Mathematical Biosciences and Engineering

MBE, 20(4): 6422–6467.
DOI: 10.3934/mbe.2023278
Received: 14 October 2022
Revised: 05 January 2023
Accepted: 17 January 2023
Published: 01 February 2023

http://www.aimspress.com/journal/MBE

Research article

An enhanced aquila optimization algorithm with velocity-aided global search mechanism and adaptive opposition-based learning

Yufei Wang[1], Yujun Zhang[1], Yuxin Yan[2], Juan Zhao[1,3] and Zhengming Gao[3,4,5,]*

[1] School of Electronics and Information Engineering, Jingchu University of Technology, Jingmen 448000, China
[2] Academy of Arts, Jingchu University of Technology, Jingmen 448000, China
[3] Institute of Intelligent Computing Technology, Jingchu University of Technology, Jingmen 448000, China
[4] School of Computer Engineering, Jingchu University of Technology, Jingmen 448000, China
[5] Hubei Engineering Research Center for Specialty Flowers Biological Breeding, Jingmen 448000, China

* **Correspondence:** Email: gaozming@jcut.edu.cn.

Abstract: The aquila optimization algorithm (AO) is an efficient swarm intelligence algorithm proposed recently. However, considering that AO has better performance and slower late convergence speed in the optimization process. For solving this effect of AO and improving its performance, this paper proposes an enhanced aquila optimization algorithm with a velocity-aided global search mechanism and adaptive opposition-based learning (VAIAO) which is based on AO and simplified Aquila optimization algorithm (IAO). In VAIAO, the velocity and acceleration terms are set and included in the update formula. Furthermore, an adaptive opposition-based learning strategy is introduced to improve local optima. To verify the performance of the proposed VAIAO, 27 classical benchmark functions, the Wilcoxon statistical sign-rank experiment, the Friedman test and five engineering optimization problems are tested. The results of the experiment show that the proposed VAIAO has better performance than AO, IAO and other comparison algorithms. This also means the introduction of these two strategies enhances the global exploration ability and convergence speed of the algorithm.

Keywords: aquila optimizer; simplified aquila optimization algorithm; swarm intelligence algorithm; velocity-aided global search; adaptive opposition-based learning; engineering problem

1. Introduction

With the development of science and technology in human society, traditional technology can no longer not be used to solve complex practical problems. Therefore, the meta-heuristic algorithm came into being. Meta-heuristic algorithms are widely used in these complex practical problems because of their simple concept and flexible parameters. At present, the proposed algorithms are mainly divided into four categories [1], The first category is swarm-based algorithms, such as Artificial Bee Colony (ABC) [2]. The second category is human-based algorithms, such as Teaching based learning algorithm (TBLA) [3]. The third category is physics-based algorithms, such as Fireworks Algorithm (FWA) [4]. The last category is evolutionary algorithms, such as Tree Growth Algorithm (TGA) [5]. The classification of the meta-heuristic algorithm is shown in Table 1.

Table 1. Classification of meta-heuristic algorithm.

Classes	Algorithm
Evolutionary	Tree Growth Algorithm (TGA) [5]
	Arithmetic Optimization Algorithm (AOA) [6]
	Genetic Algorithm (GA) [7]
	Differential evolution (DE) [8]
	Genetic Programming (GP) [9]
	Evolutionary Strategies (ES) [10]
Human-based	Teaching based learning algorithm (TBLA) [3]
	Harmony Search (HS) [11]
	Imperialist Competitive Algorithm (ICA) [12]
	Fireworks Algorithm (FWA) [4]
	Collective Decision Optimization (CSO) [13]
	Socio Evolution & Learning Optimization Algorithm (SELOA) [14]
	Political Optimizer (PO) [15]
Physics-based	Henry Gas Solubility Optimization (HGSO) [16]
	Big Bang–Big Crunch (BBBC) [17]
	Multi-verse Optimizer (MVO) [18]
	Electromagnetic Field Optimization (EFO) [19]
	Gravitational Search Algorithm (GSA) [20]
	Thermal Exchange Optimization (TEO) [21]
	Central Force Optimization (CFO) [22]
Swarm-based	Artificial Bee Colony (ABC) [23]
	Particle Swarm Optimization (PSO) [24]
	Moth Flame Optimization (MFO) [25]
	Ant Colony Optimization (ACO) [26]
	Marine Predators Algorithm (MPA) [27]
	Seagull optimization algorithm (SOA) [28]
	Sooty Tern Optimization Algorithm (STOA) [29]
	Aquila Optimizer (AO) [30]

Continued on next page

Classes	Algorithm
Swarm-based	Grey Wolf Optimizer (GWO) [31]
	Salp Swarm Algorithm (SSA) [32]
	Harris Hawks Optimization (HHO) [33]
	Monarch Butterfly Optimization (MBO) [34]
	Slime Mould Algorithm (SMA) [35]
	Moth Search Algorithm (MSA) [36]
	Hunger Games Search (HGS) [37]
	Runge Kutta Method (RUN) [38]
	Colony Predation Algorithm (CPA) [39]
	Welghted mean of vectors (INFO) [40]
	Fick's Law Algorithm(FLA) [41]
	Ant Lion Optimization [42]

However, not all problems can be solved [43] by these algorithms, especially the multidisciplinary problems that have become increasingly complex in recent years. Therefore, we need to find better algorithms for solving multidisciplinary and complex problems.

The improvement for AO is less because the time to propose the algorithm is not long, so far there is not enough research on AO algorithms. Wang et al. [44] proposed a hybrid improvement of the AO algorithm and the HHO algorithm and added hunting strategies and adversarial-based learning rules to them. It has been verified by experiments that the improved algorithm has a very good performance. Elaziz et al. [45] proposed a Mobile Net V3 AO hybrid model. He uses the AO algorithm to reduce the dimensionality of the image, thereby improving the accuracy of the data. Zhang et al. [46] proposed the hybridization of the AO algorithm and the AOA algorithm. Zhao et al. [47] applied the JAYA algorithm to the parameter extraction of photovoltaic models and added a chaotic adaptive strategy. Zhang et al. [48] used the enhanced adaptive comprehensive learning hybrid algorithm of Rao-1 and JAYA algorithm for photovoltaic model parameter extraction. Zhou et al. proposed [49] metaphor-free dynamic spherical evolution for parameter estimation of photovoltaic modules, it helps the research on the application of swarm intelligence algorithm in photovoltaic modules. We believe that the enhanced AO algorithm can be used in photovoltaic model parameter extraction. In the future, we will also conduct relevant research. Simrandeep Singh et al. [50] proposed Arithmetic optimization based image segmentation. Hussien et al. [51] proposed comparative research and application of Harris Eagle optimization research progress. Wang et al. [52] proposed an enhanced remora optimization algorithm. Hashim et al. [53] proposed a snake optimizer. Zheng et al. [54] proposed an improved wild horse optimizer. Hussien et al. [55] proposed an enhanced COOT optimization algorithm for dimensionality reduction. Yu et al. [56] proposed an enhanced aquila optimizer for global optimization and constrained engineering problems. Yang et al. [57] proposed an efficient DBSCAN optimized by the arithmetic optimization algorithm. Cui et al. [58] proposed a modified slime mold algorithm via levy flight. Yang et al. proposed [59] an opposition learning and spiral modelling based arithmetic optimization algorithm. Hussien et al. [60] proposed boosting whale optimization with an evolution strategy and Gaussian random walks, and used it to image segmentation. Hussien et al. [61] proposed a comprehensive review of moth-flame optimization: variants, hybrids, and applications. Also, swarm intelligence algorithms are used to solve real-world engineering problems. Yu et al. [62] proposed laplace crossover and random replacement strategy boosted Harris hawks optimization. Qi et al. [63]

proposed directional mutation and crossover for the immature performance of whale algorithm with application to engineering optimization. Zhao et al. [64] proposed opposition-based ant colony optimization with an all-dimension neighborhood search for engineering design. Zhou et al. [65] proposed advanced orthogonal learning and Gaussian barebone hunger games for engineering design.

Considering the powerful global exploration capability of the AO and early convergence [44] caused by insufficient exploration and exploitation in the later phases, an enhanced AO with velocity-aided global search mechanism and adaptive opposition-based learning, called VAIAO. The proposed VAIAO has better search ability than the original algorithm, and it is got better performance easily. This article was inspired by the velocity-aided global search mechanism [66], and this paper applies the velocity-aided global search Mechanism to the AO. On the other land, the adaptive opposition-based learning rule is introduced to avoid being trapped in the local optimum.

The performance of the VAIAO is tested by using 27 classical benchmark functions, the Wilcoxon statistical test, and five engineering optimization problems. In order to compare the various experimental results, IAO [67], AO [30] and some well-known algorithms including Arithmetic AOA [6], HHO [33], STOA [29], ChOA [68], SOA [28], PSO [24], DE [8] are introduced. The results of the experiment show that the proposed VAIAO is better than another algorithm.

The basic AO and improved strategies will be introduced later. And carry out simulation experiments and engineering optimization problems using VAIAO and comparison algorithms to solve. Finally, the conclusion is given at the end of the article.

2. Aquila optimizer

When the Aquila is preying, it can switch four strategies in different situations when facing the prey in different situations to escape the eagle's predation strategy, which can be represented by the following four strategies.

2.1. Strategy 1: Flying high for searching prey

In this strategy, Aquila flies through high-altitude hunting areas to find the best spot for prey, and once it finds the best spot, Aquila swoops vertically at prey. This behavior can be expressed by the formal.

$$X(t+1) = X_{best}(t) \times \left(1 - \frac{t}{T}\right) + (X_M(t) - X_{best}(t) \times rand) \tag{1}$$

where, $X_{best}(t)$ represent the global optimal position, $X_M(t)$ represents the current mean position, and t and T represent the current number of iterations and maximum number of iterations.

2.2. Short glide attack and fight

In this strategy, Aquila will switch from flying at high altitude to hovering on the head of the prey, preparing for Aquila.

$$X(t+1) = X_{best}(t) \times LF(D) + X_R(t) + (\mathrm{y} - \mathrm{x}) \times \mathrm{rand} \tag{2}$$

where, the Aquila's random position is $X_R(t)$, and D is the size of dimension. LF is Levy flight function. The y and x represents the shape of the search, which can be expressed by the formula:

$$\begin{cases} x = (r_1 + 0.00565 \times D_1) \times sin\left(-\omega \times D_1 + \frac{3\times\Pi}{2}\right) \\ y = (r_1 + 0.00565 \times D_1) \times cos\left(-\omega \times D_1 + \frac{3\times\Pi}{2}\right) \end{cases} \tag{3}$$

$$LF(x) = 0.01 \times \frac{\mu\times\sigma}{|v|^{\frac{1}{\beta}}},\ \sigma = \left(\frac{r(1+\beta)\times sin\left(\frac{\Pi\beta}{2}\right)}{r\left(\frac{1+\beta}{2}\right)\times\beta\times 2^{\left(\frac{\beta-1}{2}\right)}}\right)^{\frac{1}{\beta}} \tag{4}$$

where, r_1 is Aquila's search cycles which number between $[1, 20]$, D_1 is a random integer from one to D dimensions, and ω is a constant of 0.005.

