Neural Computing and Applications (2020) 32:3803–3824
https://doi.org/10.1007/s00521-018-3871-9

ORIGINAL ARTICLE

A novel improved antlion optimizer algorithm and its comparative performance

Haydar Kilic[1] · Ugur Yuzgec[1] · Cihan Karakuzu[1]

Received: 17 February 2018 / Accepted: 9 November 2018 / Published online: 21 November 2018

Abstract
In this study, the improvement of the ant lion optimization which is inspired by ant lion's hunting strategy is dealt with. The most disadvantageous property of this algorithm is its having a long run time due to the random walking process. In order to overcome this drawback, we proposed the improved random walking model, tournament selection method instead of the roulette wheel selection method, and reproduction mechanism at the boundary values. The performance of improved ant lion optimization algorithm based on the tournament selection (IALOT) is evaluated in comparison with the commonly known and used heuristic algorithms for ten benchmark functions. Furthermore, we have tested the performance of IALOT on the training of ANFIS known as a difficult optimization problem. The benchmark and ANFIS test results show that IALOT algorithm exhibits better performance than that of the ALO algorithm.

Keywords Tournament selection · Heuristic optimization · Improved antlion · ANFIS

1 Introduction

Different methods have been used in solution of optimization problems for many years. One of these is to solve the problem with heuristic optimization algorithms. In recent years, heuristic algorithms, inspired by the nature, have an important role in solving optimization problems. In these algorithms, the animals' behaviors are generally examined and the perfect behavior mechanisms such as hunting, feeding and mating lead scientists to propose such algorithms.

For the heuristic algorithms classification, there are a number of classification criteria in the literature. According to these criteria, heuristic algorithms are divided into two main groups: single-solution based algorithms known as trajectory methods and population-based algorithms. Simulated Annealing (SA) algorithm is one of the best-known examples of the single-solution based heuristic algorithms. SA algorithm was presented by Kirkpatrick et al. in 1983, then was reviewed by Rutenbar in 1989. It was based on the principle that the crystals are crystallized by heating and then they are slowly cooled. The temperature value is an important factor in searching the global solution [24, 37]. Some well-known examples of the population-based algorithms are the Differential Evolution (DE) algorithm, Particle Swarm Optimization (PSO) algorithm, Artificial Bee Colony (ABC) algorithm and Ant Colony Optimization (ACO) algorithm. Differential Evolution (DE) algorithm was developed by Price and Storn in 1995. This algorithm is based on adding the difference value between two individuals to a third individual in the population [43, 44]. The Particle Swarm Optimization (PSO) algorithm is another population-based heuristic optimization technique developed by Kennedy in 1995. The original idea of this algorithm is based on the principle that animals living in flocks, such as birds and fishes, share information by finding rich food and avoiding hunting [23]. The Touring Ant Colony Optimization (TACO) algorithm, proposed by Hiroyasu et al., in 2000, is an updated version of Dorigo's Ant Colony Optimization (ACO) algorithm. In the TACO's basic idea, ants use only the pheromone information and search for each bit in the candidate solution string of binary bits [11, 14]. The Artificial Bee Colony (ABC) algorithm was presented by Karaboga in 2005. It is an heuristic algorithm inspired by foraging behaviors

✉ Ugur Yuzgec
ugur.yuzgec@bilecik.edu.tr

[1] Department of Computer Engineering, Bilecik Seyh Edebali University, Bilecik, Turkey

of bees [18]. Besides these algorithms, there are a lot of studies about improvement of heuristic algorithms and their implementation on the different application areas in the literature and such studies will certainly come in the future. A search space-based evolutionary algorithm was proposed by Medhane and Sangaiah [26] for multiobjective optimization problems. Abdel-Basset et al. [2] presented application of a novel modified version of whale optimization algorithm for Merkle–Hellman Knapsack Cryptosystem. In [3], a novel improved version of the whale optimization algorithm was proposed for solving 0–1 knapsack problem. This improved algorithm is based on two basic strategies: local search strategy and the Lévy flight walks. Srikanth et al. [42] presented a quantum inspired binary grey wolf optimizer (QI-BGWO) based on the combination the hunting behavior of grey wolf and principles of quantum computing for unit commitment problem. For the multi-trip capacitated arc routing problem, a hybrid algorithm was proposed by Tirkolaee et al. [45]. The proposed hybrid algorithm is an efficient ant colony optimizer based on max–min ant system. In the study of Abdel-Basset et al. [1], a modified version of the flower pollination algorithm was developed for the multidimensional knapsack problems.

In this study, Antlion Optimizer Algorithm (ALO), inspired by the hunting behaviors of ant lions and presented in 2015 by Mirjalili [27], is examined. In the ALO algorithm, the interactions between antlions and ants are imitated basically. To model these interactions, five main steps are realized such as random walking of ants, building trap, trapping in the antlion's pits, sliding ants toward antlion, and catching the prey and rebuilding the pit. A number of studies, including antlion optimization algorithm for engineering problems reported in the literature chronologically: Raju et al. [34] proposed the automatic generation control (AGC) of an unequal three area thermal system. ALO algorithm was used for determining gains of the controller such as I, PI, PID and PID + DD controllers. Kamboj et al. [17] presented the implementation of ALO algorithm to the dynamic economic load dispatch problem of electric power system. Petrovic et al. [33] used ALO algorithm for optimization of the process planning, and the performance of ALO algorithm was compared with the other bio-inspired algorithms. Nischal and Mehta [30] presented optimal load dispatch problem by using ALO algorithm. In [48], Yao and Wang proposed the Levy flight-based random walk for ALO algorithm. The proposed dynamic adaptive ALO algorithm was tested for finding the optimal route planning of unmanned aerial vehicle [48]. Chopra and Mehta performed ALO algorithm in multiobjective optimum generation scheduling problem in different test power systems [10]. In the study of Gupta and Saxena [12], the automatic generation control (AGC) of two-area interconnected power system was designed by ALO algorithm, and the comparison results of the regulator performances were obtained from the proposed ALO-, GA-, PSO-, GSA-based regulators. Babers et al. [5] carried out ALO algorithm to detect the number of communities in the networks. In [39], Satheeshkumar et al. presented ALO algorithm to adjust the gain parameters of PI controllers for load frequency control problem. Rebecca et al. [36] proposed the application of ALO algorithm to solve optimal reactive power dispatch problem. Tung and Chakravorty presented the implementation of ALO algorithm on electrical economic power dispatch planning problem [47]. In [46], Trivedi et al. compared the performance of ALO algorithm with PSO and FA algorithms for solving the optimal power flow problem on a IEEE 30 bus system. Nair et al. [28] proposed ALO algorithm based system identification for determining the optimal coefficients of IIR filters with different orders.

Although ALO gives effective results for different optimization on engineering problems, it has some disadvantages. The greatest drawback of ALO algorithm is that it has got long run time due to the random ant walking model especially. In improvement studies about ALO algorithm in here, the random walking distance for modeling ant's movement is changed. The distance of the random walk is determined as twenty percentage of maximum iteration rather than the maximum iteration number in the original ALO code. In Mirjalili's study [27], every ant randomly walks around an antlion selected by the roulette wheel method and the elite antlion simultaneously, but the roulette wheel method is more efficient method for maximization problems [35]. Because of this reason, tournament selection method which is more useful for minimization problems, is preferred in this study rather than the roulette wheel method. Besides these revisions on ALO algorithm, some movements between lower and upper boundaries around the ant lion in the phase of trapping on ant lion pits are added. These movements ensured that ants walk more effectively around the selected antlion in the search space. In the proposed improved ALO algorithm based on the tournament selection (IALOT), the boundary checking process and the procedure about the catching prey and rebuilding the pit are improved.

The proposed algorithm is used in neuro-fuzzy system training of antecedent and consequent parameters. As it is known neuro-fuzzy systems (NFS) consist of different combinations of neural network and fuzzy logic, that was proposed by Jang [15] in 1993. The idea behind the neuro-fuzzy system is that a fuzzy system is trained by a learning algorithm derived from neural network. These systems are capable of modeling nonlinear complex correlation between input and output of a system [9]. The neuro-fuzzy systems are used in many sectors in social and technological life. The antecedent parameters of ANFIS were

optimized by PSO algorithm and the consequent parameters of ANFIS were tuned by extended Kalman filter in Shoorehdeli et al. work [40, 41]. Carrano et al. [8] proposed genetic algorithm for the multiobjective training of anfis fuzzy networks. In [21], the parameter tuning method based on particle swarm optimization algorithm was proposed for fuzzy sliding mode controller. Zangeneh et al. [49] presented differential evolution algorithm to train the antecedent part parameters of ANFIS structure. In the paper of Jiang et al. [16], an ANFIS approach tuned by particle swarm optimization (PSO) was proposed for modeling customer satisfaction structure. In [20], Karaboga and Kaya presented the ABC algorithm for training ANFIS model and compared its performance with backpropagation and hybrid learning methods. Karakuzu examined performance of commonly known and used six heuristic algorithms on dynamic system identification problem based on NFS [22].

In this study, we propose the improved ALO algorithm via tournament selection method (IALOT) to obtain superior performance on run time. By the proposed IALOT algorithm, the long run time is decreased which is one of the handicaps of ALO algorithm. The performance of the proposed IALOT algorithm is compared with PSO, ABC, SA, DE, TACO and ALO algorithms on ten different benchmark functions taken from the literature. In addition, IALOT's performance is tested on the ANFIS-based dynamic system modeling problem. For this purpose, five dynamic benchmark systems are used. The rest of the paper is organized as follows: Sect. 2 presents the introduction of the classic ALO algorithm. The proposed IALOT algorithm and its novelty are provided in Sect. 3. In Sect. 4, ANFIS structure is presented briefly. Test results and the performance of the proposed IALOT algorithm are given in Sect. 5. In the last section, conclusion and the future works are drawn.

2 Ant lion optimizer (ALO)

The life stages of antlions are composed of two parts as larval and adulthood. Larval periods last about 2 years, while adult periods are as short as 6 months. Adult antlions resemble a beautiful dragonfly, but the larvae looks for a place with curved marks in the sand to form a death trap. The larva throws sand and small rocks until it forms a conical pit with increasingly shrinking round circles and buries itself under the trap to catch the prey as shown in Fig. 1. When the ants come into the antlion's trap, it throws sand at the ants to speed up their death, and the ants slip into the trap center with the fact that antlion forms a small landslide in the trap.

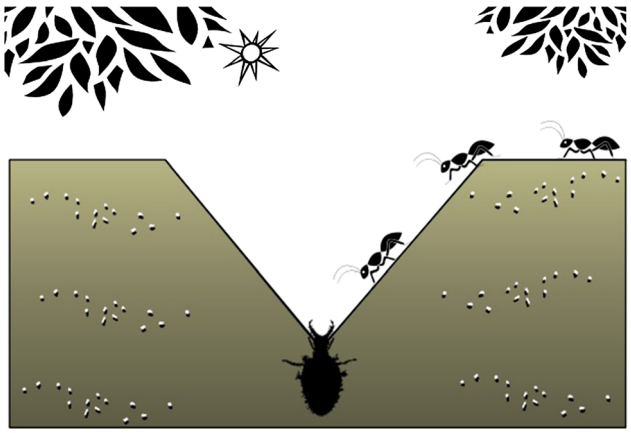


Fig. 1 Antlion's hunting strategy [27]

This little monster's method of hunting instincts actually works in a wonderful mathematical context. This hunting mechanism in the larval periods of antlions was transformed into an optimization algorithm by Seyedali Mirjalili and called Ant Lion Optimizer (ALO).

