Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **8**, Number **3**, August **2023**

An Improved Archimedes Optimization Algorithm (IAOA)

Yan-Jiao Wang

School of Electrical Engineering, Northeast Electric Power University
169 Changchun Road, Jilin, 132012, China
wangyanjiao1028@126.com

Ming-Chi Chen

School of Electrical Engineering, Northeast Electric Power University
169 Changchun Road, Jilin, 132012, China
chenmingchi925@126.com

Chin Soon Ku

Department of Computer Science, Universiti Tunku
Abdul Rahman, Kampar 31900, Malaysia
kucs@utar.edu.my

*Corresponding author: Ming-Chi Chen

Received November 14, 2022, revised February 23, 2023, accepted May 4, 2023.

Abstract. *The Archimedes optimization algorithm (AOA) is a new meta-heuristic inspired by Archimedes' principle in physics. However, when solving complex optimization problems, there are some defects, such as slow convergence and easy falling into local optimum. In order to further improve the convergence performance of AOA, an improved Archimedes optimization algorithm (IAOA) is proposed in this paper. This paper mainly improves the learning factor and the acceleration updating formula of individual attributes in the exploration stage to further maintain the diversity of the population and improve the early convergence speed of the algorithm. At the same time, this paper proposes a mechanism to update the individual position according to the probability and introduces an adjustment factor in the development stage to improve the development ability of the algorithm. In addition, the simplex method mechanism is used to correct the poor individuals and improve the convergence speed and accuracy of the algorithm. The experimental results of the CEC2013 test set show that IAOA has significant advantages in terms of convergence speed, accuracy, and stability compared with the original AOA algorithm and the four most excellent meta-heuristics algorithms so far.*
Keywords: Archimedes optimization algorithm, meta-heuristic algorithm

1. **Introduction.** Optimization problems exist widely in industrial design, cloud computing, internet communication, and other fields, such as the lowest cost, the best parameters, and the lowest energy consumption. For example, the traditional optimization methods, such as the gradient descent method, usually have poor robustness and require the optimization problem to be guided. Moreover, when the optimization problem is complex, there are many constraints and multiple extreme values, the convergence speed and convergence accuracy can hardly meet the actual demand. The meta-heuristic algorithm, proposed by simulating biological habits or physical phenomena in nature, has good exploration and development ability because of its randomness and intelligence.

Meta-heuristic algorithms mainly include the following three types. 1) A meta-heuristic algorithm based on biological behavior in nature. For example, ant colony optimization algorithm (ACO) [1], whale optimization algorithm (WOA) [2], Particle Swarm Optimization(PSO) [3], Bat Algorithm(BA) [4], Grey Wolf Optimizer(GWO) [5], hunting search algorithm(Hus) [6]. Many meta-heuristic algorithms of this type have been proposed in recent years. In 2018, Cheraghalipour et al. [7] proposed the Tree Growth Algorithm (TGA) based on the competitive behavior of trees for access to light and food, which uses trees as individuals to search for optimal solutions through intensification and diversification stages; In 2021, Abdollahzadeh et al. [8] proposed the African Vulture Optimization Algorithm (AVOA) based on the foraging and navigation behavior of African vultures, in which the weakest and hungriest vulture is identified as the worst solution and the optimal solution is sought by approaching the strongest and fullest vulture; In 2021, Naruei et al. [9] proposed the Wild Horse Optimization Algorithm (WHO) based on the social life behavior of horses, which designs a wild horse optimizer to search for the optimal solution by using group behavior, herding, mating, dominance and leadership; 2) A meta-heuristic algorithm based on physical phenomena in nature. For example, harmonious search algorithm (HS) [10], vibrating particle system algorithm (VPS) [11], ray optimization algorithm (Ray) [12], magnetic system search algorithm (MCSS) [13], Simulated Annealing Algorithm (SA) [14]. Many meta-heuristic algorithms of this type have also been proposed in recent years. In 2020 Rahmanzadeh et al. [15] inspired by the electron discharge mechanism, proposed the Electron Radar Search Algorithm (ERSA), which simulates the behavior of electrons searching for the best path in a medium and evaluates the surrounding environment through a radar mechanism to select the best way in the next step; In 2021, Hashim et al. [16] proposed the Archimedes optimization algorithm (AOA), based on Archimedes' principle in physics, which updates the optimal solution by simulating the behavior of an object seeking equilibrium in a liquid; 3) A meta-heuristic algorithm based on biological evolution in nature. For example, Genetic Algorithm (GA) [17], Difference Algorithm (DE) [18]. In 2022, Oyelade et al. proposed the Ebola Optimization Search Algorithm (EOSA) [19], based on the transmission behavior of the Ebola virus, which updates the solution through dynamic mechanisms such as susceptibility, infection, isolation, recovery and Inpatient area.

In recent years, further improving the convergence performance of existing evolutionary algorithms has become a research focus in the field of evolution. The convergence performance of the AOA algorithm is significantly better than WOA, CSA, HHO, EO, etc., and further improving the performance of AOA has attracted the attention of some scholars. In this context, to further improve the convergence accuracy and speed of the AOA algorithm for solving complex optimization problems, this paper proposes an improved Archimedean optimization algorithm (IAOA). The main points of innovation and motivation are as follows:

1. To improve the exploration phase of the algorithm. The learning factor and the acceleration updating formula of individual attributes are improved. At the same time, a regulatory factor is introduced to enhance the learning ability of the acceleration of individual attributes and maintain population diversity while maintaining the algorithm's convergence rate.

2. To improve the development phase of the algorithm. A mechanism for updating individual positions according to probability is proposed. The present individual learns from the current best individual to improve the convergence speed of the algorithm and learns from the individual itself to effectively maintain population diversity. At the same time, this phase introduces an adjustment factor to learn from both the individual itself

and the best individual to further balance the convergence speed and population diversity of the algorithm.

3. The simplex method correction strategy is used to correct the positions of the poor individuals to enhance the exploration capability of the algorithm in the early stages of evolution and to improve the convergence speed.

The test results on the CEC 2013 test set show that compared with AOA and four more representative optimization algorithms, the IAOA proposed in this paper has significant advantages in terms of convergence speed, accuracy, and stability.

The rest of the paper is organized as follows: Section 2 summarizes the research of some improved algorithms and AOA algorithm. Section 3 describes the working principle and flow of the AOA algorithm. Section 4 analyses the shortcomings of the original AOA algorithm and further proposes an improved IAOA algorithm. Section 5 shows the simulation results and analysis of the IAOA algorithm with the original AOA algorithm and other more mainstream improved algorithms on the CEC2013 test function set. Section 6 concludes the algorithm proposed in this paper.

2. **Related work.** To further improve the optimization effect, researchers have done a lot of work on the performance improvement of existing meta-heuristic algorithms. In 2010, Cheng et al. [20] proposed an improved ant colony optimization algorithm (IACO), which uses the metropolis criterion to select the paths of ants, overcoming the immature convergence of the algorithm and improving the algorithm's ability to find the best; In 2014, Layeb et al. [21] proposed a new Firefly Optimization algorithm (FCO), which is a distributed greedy metaheuristic algorithm that uses positive feedback to construct greedy optimal solutions, avoiding premature convergence and increasing the ability of the algorithm to jump out of the local optimum; In 2014, Jia et al. [22] proposed an improved max-min ant system algorithm (MMAS), which incorporates a local search algorithm for better performance; In 2018, Zhang et al. [23] proposed the DLPSO algorithm, which selects suitable vectors from those distributed in the search space, forms a new vector and moves towards the better vector position, improving the possibility of jumping out of the local optimum to a greater extent; In 2018, Abdel-Basset et al. [24] proposed an improved Levy flight-based whale optimization algorithm (ILWOA) that introduces Lévy flights to model the behavior of whale movements, while adding a new variational phase to improve convergence speed. Finally, using chaotic logistic mapping balances the exploration and exploitation capabilities of the algorithm; In 2019, Kang et al. [25] proposed a caching optimization method based on cloud computing for communication systems to reduce network traffic; In 2020, Truong et al. [26] proposed a quasi-oppositional chaotic symbiotic biological search algorithm (QOCSOS), which introduces two search strategies: quasi-oppositional learning and chaotic local search to increase the search exploitation capability of the algorithm and thus achieve better performance; In 2020, Zhang et al. [27] proposed a short-term traffic flow prediction algorithm based on quantum genetic algorithms-Learning vector Quantization (QGA-LVQ) neural network for predicting changes in traffic flow; In 2022 Shaik et al. [28] proposed a Gauss mutation - Spider monkey optimization (GM-SMO) algorithm, in which Gauss mutation increased the diversity of the population and enabled GM-SMO to achieve a better balance in exploration and development; In 2023, Chen et al. [29] proposed a genetic algorithm combined with simulated annealing to solve the Vehicle Routing Problem.

