achieved with the usage of these parameters are displayed in Table 7. According to the results, the best training and test results are generally found when $\alpha>=1.0$ and $\gamma>=0.7$. In Fig. 3 and Fig. 4, the convergence graphics provided for all examples are shown. In order to provide these graphics, different ($\gamma$, $\alpha$) values are used. According to the graphics, it is seen that the convergence speed of aABC algorithm is better than the standard ABC algorithm for all examples.

By using Wilcoxon signed rank test, the performances of aABC and ABC algorithms are compared over ANFIS

training. Related to the comparison results, the average difference and p-Value are given in Table 8. As can be seen from Table 8, p-Values achived for training results in all examples is 0.000. The p-Values obtained for testing results in Example 1–4 are found successively as 0.001, 0.001, 0.000 and 0.000. Examining the obtained p-Values, a significant difference is found between aABC algorithm and ABC algorithm due to the fact

**Table 8**
Wilcoxon signed rank test results for nonlinear dynamic systems.

| Example | Train | | Test | |
|---|---|---|---|---|
| | Mean Difference | p-Value | Mean Difference | p-Value |
| 1 | 0.0042 | 0.000 | 0.0026 | 0.001 |
| 2 | 0.0011 | 0.000 | 0.0034 | 0.001 |
| 3 | 0.0072 | 0.000 | 0.0073 | 0.000 |
| 4 | 0.0089 | 0.000 | 0.0140 | 0.000 |

that $p<0.05$. Accordingly, in the ANFIS training conducted to identify the nonlinear systems, it is seen that aABC algorithm is more successful than ABC algorithm.

The results achieved are compared with the different neuro-fuzzy based methods taken from [88,94] in Table 9. Additionally, the results provided with the usage of ABC algorithm are also given in this table. In the Example 1, the best training error value is achieved with RSEFNN-LF method.The best testing error value is found with the usage of aABC algorithm. Moreover, the least parameter is used in aABC algorithms for the Example 1. In the Example 2, the best training error value is achieved with TRFN-S and the best testing error value is found with aABC algorithm. As In Example-1, the least parameter is used in aABC algorithm for the Example 2. In the Example 3, the best training error value is obtained with the GDFNN method. The best error value other than GDFNN is found with the aABC algorithm. The best testing error value is achieved with the usage of aABC algorithm. In the Example-4, the best training error value is obtained with aABC algorithm. The best testing error value is reached with the RSEFNN-LF method. The best testing error value other than RSEFNN-LF is found with the aABC algorithm. It can be seen that the results which are provided with aABC algorithm are better than ABC algorithm for all the examples. Along with the fact that it changes according to the examples; with the usage of aABC algorithm, a performance increase between 4.51% and 33.33% is provided compared to the ABC algorithm. Accordingly, for the training error values of the Example 1–4 a performance increase respectively with the rates of 10.88%, 4.51%, 32.14% and 30.27% is provided. As for the testing error values, a performance increase successively with the rates of 9.25%, 14.40%, 33.33% and 32.18% is achived.

The results obtained with the aABC algorithm for 2000 iterations are compared to different methods in Table 9. aABC algorithm finds more effective results than ABC algorithm in 2000 iterations. Graphics of convergence in the lower iteration are given in Fig. 3 and 4. Even though aABC algorithm appears to be more successful