2.3. Approaching prey and attack

In this strategy, Aquila first finds and determines the approximate location of the prey through strategy 2. After finding the prey, Aquila descends vertically for initial hunting. This behavior can be expressed by the formula:

$$X(t+1) = (X_{best}(t) - X_M(t) \times \alpha - rand + \left((UB - LB)rand + LB\right) \times \delta \tag{5}$$

where α and δ are adjustment parameters during the development process, fixed at 0.1, UB and LB are the upper and lower bounds of the search space, respectively.

2.4. Catching prey on land

In this strategy, Aquila will follow the prey's escape track to land and chase it, then attack it. This behavior be expressed by the formula:

$$(t+1) = QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times LF(D) + rand \times G_1 \tag{6}$$

$$QF(t) = t^{\frac{2\times rand-}{(1-T)^2}} \tag{7}$$

$$\begin{cases} G_1 = 2 \times rand - 1 \\ G_2 = 2 \times \left(1 - \frac{t}{T}\right) \end{cases} \tag{8}$$

where, G_1 is the random number which between [1,–1], and G_2 is the flight slope when Aquila tracking the prey between [2,0].

3. The simplified IAO

The IAO [67] is a simplification of the AO [69]. Although the AO algorithm has better convergence speed and exploration ability, the AO algorithm has poor convergence performance in the later stage. For solving this problem, the IAO algorithm selects only strategies 1 and 2 based on the Aquila algorithm. In this way, IAO algorithm can avoid the situation that the convergence speed decreased in the later period. The simplified improved Aquila will be easier than ever.

4. The proposed VAIAO algorithm

During exploration, individuals perform fast flights and hunts of Aquila in the search space. Even though the simplified algorithm speeds up the convergence and improves the search ability. Individuals can still easily fall into local optima. And the speed of convergence and exploration ability of AO can be strengthened That's why we should improve and overcome these shortcomings.

4.1. Velocity-Aided global search mechanism

When Aquila individuals move in the search space without considering the two features of velocity and acceleration, these movements do not always occur continuously or smoothly. The movement of search agents in the search space may break down, then leading to potential drift. Due to this reason, search agents may drift when executing search strategies. Considering the velocity term can avoid premature convergence by helping the search agent maintain its unique trajectory, enhancing the search agent's exploration capabilities, and balancing the exploration and development phases of the optimization process. In VAIAO, an initial random velocity is defined for each Aquila search agent as it performs a search policy move, and an initial random position is defined for each search agent in the search space. Therefore, each search agent has a velocity and position in each dimension of the optimization problem. This idea be expressed by this formula:

$$V_j^{t+1} = k \times \left(sgn\left(A_j^t\right) \times \left|V_j^t\right|\right) + A_j^t \times D_j^t \tag{9}$$

where, V_j^t represent the velocity of the Aquila search agent in the jth dimension. sgn is the sign function. A_j^t represents the acceleration terms of the search agents, and A_j^t be expressed by this formula:

$$A_j^t = (2 \times r_1 - 1) \times a^2 \tag{10}$$

where, r_1 represents a random number between 1 and 0, Furthermore, a is a linearly decreasing parameter determined as follows:

$$a = a_{max} - (a_{max} - a_{min}) \times \frac{t-1}{t_{max}-1} \tag{11}$$

D_j^t a represents the modified distance between the focused search agent and the leading Aquila. These distances in each dimension can be calculated as follows:

$$D_j^t = \left|C_j^t \times X_{1,j}^t - X_{i,j}^t\right| \tag{12}$$

$$C_j^t = 1 + (2 \times r_2 - 1) \times c^2 \tag{13}$$

$$c = c_{max} - (c_{max} - c_{min}) \times \frac{t-1}{t_{max}-1} \tag{14}$$

where, C_j^t can take these uncertainties into account and help the algorithm better perform the exploration phase. Especially in the early iterations of the optimization process. c is an adaptively determined parameter as follows. r_2 represents a random number between 0 and 1.

k acts as a tuning parameter, acting as an inertia weight, while facilitating a proper and reliable transition from exploration to exploitation, and is iterated through iterative computation as follows:

$$k = k_{max} - (k_{max} - k_{min}) \times \frac{t-1}{t_{max}-1} \tag{15}$$

Finally, the next position of the search agent can be expressed by this formula:

$$X_{1,j}^{t+1} = X_j^{t+1} - V_j^{t+1} \tag{16}$$

The new position of search agent is calculated by this formula:

$$X_{i,j}^{t+1} = \frac{\left(X_{1,j}^t - X_{1,j}^{t+1}\right)}{2} \tag{17}$$

where, is the position of the ith search agent (Aquila) in the jth dimension in the (t+1)th iteration.

4.2. Adaptive opposition-based learning

The starting point of introducing reverse learning into IAO is to get rid of the precocious dilemma when it falls into local convergence through reverse learning. Introducing the reverse learning probability of reverse learning requires a large amount of experimental data to determine its value in a hybrid algorithm designed by the original reverse learning and other algorithms. Hussien et al. [70] proposed opposition-based learning and chaotic local search strategy are used in Harris Hawks optimization algorithm for global optimization and feature selection. Hussien et al. [71] An enhanced opposition-based Salp Swarm Algorithm. Therefore, we find that the adaptive opposition-based learning rules can help the algorithm jump out of the local optimum. That’s why we introduce adaptive opposition-based learning rules to improve the IAO algorithm. It can be expressed by this formula:

$$X_{jOBL}(t+1) = LB_j + UB_j - X_j \tag{18}$$

where, LB_j is the lower bound of the current problem in the jth dimension, and UB_j is the upper bound of the current problem in the jth dimension.

4.3. Overview of the proposed VAIAO algorithm

The flowchart of the VAIAO algorithm is given in Figure 1. The pseudo-code for the VAIAO algorithm is shown in Table 2. The computational time complexity is O (T $\times$ D $\times$ N).

Table 2. The pseudo-code of the VAIAO algorithm.

```
Initialization of population size N and the positions of Individuals  Xi(i = 1,2,······, n)
While(t ≤ T)
   For i = 1:N
      Updata X(t)
      Updata the parameters
      If rand<0.5
      Updata X(t + 1) with Eq.(1)
      Else
      Updata X(t + 1) with Eq.(2)
      End
   For i = 1:N
      Updata X(t + 1) with Eq.(9)-Eq.(17)
   End For
   For i = 1:N
      Updata X(t + 1) with Eq.(18)
   End For
   For i = 1:N
      Check boundaries
   Calculate fitness of X(t)
      Updata X_best(t)
   End For
      t+t+1
   End While
Return X_best(t)
```

5. Experiments and results

In this section, the 27 classic functions, the Wilcoxon test and Friedman is used to test the performance of VAIAO algorithm. The results of this experimental show that the VAIAO algorithm has better performance than the original AO algorithm.

5.1. Experimental setup

In this section, some well-known algorithms are selected to verify the performance of the proposed VAIAO algorithm, such as IAO [67], AO [30], AOA [6], HHO [33], STOA [29], ChOA [68], SOA [28], PSO [24], DE [8]. To ensure the fairness of the experimental, set the population data to 30, and set the number of iterations is 500. The worst value, average value, optimal value, median value, and standard deviation of the output are compared to set the parameters of the algorithm according to the original version. The parameter settings for the comparative algorithms are shown in Table 3.

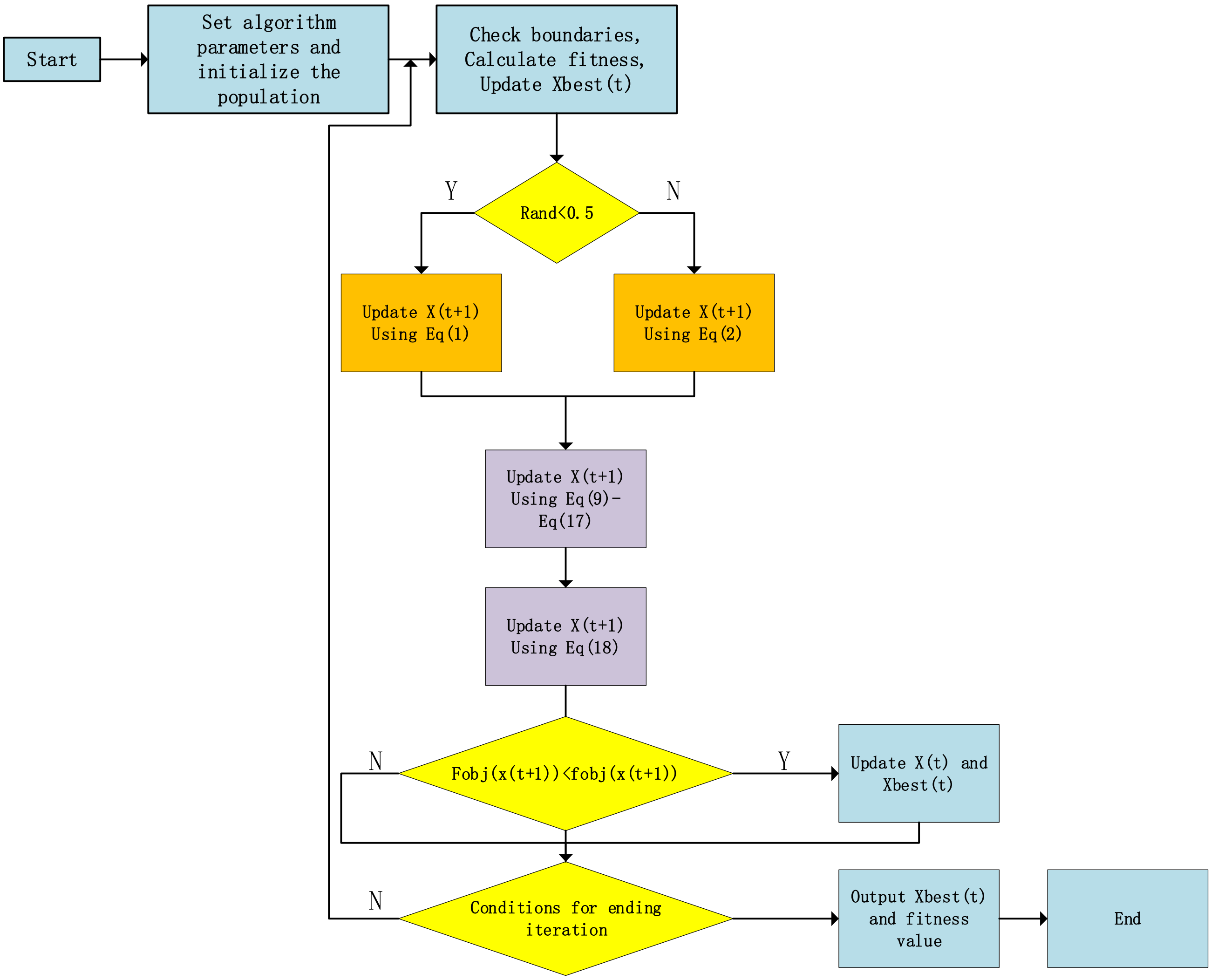

Figure 1. The flowchart of the VAIAO algorithm.