Some scholars have done some work to improve the performance of the AOA algorithm. Desuky et al. [30] proposed an enhanced Archimedes optimization algorithm (EAOA) for feature selection in classification, which adds a new parameter depending on the step

size of each individual while modifying the individual positions. This improvement improves the balance of AOA exploration and exploitation and improves the classification performance of feature selection problems in real-world datasets; Houssein et al. [31] proposed to introduce local escape operators (LEO) and orthogonal learning (OL) into the Archimedes optimization algorithm to increase the local and global search capabilities of the algorithm, respectively; Sun et al. [32] proposed an improved Archimedes optimization algorithm. The opposition-based learning (OBL) mechanism is added to increase the diversity of the population and accelerate the convergence rate. Chaotic learning theory was used to generate some new solutions, thereby increasing the algorithm's ability to jump out of the local optimal; Yao et al. [33] adopted chaotic logic mapping to change random numbers in the initialization stage of the population and carried out Lévy flight operation to update individual positions. Thus, these operations increase the optimization ability and convergence speed of Archimedes algorithm.

It can be seen from the above research on AOA that compared with AOA, the convergence speed and convergence accuracy of the improved AOA algorithm have been improved to a certain extent. However, when solving complex problems, there is still much room for improvement in convergence performance. Compared with the existing improved AOA algorithm, the IAOA proposed in this paper has the following advantages: (1) It can better maintain the diversity of the population and balance the exploration and development stages of the algorithm, which is conducive to the further improvement of the algorithm's convergence accuracy, convergence speed and stability; (2) It is more competitive in solving complex problems.

3. Archimedes optimization algorithm.

In physics, Archimedes' principle states that when an object is immersed in a stationary fluid, it will experience a buoyant force, the direction of which is vertically upward and the size of which is equal to the weight of the fluid displaced by the object. Inspired by the law, Hashim et al. [16] proposed an Archimedes optimization algorithm with the pseudo-code shown in Algorithm 1, which mainly contains three critical operations: initialization, individual position update, and individual attribute update, as follows.

3.1. Initialization.

In AOA, each individual contains three attribute information in addition to position information: volume, density, and acceleration. Suppose the number of individuals in the iterative population X is N and the dimensionality of the optimization problem is D. In the initialization phase, the position $x_i^0 = \left[x_{i,1}^0, x_{i,2}^0, \ldots, x_{i,j}^0 \ldots, x_{i,D}^0\right]$, volume $vol_i^0 = \left[vol_{i,1}^0, vol_{i,2}^0, \ldots, vol_{i,j}^0 \ldots, vol_{i,D}^0\right]$, density $den_i^0 = \left[den_{i,1}^0, den_{i,2}^0, \ldots, den_{i,j}^0 \ldots, den_{i,D}^0\right]$, and acceleration $acc_i^0 = \left[acc_{i,1}^0, acc_{i,2}^0, \ldots, acc_{i,j}^0 \ldots, acc_{i,D}^0\right]$ of each individual are randomly generated. The details are shown in Equations (1) – (4).

$$X_{i,j}^0 = X_L + rand_1 \times (X_U - X_L) \tag{1}$$

$$vol_{i,j}^0 = rand_2 \tag{2}$$

$$den_{i,j}^0 = rand_3 \tag{3}$$

$$acc_{i,j}^0 = X_L + rand_4 \times (X_U - X_L) \tag{4}$$

Where X_U and X_L correspond to the upper and lower limits of the jth dimensional search space in the optimization problem, respectively, and $rand_1$, $rand_2$, $rand_3$, $rand_4$ are different random numbers uniformly distributed within $[0, 1]$.

Algorithm 1 AOA

Input: population size popsize, maximum number of iterations tmax, Dimensionalities of optimizing problem Dim, fixed constants $C1$, $C2$, $C3$, $C4$

Output: The optimal solution and its fitness value

Initialization of parameters($popsize$, $tmax$, Dim, $C1$, $C2$, $C3$, $C4$)
Generate the initial population X and the density deni, volume voli and acceleration acci of each individual X_i in the population according to section 3.1
Calculate the fitness value Fitnessi of each individual X_i
while $t <= tmax$ **do**
    Identify the optimal individual Xbest so far in the iteration, and its corresponding denbest, volbest and accbest
    Use Equation (5) to calculate the migration factor
    for each individual X_i in population X **do**
        if $TF <= 0.5$ **then**
            Individual X_i performs the exploration phase of Section 3.2 to update its position
        else
            Individual X_i performs the development phase of Section 3.2 to update its position
        end if
    end for
    Update the volume vol_i, density den_i, and acceleration acc_i of each individual X_i according to section 3.3
    $t = t + 1$
end while
Output the global optimal solution and its fitness value

3.2. **Update individual position.** The individuals in AOA go through the exploration and development phases to achieve self-renewal. And the migration operator, as shown in Equation (5), realizes the transition between the two phases as follows: when $TF^t < 0.5$, the algorithm enters the exploration phase; Otherwise, it will enter the development phase. The specific updating mechanism is as follows.

$$TF^t = \exp\left(\frac{t - t_{\max}}{t_{\max}}\right) \tag{5}$$

Where t and $t_{\max}$ represent the current and maximum iteration times, respectively.

(1) exploration phase

In the exploration phase, individual x_i^t is updated according to Equation (6). If it exceeds the boundary setting in dimensionality j, $x_{i,j}^t$ is directly placed near the boundary.

$$x_i^{t+1} = x_i^t + C1 \times rand(1, D) \times acc_{norm}^t \times d^t \times \left(x_{k1}^t - x_i^t\right) \tag{6}$$

Where x_i^t and x_i^{t+1} represent the positions of individuals before and after updating, respectively, $C1$ is an artificially set constant, usually set to 2, $rand(1, D)$ is a vector of length D between $[0, 1]$, x_{k1}^t is the position information of a randomly selected individual $K1$ in the population, d^t is the density factor, as shown in Equation (7), and acc_{norm}^{t+1} is the normalized acceleration, as shown in Equation (8).

$$d^t = e^{\frac{t_{\max} - t}{t_{\max}}} - \frac{t}{t_{\max}} \tag{7}$$

$$acc_{norm}^{t} = u \times \frac{acc_i^t - \min(acc)}{\max(acc) - \min(acc)} + l \tag{8}$$

Where $\max(acc)$ and $\min(acc)$ are the maximum and minimum accelerations in the current population, both u and l are artificially set parameters and are usually recommended to be set to 0.9 and 0.1, or can be modified according to specific problems.

After the individual renewal, the final individuals participating in the next iteration are determined according to Equation (9).

$$x_i^{t+1} = \begin{cases} x_i^{t+1}, & \text{if } fit\left(x_i^{t+1}\right) \le fit\left(x_i^t\right) \\ x_i^t, & \text{else} \end{cases} \tag{9}$$

Where fit $\left(x_i^{t+1}\right)$ and fit $\left(x_i^t\right)$ are the fitness values of x_i^{t+1} and x_i^t, respectively.

(2) Development phase

In the development phase, individual x_i^t is updated according to Equation (10). If it exceeds the boundary setting in dimensionality j, $x_{i,j}^t$ is directly placed near the boundary. After the update, the final individuals participating in the next iteration are determined according to Equation (9).

$$x_i^{t+1} = x_{best}^t + F \times C2 \times rand(1, D) \times acc_{norm}^t \times d^t \times \left(T \times x_{best}^t - x_i^t\right) \tag{10}$$

Where x_{best}^t represents the position of the optimal individual in the tth generation of iterations, $C2$ is artificially set constant and is usually recommended to be set to 6, d^t and acc_{norm}^{t+1} are calculated according to Equations (7) and (8), respectively, parameter T is calculated according to Equation (11), F determines the updating direction of the individual in each dimensionality and is calculated according to Equation (12).

$$T = C3 \times TF \tag{11}$$

Where $C3$ is artificially set constant, it is usually recommended to be set to 2.

$$F_j = \begin{cases} +1, & \text{if } p \le 0.5 \\ -1, & \text{else} \end{cases} \tag{12}$$

Where $p = 2 \times rand - C4$, $C4$ is artificially set constant, it is usually recommended to be set to 0.5.