**Table 9**
Comparison of the proposed algorithm with other algorithms [88,94].

| Algorithm | Example-1 | | | Example-2 | | | Example-3 | | | Example-4 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Number of Parameters | RMSE Train | RMSE Test | Number of Parameters | RMSE Train | RMSE Test | Number of Parameters | RMSE Train | RMSE Test | Number of Parameters | RMSE Train | RMSE Test |
| RSONFIN | 36 | 0.0248 | 0.078 | 36 | 0.03 | 0.06 | – | – | – | – | – | – |
| RFNN | 112 | 0.114 | 0.0575 | 49 | 0.072 | 0.128 | – | – | – | 60 | 0.463 | 0.469 |
| TRFN-S | 33 | 0.0084 | 0.0346 | 33 | **0.0067** | 0.0313 | – | – | – | 66 | 0.0362 | 0.032 |
| DFNN | 39 | – | 0.05 | – | – | – | 48 | 0.0283 | – | – | – | – |
| HO-RNFS | 45 | 0.0542 | 0.0815 | – | – | – | – | – | – | – | – | – |
| RSEFNN-LF (zero) | 34 | 0.0246 | 0.03 | 28 | 0.0221 | 0.0383 | – | – | – | 48 | 0.0568 | 0.0409 |
| RSEFNN-LF (first) | 32 | **0.0199** | 0.0397 | 30 | 0.0156 | 0.0279 | – | – | – | 35 | 0.0320 | **0.0209** |
| WRFNN | – | – | – | 55 | 0.0574 | 0.083 | – | – | – | 70 | 0.191 | 0.188 |
| RBF-AFS | – | – | – | – | – | – | 208 | 0.1384 | – | – | – | – |
| OLS | – | – | – | – | – | – | 326 | 0.0288 | – | – | – | – |
| GDFNN | – | – | – | – | – | – | 56 | **0.0108** | – | – | – | – |
| FAOS-PFNN | – | – | – | – | – | – | 25 | 0.0252 | – | – | – | – |
| GOSFNN | – | – | – | – | – | – | 72 | 0.0228 | – | – | – | – |
| ANFIS (ABC) | 24 | 0.0386 | 0.0281 | 24 | 0.0244 | 0.0236 | 50 | 0.0224 | 0.0219 | 72 | 0.0294 | 0.0435 |
| ANFIS (Proposed aABC) | 24 | 0.0344 | **0.0255** | 24 | 0.0232 | **0.0202** | 50 | 0.0152 | **0.0146** | 72 | **0.0205** | 0.0295 |

The best results is given bold.