The mathematical model of antlion hunting starts with random walks in search space. The Eq. 1 is a mathematical model of random walks.

$$X(t) = [0, \text{cumsum}(2r(t_1) - 1), \text{cumsum}(2r(t_2) - 1), \ldots, \text{cumsum}(2r(t_n) - 1)] \quad (1)$$

where cumsum represents cumulative sum, n denotes the maximum number of iteration, $X(t)$ denotes the random walk at t step, and rand is the number in interval [0, 1], $r(t)$ refers the stochastic function as defined:

$$r(t) = \begin{cases} 1 & \text{if rand} > 0.5 \\ 0 & \text{if rand} \leq 0.5 \end{cases} \quad (2)$$

The random walkways described in this model are illustrated with five different random walkways in Fig. 2.

A max–min normalization is done as follows to make sure that the random walk is in the search space.

$$X_i^t = \frac{(X_i^t - a_i)(d_i^t - c_i^t)}{b_i - a_i} + c_i^t \quad (3)$$

where i is the variable index, t is the iteration number, a is the minimum random walk, b is the maximum random walk value, c is minimum variable value, and d is maximum variable value of antlion's position that is updated at each iteration as follows.

The ants that fall in the antlion's trap instinctively want to get out of there. The ant lion, begins to throw sand at the ants hindering their escape and to move them toward the bottom of the pit. Equations 4, 5, 6, 7 are the mathematical explanations of the antlions shifting the ants from a certain shift rate toward the bottom of the pit.

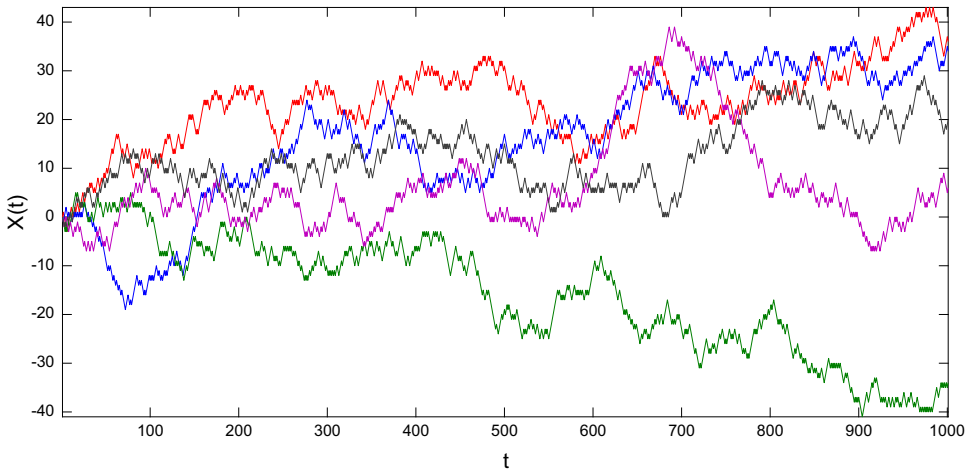


Fig. 2 Ants' five different random walkways, each color shows the random walk of each ant

$$c_i^t = \text{Antlion}^t + c^t \tag{4}$$

$$d_i^t = \text{Antlion}^t + d^t \tag{5}$$

$$c^t = \frac{c^t}{J} \tag{6}$$

$$d^t = \frac{d^t}{J} \tag{7}$$

Antlion^t is the position of the selected antlion at t-th iteration, J is the sliding ratio, value of J can be changed in different scenarios as follows,

$$J = \begin{cases} 1 + 10^2 \dfrac{t}{T_{\max}} & \text{if } 0.1\ T_{\max} < t < 0.5\ T_{\max} \\ 1 + 10^3 \dfrac{t}{T_{\max}} & \text{if } 0.5\ T_{\max} < t < 0.75\ T_{\max} \\ 1 + 10^4 \dfrac{t}{T_{\max}} & \text{if } 0.75\ T_{\max} < t < 0.9\ T_{\max} \\ 1 + 10^5 \dfrac{t}{T_{\max}} & \text{if } 0.9\ T_{\max} < t < 0.95\ T_{\max} \\ 1 + 10^6 \dfrac{t}{T_{\max}} & \text{if } 0.95\ T_{\max} < t < T_{\max} \\ 1 & \text{otherwise.} \end{cases} \tag{8}$$

where t is the current iteration, and $T_{\max}$ is the maximum iteration. According to the ants' random walking mechanism, ants walk around the antlion selected by roulette selection, and the elite antlion. The positions of the ants are updated at each iteration as in Eq. 9

$$\text{Ant}_i^t = \frac{R_{\text{A}}^t + R_{\text{E}}^t}{2} \tag{9}$$

where R_{A}^t represents the position of ant around the antlion chosen by roulette wheel, and R_{E}^t stands for ant's random walks around the elite antlion. After the antlion eats the fallen ant in its trap, antlion updates its position according to i-th ant at t-th iteration Ant_i^t by the following equation:

$$\text{Antlion}_j^t = \text{Ant}_i^t \quad \text{if} \quad f\left(\text{Ant}_i^t\right) < f\left(\text{Antlion}_j^t\right) \tag{10}$$

where f stands for the fitness function.

It is observed that the ALO algorithm given in the following pseudo-code [27] is a clumsy algorithm by comparison with other algorithms in this study. In order to shed light of future works, it was necessary to speed up the ALO algorithm and improve its performance. For this reason, detailed information about the innovations made can be found in the next section.

Original ant lion optimizer algorithm	
Step 1:	Initialize antlion's positions in the population
Step 2:	Calculate fitness values for each of antlion and sort them
Step 3:	Ant's random walking mechanism is started around the elite antlion and the antlion selected by roulette wheel in the search space
Step 4:	If the ants get better fitness, an antlion relocate its position
Step 5:	Update elite antlion
Step 6:	Until stop criterion satisfied repeat Step 2 to Step 6
Step 7:	Return elite antlion's position as global optimum point

3 Improved ant lion optimizer via tournament selection (IALOT)

3.1 Tournament selection

In heuristic algorithms, the selection procedures, such as roulette wheel method, truncation selection, linear ranking selection, exponential ranking selection and tournament selection method, are used for individual selection among a

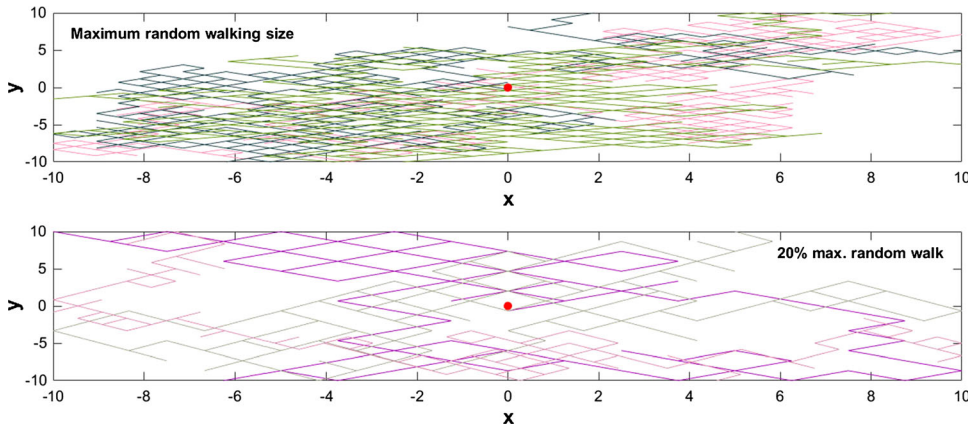


Fig. 3 Ants' random walk modeling for different random walking size values of ALO and IALOT algorithms

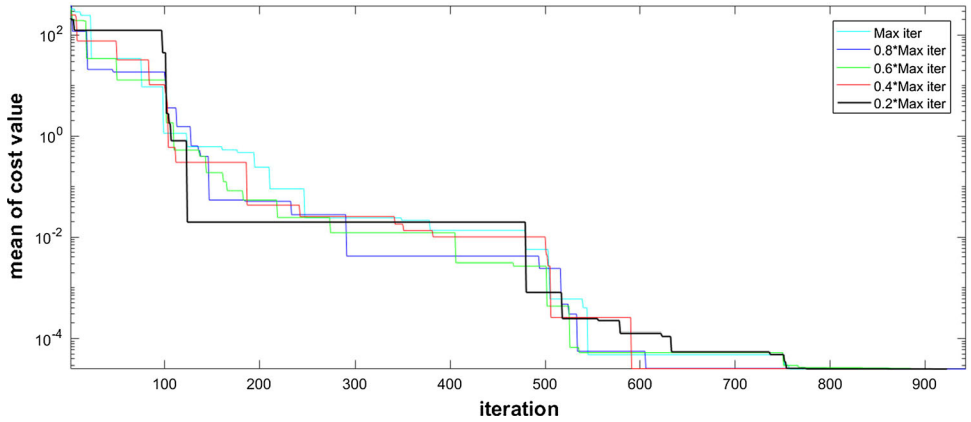


Fig. 4 Convergence curves with different random walking sizes for Schwefel function

Table 1 CPU time (s) values with different random walking sizes for Schwefel function

	Random walking size				
	MaxIt	0.8 × MaxIt	0.6 × MaxIt	0.4 × MaxIt	0.2 × MaxIt
CPU time (s)	9.661	8.541	8.122	7.179	6.887

population. Tournament selection is the most effective one of them [7]. This method arranges a tournament among randomly selected individuals from the population, and the best individual having the best fitness is the winner of the tournament. The parameter of tournament selection method is the tournament size which is known as "tour". Tour takes values ranging from 2 to number of population. While tournament size increases, selection intensity of the population increases. In this study, the tournament size is 2, so the tournament method randomly selects two groups from the population and the size of these groups is calculated by division of number of the population to tournament size.

In the roulette wheel selection method, fitness values of an individual form roulette wheel segments, and the probability of selecting the best individual are directly proportional to the size of the slice. Because of this reason, for finding the best individual through the tournaments including individuals from the population in the tournament would be a better choice instead of the probabilistic process on the roulette wheel [4]. In Mirjalili's work [27], the roulette wheel method was used for selecting the antlion, which every ant randomly walks around. But the roulette wheel method is not more efficient than the other method used for minimization problems [35]. Therefore, the tournament selection method, which is more useful in