3.3. **Update individual attribute information.** In AOA, the density den_i^t and volume vol_i^t in individual attribute information are updated according to Equations (13) and (14), respectively. If the density den_i^t and volume vol_i^t exceed the boundary range, they will be corrected according to Equations (15) and (16), and the acceleration acc_i^t will be updated in different ways according to different phases of individual evolution. The details are as follows:

When the individual performs the exploration phase, the acceleration acc_i^t is updated according to Equation (17); When the individual performs the development phase, the acceleration acc_i^t is updated according to Equation (18).

$$den_i^{t+1} = den_i^t + rand(1, D) \times \left(den_{\text{best}} - den_i^t\right) \tag{13}$$

$$vol_i^{t+1} = vol_i^t + rand(1, D) \times \left(vol_{best} - vol_i^t\right) \tag{14}$$

Where den_{best} and $v_{ol}l_{best}$ are the density and volume of the optimal individual so far in the iteration.

$$den_{i,j}^{t+1} = \begin{cases} 1, & \text{if } den_{i,j}^{t+1} > 1 \\ 0, & \text{esleif } den_{i,j}^{t+1} < 0 \end{cases} \tag{15}$$

$$vol_{i,j}^{t+1} = \begin{cases} 1, & \text{if } vol_{i,j}^{t+1} > 1 \\ 0, & \text{esleif } vol_{i,j}^{t+1} < 0 \end{cases} \tag{16}$$

$$acc_i^{t+1} = \frac{den_{k_2}^t + vol_{k_2}^t \times acc_{k_2}^t}{den_i^t \times vol_i^t} \tag{17}$$

Where $den_{k_2}^t$, $vol_{k_2}^t$, and $acc_{k_2}^t$ are the density, volume, and acceleration corresponding to a randomly selected individual $x_{k_n}^t$ in the population.

$$acc_i^{t+1} = \frac{den_{best} + vol_{best} \times acc_{best}}{den_i^{t+1} \times vol_i^{t+1}} \tag{18}$$

Where den_{best}, vol_{best}, and acc_{best} are the density, volume, and acceleration of the optimal individual so far in the iteration.

4. **Improved Archimedes optimization algorithm(IAOA).** In order to further improve the convergence performance of AOA, an improved Archimedean optimization algorithm (IAOA) is proposed in this section, and its pseudo-code is shown in Algorithm 2.

Algorithm 2 IAOA

Input: population size popsize, maximum number of iterations $tmax$, Dimensionalities of optimizing problem Dim, fixed constants $C1$, $C2$, $C3$, $C4$

Output: The optimal solution and its fitness value

Initialization of parameters($popsize$, $tmax$, Dim, $C1$, $C2$, $C3$, $C4$)
Generate the initial population X and the density den_i, volume vol_i and acceleration acc_i of each individual X_i in the population according to section 3.1
Calculate the fitness value $Fitness_i$ of each individual X_i
while $t <= tmax$ **do**
    Identify the optimal individual X_{best} so far in the iteration, and its corresponding den_{best}, vol_{best} and acc_{best}
    Use Equation (5) to calculate the migration factor
    for each individual X_i in population X **do**
        if $TF <= 0.5$ **then**
            Individual X_i performs the improved exploration phase of section 4.1 to update its position
        else
            Individual X_i performs the improved development phase of section 4.2 to update its position
        end if
    end for
    Update the volume vol_i, density den_i, and acceleration acc_i of each individual X_i according to section 3.3
    Execute the simplex method strategy of section 4.3 to revise and improve the poor 0.2 popsize individuals
    $t = t + 1$
end while
Output the global optimal solution and its fitness value

4.1. **Improved exploration phase.** It can be seen from Section 3.2 that all individuals entered the exploration phase in the early stage of AOA evolution. Generally, in the early stage of evolution, the evolutionary algorithm can improve its convergence speed as much as possible on the premise of ensuring sufficient exploration ability to realize the global search in the search range. Through in-depth analysis of the individual renewal

mode in the exploration phase of AOA shown in Equation (6), it can be found that its individual renewal can be essentially understood as "self-cognition + learning factor × social learning". Among them, the self-cognition part retains the evolutionary information carried by individuals themselves, and the social learning part makes each individual have certain opportunities to learn from any individual in the population, which maintains the diversity of the population to a certain extent and is conducive to the realization of global search. The learning factor $C1 \times rand(1, D) \times acc^t_{norm} \times d^t$ determines the degree of learning. A large number of experimental studies have shown that the convergence speed of AOA needs to be further improved at the early phase of evolution. In order to avoid excessive aggregation of individuals due to the improvement of convergence speed and retain the ability of self-cognition and social learning to maintain population diversity, $C1$ in the learning factor and acc^t_i required in the calculation of acc^t_{norm} are improved, as shown in Equations (19) and (20).

$$C1 = \lambda \times \exp\left(\frac{t_{\max} - t}{t_{\max}}\right) - \frac{\gamma \times Fitness_i}{sum(Fitness) + m} \tag{19}$$

Where λ and γ are generally set to 2 and 5 to achieve the best effect, or can be modified according to the specific problem. $sum(Fitness)$ represents the sum of the fitness values of all individuals in the current iteration population, and m is a minimal number to ensure that the denominator is not 0.

$$acc^{t+1}_i = \sigma \times \frac{den^t_{k_2} + vol^t_{k_2} \times acc^t_{k_2}}{rand \times den^t_i \times vol^t_i} + (1 - \sigma) \times \left(acc_{best} - accl^t_i\right) \tag{20}$$

Where σ is the regulatory factor, as shown in Equation (21).

$$\sigma = 1 - \left(\frac{t}{t_{\max}}\right)^2 - \frac{2\arctan\left(\frac{Fitness_i}{10}\right)}{\pi} \tag{21}$$

In summary, the adaptive learning factor proposed in this section has the following advantages: Firstly, it can be seen from Equation (19) that the maximum value of $C1$ is 2e, which can fully ensure sufficient exploration capacity for global search. Meanwhile, $C1$ adaptively changes according to the individual's own fitness value and evolutionary stage. When it is closer to the end of evolution, those individuals with better fitness value, the larger $C1$, they learn from the society to a greater extent and carry out greater global search. The population diversity will not decrease too much. Thus, the convergence speed and population diversity of the algorithm are well balanced; Secondly, it can be seen from Equations (20) and (21) that the learning of the optimal individual acceleration is added to the updating formula of acceleration at the initial stage of the algorithm. Meanwhile, the regulatory factor σ is introduced, it adaptively changes according to the individual's own fitness value and evolutionary stage. When the distance from the evolutionary termination is further, those individuals with worse fitness values have the smaller σ, the degree of individual acceleration learning from the optimal individual acceleration acc_{best} is higher than that from the random individual acceleration. The evolutionary information of the optimal individual acceleration is effectively absorbed by those individuals with worse fitness values, which can better improve the convergence speed of the algorithm in the early stage.