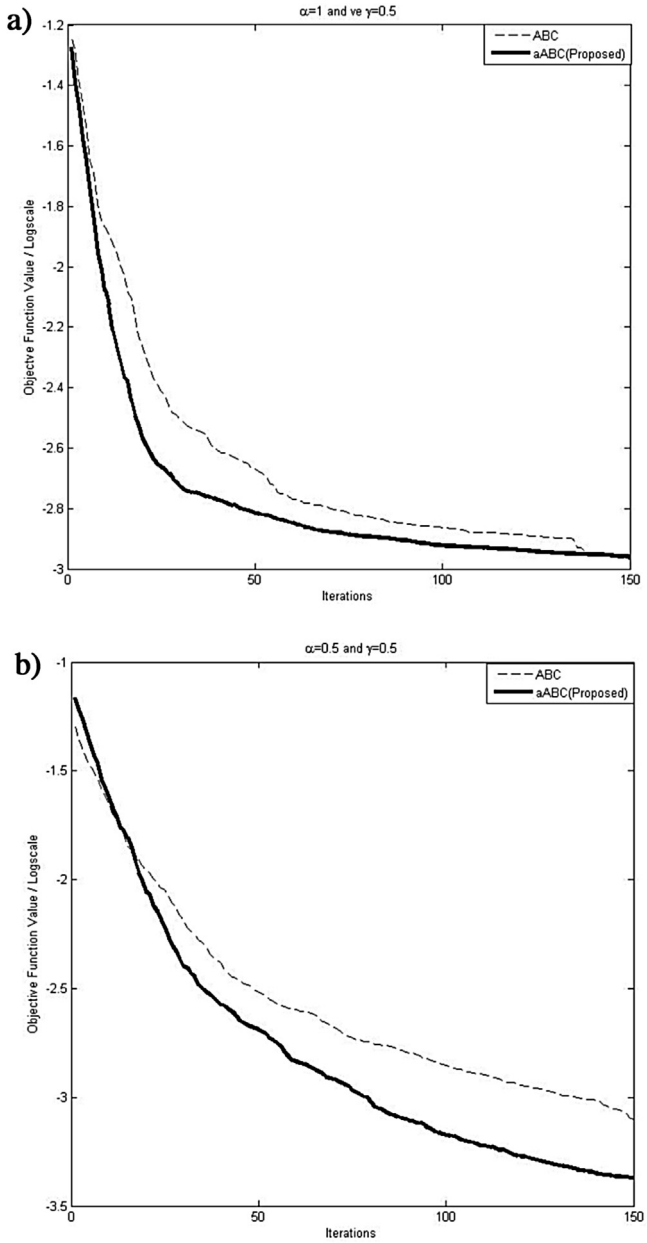


**Fig. 3.** Comparison of ABC and aABC algorithms' convergence on a) Example-1 b) Example-2.

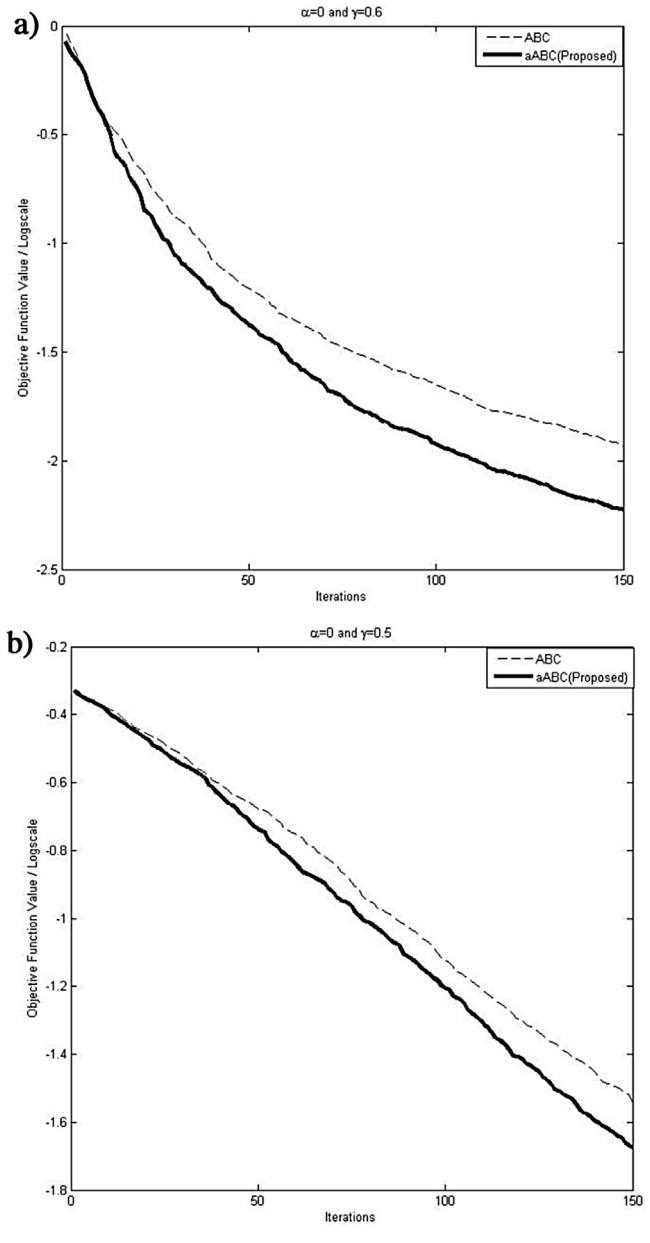


**Fig. 4.** Comparison of ABC and aABC algorithms' convergence on a) Example-3 b) Example-4.

**Table 10**
Comparison of the obtained results aABC and ABC algorithm in training ANFIS (the number of iterations = 100).

| Example | ABC | | aABC (Proposed) | | p Value | |
|---|---|---|---|---|---|---|
| | Train | Test | Train | Test | Train | Test |
| 1 | 0.057145 | 0.059116 | **0.053800** | **0.040086** | 0.000 | 0.000 |
| 2 | 0.051785 | 0.045051 | **0.038686** | **0.034339** | 0.000 | 0.000 |
| 3 | 0.175705 | 0.175071 | **0.154403** | **0.154511** | 0.020 | 0.021 |
| 4 | 0.330142 | 0.293924 | **0.282344** | **0.253701** | 0.032 | 0.045 |

The best results is given bold.

on the graphics, analysing error values of convergence graphics is required. Therefore, the results obtained by using aABC and ABC algorithm in 100 iterations are presented in Table 10. Besides, p-Values are calculated by utilizing Wilcoxon signed rank test and it is shown in this table. When the table is analyzed, it can be seen that the train and test error values by using aABC algorithm is better than ABC algorithm in 100 iterations. Additionally, p-Values are lower than 0.05, which shows that convergence graphics are significant. Namely, aABC algorithm converges faster than ABC algorithm in training ANFIS. When all tables and figures given for training ANFIS are considered together, more successful results with aABC algorithm are found. In obtaining these results, adaptive and hybrid structures of aABC algorithm are effective.