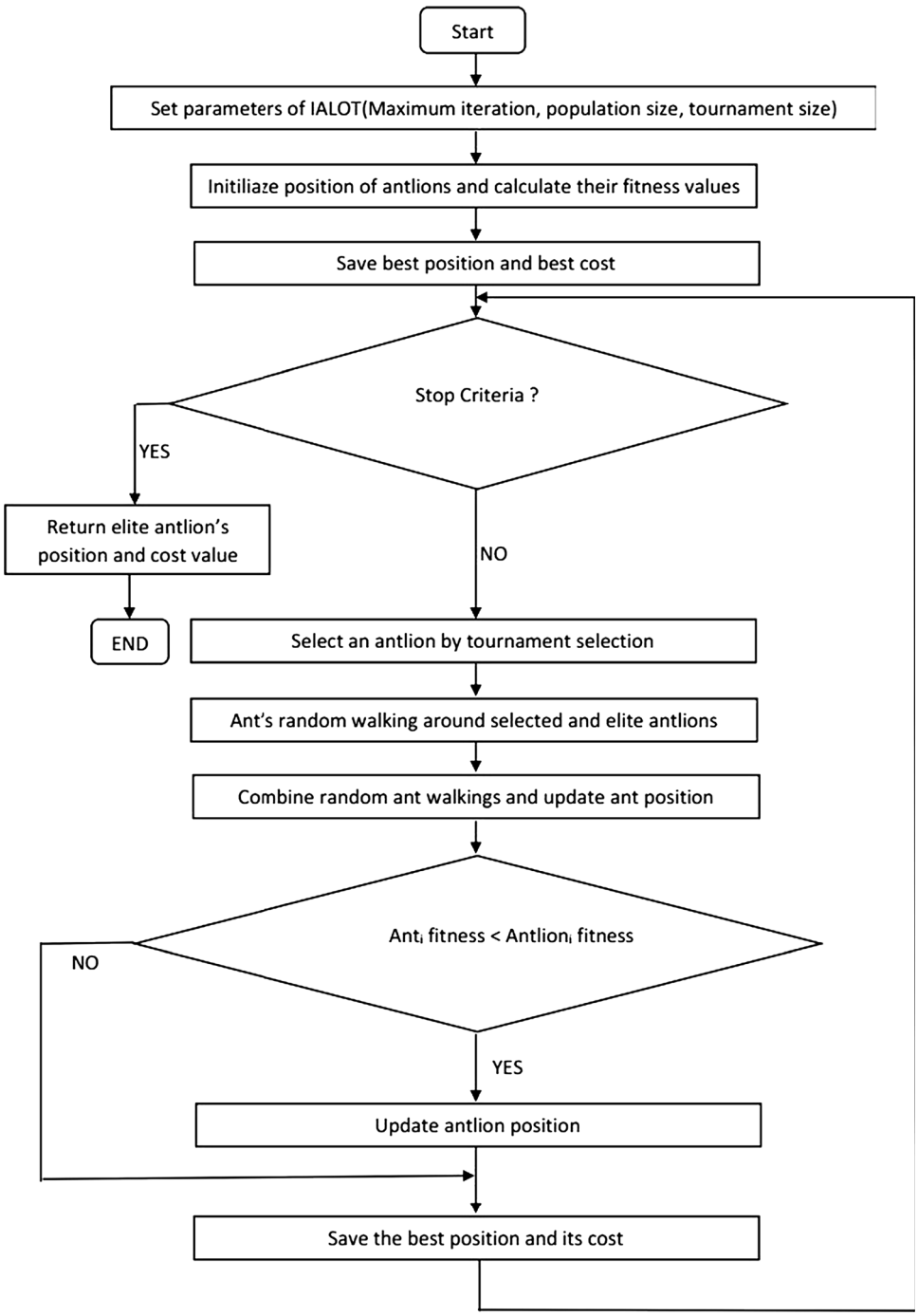


Fig. 5 IALOT algorithm flowchart

minimization problems, is preferred rather than the roulette wheel method in this study. The following pseudo-code summarizes the working mechanism of the tournament method.

Tournament selection method	
Step 1:	Select tournament size (T) less than population size (P)
Step 2:	Create $N = P/T$ tournament groups
Step 3:	Set shuffle pool
Step 4:	Find winner of each group according to fitness values. Send to shuffle pool
Step 5:	Go to Step 2 until reach the tournament size

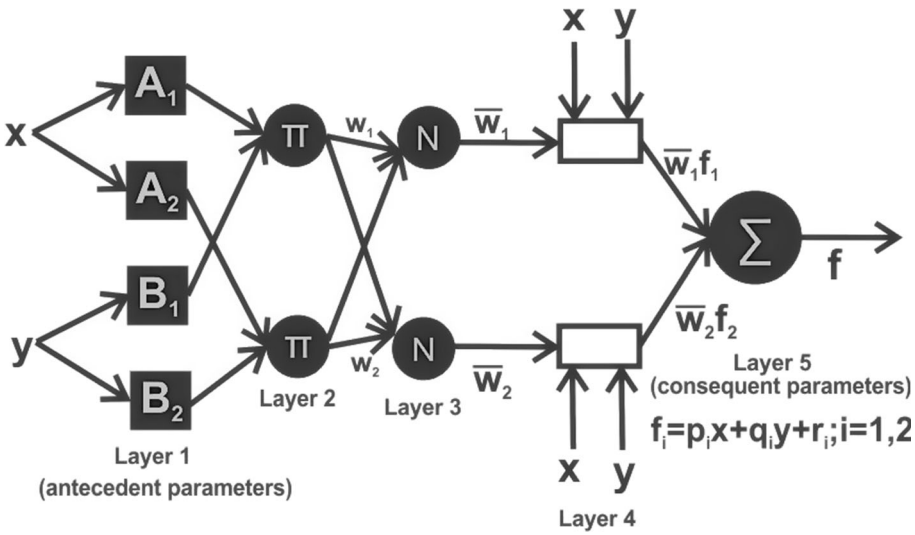


Fig. 6 General ANFIS structure with two inputs

Table 2 Parameters of heuristic algorithms used in this study

Algorithms	Parameters
PSO [23]	Learning coefficients = 2.05, constriction factor = 0.7298
ABC [18, 19]	Number of food sources = 50, limit cycle = 100
SA [24, 37]	Temperature = current iteration/maximum iteration number
DE [43, 44]	Crossover probability = 0.5, differential weight = 0.8, differential strategy = DE/rand/1/bin
TACO [14]	Vaporing = 0.1, number of bit = 18
ALO [27]	Search agent = 100
IALOT	Search agent = 100, random walk size = Max Iter/5

3.2 The other innovations on ALO algorithm

Even though ALO algorithm has impressive results for different benchmark functions and engineering design problems, it has some disadvantages. The greatest drawback of ALO algorithm is that it has a long run time on account of the random ant walking model especially. The leading innovation made on ALO algorithm is the one regarding the random walking mechanism for ants' movement. In proposed IALOT algorithm, the random walking distance for modeling ants' movement is changed. The distance of the random walk is determined as 20% of maximum iteration number rather than the maximum iteration number in the original ALO.

In original ALO algorithm, the long running times are encountered due to the fact that the random walking size is the maximum iteration number. Figure 3 shows the ants' random walks around the selected antlion for maximum random walking size and 20% of this size. In this figure, the red dot denotes the antlion at (0, 0) position. There are three different random walking routes in each figure. In Fig. 4, the convergence curves with different random walking size values (1, 0.8, 0.6, 0.4, 0.2 × maximum iteration) are shown for Schwefel function.

Table 1 summarizes the CPU time values obtained for Schwefel function. As can be seen from this table, while decreasing the random walking size, the CPU time values decreases. For the random walking size values less than 0.2 × maximum iteration, there is no exploration and exploitation in search space. In this study, we took the random walking size as 20% of the maximum iteration number due to these reasons.

Besides, some movements are added between lower and upper boundaries of the antlion in the phase of trapping on antlion pits. These proposed new movements ensure that the ants walk more effectively around the selected antlion in the search space. In addition, the selection mechanism in the last section of the classic ALO algorithm is improved. The old mechanism works with sorting all the ants and antlions in the population in order of fitness values. Thanks to the proposed selection mechanism, the next population is determined by comparing the ants and antlions according to their fitness values step by step. If the fitness value of the ant is better than the fitness value of the antlion, the antlion consumes the ant; thus, it takes the position of the ant. As a final improvement, the boundary checking mechanism is changed. By this mechanism, the positions of ants are brought back inside the search space if the ants go beyond the boundaries of the search space. Thus, the algorithm performance is increased in terms of the optimality and accuracy metrics.

Innovations made in the antlion's pit, and the shifting of the ants by throwing sand can be explained by the following mathematical model.

Table 3 Benchmark test results by 50 independent runs for heuristic algorithms

Measurements	Algorithms	F1	F2	F3	F4	F5
Mean best (STD)	PSO	4.211e+00(8.906e−01)	5.131e−01(1.440e−01)	2.219e−01(1.889e−01)	1.477e+01(5.772e+00)	1.078e+01(4.951e+00)
	ABC	4.998e−10(6.423e−10)	2.469e−03(4.373e−03)	1.299e−12(5.438e−12)	3.439e−14(9.002e−14)	1.855e−01(1.691e−01)
	SA	1.571e+01(2.168e+00)	1.145e+00(1.186e−01)	2.342e+00(8.741e−01)	2.295e+01(5.661e+00)	5.919e+01(2.381e+01)
	DE	5.966e−07(1.310e−07)	1.201e−01(2.303e−02)	1.473e−07(4.386e−08)	6.135e−01(8.683e−01)	2.743e+00(1.388e−01)
	TACO	1.508e+01(1.118e+00)	1.245e+00(1.652e−01)	1.377e+00(1.196e+00)	4.746e+01(1.037e+01)	2.928e+01(3.341e+01)
	ALO	2.143e−01(5.052e−01)	1.420e−01(7.520e−02)	2.498e−01(3.495e−01)	1.761e+01(7.441e+00)	4.938e+00(2.221e+00)
	IALOT	0.000e+00(0.000e+00)	0.000e+00(0.000e+00)	5.044e−06(1.162e−05)	0.000e+00(0.000e+00)	6.881e−05(1.400e−04)
NFE (CPU time)	PSO	23,824.00(0.804)	22,404.00(0.763)	18,928.00(0.507)	23,770.00(0.807)	20,678.00(0.600)
	ABC	50,347.36(1.804)	51,004.94(1.853)	30,893.06(0.922)	51001.06(1.832)	51010.26(1.577)
	SA	100,000.00(4.214)	100,000.00(4.275)	100,000.00(3.654)	100,000.00(4.237)	100,000.00(3.696)
	DE	69,488.00(2.369)	100,000.00(3.446)	41,192.00(1.121)	100000.00(3.446)	100000.00(2.920)
	TACO	100,000.00(28.619)	100,000.00(28.904)	100,000.00(28.735)	100,000.00(29.086)	100,000.00(27.852)
	ALO	99,258.00(49.881)	95,216.00(48.518)	90,324.00(46.103)	95,324.00(48.447)	99,812.00(49.180)
	IALOT	98,158.00(5.745)	74,826.00(4.435)	76,242.00(3.984)	77,482.00(4.564)	82,128.00(4.421)
Optimality	PSO	0.811	0.922	0.998	0.817	0.998
	ABC	1.000	1.000	1.000	1.000	1.000
	SA	0.296	0.827	0.975	0.715	0.990
	DE	1.000	0.982	1.000	0.992	1.000
	TACO	0.324	0.812	0.986	0.411	0.995
	ALO	0.990	0.979	0.997	0.781	0.999
	IALOT	1.000	1.000	1.000	1.000	1.000
Accuracy	PSO	0.989	0.973	0.989	0.932	0.797
	ABC	1.000	0.999	1.000	1.000	0.985
	SA	0.916	0.963	0.955	0.907	0.840
	DE	1.000	0.986	1.000	0.998	0.901
	TACO	0.933	0.959	0.966	0.888	0.809
	ALO	1.000	0.973	0.992	0.900	0.858
	IALOT	1.000	1.000	1.000	1.000	1.000

Table 3 (continued)