5.2. Test function

In this section, there are 27 typical benchmark selection functions, including 9 single-peak functions (shown in Table 4), 9 two-dimensional multi-channel functions (shown in Table 5), and 9 multidimensional multi-peak functions (shown in Table 6). These benchmark functions are used to test the basic performance of the proposed OLAO algorithm. All simulation experiments will be conducted on Intel(R) Core (TM) i5-10300H CPU @2.50GHz 3106×2 core and MATLAB 2017b software will be used.

Table 3. Para settings for the comparative algorithm.

Algorithm	Parameters
VAIAO	$a \in [\sqrt{2}, 0]; c \in [1,0]; k = 0.9,0.4$
IAO	$r \in [1,0]$
AO	$U = 0.00565; r_1 = 10; \omega = 0.005; \alpha = 0.1; \delta = 0.1; G1 \in [-1,1]; G2 \in [2,0]$
AOA	$r_1 \in [1,0]; r_2 \in [1,0]; r_3 \in [1,0]$
HHO	$q \in [0,1]; r \in [0,1]; E0 \in [-1,1]; E1 \in [2,0]; E \in [-2,2]$
STOA	$Sa \in [2,0]; r1 \in [0.5,0]; r2 \in [0.5,0]; b = 1$
ChOA	$f \in [2,0]; r1 \in [0,1]; r2 \in [0,1]$
SOA	$Fc \in [2,0]; r1 \in [1,0]; r2 \in [1,0]; b = 1$
PSO	$C_1 = 1.49445, C_2 = 1.49445, v \in [-0.5,0.5]$
DE	$F_0 = 0.5, C_R = 0.7$

Table 4. Unimodal benchmark functions.

Functions	Description	Dimensions	Range	fmin
F1	$f(x) = \sum_{i=1}^{D} x_i^2$	30,60,100,300,500	[−100,100]	0
F2	$f(x) = \sum_{i=1}^{D} \lvert x_i \rvert$	30,60,100,300,500	[−100,100]	0
F3	$f(x) = 10^{6\frac{i-1}{D-1}} \times \sum_{i=1}^{D} x_i^2$	30,60,100,300,500	[−100,100]	0
F4	$f(x) = \sum_{i=1}^{D} (x_{i+1}^2 * x_i^2)$	30,60,100,300,500	[−100,100]	0
F5	$f(x) = max\{\lvert x_i \rvert, 1 \leq i \leq D\}$	30,60,100,300,500	[−100,100]	0
F6	$f(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30,60,100,300,500	[−100,100]	0
F7	$f(x) = \sum_{i=1}^{D} i x_i^2$	30,60,100,300,500	[−100,100]	0
F8	$f(x) = f(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} i x_i^2)^2$	30,60,100,300,500	[−100,100]	0
F9	$f(x) = \sum_{i=1}^{D} i x_i^4$	30,60,100,300,500	[−100,100]	0

Table 5. Two-dimensional multimodal functions.

Functions	Descriptions	Dimensions	Ranges	fmin
F10	$f(x) = 7x_1^2 - 6\sqrt{3}x_1x_2 + 13x_2^2$	2	[−100,100]	0
F11	$f(x) = 195.6316 - 200e^{\left(-0.02\sqrt{x_1^2+x_2^2}\right)} + 5e^{(cos3x_1+sin3x_2)}$	2	[−100,100]	0
F12	$f(x) = (x_1^2 + x_2^2)^{0.25} \times (1 + \sin(50(3x_1^2 + x_2^2)^{0.1})^2)$	2	[−100,100]	0
F13	$f(x) = 0.5 + \frac{(sin\left(\sqrt{x_1^2 + x_2^2}\right)^2 - 0.5)}{(1 + 0.001(x_1^2 + x_2^2))^2}$	2	[−5,5]	0
F14	$f(x) = 0.1 + \sin(x_1)^2 + \sin(x_2)^2 - 0.1e^{(-x_1^2-x_2^2)}$	2	[−10,10]	0
F15	$f(x) = \left(sin4 - \frac{\sin\left(2\sqrt{x_1^2 + x_2^2}\right)}{2} + \frac{\sin\left(3\sqrt{x_1^2 + x_2^2}\right)}{3} - \frac{\sin\left(4\sqrt{x_1^2 + x_2^2}\right)}{4} + 4\right) \times \cos\left(argtan\frac{x_2}{x_1} - \pi\right) + 2$	2	[−100,100]	0
F16	$f(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$	2	[−100,100]	0
F17	$f(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	2	[−100,100]	0
F18	$f(x) = 0.5 + \frac{(\left(sin\left(\sqrt{x_1^2 + x_2^2}\right)^2 - 0.5\right))}{(1 + 0.001(x_1^2 + x_2^2))^2}$	2	[−20,20]	0

Table 6. Multi-dimensional multimodal functions.

Functions	Descriptions	Dimensions	Ranges	fmin
F19	$f(x) = 1 + (\sum_{i=1}^{D} \sin(x_i)^2 - e^{(-\sum_{i=1}^{D} x_i^2)} \times e^{(-\sum_{i=1}^{D} \sin(\sqrt{\lvert x_i \rvert})^2)}$	30,60,100,300,500	[−100,100]	0
F20	$f(x) = \sum_{i=1}^{D} x_i^6(2 + \sin(\frac{1}{x_i}))$	30,60,100,300,500	[−100,100]	0
F21	$f(x) = \sum_{i=1}^{D} [100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2]$	30,60,100,300,500	[−100,100]	0
F22	$f(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	30,60,100,300,500	[−100,100]	0

Continued on next page

Functions	Descriptions	Dimensions	Ranges	fmin
F23	$f(x)=\frac{D}{10}+\sum_{i=1}^{D}x_i^2-\frac{\sum_{i=1}^{D}\cos(5\pi x_i)}{10}$	30,60,100,300,500	[−100,100]	0
F24	$f(x)=1+\frac{\sum_{i=1}^{D}x_i^2}{4000}-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})$	30,60,100,300,500	[−100,100]	0
F25	$f(x)=1-\cos(2\pi\sqrt{\sum_{i=1}^{D}x_i^2}+0.1\sqrt{\sum_{i=1}^{D}x_i^2})$	30,60,100,300,500	[−100,100]	0
F26	$f(x)=10D+\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)]$	30,60,100,300,500	[−100,100]	0
F27	$f(x)=-20e^{\left(-2\sqrt{\frac{\sum_{i=1}^{D}x_i^2}{D}}\right)}-e^{\left(\frac{\sum_{i=1}^{D}\cos(2\pi x_i)}{D}\right)}+20+e$	30,60,100,300,500	[−100,100]	0

5.3. Experimental series 1: Intensification capability experiments

In this section, the test function contains an unimodal test function and a multimodal test function. The algorithm tests the single peak function to verify the algorithm's exploration ability is effective, because the unimodal test function only has one global optimal value. The results of the experiment are shown in Table 7. In most instances, the VAIAO algorithm has the best convergence performance in almost all case and has better performance on unimodal functions than other algorithm.

The multimodal test functions include many local optimal solutions and global optimal solutions. With the increase of the dimension, the algorithm with poor performance can easily fall into local optimum. Therefore, it is very effective to use these functions to test the diversity of algorithms. The results of the multimodal two-dimensional in this section are shown in Tables 8 and 9. In the low-dimensional diversification experiment, the VAIAO algorithm proposed in this paper can converge to 0 many times, and obtain the optimal value through the data in Tables 8 and 9.

Table 7. Test results of Unimodal function (F1–F9), the dimension is fixed to 30.

Fun	Items	SOIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F1	Average	**0.0000**	1.3374×10^{-291}	5.2757×10^{-152}	2.4485×10^{-93}	3.6079×10^{-102}	1.9011×10^{-6}	1.3008×10^{-5}	2.8797×10^{-12}	4.1938×10^{-4}	4.5181×10^{-4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F2	Average	$\mathbf{3.7644 \times 10^{-147}}$	9.7308×10^{-146}	6.5298×10^{-59}	6.3692×10^{-52}	6.4248×10^{-51}	1.7544×10^{-4}	2.1821×10^{-4}	2.7349×10^{-7}	2.0394×10^{-2}	9.6032×10^{2}
	Std	$\mathbf{8.4104 \times 10^{-147}}$	1.2976×10^{-145}	1.4601×10^{-58}	1.4242×10^{-51}	1.2502×10^{-50}	1.1095×10^{-4}	2.4623×10^{-4}	1.9075×10^{-7}	8.9718×10^{-3}	38.142
F3	Average	$\mathbf{1.1656 \times 10^{-300}}$	4.2723×10^{-296}	1.4097×10^{-141}	3.6188×10^{-3}	1.5452×10^{-94}	74.685	1.4667×10^{2}	6.1053×10^{-6}	2.1579×10^{3}	6.7414×10^{11}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F4	Average	0.0000	0.0000	1.2913×10^{-296}	0.0000	2.3320×10^{-189}	6.1574×10^{-18}	3.5583×10^{-16}	5.9353×10^{-28}	2.4914×10^{-8}	5.1324×10^{7}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	1.0005×10^{-17}	5.0628×10^{-16}	9.0818×10^{-28}	5.3972×10^{-8}	7.9633×10^{6}
F5	Average	3.7482×10^{-145}	$\mathbf{3.1301 \times 10^{-150}}$	1.1888×10^{-52}	2.7044×10^{-2}	2.0140×10^{-49}	4.8732×10^{-2}	0.34245	1.2416×10^{-2}	1.7174	74.286
	Std	8.3764×10^{-145}	$\mathbf{4.4009 \times 10^{-150}}$	2.6583×10^{-52}	2.5004×10^{-2}	2.8430×10^{-49}	6.8444×10^{-2}	0.35991	2.5710×10^{-2}	0.40172	4.1678
F6	Average	$\mathbf{3.6797 \times 10^{-143}}$	3.8905×10^{-142}	4.3251×10^{-71}	0.0000	2.3336×10^{-49}	5.5061×10^{-6}	6.6415×10^{-5}	2.6600×10^{-8}	1.9753×10^{-2}	5.4108×10^{8}
	Std	$\mathbf{8.2280 \times 10^{-143}}$	8.6993×10^{-142}	9.6703×10^{-71}	0.0000	5.0420×10^{-49}	4.7519×10^{-6}	1.1828×10^{-4}	8.7207×10^{-9}	1.1563×10^{-2}	7.3281×10^{8}
F7	Average	$\mathbf{8.8651 \times 10^{-308}}$	9.0167×10^{-263}	6.9932×10^{-110}	3.5958×10^{-30}	1.0178×10^{-94}	6.0971×10^{-6}	8.9469×10^{-5}	2.2386×10^{-10}	8.4813×10^{-4}	6.4148×10^{5}
	Std	0.0000	0.0000	1.5637×10^{-109}	8.0405×10^{-30}	2.2759×10^{-94}	4.7054×10^{-6}	1.3921×10^{-4}	3.3203×10^{-10}	8.9580×10^{-4}	6.9981×10^{4}