The different($\alpha$, $\gamma$) couples are effective for increasing the convergence speed or for reaching the optimum solution in ANFIS training conducted with aABC algorithm. In general, it is observed that the convergence decreases as the value of $\gamma$ gets closer to 1 and convergence increases as it gets closer to 0.5. With the addition of $\gamma$ parameter, better solutions are provided with lower iterations. In cases that the value of $\alpha$ parameter is lower, the influence of the $\gamma$ parameter can be observed more clearly. The reason is that as the value of $\alpha$ increases, the magnitude of $\Phi_{ij}$ the random number decreases and the influence of arithmetic crossover abolishs. Furthermore, as the value of $\gamma$ parameter gets closer to 1 it goes back to the solution search equation of standard ABC algorithm. In this case, only the influence of $\alpha$ parameter can be observed. Only with the usage of $\alpha$ parameter, it is enabled that better solutions can be achieved with lower step magnitude. Under the light of what is explained here; it can be seen that the best convergence in ANFIS training is generally found when $\alpha <= 1.5$ and $\gamma <= 0.6$. Moreover, examining the results of Wilcoxon signed rank test and the comparison results given in Table 9, it is seen that aABC algorithm is much more efficient than standard ABC algorithm in terms of reaching the optimum solution.

## 5. Conclusion

In this study, the standard ABC algorithm is modified for training ANFIS and a variant of ABC named adaptive and hybrid ABC −aABC- is suggested. The adaptivity value which is developed basing on failure counter and the arithmetic crossover is added to the solution generating mechanism of aABC algorithm. In this manner, two new control parameters are adapted as crossover rate ($\gamma$) and adaptivity coefficient ($\alpha$). The success of the suggested variant of ABC is tested on benchmark unconstrained numerical function and ANFIS training to identify nonlinear dynamical systems. The results are compared with other methods existing in the literature. In case that $\gamma$ and $\alpha$ parameters are settled properly on both application groups, it is seen that the convergence speed increases and better solutions are provided.