4.2. **Improved development phase.** In the later stage of AOA evolution, all individuals enter the development phase and realize self-renewal according to Equation (10). It can be seen from formula (10) that all individuals learn from the optimal individuals so far, which speeds up the convergence of the algorithm to a certain extent and strengthens

the fine-grained development of the algorithm near the optimal individuals. But it will undoubtedly greatly reduce the diversity of the algorithm. If the optimal individual deviates from the actual theoretical optimal position and is in a local peak, the algorithm is very likely to fall into the local optimum due to the extreme similarity of individuals, and the phenomenon of algorithm stagnation occurs. According to the fact that each individual has evolved to a certain extent in the exploration stage, they have carried certain evolutionary information when entering the development stage. Although this evolutionary information is inferior to the evolutionary information of the optimal individual, they are very likely to carry other directional search information, which is extremely important for the development of the global optimal position. Therefore, the individual renewal method of the development stage is proposed, as shown in Equation (22).

$$x_i^{t+1} = \begin{cases} x_{\text{best}}^t + F \times C2 \times rand(1,D) \times acc_{norm}^t \times d^t \times (x_{k3}^t - x_i^t) & \text{if } rand < 0.2 \\ \beta \times x_{\text{best}}^t + (1-\beta) \times x_i^t + F \times C2 \times rand(1,D) \times acc_{norm}^t \times d^t & \\ \quad \times (x_{k3}^t - x_i^t) & \text{if } 0.2 \le rand \le 0.6 \\ x_i^t + F \times C2 \times rand(1,D) \times acc_{norm}^t \times d^t \times (x_{k3}^t - x_i^t) & \text{if } rand > 0.6 \end{cases} \tag{22}$$

Where x_{k3}^t is a randomly selected individual in the population that is different from individual x_i^t, β is an adjustment factor, and the specific calculation method is shown in Equation (23).

$$\beta = 0.8e^{-\left(\frac{sum(Fitness)+m}{5\times Fitness_i+m}\right)\times\left(1.5-\frac{t}{t_{\max}}\right)^2} \tag{23}$$

Compared with the individual updating method in AOA's development phase, the new updating method proposed in this section has the following advantages: Firstly, the individual retains a certain possibility of learning from the optimal individual x_{best}^t, which does not reduce the convergence speed of the algorithm too much. On this basis, it increases the opportunity to learn from individual itself x_i^t and other individuals x_{k3}^t, and with the help of evolutionary information carried by them, the population diversity is effectively maintained, and the possibility of the algorithm falling into a local optimum is greatly reduced; Secondly, the adjustment factor β adaptively changes according to the individual's own fitness value and evolutionary stage. The further away from the end of evolution, those individuals with better fitness values have the smaller β, the degree of individual learning from the optimal individual x_{best}^t is weaker than the degree of individual learning from the individual x_i^t. That effectively preserves the excellent evolutionary information of the algorithm itself and further balances the convergence speed and population diversity of the algorithm.

4.3. **Poor individual correction strategy based on the simplex method.** Generally, poor solutions in evolutionary algorithms slow down the convergence speed of the algorithm. Literature [34] shows that the simplex method can improve the worst point in the simplex and form a new simplex by internal compression, external compression, mapping, and expansion. Then the worst point will continuously approximate the optimal point, which is helpful to improve the convergence speed of the evolutionary algorithm. In view of this, the simplex method strategy is introduced in this section to improve 0.2 N poor individuals in each generation. The details are as follows:

Firstly, for individual x_s who is to use the simplex method strategy, its reflection point x_r is calculated according to Equation (24).

$$x_r = x_c + a \times (x_c - x_s) \tag{24}$$

Where a is the reflection coefficient, generally set to 1, x_c is the center point of the optimal individual x_{hest} and the suboptimal individual x_{er} searched so far, as shown in Equation

(25)

$$x_c = \frac{x_{best} + x_{er}}{2} \tag{25}$$

Then, the relationship between x_s's fitness $Fitness_{x_s}$ and x_r's fitness $Fitness_{x_r}$ is compared, and a certain operation is selected to generate a new individual x'_s. The details are as follows: When $Fitness_{x_r} < Fitness_{x_{best}}$, the expansion operation is carried out according to Equation (26); When $Fitness_{x_r} > Fitness_{x_s}$, the compression operation is carried out according to Equation (27); When $Fitness_{x_{best}} \leq Fitness_{x_r} \leq Fitness_{x_s}$, the contraction operation is carried out according to Equation (28). If x'_s is better than x_s, replace x_s. Otherwise, retain x_s unchanged.

$$x'_s = x_c + \lambda \times (x_r - x_c) \tag{26}$$

Where λ is the expansion coefficient, it is generally set to 2.

$$x'_s = x_c + \eta \times (x_s - x_c) \tag{27}$$

Where η is the compression coefficient, it is generally set to 0.5.

$$x'_s = x_c - \mu \times (x_s - x_c) \tag{28}$$

Where μ is the shrinkage coefficient, it is generally set to 0.5.

5. **Experimental Results and Analysis.** To fully validate the performance of the IAOA algorithm, this section compares it with the AOA algorithm and the four evolutionary algorithms that have been excellent so far on the CEC2013 test set [35]. These include the sine cosine algorithm (ESCA) based on backward learning enhancement [36], the artificial tree algorithm (IATTP) based on bi-population [37], the improved crow search algorithm (ICSA) [38], and the improved Archimedes classification feature selection algorithm EAOA algorithm [30]. All the above experiments are run on a PC with Windows 10 operating system and i5-8265U CPU, and are programmed using MATLAB R2021a.

To ensure fairness of comparison, the number of population N = 50 for all algorithms, the dimensionalities D of the optimization problem is 30 for all, and the maximum number of function evaluations is MaxFEs = 100000. The other parameter settings for each algorithm are shown in Table 1, where the parameter values for each comparison algorithm are taken as in the original.

Table 1. Related parameter settings of each algorithm.

Algorithm	*Parameter*
AOA	$C1 = 2$; $C2 = 6$; $C3 = 2$; $C4 = 0.5$
ESCA	$h1 = h2 = h3 = 0.5$, $h4 = 0.8$, $m = 50$, $q = 0.8$
IATTTP	$Pc = 0.6$
ICSA	$AP = 0.1$; $FL = 1.5$
EAOA	$C1 = 2$; $C2 = 6$; $C3 = 2$; $C4 = 0.5$
IAOA	$C2 = 6$; $C3 = 2$; $C4 = 0.5$

Table 2 gives the mean and standard deviation of 30 independent experiments for each algorithm on the 30-dimensional CEC2013 test set, and the results of the algorithm that achieved the best optimization on the same function are blacked out. Table 3 gives the results of the Friedman test, and it gives the results of the Wilcoxon rank sum test of each algorithm against the IAOA at a significance level of 5%.

TABLE 2. Mean and variance of each algorithm on the 30-dimensional CEC2013 test set