Continued on next page

Fun	Items	SOIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F8	Average	0.0000	0.0000	3.9573×10^{-276}	4.4146×10^{-36}	3.9112×10^{-193}	1.5903×10^{-10}	2.1632×10^{-6}	2.0882×10^{-18}	5.0434×10^{-6}	1.7330×10^{8}
	Std	0.0000	0.0000	0.0000	9.8714×10^{-36}	0.0000	2.5138×10^{-10}	4.8222×10^{-6}	4.0422×10^{-18}	4.4984×10^{-6}	2.7526×10^{7}
F9	Average	0.0000	0.0000	2.2060×10^{-282}	9.5734×10^{-119}	6.6071×10^{-207}	4.0249×10^{-9}	2.2641×10^{-5}	1.2280×10^{-19}	1.4946×10^{-4}	2.3440×10^{9}
	Std	0.0000	0.0000	0.0000	2.1407×10^{-118}	0.0000	6.8554×10^{-9}	4.7015×10^{-5}	1.2419×10^{-19}	2.5801×10^{-4}	2.6115×10^{8}

Table 8. Test results of multimodal two-dimensional function (F10–F18), the dimension is fixed to 30.

Fun	Items	SOIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F10	Average	0.0000	0.0000	6.1912×10^{-2}	10.000	0.0000	10.000	10.000	10.000	10.000	10.000
	Std	0.0000	0.0000	5.7285×10^{-2}	6.1740×10^{-10}	0.0000	1.6385×10^{-11}	1.9981×10^{-9}	1.5984×10^{-10}	6.4737×10^{-16}	1.2519×10^{-8}
F11	Average	0.0000	0.0000	0.0000	2.0226×10^{-50}	3.0654×10^{-286}	1.9485×10^{-11}	1.4912×10^{-8}	2.6768×10^{-23}	1.1258×10^{-4}	1.4997×10^{12}
	Std	0.0000	0.0000	0.0000	4.5226×10^{-50}	0.0000	3.6242×10^{-11}	3.3336×10^{-8}	3.0150×10^{-23}	2.1042×10^{-4}	4.1024×10^{11}
F12	Average	2.5155×10^{-2}	4.3679×10^{-1}	$\mathbf{6.5901 \times 10^{-3}}$	71.927	6.2121×10^{-3}	73.546	89.006	73.509	75.565	11.242
	Std	5.4232×10^{-2}	9.5290×10^{-1}	$\mathbf{1.0058 \times 10^{-2}}$	0.49991	1.0974×10^{-2}	0.41366	0.12286	0.42066	15.983	6.4118×10^{8}
F13	Average	3.3299×10^{-38}	$\mathbf{9.8432 \times 10^{-124}}$	9.7428×10^{-5}	6.6943×10^{-24}	2.8332×10^{-48}	0.23757	**0.10823**	2.1251×10^{-3}	6.3621×10^{-2}	5.1495×10^{2}
	Std	7.4458×10^{-38}	$\mathbf{2.2010 \times 10^{-123}}$	2.1786×10^{-4}	1.4969×10^{-23}	6.3024×10^{-48}	0.40764	0.20948	$\mathbf{1.4765 \times 10^{-3}}$	9.5643×10^{-2}	25.793
F14	Average	0.0000	0.0000	0.0000	0.0000	0.0000	2.9961×10^{-2}	4.4001×10^{-6}	1.2201×10^{-11}	0.47933	4.4415×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	6.6979×10^{-2}	3.7691×10^{-6}	1.7271×10^{-11}	0.35923	3.7541×10^{3}
F15	Average	0.0000	0.0000	0.0000	0.0000	0.0000	2.2648×10^{-2}	3.6068×10^{-2}	8.5477×10^{-3}	3.9600×10^{-3}	12.304
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	2.4951×10^{-2}	2.3192×10^{-2}	1.9113×10^{-2}	5.3962×10^{-3}	2.0706
F16	Average	$\mathbf{2.1276 \times 10^{-139}}$	1.3235×10^{-129}	4.9230×10^{-37}	9.9873×10^{-2}	1.8757×10^{-50}	0.25987	0.15301	0.13987	0.46473	22.130
	Std	$\mathbf{4.7576 \times 10^{-139}}$	2.9594×10^{-129}	1.1007×10^{-36}	1.5245×10^{-8}	3.0802×10^{-50}	5.4772×10^{-2}	5.0490×10^{-2}	5.4772×10^{-2}	8.5849×10^{-2}	**1.0707**
F17	Average	0.0000	0.0000	0.0000	3.4987×10^{-3}	0.0000	22.912	17.773	84.342	65.628	4.8665×10^{4}
	Std	0.0000	0.0000	0.0000	7.8233×10^{-3}	0.0000	10.347	14.383	7.9822	43.406	2.4056×10^{3}
F18	Average	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	20.000	20.000	20.000	12.575	21.266
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	10.815	3.2831×10^{-2}

Table 9. Test results of Multi-peak multidimensional function (F19–F27), the dimension is fixed to 30.

Fun	Items	SOIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F19	Average	3.7921×10^{-284}	4.2388×10^{-152}	**0.0000**	7.5115×10^{-123}	5.3123×10^{-60}	8.4732×10^{-109}	1.2217×10^{-94}	5.7790×10^{-43}	$1.1176 \times 10^{+2}$	3.7921×10^{-284}
	Std	0.0000	6.8746×10^{-152}	0.0000	1.4946×10^{-122}	1.1458×10^{-59}	1.7112×10^{-108}	2.7296×10^{-94}	1.2922×10^{-42}	$1.5591 \times 10^{+2}$	0.0000
F20	Average	$\mathbf{3.1780 \times 10^{-5}}$	6.1035×10^{-4}	3.6168×10^{-3}	2.6850×10^{-3}	2.5718×10^{-3}	2.5765×10^{-3}	2.6067×10^{-3}	2.5818×10^{-3}	2.5717×10^{-3}	12.733
	Std	$\mathbf{9.3107 \times 10^{-5}}$	3.0590×10^{-4}	4.8087×10^{-4}	1.1191×10^{-4}	1.1817×10^{-7}	2.0037×10^{-6}	2.8916×10^{-5}	6.9897×10^{-6}	1.4845×10^{-15}	6.6951
F21	Average	7.4397×10^{-67}	1.0670×10^{-54}	8.0752×10^{-37}	**0.0000**	2.0674×10^{-28}	1.5729×10^{-16}	1.1379×10^{-32}	6.3436×10^{-27}	8.1781×10^{-15}	2.5053
	Std	1.6082×10^{-66}	2.3860×10^{-54}	1.6215×10^{-36}	**0.0000**	2.9078×10^{-28}	1.9113×10^{-16}	2.5444×10^{-32}	7.8475×10^{-27}	1.1609×10^{-14}	0.81600
F22	Average	0.0000	0.0000	0.0000	0.0000	0.0000	1.9432×10^{-3}	6.3022×10^{-3}	3.8864×10^{-3}	1.9432×10^{-3}	9.7478×10^{-3}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	4.3451×10^{-3}	4.7119×10^{-3}	5.3216×10^{-3}	4.3452×10^{-3}	5.5634×10^{-5}
F23	Average	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	8.0236×10^{-2}	2.0024×10^{-2}	9.9192×10^{-2}	8.7557×10^{-2}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	4.4853×10^{-2}	4.4775×10^{-2}	1.7146×10^{-1}	3.0053×10^{-2}
F24	Average	6.1118×10^{-145}	1.7857×10^{-138}	3.2553×10^{-53}	**0.0000**	6.3114×10^{-54}	2.7565	4.5736	1.8520	4.6233	4.6436
	Std	1.3661×10^{-144}	3.9697×10^{-138}	7.2790×10^{-53}	**0.0000**	1.1169×10^{-53}	2.5178	0.16283	2.5359	5.9274×10^{-2}	6.9033×10^{-2}
F25	Average	0.0000	0.0000	1.7368×10^{-156}	0.0000	2.6981×10^{-146}	1.2028×10^{-54}	9.2557×10^{-69}	5.9487×10^{-84}	4.0707×10^{-49}	0.72193
	Std	0.0000	0.0000	1.2975×10^{-139}	0.0000	1.4173×10^{-107}	9.7993×10^{-64}	2.4634×10^{-140}	7.0872×10^{-100}	1.2175×10^{-56}	33.863
F26	Average	0.0000	0.0000	1.7368×10^{-156}	0.0000	2.6981×10^{-146}	1.2028×10^{-54}	9.2557×10^{-69}	5.9487×10^{-84}	4.0707×10^{-49}	0.72193
	Std	0.0000	0.0000	1.2975×10^{-139}	0.0000	1.4173×10^{-107}	9.7993×10^{-64}	2.4634×10^{-140}	7.0872×10^{-100}	1.2175×10^{-56}	33.863
F27	Average	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	2.0085
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.58149

5.4. Experiment series 3: Accelerated convergence experiments

In order to understand the performance of the algorithm, this section will perform accelerated convergence analysis of these algorithms on 27 classic benchmark functions to compare the performance of the algorithm, the results are shown in Figures 2–5. The results in Figure 2 show the VAIAO algorithm has faster convergence s peed, higher convergence accuracy, and more stable convergence curve.