## References


[1] J.S. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybern. 23 (3) (1993) 665–685.
[2] J.P.S. Catalão, H.M.I. Pousinho, V.M.F. Mendes, Hybrid Wavelet-PSO-ANFIS approach for short-term electricity prices forecasting, IEEE Trans. Power Syst. 26 (1) (2011) 137–144.
[3] A. Chatterjee, K. Watanabe, An optimized Takagi-Sugeno type neuro-fuzzy system for modeling robot manipulators, Neural Comput. Appl. 15 (1) (2006) 55–61.
[4] A. Sarkheyli, A.M. Zain, S. Sharif, Robust optimization of ANFIS based on a new modified GA, Neurocomputing 166 (2015) 357–366.
[5] L.Y. Wei, A GA-weighted ANFIS model based on multiple stock market volatility causality for TAIEX forecasting, Appl. Soft Comput. 13 (2) (2013) 911–920.
[6] P. Liu, W. Leng, W. Fang, Training ANFIS model with an improved quantum-behaved particle swarm optimization algorithm, Math. Probl. Eng. 2013 (2013) (2013).
[7] J.P.S. Catalão, H.M.I. Pousinho, V.M.F. Mendes, Hybrid Wavelet-PSO-ANFIS approach for short-term electricity prices forecasting, IEEE Trans. Power Syst. 26 (1) (2011) 137–144.
[8] S. Suja Priyadharsini, S. Edward Rajan, S. Femilin Sheniha, A novel approach for the elimination of artefacts from EEG signals employing an improved Artificial Immune System algorithm, J. Exp. Theor. Artif. Intell. (2015) 1–21.
[9] M. Gunasekaran, K.S. Ramaswami, A fusion model integrating ANFIS and artificial immune algorithm for forecasting indian stock market, J. Appl. Sci. 11 (16) (2011) 3028–3033.
[10] M. Asadollahi-Baboli, In silico prediction of the aniline derivatives toxicities to Tetrahymena pyriformis using chemometrics tools, Toxicol. Environ. Chem. 94 (10) (2012) 2019–2034.
[11] R. Teimouri, H. Baseri, Optimization of magnetic field assisted EDM using the continuous ACO algorithm, Appl. Soft Comput. 14 (2014) 381–389.
[12] G.S. Varshini, S.C. Raja, P. Venkatesh, Design of ANFIS controller for power system stability enhancement using FACTS device, in: Power Electronics and Renewable Energy Systems, Springer, India, 2015, pp. 1163–1171.
[13] M. Bhavani, K. Selvi, L. Sindhumathi, Neuro fuzzy load frequency control in a competitive electricity market using BFOA tuned SMES and TCPS, in: Swarm, Evolutionary, and Memetic Computing, Springer International Publishing, 2014, pp. 373–385.
[14] K. Zarei, M. Atabati, K. Kor, Bee algorithm and adaptive neuro-fuzzy inference system as tools for QSAR study toxicity of substituted benzenes to Tetrahymena pyriformis, Bull. Environ. Contam. Toxicol. 92 (6) (2014) 642–649.
[15] M. Azarbad, H. Azami, S. Sanei, A. Ebrahimzadeh, New neural network-based approaches for GPS GDOP classification based on neuro-fuzzy inference system, radial basis function, and improved bee algorithm, Appl. Soft Comput. 25 (2014) 285–292.
[16] M.A.F. Rani, B. Sankaragomathi, Performance enhancement of PID controllers by modern optimization techniques for speed control of PMBL DC motor, Res. J. Appl. Sci. Eng. Technol. 10 (10) (2015) 1154–1163.
[17] J.F. Chen, Q.H. Do, A cooperative cuckoo search–hierarchical adaptive neuro-fuzzy inference system approach for predicting student academic performance, J. Intell. Fuzzy Syst. 27 (5) (2014) 2551–2561.
[18] S.M. Khazraee, A.H. Jahanmiri, S.A. Ghorayshi, Model reduction and optimization of reactive batch distillation based on the adaptive neuro-fuzzy inference system and differential evolution, Neural Comput. Appl. 20 (2) (2011) 239–248.
[19] A.Z. Zangeneh, M. Mansouri, M. Teshnehlab, A.K. Sedigh, Training ANFIS system with DE algorithm, in: Advanced Computational Intelligence (IWACI), 2011 Fourth International Workshop on, IEEE, October, 2011, pp. 308–314.
[20] J. Wang, Gao, X.Z. Tanskanen, J.M.P. Guo, Epileptic EEG signal classification with ANFIS based on harmony search method, in: Computational Intelligence and Security (CIS), 2012 Eighth International Conference on, IEEE, November, 2012, pp. 690–694.
[21] O.F. Lutfy, M. Noor, S. Bahari, M.H. Marhaban, K.A. Abbas, Utilizing global-best harmony search to train a PID-like ANFIS controller, Aust. J. Basic Appl. Sci. 4 (12) (2010) 6319–6330.
[22] D. Karaboga, E. Kaya, Training ANFIS using artificial bee colony algorithm, 2013 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) (2013) 1–5.
[23] D. Karaboga, E. Kaya, Training ANFIS using artificial bee colony algorithm for nonlinear dynamic systems identification, Signal Processing and Communications Applications Conference (SIU) (2014) 493–496.
[24] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
[25] D. Karaboga, Artificial bee colony algorithm, Scholarpedia 5 (3) (2010) 6915.
[26] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471.
[27] D. Karaboga, C. Ozturk, Neural networks training by artificial bee colony algorithm on pattern classification, Neural Netw. World 19 (3) (2009) 279–292.
[28] W.Y. Szeto, Y. Wu, S.C. Ho, An artificial bee colony algorithm for the capacitated vehicle routing problem, Eur. J. Oper. Res. 215 (1) (2011) 126–135.
[29] R.V. Rao, P.J. Pawar, Modelling and optimization of process parameters of wire electrical discharge machining, Proc. Inst. Mech. Eng. B J. Eng. Manuf. 223 (11) (2009) 1431–1440.