Measurements	Algorithms	F6	F7	F8	F9	F10
Mean best (STD)	PSO	1.684e+03(3.264e+02)	1.129e−01(6.549e−02)	-3.379e+01(2.309e+00)	2.264e+00(1.586e+00)	6.858e+00(7.276e+00)
	ABC	1.273e−04(3.632e−08)	1.043e−11(4.096e−11)	− 3.917e+01(7.713e−14)	1.494e−12(4.129e−12)	9.460e+00(5.399e+00)
	SA	8.753e+02(1.179e+02)	1.895e+00(7.545e−01)	− 3.534e+01(8.019e−01)	3.390e+01(1.472e+01)	6.699e+01(1.887e+01)
	DE	1.274e−04(4.530e−08)	1.511e−07(4.560e−08)	− 3.917e+01(4.400e−08)	1.551e−07(5.455e−08)	1.778e−02(1.189e−02)
	TACO	1.127e+03(3.077e+02)	2.293e−01(9.581e−01)	− 3.013e+01(1.703e+00)	2.280e+00(6.065e+00)	2.009e+01(6.186e+00)
	ALO	1.615e+03(5.727e+02)	8.677e−09(3.487e−09)	− 3.659e+01(2.452e+00)	3.823e−08(3.199e−08)	6.129e−10(2.280e−10)
	IALOT	4.132e−01(9.657e−01)	1.926e−35(9.531e−35)	− 3.917e+01(5.683e−05)	0.000e+00(0.000e+00)	6.924e−14(4.896e−13)
NFE (CPU time)	PSO	28,058.00(0.863)	16,162.00(0.331)	21,188.00(0.577)	20,954.00(0.424)	30,654.00(0.634)
	ABC	51006.98(1.658)	26837.24(0.624)	44658.24(1.305)	30471.00(0.701)	51,001.98(1.185)
	SA	100,000.00(3.883)	100,000.00(2.839)	100,000.00(3.490)	100,000.00(2.825)	100,000.00(2.842)
	DE	85238.00(2.922)	36086.00(0.755)	44556.00(1.232)	41428.00(0.856)	100,000.00(2.104)
	TACO	100,000.00(28.826)	100,000.00(28.064)	72,678.00(20.589)	100,000.00(27.983)	96,774.00(26.741)
	ALO	98,496.00(48.897)	90,288.00(45.223)	90,250.00(45.690)	91,670.00(45.966)	95,200.00(47.429)
	IALOT	94,900.00(5.228)	53,814.00(2.465)	75,756.00(4.016)	71,386.00(3.256)	79,666.00(3.658)
Optimality	PSO	0.005	0.998	0.967	0.992	1.000
	ABC	1.000	1.000	1.000	1.000	1.000
	SA	0.478	0.964	0.977	0.887	0.999
	DE	1.000	1.000	1.000	1.000	1.000
	TACO	0.327	0.996	0.945	0.992	1.000
	ALO	0.036	1.000	0.984	1.000	1.000
	IALOT	1.000	1.000	1.000	1.000	1.000
Accuracy	PSO	0.671	0.992	0.867	0.990	0.961
	ABC	1.000	1.000	1.000	1.000	0.955
	SA	0.705	0.969	0.907	0.967	0.861
	DE	1.000	1.000	1.000	1.000	0.998
	TACO	0.777	0.995	0.824	0.994	0.925
	ALO	0.390	1.000	0.897	1.000	1.000
	IALOT	1.000	1.000	1.000	1.000	1.000

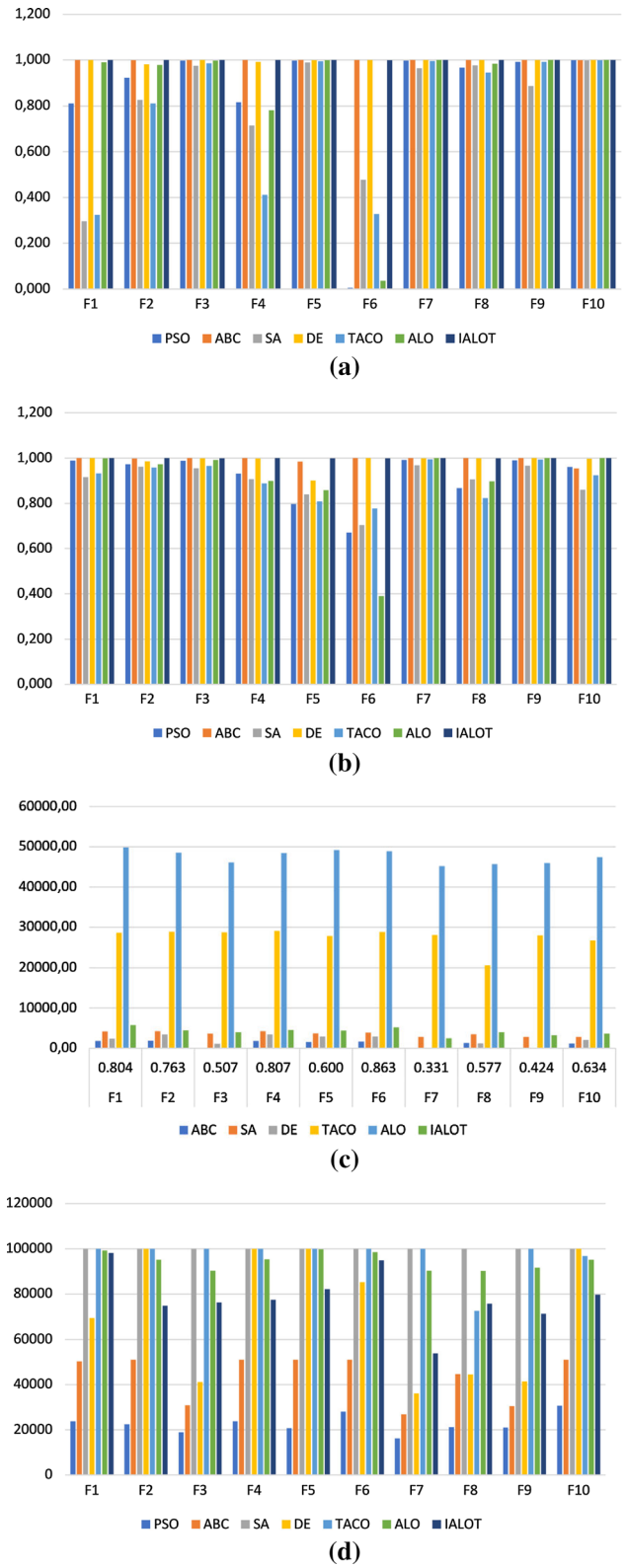


Fig. 7 Comparison results of IALOT and the other algorithms. **a** Optimality, **b** accuracy, **c** CPU time, **d** number of functions evaluation

$$\left.\begin{aligned} c_i^t &= \text{Antlion}_i^t + c_i^t \\ d_i^t &= \text{Antlion}_i^t + d_i^t \end{aligned}\right\} \quad \text{if } 0.75 \le \delta < 1 \tag{11}$$

$$\left.\begin{aligned} c_i^t &= \text{Antlion}_i^t - c_i^t \\ d_i^t &= \text{Antlion}_i^t - d_i^t \end{aligned}\right\} \quad \text{if } 0.5 \le \delta < 0.75 \tag{12}$$

$$\left.\begin{aligned} c_i^t &= -\text{Antlion}_i^t + c^t \\ d_i^t &= -\text{Antlion}_i^t + d^t \end{aligned}\right\} \quad \text{if } 0.25 \le \delta < 0.5 \tag{13}$$

$$\left.\begin{aligned} c_i^t &= -\text{Antlion}_i^t - c^t \\ d_i^t &= -\text{Antlion}_i^t - d^t \end{aligned}\right\} \quad \text{otherwise} \tag{14}$$

where δ is a randomly chosen number from [0, 1] interval. When the ants exit the search space, they are taken to the search space again with the following mathematical model. This innovation, unlike the original ALO algorithm, leaves the ants randomly in the search space.

$$\begin{aligned} &\text{Ant}_i^t = b_{\text{low}} + \text{rand} \times \left(b_{\text{up}} - b_{\text{low}}\right) \quad \text{if} \\ &\left(\text{Ant}_i^t > b_{\text{up}}\right) \text{OR} \left(\text{Ant}_i^t < b_{\text{low}}\right) \end{aligned} \tag{15}$$

where rand is a randomly chosen number in interval [0, 1], b_{low} is lower boundary, b_{up} is upper boundary. The flowchart given in Fig. 5 shows the fundamental operations of the proposed IALOT algorithm.

4 ANFIS structure

Adaptive neuro-fuzzy inference system (ANFIS) was presented by Jang [15]. The ANFIS structure is an intelligent neuro-fuzzy structure that combines the fuzzy logic system and neural network system. In fact, antecedent and consequent parameters of ANFIS are trained to make a fuzzy conclusion. The ANFIS structure consists of five layers. The description of these layers is given below and ANFIS structure with two inputs is given in Fig. 6. The ANFIS's layers are explained in following equations.

Layer 1:

$$O_{1,i} = \mu_{A_i}(x) = \exp\left(-\frac{(x-a_i)^2}{(2b_i)^2}\right); \quad i = 1,2 \tag{16}$$

$$O_{1,i} = \mu_{B_{i-2}}(y) = \exp\left(-\frac{(y-a_i)^2}{(2b_i)^2}\right); \quad i = 3,4 \tag{17}$$

In this layer, a_i and b_i are called antecedent parameters, μ is called membership function.

Layer 2:

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1,2 \tag{18}$$

The output of each node represents the firing strength w_i of the rule, usually the algebraic product T-norm operator is applied as the node function.

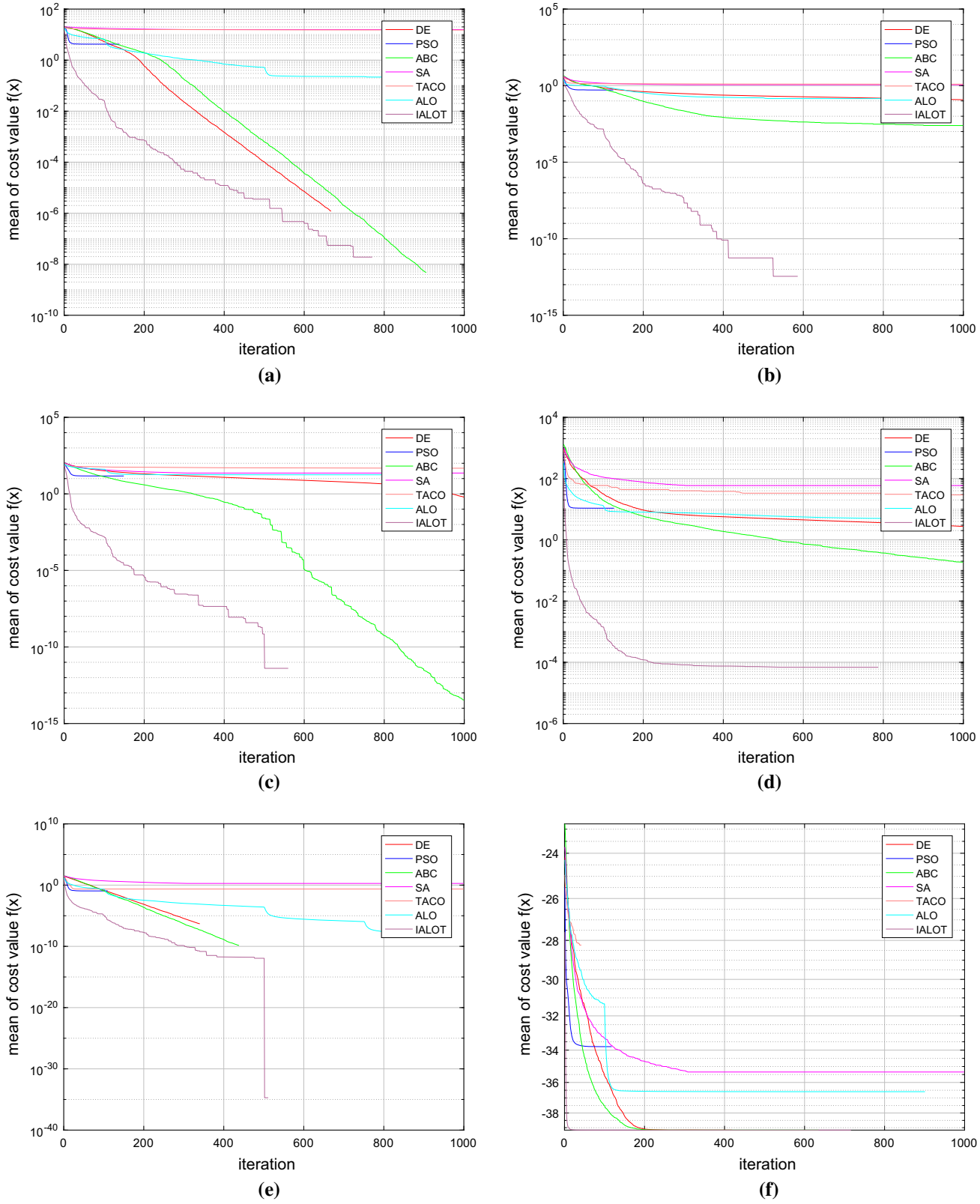


Fig. 8 Mean best fitness curves for 50 independent runs of some benchmark functions. **a** F1, **b** F2, **c** F4, **d** F5, **e** F7, **f** F8

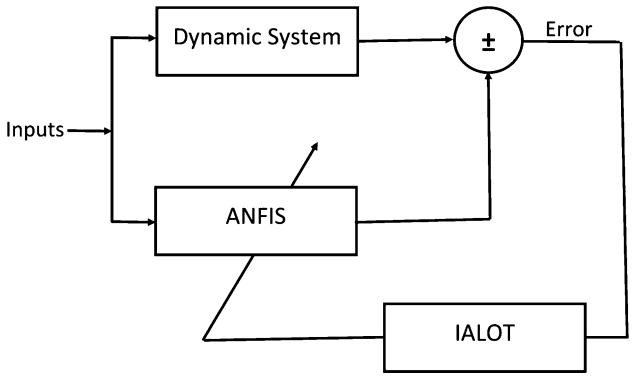


Fig. 9 ANFIS training by IALOT algorithm

Layer 3:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{19}$$

In each node, the ratio of rule firing strength to sum of all the firing strength is found. The output of this layer is called normalized firing strengths.