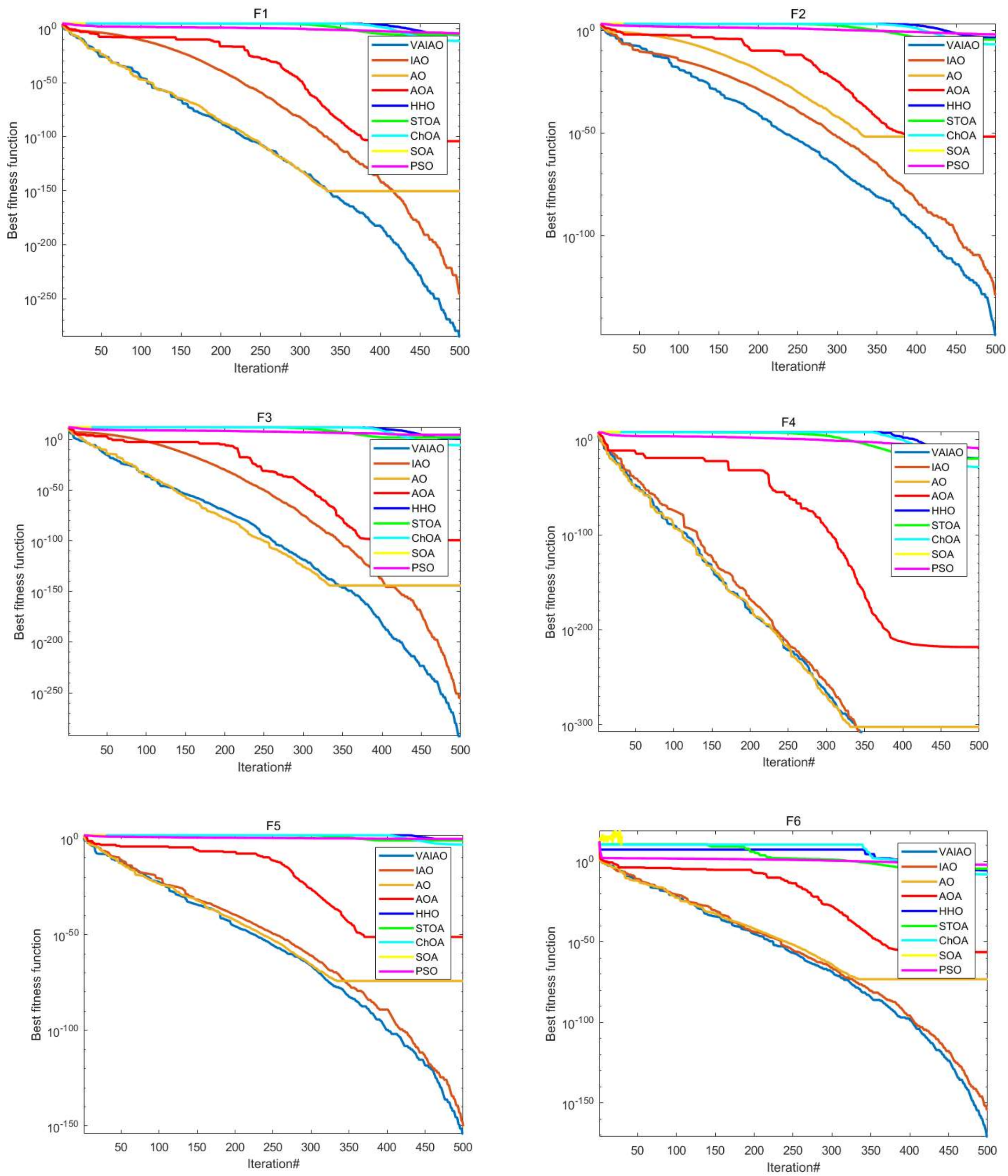

Figure 2. Convergence curves of 27 benchmark functions (F1–F6).

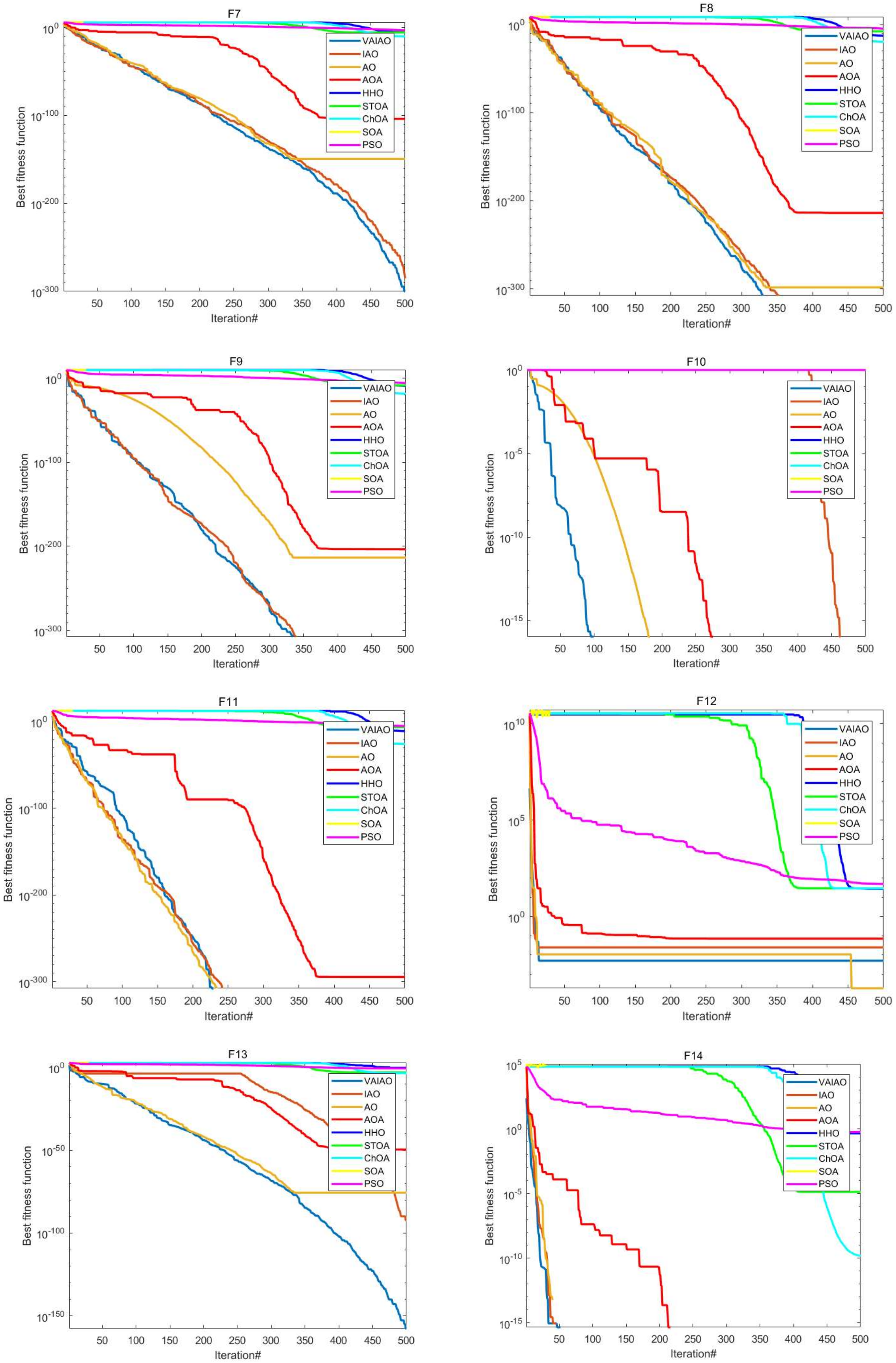

Figure 3. Convergence curves of 27 benchmark functions (F7–F14).

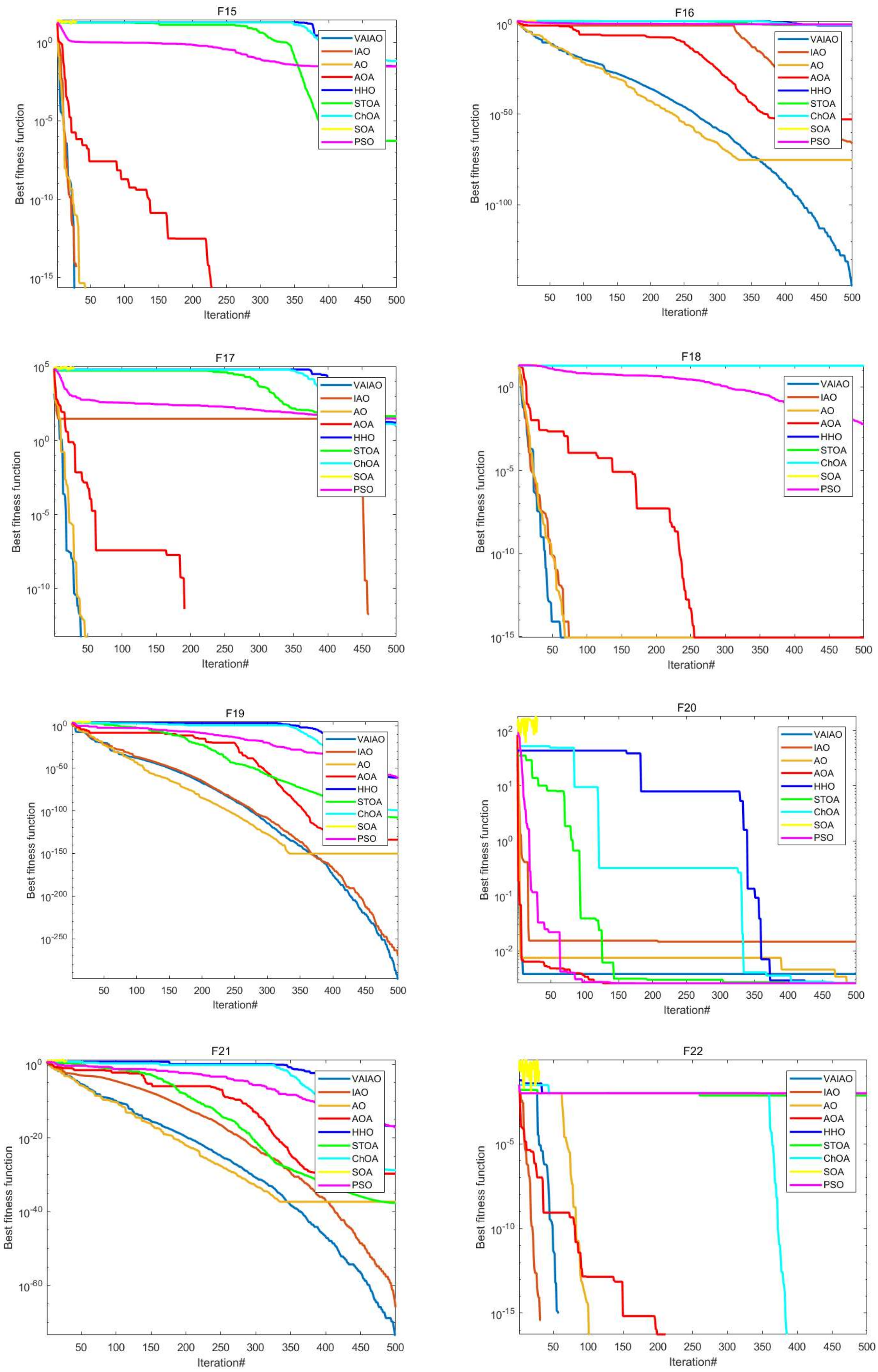

Figure 4. Convergence curves of 27 benchmark functions (F15–F22).