[30] A. Yousefi-Talouki, S.A. Gholamian, M. Hosseini, S. Valiollahi, Optimal power flow with unified power flow controller using artificial bee colony algorithm, Int. Rev. Electr. Eng. 5 (6) (2010) 2773–2778.
[31] N. Karaboga, A new design method based on artificial bee colony algorithm for digital iir filters, J. Franklin Inst. 346 (4) (2009) 328–348.
[32] D. Karaboga, B. Akay, Proportional-integral-derivative controller design by using artificial bee colony, harmony search, and the bees algorithms, Proc. Inst. Mech. Eng. H 224 (I7) (2010) 869–883.
[33] S.K. Mandal, F.T.S. Chan, M.K. Tiwari, Leak detection of pipeline: an integrated approach of rough set theory and artificial bee colony trained svm, Expert Syst. Appl. 39 (3) (2012) 3071–3080.
[34] D.J. Mala, V. Mohan, M. Kamalapriya, Automated software test optimisation framework—an artificial bee colony optimisation-based approach, IET Softw. 4 (5) (2010) 334–348.
[35] M.H. Horng, Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation, Expert Syst. Appl. 38 (11) (2011) 13785–13791.
[36] D. Karaboga, C. Ozturk, A novel clustering approach: artificial bee colony (abc) algorithm, Appl. Soft Comput. 11 (1) (2011) 652–657.
[37] C. Ozturk, D. Karaboga, B. Gorkemli, Artificial bee colony algorithm for dynamic deployment of wireless sensor networks, Turk. J. Electr. Eng. Comput. Sci. 20 (2) (2012) 1–8.
[38] H.A.A. Bahamish, R. Abdullah, R.A. Salam, Protein tertiary structure prediction using artificial bee colony algorithm, 2009 Third Asia International Conference on Modelling and Simulation 1 and 2 (2009) 258–263.
[39] J. Zhang, M. Dolg, Global optimization of clusters of rigid molecules using the artificial bee colony algorithm, Phys. Chem. Chem. Phys. 18 (4) (2016) 3003–3010.
[40] S. Santander-Jimenez, M.A. Vega-Rodriguez, Performance analysis of multiobjective artificial bee colony implementations for phylogenetic reconstruction, in: Nature and Biologically Inspired Computing (NaBIC), 2014 Sixth World Congress on, IEEE, July, 2014, pp. 35–40.
[41] S. Uehara, K.J. Fujimoto, S. Tanaka, Protein–ligand docking using fitness learning-based artificial bee colony with proximity stimuli, Phys. Chem. Chem. Phys. 17 (25) (2015) 16412–16417.
[42] P.W. Tang, Y.W. Choon, M.S. Mohamad, S. Deris, S. Napis, Optimising the production of succinate and lactate in *Escherichia coli* using a hybrid of artificial bee colony algorithm and minimisation of metabolic adjustment, J. Biosci. Bioeng. 119 (3) (2015) 363–368.
[43] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, The best-so-far selection in artificial bee colony algorithm, Appl. Soft Comput. 11 (2) (2011) 2888–2901.
[44] F. Kang, J. Li, Z. Ma, Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, Inform. Sci. 181 (16) (2011) 3508–3531.
[45] L. Dos Santos Coelho, P. Alotto, Gaussian artificial bee colony algorithm approach applied to Loney's solenoid benchmark problem, IEEE Trans. Magn. 47 (5) (2011) 1326–1329.
[46] A. Rajasekhar, A. Abraham, M. Pant, Levy mutated artificial bee colony algorithm for global optimization, IEEE International Conference on Systems Man and Cybernetics (2011) 655–662.
[47] W.F. Gao, S.Y. Liu, F. Jiang, An improved artificial bee colony algorithm for directing orbits of chaotic systems, Appl. Math. Comput. 218 (7) (2011) 3868–3879.
[48] B. Babayigit, R. Ozdemir, A modified artificial bee colony algorithm for numerical function optimization, in: Computers and Communications (ISCC), 2012 IEEE Symposium on, IEEE, July, 2012, pp. 000245–000249.
[49] B. Akay, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Inform. Sci. 192 (2012) 120–142.
[50] M. El-Abd, Generalized opposition-based artificial bee colony algorithm, in: Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE, June, 2012, pp. 1–4.