	ESCA		IATTP		ICSA		AOA		EAOA		IAOA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	5.20E+03	1.14E+03	1.38E+01	4.93E+00	3.54E+01	3.33E+01	2.13E+04	5.73E+03	2.62E+03	6.73E+02	**0.00E+00**	**0.00E+00**
F2	7.75E+07	1.73E+07	1.59E+07	4.91E+06	1.90E+07	8.38E+06	1.32E+08	7.45E+07	9.34E+07	2.84E+07	**1.27E+06**	**1.24E+06**
F3	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.07E+12	4.92E+12	1.87E+08	6.96E+08	**0.00E+00**	**0.00E+00**
F4	5.62E+04	5.19E+03	**3.17E+03**	**9.28E+02**	2.30E+04	4.84E+03	4.17E+04	1.33E+04	5.41E+04	8.83E+03	8.04E+03	9.16E+03
F5	9.72E+02	2.23E+02	4.68E+01	1.83E+01	2.72E+02	2.26E+02	1.05E+04	6.30E+03	1.53E+03	4.81E+02	**0.00E+00**	**0.00E+00**
F6	4.66E+02	1.15E+02	8.93E+01	3.01E+01	1.25E+02	2.24E+01	1.84E+03	7.23E+02	2.63E+02	6.51E+01	**2.76E+01**	**2.38E+01**
F7	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.85E+02	1.02E+03	1.30E+01	2.05E+01	**0.00E+00**	**0.00E+00**
F8	**2.10E+01**	**5.68E-02**	**2.10E+01**	**6.46E-02**	**2.10E+01**	**3.85E-02**	**2.10E+01**	**4.49E-02**	**2.10E+01**	**5.20E-02**	**2.10E+01**	**4.91E-02**
F9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.52E+01	8.80E+00	4.74E+00	8.40E+00	**0.00E+00**	**0.00E+00**
F10	9.30E+02	1.79E+02	4.60E+01	1.81E+01	1.30E+02	6.83E+01	2.47E+03	6.58E+02	7.79E+02	2.50E+02	**8.06E-02**	**7.42E-02**
F11	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.72E+00	2.83E+00	4.78E+00	2.79E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F12	6.21E+01	8.27E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.33E+02	4.52E+01	1.62E+02	6.05E+01	**0.00E+00**	**0.00E+00**
F13	3.91E+01	7.01E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.39E+02	6.86E+01	1.15E+02	8.80E+01	**0.00E+00**	**0.00E+00**
F14	6.79E+03	3.67E+02	6.90E+03	3.72E+02	2.80E+03	8.27E+02	3.81E+03	5.20E+02	4.46E+03	5.45E+02	**2.17E+03**	**4.12E+02**
F15	7.48E+03	2.42E+02	7.12E+03	5.38E+02	6.97E+03	3.43E+02	5.96E+03	5.50E+02	**5.79E+03**	**5.35E+02**	6.92E+03	4.94E+02
F16	2.82E+00	2.98E-01	2.71E+00	3.78E-01	2.69E+00	3.03E-01	**2.04E+00**	**3.45E-01**	2.51E+00	4.53E-01	2.80E+00	3.61E-01
F17	5.55E+02	4.95E+01	2.18E+02	1.68E+01	9.27E+01	3.67E+01	1.07E+03	2.18E+02	4.39E+02	4.68E+01	**4.37E+01**	**2.07E+01**
F18	5.61E+02	8.01E+01	2.30E+02	1.82E+01	2.35E+02	3.54E+01	1.08E+03	2.30E+02	4.85E+02	6.41E+01	**1.81E+02**	**2.66E+01**
F19	1.12E+03	9.16E+02	1.94E+01	1.92E+00	1.22E+01	4.82E+00	2.49E+04	2.57E+04	2.84E+02	4.78E+02	**1.18E+01**	**2.87E+00**
F20	5.20E+00	5.60E+00	1.44E+00	3.30E+00	7.76E+00	4.44E+00	1.26E+01	2.23E+00	1.31E+01	2.68E+00	**0.00E+00**	**0.00E+00**
F21	6.63E+02	4.05E+01	4.05E+02	1.13E+00	4.26E+02	2.42E+01	1.53E+03	2.28E+02	6.26E+02	7.01E+01	**3.93E+02**	**3.59E+01**
F22	7.14E+03	3.89E+02	7.14E+03	3.27E+02	2.14E+03	6.26E+02	4.68E+03	8.61E+02	5.41E+03	5.10E+02	**1.93E+03**	**5.04E+02**
F23	7.63E+03	3.61E+02	7.77E+03	3.57E+02	**5.95E+03**	**1.10E+03**	6.39E+03	8.43E+02	6.74E+03	6.35E+02	6.65E+03	5.54E+02
F24	2.04E+02	3.00E+00	2.50E+02	1.05E+01	**2.01E+02**	**3.98E-01**	2.86E+02	3.13E+01	2.38E+02	2.57E+01	2.29E+02	3.38E+01
F25	2.78E+02	3.61E+01	**2.31E+02**	**2.91E+01**	2.70E+02	2.78E+01	3.29E+02	1.90E+01	3.02E+02	1.10E+01	2.82E+02	1.93E+01
F26	2.95E+02	6.62E-01	**2.38E+02**	**4.76E+01**	2.92E+02	3.11E+01	3.49E+02	5.17E+01	3.46E+02	4.44E+01	3.23E+02	2.78E+01
F27	2.05E+03	2.33E+02	**3.40E+02**	**9.44E+01**	3.62E+02	8.43E+01	1.08E+03	2.01E+02	9.65E+02	1.83E+02	8.68E+02	2.38E+02
F28	2.29E+03	2.46E+02	1.08E+03	3.95E+01	1.77E+03	6.05E+02	3.64E+03	4.34E+02	3.32E+03	1.06E+03	**1.05E+03**	**5.10E+02**

As can be seen from Tables 2 and 3, for the 30-dimensional optimization problem: IAOA achieved theoretical optimal values on nine functions, including F1, F3, F5, F7, F9, F11, F12, F13, and F20; ESCA achieved the theoretical optimal values on only four functions, including F3, F7, F9, and F11; IATTP obtained the theoretical optimal values on six functions, including F3, F7, F9, F11, F12, and F13; ICSA obtained the theoretical optimal values on five functions, including F3, F7, F9, F12, and F13; EAOA obtained a theoretical optimal value on F11 only; And the AOA did not converge to the theoretical optimal values of either function. It can be seen that IAOA obtains the most theoretical optimal values compared to AOA and the other four algorithms. Compared to IAOA, ESCA performs significantly better only on F27 but significantly worse on the 18 tested functions; IATTP performs similarly on the nine tested functions and performs significantly better performance on the four tested functions but significantly worse performance on the remaining 15 functions; ICSA performs significantly better only on F23, F26, and F27 but significantly worse on the 13 tested functions; AOA performs significantly better only on F16 but significantly worse on the 25 tested functions; EAOA performs significantly better only on F15 but significantly worse on the 23 tested functions. Combined with the Wilcoxon rank sum test results, it can be seen that IAOA has the best overall performance among the six optimization algorithms, and its advantage is the most obvious. IATTP has the 2nd best overall performance. ICSA has a slightly worse overall performance than IATTP. EAOA and ESCA are tied for 4th place. And AOA has the worst overall performance.

For a more intuitive comparison of the speed of convergence between the algorithms, the convergence curves for each algorithm on each test function are given in Figure 1, where the horizontal coordinate is the number of function evaluations, and the vertical coordinate is the logarithm of the fitness value.

Figure 1 shows that for single-mode functions F1–F5, IAOA can achieve global optimum on F1, F3, and F5 test functions, and its convergence rate on F1, F3, and F5 are all

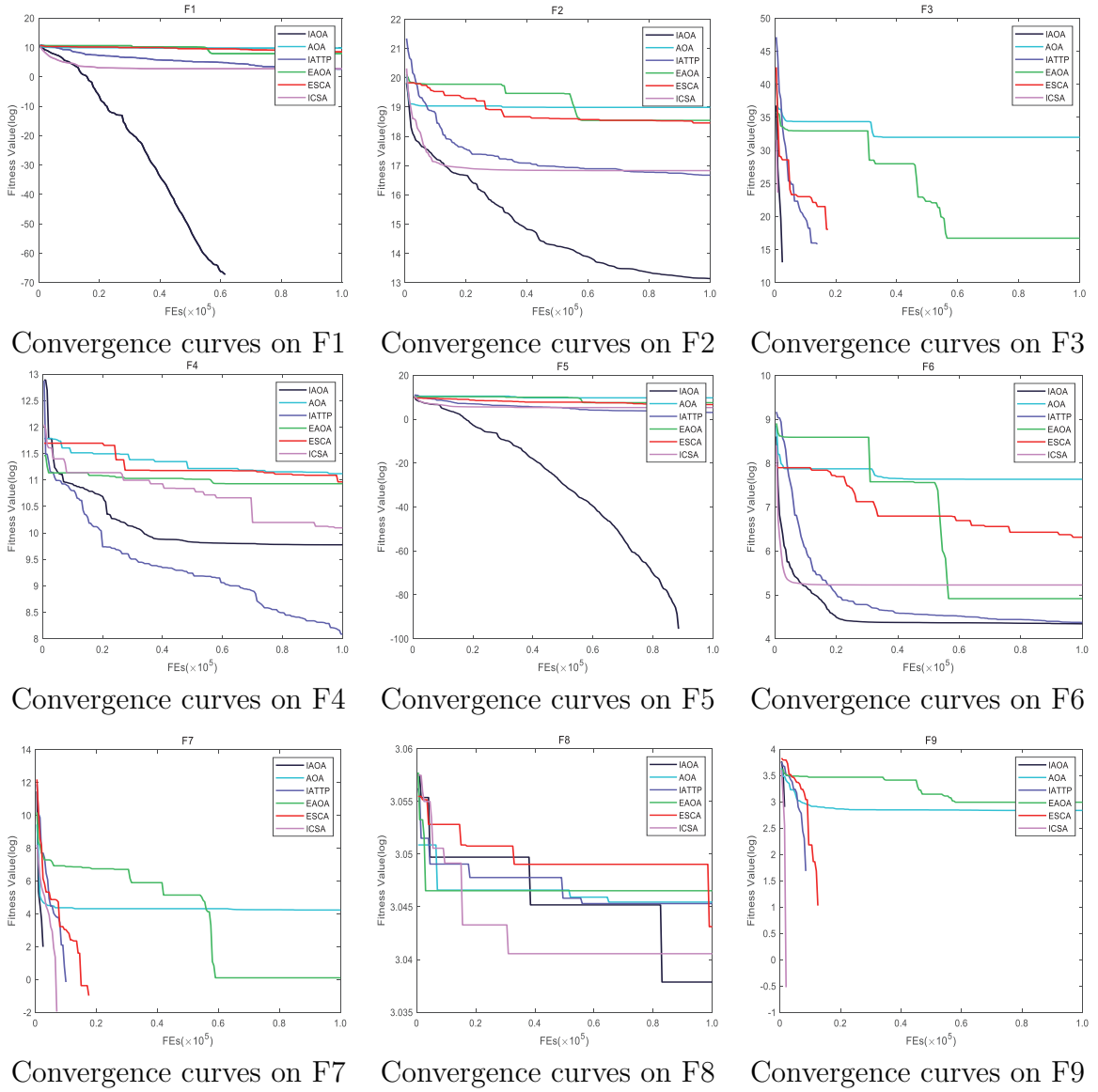
TABLE 3. Friedman test results and Wilcoxon rank sum test results of IAOA with other algorithms (vs.IAOA)