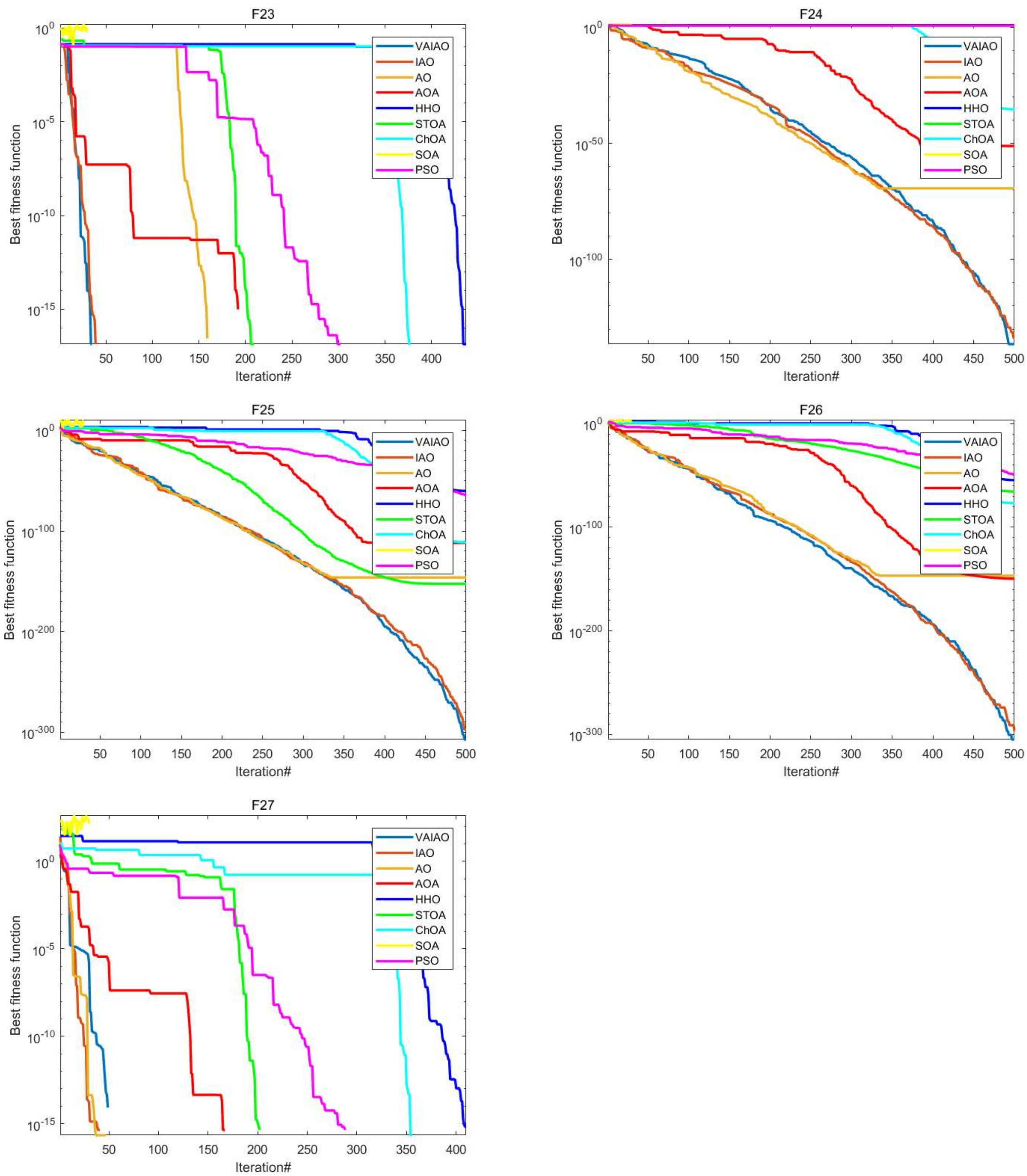

Figure 5. Convergence curves of 27 benchmark functions (F23–F27).

5.5. Experiment series 4: Scalability experiments

One of the problems which the algorithm must solve is the high-dimensional problem. The dimensions are selected as 60, 100, 300, 500 to solve this problem, and the results are shown in Tables 10–13.

Table 10. Test results of benchmark function (F1–F9, F19–F27), the dimension is fixed to 60.

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F1	Average	**4.9791×10^{-285}**	7.9894×10^{-284}	1.6265×10^{-101}	7.9491×10^{-83}	1.2280×10^{-97}	1.2062×10^{-6}	3.0010×10^{-6}	2.2287×10^{-12}	1.2383×10^{-4}	4.5836×10^{4}
	Std	0.0000	0.0000	3.6369×10^{-101}	1.4446×10^{-82}	2.7459×10^{-97}	1.8624×10^{-6}	3.2504×10^{-6}	2.0211×10^{-12}	1.0144×10^{-4}	3.3363×10^{3}
F2	Average	**1.0754×10^{-144}**	1.0762×10^{-142}	4.4908×10^{-62}	8.6497×10^{-43}	4.7376×10^{-50}	7.2859×10^{-5}	5.4243×10^{-4}	2.9238×10^{-7}	2.8354×10^{-2}	9.3463×10^{2}
	Std	**2.4046×10^{-144}**	2.4064×10^{-142}	1.0042×10^{-61}	1.9341×10^{-42}	9.9176×10^{-50}	9.0603×10^{-5}	6.9310×10^{-4}	2.7641×10^{-7}	1.7438×10^{-2}	85.440
F3	Average	**0.0000**	6.4756×10^{-295}	4.0162×10^{-130}	4.2837×10^{-127}	7.4047×10^{-96}	1.1059×10^{-1}	7.9538	1.6480×10^{-4}	2.0313×10^{4}	6.7097×10^{11}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F4	Average	0.0000	0.0000	1.6672×10^{-227}	0.0000	2.9404×10^{-204}	9.4603×10^{-18}	7.5156×10^{-16}	8.3905×10^{-28}	1.8487×10^{-9}	5.9847×10^{7}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	1.2690×10^{-17}	1.1246×10^{-15}	1.4975×10^{-27}	3.3947×10^{-9}	4.5947×10^{6}
F5	Average	**5.3277×10^{-150}**	5.8938×10^{-150}	2.8218×10^{-67}	3.3871×10^{-2}	3.2133×10^{-50}	6.5490×10^{-2}	0.41592	1.0360×10^{-3}	1.2916	76.030
	Std	**1.2621×10^{-149}**	1.3179×10^{-149}	6.2674×10^{-67}	1.8970×10^{-2}	4.2164×10^{-50}	8.9912×10^{-2}	0.43917	7.9839×10^{-4}	0.43288	2.0916
F6	Average	3.3763×10^{-149}	1.1833×10^{-145}	5.5194×10^{-51}	**0.0000**	5.3515×10^{-52}	1.4184×10^{-5}	6.6697×10^{-5}	1.0693×10^{-8}	1.9238×10^{-2}	1.9707×10^{8}
	Std	7.5442×10^{-149}	2.6406×10^{-145}	1.2342×10^{-50}	**0.0000**	9.5332×10^{-52}	1.4988×10^{-5}	7.7537×10^{-5}	4.9822×10^{-9}	3.6831×10^{-3}	2.2429×10^{8}
F7	Average	**5.7989×10^{-283}**	2.7115×10^{-281}	9.1577×10^{-135}	4.6576×10^{-87}	8.4403×10^{-98}	5.6128×10^{-6}	1.2090×10^{-5}	1.6166×10^{-10}	9.5755×10^{-4}	6.5491×10^{5}
	Std	0.0000	0.0000	2.0477×10^{-134}	1.0268×10^{-86}	1.5796×10^{-97}	6.8124×10^{-6}	2.0955×10^{-5}	1.4812×10^{-10}	9.0545×10^{-4}	4.1739×10^{4}
F8	Average	0.0000	0.0000	6.1451×10^{-295}	1.1543×10^{-99}	1.8381×10^{-186}	3.4122×10^{-11}	1.0365×10^{-7}	2.7425×10^{-19}	2.1931×10^{-5}	1.7272×10^{-8}
	Std	0.0000	0.0000	0.0000	2.5810×10^{-99}	0.0000	3.4613×10^{-11}	1.0793×10^{-7}	4.6732×10^{-19}	3.1971×10^{-5}	4.2488×10^{7}
F9	Average	0.0000	0.0000	1.2212×10^{-255}	6.2093×10^{-110}	1.2294×10^{-200}	3.1520×10^{-8}	8.8864×10^{-7}	1.5077×10^{-18}	6.5741×10^{-4}	2.0827×10^{9}
	Std	0.0000	0.0000	0.0000	1.3884×10^{-109}	0.0000	6.5054×10^{-8}	1.4509×10^{-6}	3.2314×10^{-18}	1.1546×10^{-3}	1.8815×10^{8}
F10	Average	0.0000	0.0000	0.0000	1.0000	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Std	0.0000	0.0000	0.0000	1.1073×10^{-9}	0.0000	2.4964×10^{-11}	4.8343×10^{-9}	4.7418×10^{-11}	2.1617×10^{-8}	1.5491×10^{-8}
F11	Average	0.0000	0.0000	0.0000	1.1151×10^{-9}	3.6025×10^{-293}	4.7543×10^{-12}	5.2735×10^{-5}	3.6730×10^{-25}	1.0178×10^{-2}	1.8080×10^{12}
	Std	0.0000	0.0000	0.0000	2.4935×10^{-9}	0.0000	6.9713×10^{-12}	1.1788×10^{-4}	8.1060×10^{-25}	1.3883×10^{-2}	3.1086×10^{11}
F12	Average	1.1053	1.5749×10^{-2}	2.9848×10^{-3}	7.1238	**1.6186×10^{-3}**	7.7015	8.7626	7.7813	5.6736	1.0216×10^{9}
	Std	2.4646	1.4847×10^{-2}	2.8920×10^{-3}	4.5150×10^{-1}	**1.9301×10^{-3}**	0.53349	0.38776	0.72438	4.4997	6.1546×10^{8}
F13	Average	5.1562×10^{-139}	**6.6950×10^{-145}**	2.5938×10^{-41}	2.2555×10^{-36}	9.1534×10^{-52}	0.33102	7.5026×10^{-3}	2.2971×10^{-2}	0.16973	5.2031×10^{2}
	Std	1.1530×10^{-138}	**1.4667×10^{-144}**	5.8000×10^{-41}	5.0434×10^{-36}	1.4544×10^{-51}	0.47732	4.6404×10^{-3}	4.6946×10^{-2}	0.14654	47.448