[51] A.G. Abro, J. Mohamad-Saleh, Intelligent scout-bee based artificial bee colony optimization algorithm, in: Control System, Computing and Engineering (ICCSCE), 2012 IEEE International Conference on, IEEE, November, 2012, pp. 380–385.
[52] Z.G. Su, P.H. Wang, J. Shen, Y.G. Li, Y.F. Zhang, E.J. Hu, Automatic fuzzy partitioning approach using Variable string length Artificial Bee Colony (VABC) algorithm, Appl. Soft Comput. 12 (11) (2012) 3421–3441.
[53] W.F. Gao, S.Y. Liu, L.L. Huang, A novel artificial bee colony algorithm with Powell's method, Appl. Soft Comput. 13 (9) (2013) 3763–3775.
[54] W.L. Xiang, M.Q. An, An efficient and robust artificial bee colony algorithm for numerical optimization, Comput. Oper. Res. 40 (5) (2013) 1256–1265.
[55] H. Sun, H. Luş, R. Betti, Identification of structural models using a modified Artificial Bee Colony algorithm, Comput. Struct. 116 (2013) 59–74.
[56] X. Zhang, S.Y.1 Yuen, Improving artificial bee colony with one-position inheritance mechanism, Memetic Comput. 5 (3) (2013) 187–211.
[57] H. Shayeghi, A. Ghasemi, A modified artificial bee colony based on chaos theory for solving non-convex emission/economic dispatch, Energy Convers. Manage. 79 (2014) 344–354.
[58] D. Karaboga, B. Gorkemli, A quick artificial bee colony (qABC) algorithm and its performance on optimization problems, Appl. Comput. 23 (2014) 227–238.
[59] M.S. Alam, M.M. Islam, Artificial bee colony algorithm with self-adaptive mutation: a novel approach for numeric optimization, in: TENCON 2011-2011 IEEE Region 10 Conference, IEEE, November, 2011, pp. 49–53.
[60] S.S. Hasan, F. Ahmed, Balaning explorations with exploitations in the artificial bee colony algorithm for numerical function optimization, Int. J. Appl. Inform. Syst. 9 (1) (2015) 42–48.
[61] W. Gu, M. Yin, C. Wang, Self adaptive artificial bee colony for global numerical optimization, IERI Procedia 1 (2012) 59–65.
[62] X. Liao, J. Zhou, R. Zhang, Y. Zhang, An adaptive artificial bee colony algorithm for long-term economic dispatch in cascaded hydropower systems, Int. J. Electr. Power Energy Syst. 43 (1) (2012) 1340–1345.
[63] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, Appl. Math. Comput. 217 (7) (2010) 3166–3173.
[64] E. Mezura-Montes, R.E. Velez-Koeppel, Elitist artificial bee colony for constrained real-parameter optimization, in: Evolutionary Computation (CEC), 2010 IEEE Congress on, IEEE, July, 2010, pp. 1–8.
[65] Y. Zhang, P. Zeng, Y. Wang, B. Zhu, F. Kuang, Linear weighted Gbest-guided artificial bee colony algorithm, IEEE, October, in: Computational Intelligence and Design (ISCID), 2012 Fifth International Symposium on, 2, 2012, pp. 155–159.
[66] P.N. Nagur, S. Raj, H.T. Jadhav, Modified Artificial Bee Colony algorithm for non-convex economic dispatch problems, in: Green Technologies (ICGT), 2012 International Conference on, IEEE, December, 2012, pp. 258–262.
[67] J. Luo, Q. Wang, X. Xiao, A modified artificial bee colony algorithm based on converge-onlookers approach for global optimization, Appl. Math. Comput. 219 (20) (2013) 10253–10262.
[68] W. Gao, S. Liu, Improved artificial bee colony algorithm for global optimization, Inform. Process. Lett. 111 (17) (2011) 871–882.
[69] W.F. Gao, S.Y.1 Liu, A modified artificial bee colony algorithm, Comput. Oper. Res. 39 (3) (2012) 687–697.
[70] W. Gao, S. Liu, L. Huang, A global best artificial bee colony algorithm for global optimization, J. Comput. Appl. Math. 236 (11) (2012) 2741–2753.
[71] B. Wang, L. Wang, A novel artificial bee colony algorithm for numerical function optimization, in: Computational and Information Sciences (ICCIS), 2012 Fourth International Conference on, IEEE, August, 2012, pp. 172–175.
[72] A. Rubio-Largo, D.L. Gonzalez-Alvarez, M.A. Vega-Rodríguez, J.A. Gomez-Pulido, J.M. Sanchez-Perez, MO-ABC/DE-multiobjective artificial bee colony with differential evolution for unconstrained multiobjective optimization, in: Computational Intelligence and Informatics (CINTI), 2012 IEEE 13th International Symposium on, IEEE, November, 2012, pp. 157–162.