Layer 4:

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i) \tag{20}$$

Each node in this layer is the adaptive node with their node functions. The parameters p_i, q_i, r_i in the layer are called the consequent parameters.

Layer 5 :

$$O_{5,1} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{21}$$

The single node in the last layer is known as a fixed node that calculates the sum of the all previous layers' outputs, where f_i are rules.

5 Performance comparison of IALOT

New heuristic algorithms in engineering sciences are usually tested with benchmark functions and their performances in application problems are examined. The IALOT algorithm introduced in this study is compared with ALO and other popular algorithms. IALOT is tested with ten-dimensional benchmark functions and the comparison results are presented in tables and graphs. Then, training of ANFIS parameters is done by IALOT and other algorithms, and algorithm performance for five popular dynamic systems is observed.

5.1 Benchmark performance of IALOT

When we look at the heuristic algorithms up to this point, it is seen that they are tested with benchmark test functions and the performance evaluation of these algorithms is based on different metrics such as average/mean, standard deviation, optimality, accuracy, etc. It is also compared with other well-known heuristic algorithms. In this study, performance measurement using benchmark test functions is evaluated for *optimality, accuracy, standard deviation, and best average metrics*. In addition, the performance of IALOT is compared with PSO, ABC, SA, DE, TACO algorithms other than ALO. Below are the mathematical expressions of the metrics used for comparison.

$$f : X \subseteq \mathbb{R}^n \rightarrow F \tag{22}$$

where n is the dimension of possible solution space or the search space. Let $x_0 \in X$ is the solution, $f(x_0) = f_0$ is said to be solution of the optimization problem, and $f(\hat{x_0}) = \hat{f_0}$

Table 4 Dynamic systems for ANFIS training and testing

System no.	Dynamic system	Training set	Testing set
1	$y(k) = \frac{y(k-1)y(k-2)(y(k-1)+2.5)}{1+y^2(k-1)+y^2(k-2)} + u(k)$ [29]	$u(k) = \cos(2\pi k/100)$	$u(k) = \sin(2\pi k/25)$
2	$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k)$ [29]	$u(k) = \cos(2\pi k/100)$	$u(k) = \sin(2\pi k/25)$
3	$y(k+1) = y(k) + u(k)e^{-3\lvert y(k)\rvert}$ [6]	$[-1, 1]$	$[-1, 1]$
4	$y(k+1) = \frac{24+y(k)}{30} y(k) - 0.8\frac{u^2(k)}{1+u^2(k)} y(k-1) + 0.5u(k)$ [32]	$[-5, 5]$	$[-5, 5]$
5	$y(k+1) = 0.5(\frac{y(k)}{1+y^2(k)} + (1 + u(k))u(k)(1 - u(k)))$ [38]	$[-2, 2]$	$[-2, 2]$

Table 5 Parameter numbers of ANFIS to be model of dynamic systems

System no.	Inputs	# Membership function	# rules	# parameters
1	$u(k), y(k-2), y(k-1)$	6 {2, 2, 2}	8	44
2	$u(k), y(k), y(k-1)$	6 {2, 2, 2}	8	44
3	$u(k)$, $y(k)$	4 {2, 2}	4	20
4	$u(k), y(k), y(k-1)$	6 {2, 2, 2}	8	44
5	$u(k)$, $y(k)$	4 {2, 2}	4	20

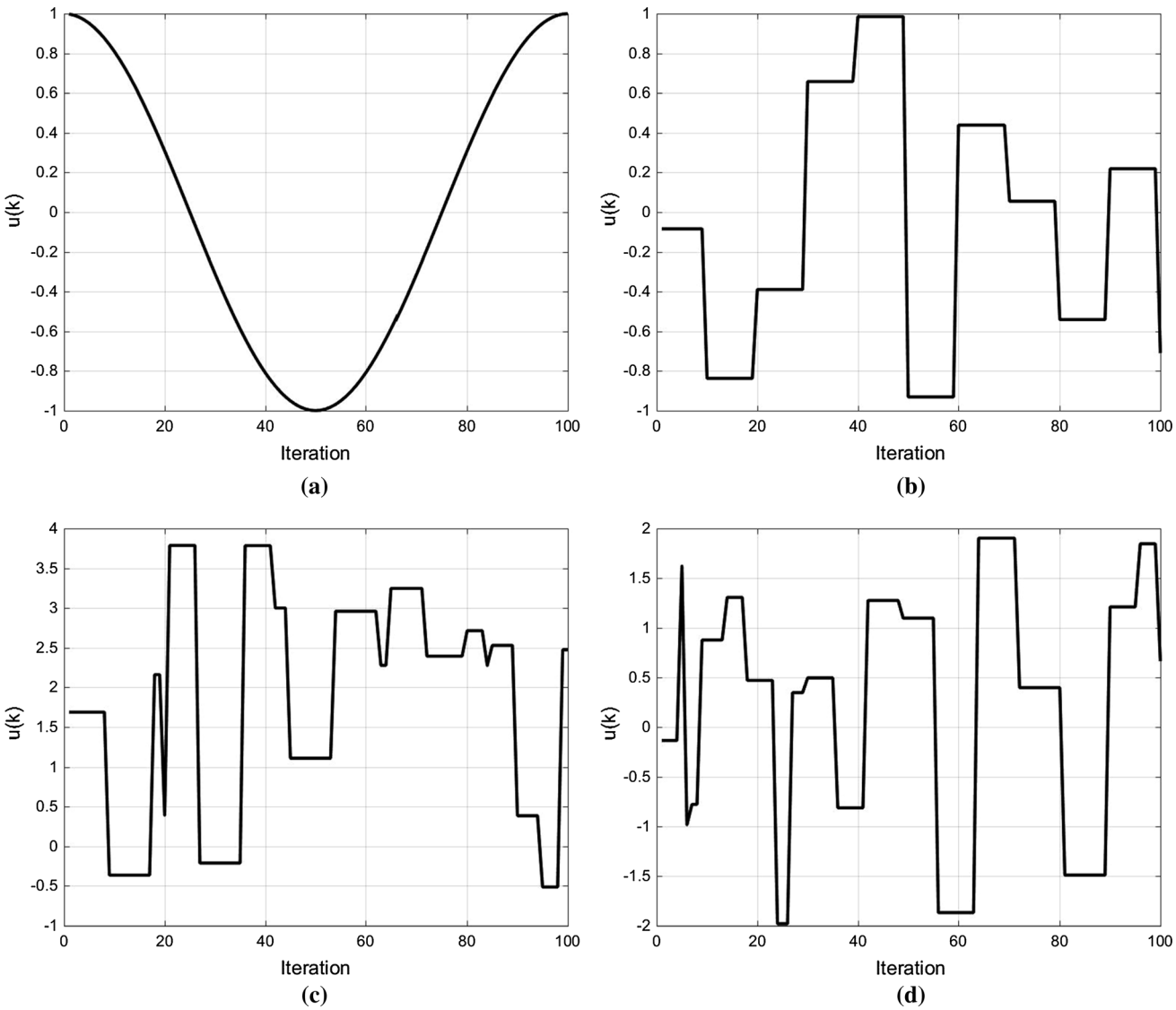


Fig. 10 Input data sets of dynamic systems for training. **a** D. System 1, 2, **b** D. System 3, **c** D. System 4, **d** D. System 5

closeness the solution found. Then used metrics are defined as follows:

$$\text{Optimality} = 1 - \frac{\|f_0 - \hat{f_0}\|}{\|\overline{f} - \underline{f}\|} \in [0, 1] \tag{23}$$

$$\text{Accuracy} = 1 - \frac{\|x_0 - \hat{x_0}\|}{\|\overline{x} - \underline{x}\|} \in [0, 1] \tag{24}$$

$$\text{Mean} = \frac{1}{N}\sum_{i=1}^{N} \hat{f_0} \tag{25}$$

$$\text{Standard deviation (STD)} = \sqrt{\frac{1}{N-1}\sum \left(\hat{f_0} - \text{Mean}\right)^2} \tag{26}$$

where $\overline{f}$ and $\underline{f}$ are lower and upper bounds of f, $\overline{x}$ and $\underline{x}$ lower and upper bounds of search space. N stands for the number of the repetition [25]. Optimality metric denotes the relative closeness to fitness value of solution. Accuracy represents the relative closeness to the solution. Mean metric stands for the average of closeness the solution found.

The 10-dimensional benchmark test functions are used in this study. The characteristics of these test functions are different from each other. Ackley, Griewank, Rastrigin, Levy, Schwefel functions are the functions which include many local minimum points. Rastrigin function is a multimodal function, Rosenbrock function is a valley-shaped unimodal function, and it is difficult to converge to a global minimum. Besides, the Sphere function is a unimodal function that has the local minimum point size. The Sum Squares function is a bowl-shaped function which has only a global minimum point. Similarly, Zakharov function has