Function	SCA	IATTP	ICSA	AOA	EAOA
F1	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F2	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F3	1.000(=)	1.000(=)	1.000(=)	0.000(−)	0.011(−)
F4	0.000(−)	0.036(+)	0.000(−)	0.000(−)	0.000(−)
F5	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F6	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F7	1.000(=)	1.000(=)	1.000(=)	0.000(−)	0.000(−)
F8	0.967(=)	0.673(=)	0.083(=)	0.059(=)	0.013(−)
F9	1.000(=)	1.000(=)	1.000(=)	0.000(−)	0.000(−)
F10	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F11	1.000(=)	1.000(=)	0.000(−)	0.000(−)	1.000(=)
F12	0.000(−)	1.000(=)	1.000(=)	0.000(−)	0.000(−)
F13	0.001(−)	1.000(=)	1.000(=)	0.000(−)	0.000(−)
F14	0.000(−)	0.000(−)	0.002(−)	0.000(−)	0.000(−)
F15	0.000(−)	0.078(=)	0.876(=)	0.000(−)	0.000(+)
F16	0.923(=)	0.411(=)	0.222(=)	0.000(+)	0.011(−)
F17	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F18	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F19	0.000(−)	0.000(−)	0.801(=)	0.000(−)	0.000(−)
F20	0.000(−)	0.021(−)	0.000(−)	0.000(−)	0.000(−)
F21	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
F22	0.000(−)	0.000(−)	0.270(=)	0.000(−)	0.000(−)
F23	0.000(−)	0.000(−)	0.021(+)	0.176(=)	0.411(=)
F24	0.793(=)	0.000(−)	0.084(=)	0.000(−)	0.051(=)
F25	0.293(=)	0.000(+)	0.145(=)	0.000(−)	0.000(−)
F26	0.562(=)	0.000(+)	0.000(+)	0.000(−)	0.000(−)
F27	0.000(+)	0.000(+)	0.000(+)	0.000(−)	0.073(=)
F28	0.000(−)	0.000(−)	0.000(−)	0.000(−)	0.000(−)
+/=/−	1/9/18	4/9/15	3/12/13	1/2/25	1/4/23
Avg.rank	4.29	2.59	2.71	5.05	4.29
sort	4	2	3	6	4

comparable to that of ICSA. IAOA shows better convergence accuracy and the fastest convergence speed than the other five algorithms on F2. The convergence accuracy and convergence speed of IAOA on the F4 are slightly inferior to IATTP, but compared with the other four algorithms, IAOA can show the best convergence accuracy. In the early stage of evolution, the IAOA algorithm shows a good convergence speed, and in the late stage of evolution, the IAOA algorithm shows a good convergence accuracy. The IAOA algorithm is very competitive in the single module function compared to other algorithms; For the multi-mode functions F6-F20, IAOA can achieve global optimum on the test functions F7, F9, F11, F12, F13, and F20. IAOA shows the fastest convergence speed on all F7, F12, and F20 and convergence rate on F9, F11, and F13 are comparable to those of ICSA. IAOA exhibits better convergence accuracy and comparable convergence speed to the ICSA algorithm on the F6, F10, and F11 than the other five algorithms. IAOA shows better convergence accuracy and better convergence speed on the F8, F14, F17, F18, and F19 test functions, but it shows poorer convergence accuracy and convergence speed on the F15 and F16 test functions. In the early stage of evolution, IAOA converges only slightly slower than the ICSA algorithm. Reaching the late stage of evolution, IAOA is

able to continue searching for better solutions on most functions of the CEC2013 test set compared to other algorithms; For the composite functions F21-F28, IAOA shows better convergence accuracy and the fastest convergence speed than the other five algorithms on the F21 and F28 test functions. IAOA shows higher convergence accuracy and faster convergence on the F22 test functions. The convergence speed and convergence accuracy of IAOA on the F25 and F27 test functions are second to the two algorithms, ICSA and IATTP. On the F23, F24, and F26 test functions, IAOA has similar convergence accuracy to the three algorithms AOA, EAOA, and ESCA, where IAOA has a faster and higher convergence speed.

In summary, it shows that IAOA has certain advantages in terms of convergence speed and convergence accuracy compared to AOA and the other four superior optimization algorithms.

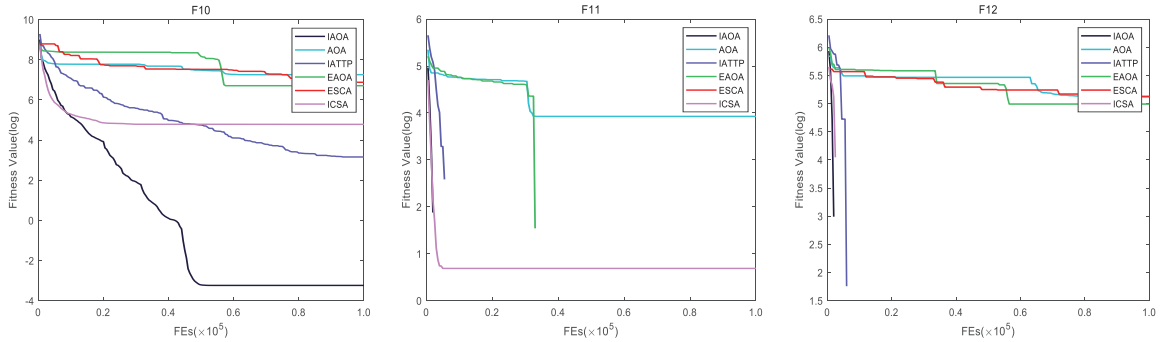


Convergence curves on F1 Convergence curves on F2 Convergence curves on F3

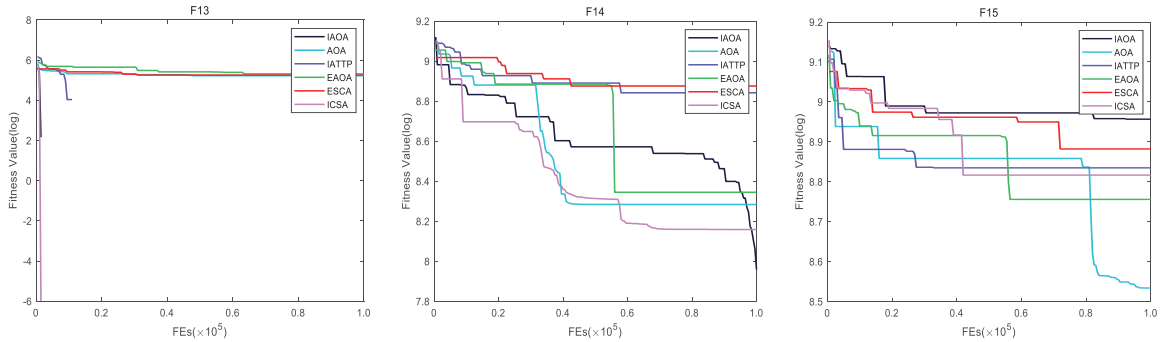
Convergence curves on F4 Convergence curves on F5 Convergence curves on F6