Continued on next page

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F14	Average	0.0000	0.0000	0.0000	0.0000	0.0000	8.2893×10^{-6}	8.4925×10^{-5}	7.4010×10^{-11}	0.47624	4.5788×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	1.4189×10^{-5}	1.3325×10^{-4}	9.5731×10^{-11}	0.31832	5.6324×10^{3}
F15	Average	0.0000	0.0000	0.0000	0.0000	0.0000	3.5722×10^{-2}	1.4915×10^{-2}	3.0534×10^{-3}	3.9471×10^{-3}	12.181
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	3.7178×10^{-2}	2.0853×10^{-2}	6.8275×10^{-3}	8.8018×10^{-3}	1.5118
F16	Average	1.0131×10^{-96}	$\mathbf{3.9346 \times 10^{-135}}$	6.6523×10^{-53}	9.9873×10^{-2}	4.6541×10^{-51}	0.23987	0.19987	0.15987	0.48059	22.228
	Std	2.2654×10^{-96}	$\mathbf{8.7981 \times 10^{-135}}$	1.4875×10^{-52}	6.9961×10^{-08}	1.0002×10^{-50}	5.4772×10^{-2}	4.4412×10^{-7}	5.4772×10^{-2}	8.3927×10^{-2}	7.9026×10^{-1}
F17	Average	0.0000	0.0000	0.0000	0.0000	0.0000	22.710	24.708	5.4079	53.609	4.8653×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	14.530	35.191	5.0580	5.4390	4.5805×10^{3}
F18	Average	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	1.5987×10^{-15}	8.8818×10^{-16}	20.000	20.000	20.000	21.217	21.211
	Std	0.0000	0.0000	0.0000	1.5888×10^{-15}	0.0000	0.0000	1.6313×10^{-9}	0.0000	7.6156×10^{-2}	1.1811×10^{-1}

Table 11. Test results of benchmark function (F1–F9, F19–F27), the dimension is fixed to 100.

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F1	Average	**2.0856×10^{-288}**	1.3525×10^{-271}	1.9209×10^{-144}	5.8406×10^{-22}	6.0336×10^{-99}	1.0055×10^{-6}	1.4164×10^{-6}	1.3739×10^{-11}	2.2488×10^{-4}	$4.7702 \times 10^{+4}$
	Std	0.0000	0.0000	4.0786×10^{-144}	1.3060×10^{-21}	1.3491×10^{-98}	5.6687×10^{-7}	1.5866×10^{-6}	2.2845×10^{-11}	1.5536×10^{-4}	$8.9312 \times 10^{+3}$
F2	Average	**5.5074×10^{-148}**	4.1899×10^{-147}	9.4978×10^{-65}	7.8562×10^{-31}	4.5422×10^{-51}	1.1767×10^{-4}	1.1669×10^{-4}	1.7749×10^{-7}	2.2129×10^{-2}	9.3520×10-^{2}
	Std	**1.2075×10^{-148}**	9.3243×10^{-147}	2.1238×10^{-64}	1.7567×10^{-30}	9.7386×10^{-51}	8.6044×10^{-5}	1.4799×10^{-4}	6.9353×10^{-8}	1.8698×10^{-2}	30.797
F3	Average	0.0000	0.0000	2.4261×10^{-144}	4.9514×10^{-82}	1.3037×10^{-101}	0.19101	0.47998	1.0582×10^{-5}	8.5986×10^{3}	$7.5167 \times 10^{+11}$
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F4	Average	0.0000	0.0000	1.8263×10^{-290}	0.0000	3.5375×10^{-204}	5.7413×10^{-18}	1.1299×10^{-15}	6.5828×10^{-27}	2.6545×10^{-10}	4.6675×10^{7}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	6.6767×10^{-18}	2.5264×10^{-15}	1.1839×10^{-26}	3.3196×10^{-10}	1.0026×10^{7}
F5	Average	**1.0731×10^{-147}**	1.1287×10^{-147}	7.7323×10^{-75}	2.6114×10^{-2}	1.3253×10^{-50}	2.5638×10^{-2}	0.27609	1.0539×10^{-2}	1.5166	75.742
	Std	**2.4467×10^{-147}**	2.5238×10^{-147}	7.1508×10^{-75}	2.3856×10^{-2}	2.9470×10^{-50}	4.3785×10^{-3}	0.17849	1.4339×10^{-2}	0.33536	2.7620
F6	Average	2.7824×10^{-154}	9.8495×10^{-149}	1.9648×10^{-64}	**0.0000**	1.6351×10^{-50}	1.3057×10^{-5}	4.3568×10^{-5}	1.2533×10^{-8}	2.4265×10^{-2}	2.9054×10^{8}
	Std	3.9894×10^{-154}	2.2024×10^{-148}	4.3933×10^{-64}	**0.0000**	3.2917×10^{-50}	1.6997×10^{-5}	5.0613×10^{-5}	6.3306×10^{-9}	1.4935×10^{-2}	3.3341×10^{8}
F7	Average	**3.4587×10^{-286}**	7.4886×10^{-262}	1.3214×10^{-115}	2.7430×10^{-30}	2.4926×10^{-102}	4.5084×10^{-6}	1.3141×10^{-4}	4.0396×10^{-11}	7.3701×10^{4}	6.9182×10^{5}
	Std	0.0000	0.0000	2.9548×10^{-115}	6.1334×10^{-30}	2.7141×10^{-102}	4.7226×10^{-6}	2.4179×10^{-4}	4.5370×10^{-11}	4.7857×10^{4}	3.9992×10^{4}
F8	Average	0.0000	0.0000	2.3053×10^{-297}	1.4862×10^{-173}	9.8413×10^{-193}	5.5288×10^{-11}	6.0337×10^{-10}	6.6808×10^{-19}	2.9822×10^{-4}	1.8184×10^{8}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	5.9165×10^{-11}	4.7341×10^{-10}	1.4725×10^{-18}	5.2236×10^{-4}	9.5505×10^{6}
F9	Average	0.0000	0.0000	3.4185×10^{-197}	2.4835×10^{-125}	4.3701×10^{-191}	6.5793×10^{-10}	5.2847×10^{-6}	1.0413×10^{-18}	3.3135×10^{-4}	2.5114×10^{9}
	Std	0.0000	0.0000	0.0000	5.5533×10^{-125}	0.0000	4.2526×10^{-10}	1.1817×10^{-5}	1.3940×10^{-18}	4.2915×10^{-4}	3.2430×10^{8}
F10	Average	0.0000	0.0000	6.1061×10^{-2}	1.0000	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Std	0.0000	0.0000	1.3458×10^{-1}	1.4541×10^{-9}	0.0000	1.7788×10^{-11}	1.8392×10^{-9}	5.4131×10^{-11}	9.9584×10^{-9}	1.7382×10^{-8}
F11	Average	0.0000	0.0000	0.0000	2.3162×10^{-40}	6.9359×10^{-291}	2.5794×10^{-12}	8.2743×10^{-10}	3.4449×10^{-25}	1.5298×10^{-4}	1.3530×10^{12}
	Std	0.0000	0.0000	0.0000	5.1792×10^{-40}	0.0000	4.9553×10^{-12}	1.2241×10^{-9}	7.6965×10^{-25}	3.1283×10^{-4}	1.4707×10^{11}
F12	Average	2.3293×10^{-2}	2.6402×10-^{2}	8.3243×10^{-3}	7.5737	**5.7798×10^{-3}**	7.3658	8.9861	7.7392	7.5901	7.9727×10^{8}
	Std	2.6047×10^{-2}	4.2889×10^{-2}	1.3581×10^{-2}	0.34765	**6.7518×10^{-3}**	0.39510	1.1386×10^{-2}	0.48710	4.9830	3.0547×10^{8}
F13	Average	**2.7741×10^{-134}**	9.5046×10^{-131}	2.0574×10^{-45}	2.9736×10^{-16}	3.6069×10^{-51}	1.1388	1.0799×10^{-2}	0.97007	0.14721	5.5924×10^{2}
	Std	**6.2031×10^{-134}**	2.1253×10^{-130}	4.6005×10^{-45}	6.6491×10^{-16}	7.9918×10^{-51}	1.9680	1.0189×10^{-2}	2.1645	0.14404	7.7151

Continued on next page

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F14	Average	0.0000	0.0000	0.0000	0.0000	0.0000	2.9766×10^{-2}	1.3756×10^{-5}	1.4978×10^{-11}	0.77284	4.7685×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	6.6554×10^{-2}	5.5149×10^{-6}	1.6901×10^{-11}	0.10194	3.1623×10^{3}
F15	Average	0.0000	0.0000	0.0000	0.0000	0.0000	2.6034×10^{-2}	1.6210×10^{-2}	2.1853×10^{-12}	1.4857×10^{-3}	11.962
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	3.6158×10^{-2}	2.3634×10^{-2}	4.5888×10^{-12}	3.3054×10^{-3}	20.858
F16	Average	**2.4308×10^{-132}**	1.4778×10^{-123}	2.0964×10^{-56}	9.9873×10^{-2}	8.6246×10^{-51}	0.23987	0.17988	0.13987	0.50070	22.327
	Std	**5.4354×10^{-132}**	3.3046×10^{-123}	4.6877×10^{-56}	1.2637×10^{-8}	1.6438×10^{-50}	5.4772×10^{-2}	$4.4713 \times 10^{-1=2}$	5.4772×10^{-2}	0.12189	0.55834
F17	Average	0.0000	0.0000	0.0000	0.0000	0.0000	21.667	23.760	5.1447	51.996	4.5831×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	22.730	21.422	4.6792	14.896	5.0338×10^{3}
F18	Average	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	2.0365×10^{-9}	8.8818×10^{-16}	20.000	20.000	20.000	12.009	21.208
	Std	0.0000	0.0000	0.0000	4.5537×10^{-9}	0.0000	3.6343×10^{-9}	0.0000	5.8102×10^{-9}	10.943	8.3420×10^{-2}

Table 12. Test results of benchmark function (F1–F9, F19–F27), the dimension is fixed to 300.