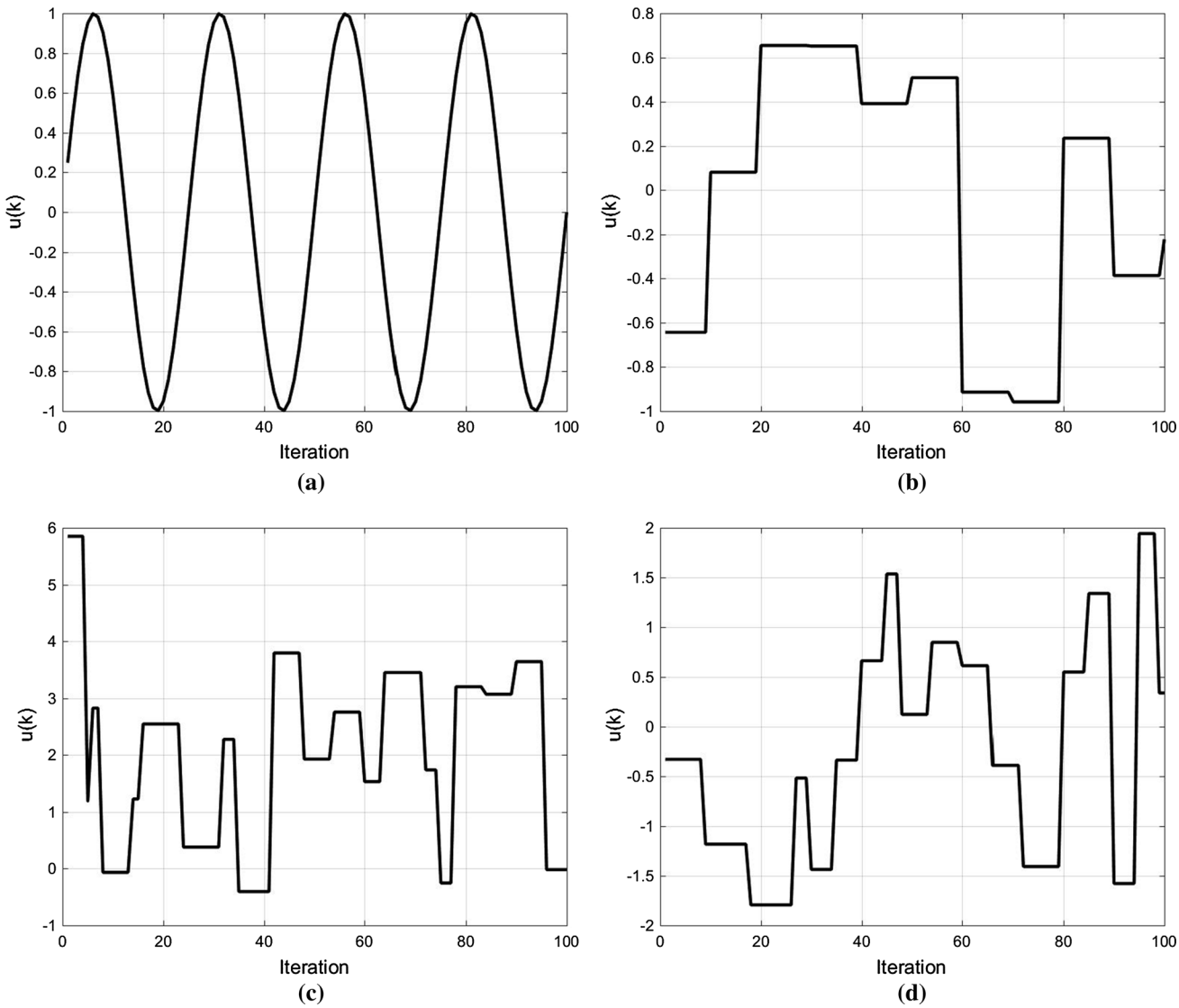


Fig. 11 Input data sets of dynamic systems for test. **a** D. System 1,2. **b** D. System 3, **c** D. System 4, **d** D. System 5

no local minima except the global one. The mathematical expressions of these benchmark functions are given below:

F1: Ackley function

$$f(x) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1) \tag{27}$$

subject to $-35 \le x_i \le 35$, the global minima is $f(x) = 0$ at $x = (0, \ldots, 0)$, $a = 20, b = 0.2, c = 2\pi$

F2: Griewank function

$$f(x) = \sum_{i=1}^{d}\frac{x_i^2}{4000} - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{28}$$

subject to $-100 \le x_i \le 100$ and global minima $f(x) = 0$ at $x = (0, \ldots, 0)$

F3: Levy function

$$f(x) = \sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i - 1)^2\left[1 + 10\sin^2(\pi w_i + 1)\right] + (w_d - 1)^2\left[1 + \sin^2(2\pi w_d)\right] \tag{29}$$

where

$$w_i = 1 + \frac{x_i - 1}{4}, \quad \text{for all } i = 1, 2, \ldots, d \tag{30}$$

subject to $-10 \le x_i \le 10$, the global minima is $f(x) = 0$ at $x = (1, \ldots, 1)$

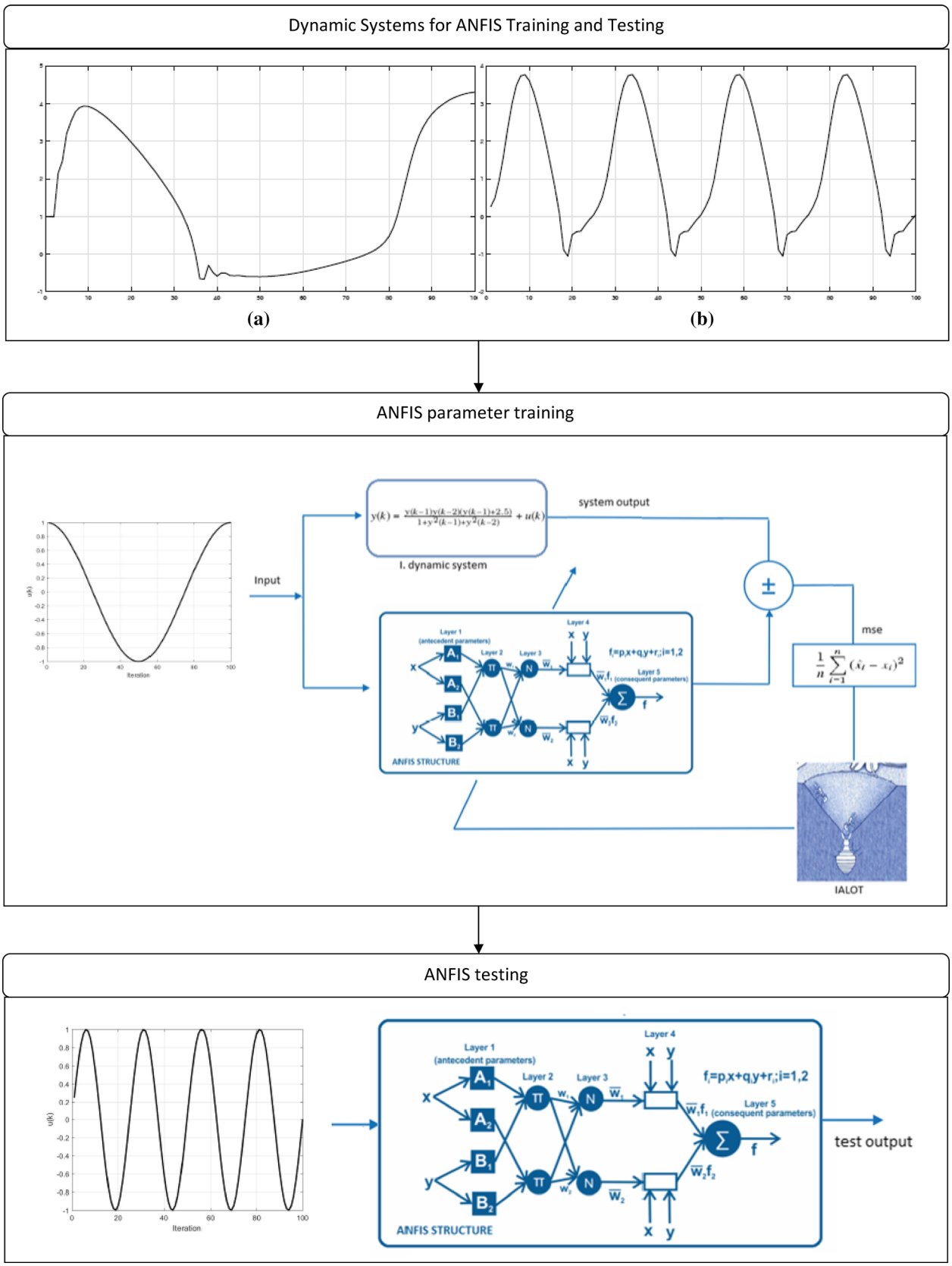


Fig. 12 The flowchart of ANFIS training and testing with IALOT algorithm

F4: Rastrigin function

$$f(x) = 10d + \sum_{i=1}^{d}\left[x_i^2 - 10\cos(2\pi x_i)\right] \tag{31}$$

subject to $-5.12 \le x_i \le 5.12$, the global minima is $f(x) = 0$ at $x = (0, 0, \ldots, 0)$

F5: Rosenbrock function

$$f(x) = \sum_{i=1}^{d-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right] \tag{32}$$

subject to $-2.3 \le x_i \le 2.3$, the global minima is $f(x) = 0$ at $x = (1, \ldots, 1)$

F6: Schwefel function

$$f(x) = 418.9829d - \sum_{i=1}^{d} x_i \sin\left(\sqrt{|x_i|}\right) \tag{33}$$

subject to $-500 \le x_i \le 500$, the global minima is $f(x) = 0$ at $x = (420.97, \ldots, 420.97)$

F7: Sphere function

$$f(x) = \sum_{i=1}^{d} x_i^2 \tag{34}$$

subject to $-5.12 \le x_i \le 5.12$, the global minima is $f(x) = 0$ at $x = (0, 0, \ldots, 0)$

F8: Styblinski–Tang function

$$f(x) = \frac{1}{2}\sum_{i=1}^{d}\left(x_i^4 - 16x_i^2 + 5x_i\right) \tag{35}$$

subject to $-5 \le x_i \le 5$, the global minima is $f(x) = -39.16$ at $x = (-2.9, \ldots, -2.9)$

F9: Sum squares function

$$f(x) = \sum_{i=1}^{d} i x_i^2 \tag{36}$$

subject to $-10 \le x_i \le 10$, the global minima is $f(x) = 0$ at $x = (0, \ldots, 0)$

F10: Zakharov function

$$f(x) = \sum_{i=1}^{d} x_i^2 + \left(\frac{1}{2}\sum_{i=1}^{d} i x_i\right)^2 + \left(\frac{1}{2}\sum_{i=1}^{d} i x_i\right)^4 \tag{37}$$

subject to $-5 \le x_i \le 10$, the global minima is $f(x) = 0$ at $x = (0, \ldots, 0)$

In performance testing of algorithms, the population size is 100, and maximum iteration number is 1000 and all algorithms are run 50 times. All codes of heuristic algorithms are run on PC with Intel(R) Core(TM) i7-6500U CPU@2.50GHz/8.00GB RAM. The parameters of heuristic algorithms that are used in this work are given as in Table 2.

Two termination criteria are used for all heuristic algorithms, one for reaching the maximum number of iterations, and the other for value to reach (VTR $= 10^{-6}$). VTR condition is given below.

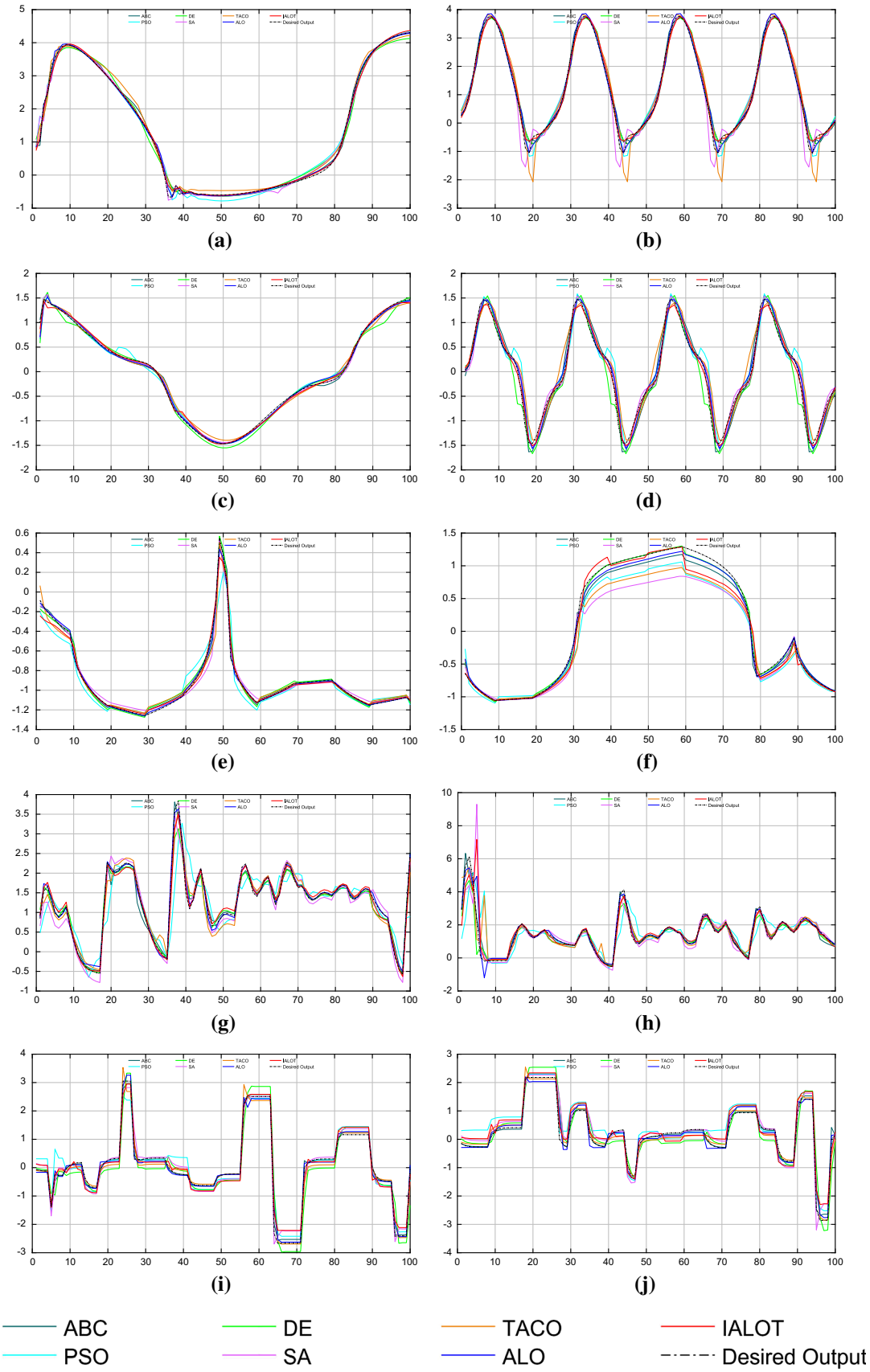
$$\text{if} \quad \text{VTR} > |\text{fitness}(\text{individual}_{\text{best}}) - \text{fitness}(\text{individual}_{\text{worst}})| \quad \text{then stop the algorithm} \tag{38}$$

In Mirjalili's work, there were no comparison results regarding the time analysis such as *CPU time and number of function evaluation (NFE)*. Also, two new metrics were added (optimality, accuracy) into the benchmark comparison works.