Convergence curves on F7 Convergence curves on F8 Convergence curves on F9

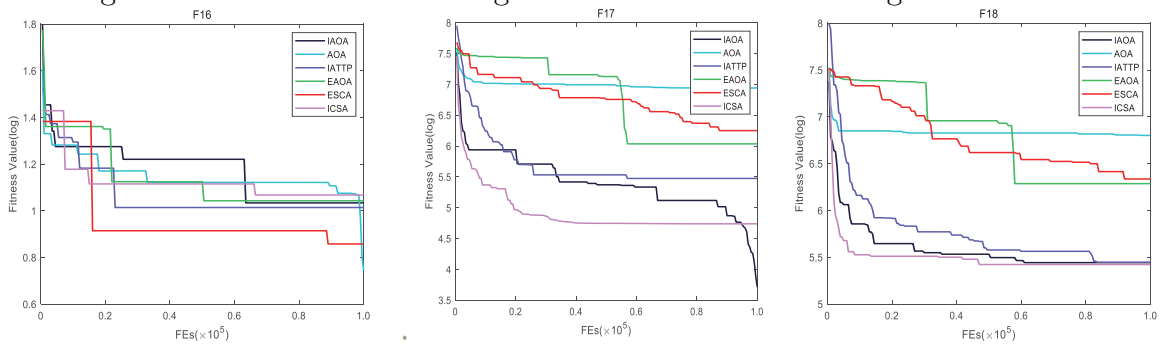
FIGURE 1. Convergence curve of each function



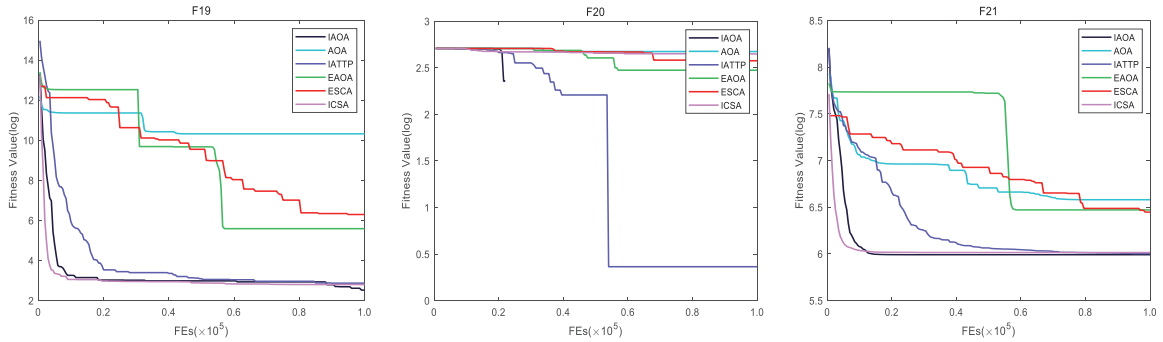
Convergence curves on F10 Convergence curves on F11 Convergence curves on F12



Convergence curves on F13 Convergence curves on F14 Convergence curves on F15

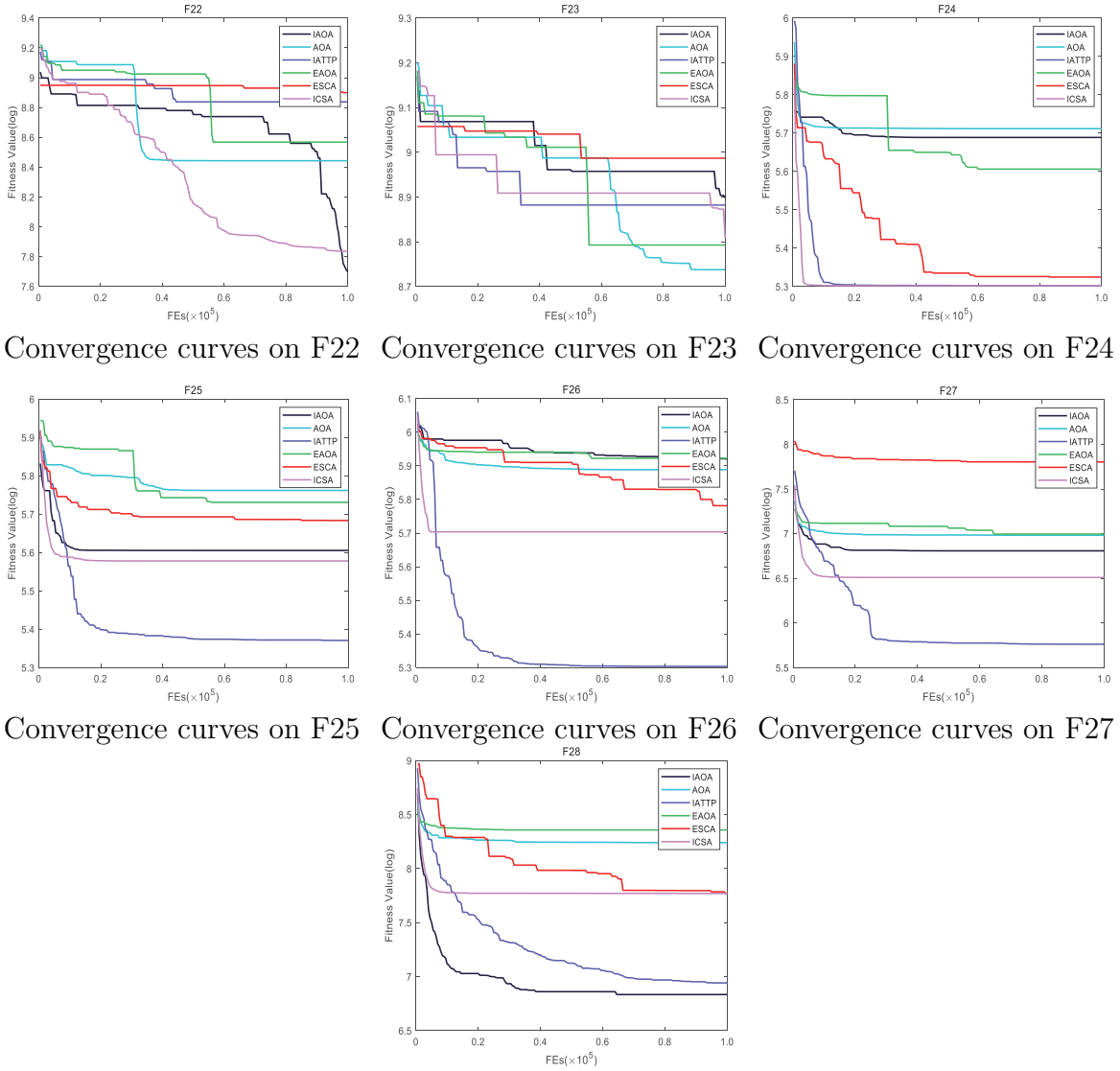


Convergence curves on F16 Convergence curves on F17 Convergence curves on F18



Convergence curves on F19 Convergence curves on F20 Convergence curves on F21

FIGURE 1. Convergence curve of each function



Convergence curves on F22 Convergence curves on F23 Convergence curves on F24

Convergence curves on F25 Convergence curves on F26 Convergence curves on F27

Convergence curves on F28

FIGURE 1. Convergence curve of each function

6. **Conclusion.** In this paper, an improved Archimedes optimization algorithm (IAOA) is proposed to further improve the convergence performance and the ability to jump out of the local optimum of the AOA algorithm. Firstly, the updating formulas of learning factor $C1$ and individual attribute acceleration in the exploration stage is improved, and the regulatory factor is introduced to improve the convergence speed of the algorithm in the early stage and maintain the population diversity; Secondly, this paper proposes a mechanism to update the individual position according to the probability and introduces an adjustment factor to improve the development ability of the algorithm; Finally, the simplex method is introduced to correct the position of poor individuals to improve the convergence speed of the algorithm. The experimental results on the CEC2013 test set show that the IAOA proposed in this paper is more competitive on single-mode and multi-mode problems compared with the other four optimization algorithms. However, the complexity of the algorithm is slightly higher. In the future research work, we can try to further reduce the complexity of AOA and use it to solve practical engineering problems.