[73] X. Li, M. Yin, Hybrid differential evolution with artificial bee colony and its application for design of a reconfigurable antenna array with discrete phase shifters, IET Microwaves Antennas Propag. 6 (14) (2012) 1573–1582.
[74] A.N. Hati, R. Darbar, N.D. Jana, J. Sil, Modified Artificial Bee Colony Algorithm using differential evolution and polynomial mutation for real-parameter optimization, in: Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on, IEEE, August, 2013, pp. 534–539.
[75] H. Wang, Z. Wu, X. Zhou, S. Rahnamayan, Accelerating artificial bee colony algorithm by using an external archive, in: Evolutionary Computation (CEC), 2013 IEEE Congress on, IEEE, June, 2013, pp. 517–521.
[76] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, Evol. Comput. IEEE Trans. 13 (5) (2009) 945–958.
[77] W.F. Gao, S.Y. Liu, L.L. Huang, Enhancing artificial bee colony algorithm using more information-based search equations, Inform. Sci. 270 (2014) 112–133.
[78] P.W. TSai, J.S. Pan, B.Y. Liao, S.C. Chu, Enhanced artificial bee colony optimization, Int. J. Innovative Comput. Inform. Control 5 (12) (2009) 5081–5092.
[79] I. Fister, I. Fister Jr, J.B. Zumer, Memetic artificial bee colony algorithm for large-scale global optimization, in: Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE, June, 2012, pp. 1–8.
[80] S.M. Chen, A. Sarosh, Y.F. Dong, Simulated annealing based artificial bee colony algorithm for global numerical optimization, Appl. Math. Comput. 219 (8) (2012) 3575–3589.
[81] W. Zhang, N. Wang, S. Yang, Hybrid artificial bee colony algorithm for parameter estimation of proton exchange membrane fuel cell, Int. J. Hydrogen Energy 38 (14) (2013) 5796–5806.
[82] S. García, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization, J. Heuristics 15 (6) (2009) 617–644.
[83] C.F. Juang, C.T. Lin, A recurrent self-organizing neural fuzzy inference network, IEEE Trans. Neural Netw. 10 (4) (1999) 828–845.
[84] C.H. Lee, C.C. Teng, Identification and control of dynamic systems using recurrent fuzzy neural networks, IEEE Trans. Fuzzy Syst. 8 (4) (2000) 349–366.
[85] C.F. Juang, A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms, IEEE Trans. Fuzzy Syst. 10 (2) (2002) 155–170.
[86] P.A. Mastorocostas, J.B. Theocharis, A recurrent fuzzy-neural model for dynamic system identification, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 32 (2) (2002) 176–190.
[87] J.B. Theocharis, A high-order recurrent neuro-fuzzy system with internal dynamics: application to the adaptive noise cancellation, Fuzzy Sets Syst. 157 (4) (2006) 471–500.
[88] C.F. Juang, Y.Y. Lin, C.C. Tu, A recurrent self-evolving fuzzy neural network with local feedbacks and its application to dynamic system processing, Fuzzy Sets Syst. 161 (19) (2010) 2552–2568.
[89] C.J. Lin, C.C. Chin, Prediction and identification using wavelet-based recurrent fuzzy neural networks, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 34 (5) (2004) 2144–2154.

[90] K.B. Cho, B.H. Wang, Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction, Fuzzy Sets and Syst. 83 (3) (1996) 325–339.
[91] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Trans. Neural Netw. 2 (2) (1991) 302–309.
[92] S. Wu, M.J. Er, Y. Gao, A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks, IEEE Trans. Fuzzy Syst. 9 (4) (2001) 578–594.
[93] N. Wang, M.J. Er, X. Meng, A fast and accurate online self-organizing scheme for parsimonious fuzzy neural networks, Neurocomputing 72 (16) (2009) 3818–3829.
[94] W. Ning, T. Yue, L. Shao-Man, A generalized online self-organizing fuzzy neural network for nonlinear dynamic system identification, 30th Chinese Control Conference (2011).