The comparative benchmark test results are given in Table 3. According to the mean of best results, IALOT algorithm shows the best performance for eight benchmark functions except of F3 and F6 functions. IALOT algorithm gives better results than ALO algorithm in all functions in terms of the standard deviation values. As can be seen from the results of the NFE and CPU time metrics, although IALOT algorithm is not the first for these metrics, the proposed IALOT algorithm is 10–20 times faster than the classic ALO algorithm. IALOT algorithm has an overall best performance at optimality and accuracy metrics. Classic ALO algorithm has 100% optimality value for F7, F9 and F10 benchmark functions. According to the accuracy metrics, ALO algorithm has the best performance with 100% for only F1, F7, F9 and F10 benchmark functions.

For the proposed IALOT algorithm and the heuristic algorithms used in benchmark tests, the comparison results are shown in Fig. 7. In this figure, four metric results, which are presented in Table 3, are visualized. In terms of first two metrics (optimality and accuracy), the proposed IALOT algorithm shows the best performance among the heuristic algorithms. As can be seen from the results of *CPU Time* metric, the classic ALO algorithm has the worst performance in comparison with the others. According to the *Number of Function Evaluation (NFE)* metric results, the proposed IALOT algorithm presents the competitive results for all benchmark functions. According to these results, ALO usually gives only successful results on easy surfaces like bowl-shaped, plate-shaped. Unlike ALO, IALOT has always been successful, even on difficult surfaces with a very local minimum point.

In Fig. 8, the logarithmic convergence curves are shown for six benchmark functions. Ackley (F1), Griewank (F2), Rastrigin (F4) functions have many local minimum points and they are difficult test functions. The Rosenbrock (F5) function is unimodal and valley shaped, with only the global minimum point. Sphere (F7) and Styblinski–Tang (F8) function are bowl-shaped functions. It can be understood here that IALOT algorithm converges faster and it stops the run when the stop criterion reaches. For these


(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
(i)
(j)
ABC
DE
TACO
IALOT
PSO
SA
ALO
Desired Output

◀**Fig. 13** ANFIS training and test results for five dynamic systems. *x*-axes: sampling period, *y*-axes: system output. **a** I. Dyn. Sys. training results, **b** I. Dyn. Sys. test results, **c** II. Dyn. Sys. training results, **d** II. Dyn. Sys. test results, **e** III. Dyn. Sys. training results, **f** III. Dyn. Sys. test results, **g** IV. Dyn. Sys. training results, **h** IV. Dyn. Sys. test results, **i** V. Dyn. Sys. training results, **j** V. Dyn. Sys. test results

benchmark functions, the mean cost curves of IALOT in all functions is better than the other algorithms.

5.2 ANFIS parameter learning by using IALOT algorithm

The parameters of ANFIS structure have been optimized with different heuristic algorithms up to this time. From time to time, these parameters have been optimized by using more than one optimization algorithm. Such models are called hybrid models. In this study, optimization of these parameters is made by comparing with IALOT, ALO and other popular algorithms. The first stage is the training phase, the parameters of ANFIS are optimized by the heuristic algorithms for modeling of five different dynamic systems in this phase, and then ANFIS structure with these adjusted parameters is tested for the systems at the second phase. The block diagram of ANFIS training phase using IALOT algorithm is presented in Fig. 9.

In Table 4, the dynamic systems to be modeled with ANFIS and the sets used for testing and training are given. The inputs of the modeled dynamic systems, the number of membership functions used, the number of rules, and the number of parameters are given in Table 5. In these dynamic models, $u(k)$ denotes input signal at k-th iteration, $y(k)$ represents output signal at k-th iteration.

The detailed information of the data sets in Table 4 can be accessed from Karakuzu's study [22]. In addition, these data sets are given in Figs. 10 and 11. Note that in the data sets 1 and 2, a cosine and sinus curves are drawn as much as the iteration size, and in the data set 2, 3, and 4 sampling is performed with random amplitudes up to the iteration in the intervals specified in Table 4.

For the first dynamic system in Table 4, there are two membership functions for each inputs, eight rules are used in 4th layer. The Gauss membership functions consist of two parameters, and there are four consequent parameters for one fuzzy rule. As a result, total 44 parameters of ANFIS are optimized by IALOT for the first dynamic system. While performing ANFIS parameter training in this study, square errors are taken from output of ANFIS at the end of each iteration, the mean of these square errors are used as a fitness value of an individual for a heuristic algorithm. Each algorithm are run 30 times to train ANFIS with maximum 500 generations. For the performance comparisons, statistical best, statistical worst, mean square error formulas are given below, where $\hat{x_i}$ is the prediction value produced by ANFIS on training at $i-th$ iteration, x_i is corresponding target value. Also f is the best solution's fitness value of each run, and m is the number of run for each algorithms.

$$\text{Mean square error} = \frac{1}{n}\sum_{i=1}^{n}(\hat{x_i} - x_i)^2 \tag{39}$$

$$\text{Statistical best} = \text{Min}_{i=1}^{m} f \tag{40}$$

$$\text{Statistical worst} = \text{Max}_{i=1}^{m} f \tag{41}$$

The population size is calculated by the following equation given in [13, 22, 31], where *round* is the function that rounds to nearest decimal or integer and D is the dimension of the problem.

$$\text{Population size} = \text{round}\left(10 + 2\sqrt{D}\right) \tag{42}$$

The entire ANFIS process is summarized in the flow diagram below in Fig. 12. Figure 13 shows ANFIS training and test results for five dynamical systems. The first column of this figure is the train results plotted according to the best individual obtained among 30 runs, and the second column of this figure shows the test results of ANFIS constructed with the best individual obtained at the training stage. The detailed results for all algorithms are given in Table 6. In the table, *R., Av.R., Av.Cum.R* are respectively refers to rank, average rank and average cumulative rank. Rank is the best degree of the algorithm among others, and the average rank is the sum of the rank of the systems divided by the number of systems for each metric. The average cumulative rank is the sum of the average rank of the metrics divided by the number of metrics.

Table 6 is organized in two parts, one of them is *Train*, and the other one is *Test* for each heuristic optimization algorithm. In the measurement column, *Best* and *Worst* refer to best solution's and worst solution's fitness values after 30 runs. *Average* is average of fitness values of all solutions, *SD* is standard deviation of them. As can be seen from this table, ABC is the best algorithm and DE is the second best algorithm for all categories of ANFIS training and test phases according to the average cumulative rank. The proposed IALOT algorithm is the third best algorithm with ALO algorithm. Furthermore, it is seen from these results that IALOT algorithm is 4–6 times faster than ALO algorithm for ANFIS training phase.

6 Conclusion and future works

This work proposes the improved Ant Lion Optimizer via tournament selection method called IALOT. The most important drawback of the classic ALO algorithm is that it

Table 6 Performance comparison of dynamic system modeling based on ANFIS trained by heuristic algorithms

Algorithms	Measurements	Sys. 1	R.	Sys. 2	R.	Sys. 3	R.	Sys. 4	R.	Sys. 5	R.	Av. R.	Av. Cum. R.
ABC	*Train*												
	Time	1.417	1	1.414	1	0.850	1	1.409	1	0.785	1	1	1.886
	Best	0.008	3	0.002	1	0.001	2	0.018	3	0.006	1	2	
	Worst	0.027	1	0.010	1	0.006	1	0.103	2	0.170	1	1.2	
	Average	0.018	3	0.006	2	0.004	2	0.040	2	0.095	1	2	
	SD	0.005	1	0.002	3	0.001	2	0.018	2	0.042	2	2	
	Test												
	Best	0.040	3	0.027	4	0.010	3	0.062	1	0.017	1	2.4	
	Worst	0.232	4	0.064	4	0.135	2	0.283	2	0.122	1	2.6	
DE	*Train*												
	Time	2.570	2	2.582	2	1.412	2	2.571	2	1.442	2	2	2.943
	Best	0.021	5	0.008	7	0.002	3	0.023	4	0.149	7	5.2	
	Worst	0.055	4	0.014	4	0.007	2	0.081	1	0.274	3	2.8	
	Average	0.036	4	0.010	5	0.004	1	0.042	3	0.216	5	3.6	
	SD	0.008	3	0.001	1	0.001	1	0.015	1	0.035	1	1.4	
	Test												
	Best	0.046	5	0.042	5	0.002	1	0.131	3	0.125	7	4.2	
	Worst	0.049	1	0.041	2	0.134	1	0.235	1	0.234	2	1.4	
PSO	*Train*												
	Time	2.644	4	2.675	3	1.519	3	2.675	3	1.463	3	3.2	6.086
	Best	0.029	7	0.004	5	0.010	7	0.210	7	0.102	6	6.4	
	Worst	2.029	7	0.045	7	0.069	7	8.300	7	0.416	7	7	
	Average	0.391	7	0.020	7	0.028	7	1.335	7	0.264	7	7	
	SD	0.420	7	0.009	7	0.016	7	1.712	7	0.073	6	6.8	
	Test												
	Best	0.046	4	0.044	6	0.045	5	0.675	7	0.116	6	5.6	
	Worst	3.498	7	0.167	7	0.790	7	11.886	7	0.267	5	6.6	
SA	*Train*												
	Time	2.619	3	2.675	4	1.544	4	2.693	4	1.535	4	3.8	4.257
	Best	0.018	4	0.002	3	0.002	5	0.036	5	0.058	4	4.2	
	Worst	0.067	5	0.010	3	0.013	4	0.209	4	0.254	2	3.6	
	Average	0.036	5	0.008	4	0.007	5	0.100	5	0.180	4	4.6	
	SD	0.012	5	0.002	2	0.002	3	0.038	4	0.055	2	3.2	
	Test												
	Best	0.119	6	0.020	3	0.074	7	0.616	6	0.075	5	5.4	
	Worst	0.288	6	0.085	6	0.391	5	0.897	5	0.249	3	5	
TACO	*Train*												
	Time	16.529	6	16.910	6	6.883	6	16.549	6	6.839	6	6	5.543
	Best	0.023	6	0.005	6	0.003	6	0.036	6	0.044	3	5.4	
	Worst	0.134	6	0.021	6	0.015	5	0.641	6	0.323	6	5.8	
	Average	0.062	6	0.013	6	0.009	6	0.203	6	0.241	6	6	
	SD	0.031	6	0.004	6	0.003	4	0.146	6	0.059	4	5.2	
	Test												
	Best	0.159	7	0.061	7	0.049	6	0.264	4	0.033	3	5.4	
	Worst	0.258	5	0.063	3	0.360	4	2.742	6	0.387	7	5	