REFERENCES

[1] Y. Liu, B. Cao, "A novel ant colony optimization algorithm with Levy flight," *IEEE Access*, vol. 8, pp. 67205–67213, 2020.

[2] F. Hemasian-Etefagh, F. Safi-Esfahani, "Group-based whale optimization algorithm," *Soft Computing*, vol. 24, no. 5, pp. 3647–3673, 2020.

[3] J. Kennedy, and R. Eberhart, "Particle swarm optimization," *IEEE*, vol. 4, pp. 1942–1948,1995

[4] X.-S. Yang, and A.-H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, 2012.

[5] S. Mirjalili, S.-M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61,2014

[6] R. Oftadeh, M.-J. Mahjoob, "A new meta-heuristic optimization algorithm: Hunting Search," *in 2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*. IEEE, 2009, pp. 1–5.

[7] A. Cheraghalipour, M. Hajiaghaei-Keshteli, and M.-M. Paydar, "Tree Growth Algorithm (TGA): A novel approach for solving optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 72, pp. 393–414, 2018.

[8] B. Abdollahzadeh, F.-S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems," *Computers & Industrial Engineering*, vol. 158, pp. 107408, 2021.

[9] I. Naruei, F. Keynia, "Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems," *Engineering with Computers*, pp. 1–32, 2021.

[10] K.-S. Lee, Z.-W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.

[11] A. Kaveh, M.-I. Ghazaan, "A new meta-heuristic algorithm: vibrating particles system," *Scientia Iranica*, vol. 24, no. 2, pp. 551, 2017.

[12] A. Kaveh, M. Khayatazad, "A new meta-heuristic method: ray optimization," *Computers & Structures*, vol. 112, pp. 283–294, 2012.

[13] A. Kaveh, M.-A. Motie Share, and M. Moslehi, "Magnetic charged system search: a new meta-heuristic algorithm for optimization," *Acta Mechanica*, vol. 224, no. 1, pp. 85–107, 2013.

[14] S. Kirkpatrick, Jr.-C.-D. Gelatt, and M.-P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[15] S. Rahmanzadeh, M.-S. Pishvaee, "Electron radar search algorithm: a novel developed meta-heuristic algorithm," *Soft Computing*, vol. 24, no. 11, pp. 8443–8465, 2020.

[16] F.-A. Hashim, K. Hussain, and E.-H. Houssein, "Archimedes optimization algorithm: a new meta-heuristic algorithm for solving optimization problems," *Applied Intelligence*, vol. 51, no. 3, pp. 1531–1551, 2021.

[17] C.-R. Houck, J. Joines, and M.-G. Kay, "A genetic algorithm for function optimization: a Matlab implementation," *Ncsu-ie tr*, vol. 95, no. 9, pp. 1–10, 1995.

[18] D. Karaboğa, and S. Ökdem, "A simple and global optimization algorithm for engineering problems: differential evolution algorithm," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 12, no. 1, pp. 53–60, 2004.

[19] O.-N. Oyelade, A.-E.-S. Ezugwu, and T.-I.-A. Mohamed, "Ebola optimization search algorithm: A new nature-inspired metaheuristic optimization algorithm," *IEEE Access*, vol. 10, pp. 16150–16177, 2022.

[20] B. Cheng, K. Li, B. Chen, "Scheduling a single batch-processing machine with non-identical job sizes in fuzzy environment using an improved ant colony optimization," *Journal of Manufacturing Systems*, vol. 29, no. 1, pp. 29–34, 2010.

[21] A. Layeb, Z. Benayad, "A novel firefly algorithm based ant colony optimization for solving combinatorial optimization problems," *International Journal of Computer Science and Applications*, vol. 11, no. 2, pp. 19–37, 2014.

[22] Z. Jia, J.-Y.-T. Leung, "An improved meta-heuristic for makespan minimization of a single batch machine with non-identical job sizes," *Computers & Operations Research*, vol. 46, pp. 49–58, 2014.

[23] G. Zhang, Y. Li, and Y. Shi, "Distributed learning particle swarm optimizer for global optimization of multimodal problems," *Frontiers of Computer Science*, vol. 12, no. 1, pp. 122–134, 2018.

[24] M. Abdel-Basset, G. Manogaran, and L. Abdel-Fatah, "An improved nature inspired meta-heuristic algorithm for 1-D bin packing problems," *Personal and Ubiquitous Computing*, vol. 22, no. 5, pp. 1117–1132, 2018.

[25] L.-L. Kang, R.-S. Chen, Y.-C. Chen, C.-C. Wang, X.-G. Li, and T.-Y. Wu, "Using Cache Optimization Method to Reduce Network Traffic in Communication Systems Based on Cloud Computing," *IEEE Access*, vol. 7, pp. 124397–124409, 2019

[26] K.-H. Truong, P. Nallagownden, and I. Elamvazuthi, "An improved meta-heuristic method to maximize the penetration of distributed generation in radial distribution networks," *Neural Computing and Applications*, vol. 32, no. 14, pp. 10159–10181, 2020

[27] F.-Q. Zhang, T.-Y. Wu, Yi. Wang, R. Xiong, G.-Y. Ding, P. Mei, and L.-Y. Liu, "Application of quantum genetic optimization of LVQ neural network in smart city traffic network prediction," *IEEE Access*, vol. 8, pp. 104555–104564, 2020

[28] A.-L.-H.-P. Shaik, M.-K. Manoharan, A.-K. Pani, R.-R. Avala, and C.-M. Chen, "Gaussian Mutation spider Monkey Optimization (GM-SMO) Model for Remote Sensing Scene Classification," *Remote Sensing*, vol. 14, no. 24, pp. 6279, 2022

[29] C.-M. Chen, S. Lv, J. Ning, and J.-M.-T. Wu, "A Genetic Algorithm for the Waitable Time-Varying Multi-Depot Green Vehicle Routing Problem," *Symmetry*, vol. 15, no. 1, pp. 124, 2023. https://doi.org/10.3390/sym15010124

[30] A.-S. Desuky, S. Hussain, and S. Kausar, "EAOA: an enhanced archimedes optimization algorithm for feature selection in classification," *IEEE Access*, vol. 9, pp. 120795–120814, 2021

[31] E.-H. Houssein, B.-E. Helmy, and H. Rezk, "An enhanced Archimedes optimization algorithm based on Local escaping operator and Orthogonal learning for PEM fuel cell parameter identification," *Engineering Applications of Artificial Intelligence*, vol. 103, pp. 104309, 2021

[32] X. Sun, G. Wang, and L. Xu, "Optimal estimation of the PEM fuel cells applying deep belief network optimized by improved archimedes optimization algorithm," *Energy*, vol. 237, pp. 121532, 2021

[33] B. Yao, and H. Hayati, "Model parameters estimation of a proton exchange membrane fuel cell using improved version of Archimedes optimization algorithm," *Energy Reports*, vol. 7, pp. 5700–5709, 2021

[34] J. A. Nelder, R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965

[35] J.-J. Liang, B.-Y. Qu, and P.-N. Suganthan, and A.-G. Hernández-Díaz, "Problem definitions an evaluation criteria for the CEC2013 special session on real-parameter optimization," *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report*, vol. 201212, no. 34, pp. 281–295, 2013

[36] Z. Feng, J. Duan, and W. Niu, "Enhanced sine cosine algorithm using opposition learning, adaptive evolution and neighborhood search strategies for multivariable parameter optimization problems," *Applied Soft Computing*, vol. 119, pp. 108562, 2022

[37] Y.-P. Xiao, H.-B. Chi, and Q.-Q. Li, "An improved artificial tree algorithm with two populations (IATTP)," *Engineering Applications of Artificial Intelligence*, vol. 104, pp. 104324, 2021

[38] J. Gholami, F. Mardukhi, and H.-M. Zawbaa, "An improved crow search algorithm for solving numerical optimization functions," *Soft Computing*, vol. 25, no. 14, pp. 9441–9454, 2021