Table 6 (continued)

Algorithms	Measurements	Sys. 1	R.	Sys. 2	R.	Sys. 3	R.	Sys. 4	R.	Sys. 5	R.	Av. R.	Av. Cum. R.
ALO	*Train*												
	Time	19.029	7	18.543	7	7.681	7	18.814	7	7.705	7	7	3.629
	Best	0.004	1	0.002	4	0.001	1	0.005	1	0.021	2	1.8	
	Worst	0.049	3	0.014	5	0.009	3	0.119	3	0.303	5	3.8	
	Average	0.015	1	0.006	3	0.005	3	0.033	1	0.148	2	2	
	SD	0.009	4	0.002	5	0.003	5	0.029	3	0.081	7	4.8	
	Test												
	Best	0.016	1	0.018	2	0.004	2	0.096	2	0.031	2	1.8	
	Worst	0.182	3	0.068	5	0.175	3	0.585	4	0.279	6	4.2	
IALOT	*Train*												
	Time	3.770	5	3.769	5	1.977	5	3.928	5	2.134	5	5	3.629
	Best	0.005	2	0.002	2	0.002	4	0.013	2	0.060	5	3	
	Worst	0.036	2	0.010	2	0.017	6	0.211	5	0.298	4	3.8	
	Average	0.016	2	0.006	1	0.006	4	0.068	4	0.180	3	2.8	
	SD	0.008	2	0.002	4	0.003	6	0.045	5	0.067	5	4.4	
	Test												
	Best	0.025	2	0.009	1	0.017	4	0.301	5	0.071	4	3.2	
	Worst	0.148	2	0.035	1	0.487	6	0.514	3	0.259	4	3.2	

has a long run time due to the standard random walking model. The proposed IALOT algorithm includes some improvements on operators of ALO algorithm, such as using tournament selection method, new boundary checking procedure, updating the selection process for each iteration, etc. The performance of the proposed IALOT algorithm is evaluated on ten benchmark functions in terms of optimality, accuracy, number of function evaluation/CPU time and mean best, in comparison with the popular and well-known heuristic algorithms, such as DE, PSO, ABC, SA, TACO and ALO algorithms. It is clearly evident that the proposed IALOT algorithm provides the best results in terms of optimality, accuracy and mean best metrics. According to the NFE/CPU time metric results, the performance of the proposed IALOT algorithm is better than classic ALO algorithm. As a result, it can be said that the proposed IALOT algorithm is able to operate as an alternative algorithm for different optimization problems.

This paper also deals with the performance of the IALOT algorithm for ANFIS parameters optimization. The training of ANFIS is a very difficult optimization problem. In this study, the proposed IALOT algorithm has been adapted for training of ANFIS parameters and its performance has been tested for five dynamic benchmark systems. The comparative results show that the proposed IALOT algorithm provides very competitive results among those of the well-known heuristic algorithms. This work is limited to solve single objective optimization problems with multi-dimensions. However, this proposed IALOT algorithm can be adapted by the future researchers for the multiobjective optimization problems. Future studies can deal with the training process of the different models using IALOT algorithm.

Compliance with ethical standards

Conflict of interest The author declares that there is no conflict of interest.

References

1. Abdel-Basset M, El-Shahat D, El-Henawy I, Sangaiah AK (2018) A modified flower pollination algorithm for the multidimensional knapsack problem: human-centric decision making. Soft Comput 22(13):4221–4239
2. Abdel-Basset M, El-Shahat D, El-henawy I, Sangaiah AK, Ahmed SH (2018) A novel whale optimization algorithm for cryptanalysis in Merkle–Hellman cryptosystem. Mob Netw Appl 23(4):723–733
3. Abdel-Basset M, El-Shahat D, Sangaiah AK (2017) A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem. Int J Mach Learn Cybern. https://doi.org/10.1007/s13042-017-0731-3
4. Anand S, Afreen N, Yazdani S (2015) A novel and efficient selection method in genetic algorithm. Int J Comput Appl 129(15):7–12
5. Babers R, Ghali NI, Hassanien AE, Madbouly NM (2015) Optimal community detection approach based on ant lion optimization. In: 2015 11th international computer engineering conference (ICENCO), pp 284–289

6. Babuška R (1997) Fuzzy systems, modeling and identification. Technical Report. http://www.dcsc.tudelft.nl/. Accessed 10 Nov 2017
7. Blickle T, Thiele L (1996) A comparison of selection schemes used in evolutionary algorithms. Evol Comput 4(4):361–394
8. Carrano EG, Takahashi RHC, Caminhas WM, Neto OM (2008) A genetic algorithm for multiobjective training of ANFIS fuzzy networks. In: 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence), pp 3259–3265
9. Cavuslu MA, Karakuzu C, Karakaya F (2012) Neural identification of dynamic systems on FPGA with improved PSO learning. Appl Soft Comput 12(9):2707–2718
10. Chopra N, Mehta S (2015) Multi-objective optimum generation scheduling using ant lion optimization. In: 2015 annual IEEE India conference (INDICON), pp 1–6
11. Dorigo M, Caro GD (1999) Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406), vol 2, p 1477
12. Gupta E, Saxena A (2016) Performance evaluation of antlion optimizer based regulator in automatic generation control of interconnected power system. J Eng 2016:14
13. Hansen N, Ros R, Schoenauer MNM, Auger A (2011) Impacts of invariance in search: when CMA-ES and PSO face ill-conditioned and non-separable problems. Appl Soft Comput 11(8):5755–5769
14. Hiroyasu T, Miki M, Ono Y, Minami Y (2000) Ant colony for continuous functions, vol 20. The Science and Engineering, Doshisha University, Kyoto
15. Jang JSR (1993) ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans Syst Man Cybern 23(3):665–685
16. Jiang HM, Kwong CK, Ip WH, Wong TC (2012) Modeling customer satisfaction for new product development using a PSO-based ANFIS approach. Appl Soft Comput 12(2):726–734
17. Kamboj VK, Bhadoria A, Bath SK (2017) Solution of non-convex economic load dispatch problem for small-scale power systems using ant lion optimizer. Neural Comput Appl 28(8):2181–2192
18. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical report, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department
19. Karaboga D, Akay B (2007) Artificial bee colony (abc) algorithm on training artificial neural networks, pp 1–4
20. Karaboga D, Kaya E (2013) Training ANFIS using artificial bee colony algorithm. In: IIS on IEEE (ed.) Innovations in intelligent systems and applications. IEEE, pp 1–5
21. Karakuzu C (2010) Parameter tuning of fuzzy sliding mode controller using particle swarm optimization. Int J Innov Comput Inform Control 6:4755–4770
22. Karakuzu C (2017) On the performance of newsworthy meta-heuristic algorithms based on point of view fuzzy modelling. Turk J Electr Eng Comput Sci 25:4706–4721
23. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings IEEE international conference on neural networks, 1995, vol 4, pp 1942–1948
24. Kirkpatrick S, Gelatt CD, Vecchi MP et al (1983) Optimization by simulated annealing. Science 220(4598):671–680
25. Luo W, Li Y (2016) Benchmarking heuristic search and optimisation algorithms in matlab. In: 22nd international conference on automation and computing (ICAC). IEEE, pp 250–255
26. Medhane DV, Sangaiah AK (2017) Search space-based multi-objective optimization evolutionary algorithm. Comput Electr Eng 58:126–143
27. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83(Supplement C):80–98
28. Nair SS, Rana KPS, Kumar V, Chawla A (2017) Efficient modeling of linear discrete filters using ant lion optimizer. Circuits Syst Signal Process 36(4):1535–1568
29. Narendra KS, Parthasarathy K (1990) Identification and control of dynamical systems using neural networks. IEEE Trans Neural Netw 1(1):4–27
30. Nischal MM, Mehta S (2015) Optimal load dispatch using ant lion optimization. Int J Eng Res Appl 5(8):10–19
31. Nobile MS, Pasi G, Cazzaniga P, Besozzi D, Colombo R, Mauri G (2015) Proactive particles in swarm optimization: a self-tuning algorithm based on fuzzy logic. In: IEEE international conference on fuzzy systems
32. Oussar Y, Rivals I, Dreyfus L (1998) Training wavelet networks for nonlinear dynamic input–output modeling. Neurocomputing 20:173–188
33. Petrovic M, Petronijevic J, Mitic M, Vukovic N, Plemic A, Miljkovic Z, Babic B (2015) The ant lion optimization algorithm for flexible process planning. JPE 18(2):65–68
34. Raju M, Saikia LC, Sinha N (2016) Automatic generation control of a multi-area system using ant lion optimizer algorithm based pid plus second order derivative controller. Int J Electr Power Energy Syst 80:52–63
35. Razali NM, Geraghty J et al (2011) Genetic algorithm performance with different selection strategies in solving TSP. Proc World Congr Eng 2:1134–1139
36. Rebecca N, Shin M, Mh S, Zuriani M (2015) Ant lion optimizer for optimal reactive power dispatch solution. J Electr Syst 3:67–74
37. Rutenbar RA (1989) Simulated annealing algorithms: an overview. IEEE Circuits Devices Mag 5(1):19–26
38. Sastry PS, Santharam G, Unnikrishnan KP (1994) Memory neuron networks for identification and control of dynamical systems. IEEE Trans Neural Netw 5(2):306–319
39. Satheeshkumar R, Shivakumar R (2016) Ant lion optimization approach for load frequency control of multi-area interconnected power systems. Circuits Syst 7(9):2357
40. Shoorehdeli MA, Teshnehlab M, Sedigh AK (2009) Training anfis as an identifier with intelligent hybrid stable learning algorithm based on particle swarm optimization and extended Kalman filter. Fuzzy Sets Syst 160(7):922–948
41. Shoorehdeli MA, Teshnehlab M, Sedigh AK, Khanesar MA (2009) Identification using anfis with intelligent hybrid stable learning algorithm approaches and stability analysis of training methods. Appl Soft Comput 9(2):833–850
42. Srikanth K, Panwar LK, Panigrahi B, Herrera-Viedma E, Sangaiah AK, Wang GG (2018) Meta-heuristic framework: quantum inspired binary grey wolf optimizer for unit commitment problem. Comput Electr Eng 70:243–260
43. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
44. Storn R, Price K (1997) Differential evolution a simple evolution strategy for fast optimization. Dr. Dobb's J 22(4):18–24 and 78
45. Tirkolaee EB, Alinaghian M, Hosseinabadi AAR, Sasi MB, Sangaiah AK (2018) An improved ant colony optimization for the multi-trip Capacitated Arc Routing Problem. Comput Electr Eng. https://doi.org/10.1016/j.compeleceng.2018.01.040
46. Trivedi IN, Parmar SA, Bhesdadiya RH, Jangir P (2016) Voltage stability enhancement and voltage deviation minimization using ant-lion optimizer algorithm. In: 2016 2nd international conference on advances in electrical, electronics, information, communication and bio-informatics (AEEICB), pp 263–267
47. Tung NS, Chakravorty S (2016) Ant lion optimizer based approach for optimal scheduling of thermal units for small scale

electrical economic power dispatch problem. Int J Grid Distrib Comput 9(7):211–224
48. Yao P, Wang H (2017) Dynamic adaptive ant lion optimizer applied to route planning for unmanned aerial vehicle. Soft Comput 21(18):5475–5488
49. Zangeneh AZ, Mansouri M, Teshnehlab M, Sedigh AK (2011) Training ANFIS system with de algorithm. In Advanced computational intelligence (IWACI) fourth international workshop on IEEE, pp 308–314