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F1	Average	**8.5034×10^{-270}**	2.3079×10^{-261}	3.3665×10^{-142}	5.1609×10^{-52}	4.3526×10^{-103}	1.8230×10^{-7}	2.7129×10^{-7}	5.2054×10^{-12}	5.1871×10^{-5}	4.6751×10^{4}
	Std	0.0000	0.0000	7.1672×10^{-142}	1.1540×10^{-51}	5.1572×10^{-103}	1.6248×10^{-7}	3.2089×10^{-7}	7.8125×10^{-12}	3.9173×10^{-5}	2.9835×10^{3}
F2	Average	**3.7912×10^{-145}**	7.2883×10^{-119}	2.4149×10^{-63}	1.3091×10^{-30}	8.5396×10^{-50}	2.1334×10^{-4}	2.9038×10^{-4}	2.8768×10^{-7}	2.0031×10^{-2}	9.2501×10^{2}
	Std	**8.4775×10^{-145}**	1.6297×10^{-118}	5.1948×10^{-63}	2.9273×10^{-30}	1.8982×10^{-49}	3.7048×10^{-4}	3.1308×10^{-4}	3.0723×10^{-7}	1.1255×10^{-2}	53.869
F3	Average	**7.3174×10^{-303}**	7.7808×10^{-294}	2.8079×10^{-141}	1.8273×10^{-47}	1.6119×10^{-99}	9.4355	18.046	2.1366×10^{-5}	$2.7337 \times 10^{+3}$	6.5635×10^{11}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F4	Average	0.0000	0.0000	1.0180×10^{-214}	0.0000	1.9773×10^{-218}	2.8943×10^{-17}	5.9752×10^{-19}	4.6456×10^{-29}	1.1970×10^{-9}	5.9402×10^{7}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	5.6868×10^{-17}	1.3144×10^{-18}	8.3512×10^{-29}	2.5358×10^{-9}	1.3212×10^{7}
F5	Average	**3.5737×10^{-147}**	$1.5840 \times 10^{\mathbf{-146}}$	3.9392×10^{-55}	2.5504×10^{-2}	1.8021×10^{-47}	4.1602×10^{-2}	1.3958×10^{-1}	2.4564×10^{-3}	1.3620	75.163
	Std	**7.9911×10^{-147}**	$2.4424 \times 10^{\mathbf{-146}}$	8.8083×10^{-55}	2.3298×10^{-2}	3.8369×10^{-47}	3.5228×10^{-2}	1.3822×10^{-1}	1.9369×10^{-3}	0.24597	1.1950
F6	Average	8.6006×10^{-154}	4.8798×10^{-138}	4.5346×10^{-59}	**0.0000**	1.4295×10^{-48}	7.3639×10^{-6}	3.8962×10^{-5}	2.4319×10^{-8}	2.7580×10^{-2}	8.9581×10^{7}
	Best	3.4887×10^{-169}	2.0631×10^{-163}	1.6957×10^{-75}	**0.0000**	1.3140×10^{-57}	3.9492×10^{-6}	8.9013×10^{-6}	7.6014×10^{-9}	$8.2748 \times 10-^{-3}$	4.1897×10^{7}
F7	Middle	8.6568×10^{-155}	1.4478×10^{-162}	2.0853×10^{-73}	0.0000	1.0650×10^{-53}	4.9859×10^{-6}	4.3948×10^{-5}	1.7779×10^{-8}	1.0212×10^{-2}	8.2422×10^{7}
	Std	1.8039×10^{-153}	1.0912×10^{-137}	1.0140×10^{-58}	0.0000	3.1961×10^{-48}	4.2023×10^{-6}	2.3412×10^{-5}	1.9170×10^{-8}	3.1222×10^{-2}	4.1768×10^{7}
F8	Average	2.6834×10^{-290}	**2.3487×10^{-291}**	5.0829×10^{-107}	2.9621×10^{-22}	1.3730×10^{-95}	3.1965×10^{-6}	7.6805×10^{-5}	1.4931×10^{-10}	6.8479×10^{-3}	6.3921×10^{5}
	Std	0.0000	0.0000	1.1366×10^{-106}	6.6234×10^{-22}	3.0614×10^{-95}	3.1031×10^{-6}	7.3720×10^{-5}	2.5066×10^{-10}	5.3827×10^{-3}	5.9461×10^{4}
F9	Average	0.0000	0.0000	8.6268×10^{-297}	1.0809×10^{-85}	9.3269×10^{-199}	2.4077×10^{-10}	1.8286×10^{-8}	8.6589×10^{-18}	6.6912×10^{-6}	1.5498×10^{8}
	Std	0.0000	0.0000	0.0000	2.4170×10^{-85}	0.0000	3.8745×10^{-10}	2.1243×10^{-8}	1.7220×10^{-17}	6.5714×10^{-6}	2.3799×10^{7}
F10	Average	0.0000	0.0000	2.1467×10^{-278}	1.5224×10^{-84}	7.5095×10^{-192}	1.6258×10^{-10}	8.2123×10^{-8}	$1.0161 \times 10-^{-17}$	1.6788×10^{-3}	2.3457×10^{9}
	Std	0.0000	0.0000	0.0000	3.4043×10^{-84}	0.0000	2.5954×10^{-10}	1.7044×10^{-7}	1.9271×10^{-17}	3.5113×10^{-3}	5.2772×10^{8}
F11	Average	0.0000	0.0000	0.0000	1.0000	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Std	0.0000	0.0000	0.0000	5.5017×10^{-9}	0.0000	1.9167×10^{-11}	7.2876×10^{-9}	4.9847×10^{-11}	5.3948×10^{-14}	1.2245×10^{-8}
F12	Average	0.0000	0.0000	0.0000	1.4957×10^{-187}	1.8815×10^{-284}	3.1720×10^{-12}	1.0875×10^{-8}	1.5745×10^{-23}	4.3439×10^{-3}	1.7600×10^{12}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	2.8874×10^{-12}	1.5643×10^{-8}	3.2855×10^{-23}	9.0737×10^{-3}	3.1814×10^{11}
F13	Average	**9.4565×10^{-3}**	2.0277×10^{-2}	4.8216×10^{-2}	7.1695	9.6104×10^{-3}	7.6772	8.9345	7.8181	34.758	7.9339×10^{8}
	Std	**6.9496×10^{-3}**	2.9629×10^{-2}	6.3765×10^{-2}	5.1244×10^{-1}	1.9708×10^{-2}	4.8319×10^{-1}	9.7361×10^{-2}	0.39107	66.473	3.7199×10^{8}

Continued on next page

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F14	Average	**1.3961×10^{-115}**	1.5967×10^{-19}	1.8333×10^{-54}	3.0679×10^{-61}	3.6483×10^{-33}	2.1743×10^{-1}	1.1566×10^{-1}	1.9336×10^{-3}	23.391	5.2977×10^{2}
	Std	**3.1217×10^{-115}**	3.5703×10^{-19}	4.0993×10^{-54}	6.8601×10^{-61}	8.1579×10^{-33}	4.1511×10^{-1}	2.4825×10^{-1}	2.3224×10^{-3}	52.087	34.131
F15	Average	0.0000	0.0000	0.0000	0.0000	0.0000	1.3587×10^{-5}	3.1783×10^{-5}	1.7297 × 10-10	0.74304	4.5561×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	2.5788×10^{-5}	4.4060×10^{-5}	3.5587×10^{-10}	0.34849	2.4965×10^{3}
F16	Average	0.0000	0.0000	0.0000	0.0000	0.0000	4.5692×10^{-2}	1.4218×10^{-2}	2.0649×10^{-2}	9.8611×10^{-3}	12.818
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	5.1199×10^{-2}	1.9480×10^{-2}	2.8551×10^{-2}	6.7337×10^{-3}	0.81075
F17	Average	5.0522×10^{-31}	**2.4786×10^{-138}**	1.0165×10^{-3}	9.9873×10^{-2}	1.3072×10^{-50}	2.3987×10^{-1}	1.7987×10^{-1}	1.3987×10^{-1}	0.48019	22.103
	Std	1.1297×10^{-30}	**4.0631×10^{-138}**	2.2730×10^{-3}	1.7264×10^{-8}	2.8236×10^{-50}	5.4772×10^{-2}	4.4721×10^{-2}	5.4772×10^{-2}	4.4542×10^{-2}	1.0527
F18	Average	0.0000	0.0000	0.0000	0.0000	0.0000	20.896	10.683	5.5791	58.920	4.2836×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	6.5017	6.6700	6.4969	18.029	8.4056×10^{3}

Table 13. Test results of benchmark function (F1–F9, F19–F27), the dimension is fixed to 500

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F1	Average	**6.6313×10^{-248}**	1.0171×10^{-241}	8.4862×10^{-144}	7.1006×10^{-48}	8.8334×10^{-95}	3.8707×10^{-7}	3.3576×10^{-6}	4.3111×10^{-12}	6.6836×10^{-5}	4.5996×10^{4}
	Std	0.0000	0.0000	1.7980×10^{-143}	1.4166×10^{-47}	1.7142×10^{-94}	3.7231×10^{-7}	2.7747×10^{-6}	1.7571×10^{-12}	5.5540×10^{-5}	5.1543×10^{3}
F2	Average	**1.7819×10^{-145}**	6.9639×10^{-136}	1.5808×10^{-69}	8.6734×10^{-53}	5.6451×10^{-49}	1.2085×10^{-4}	1.6912×10^{-4}	3.6835×10^{-7}	1.1760×10^{-1}	9.5069×10^{2}
	Std	**3.9837×10^{-145}**	1.5564×10^{-135}	3.4327×10^{-69}	1.3821×10^{-52}	1.1327×10^{-48}	4.7589×10^{-5}	1.2149×10^{-4}	3.3018×10^{-7}	1.3612×10^{-1}	39.409
F3	Average	0.0000	1.0874×10^{-291}	8.6120×10^{-109}	1.6761×10^{-93}	1.5190×10^{-97}	1.2145	8.2760×10^{-2}	1.4082×10^{-6}	$9.1394 \times 10^{+2}$	6.3988×10^{11}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
F4	Average	0.0000	0.0000	4.5259×10^{-295}	0.0000	1.7267×10^{-197}	2.4355×10^{-15}	1.1832×10^{-16}	1.9752×10^{-27}	2.5327×10^{-9}	4.5378×10^{7}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	5.4241×10^{-15}	2.6449×10^{-16}	4.1972×10^{-27}	5.3850×10^{-9}	8.1540×10^{6}
F5	Average	**1.1538×10^{-141}**	2.3397×10^{-129}	5.8828×10^{-58}	1.9148×10^{-2}	4.9412×10^{-50}	2.7255×10^{-2}	5.4022×10^{-1}	1.4371×10^{-3}	1.4054	74.030
	Std	**2.5801×10^{-141}**	5.2316×10^{-129}	1.3154×10^{-57}	1.6284×10^{-2}	1.0977×10^{-49}	1.1281×10^{-2}	5.1437×10^{-1}	1.2835×10^{-3}	2.3314×10^{-1}	1.5767
F6	Average	5.1864×10^{-148}	5.6589×10^{-142}	3.2096×10^{-68}	**0.0000**	5.1573×10^{-53}	5.5248×10^{-6}	2.1736×10^{-5}	1.7515×10^{-8}	1.3136×10^{-2}	7.1716×10^{8}
	Std	1.1597×10^{-147}	8.7834×10^{-142}	7.1768×10^{-68}	**0.0000**	7.0852×10^{-53}	3.5567×10^{-6}	2.4572×10^{-5}	1.0438×10^{-8}	1.0137×10^{-2}	1.4532×10^{9}
F7	Average	**3.0218×10^{-299}**	4.5202×10^{-280}	1.5705×10^{-146}	2.4540×10^{-28}	5.9460×10^{-96}	4.2891×10^{-6}	8.1426×10^{-5}	1.3674×10^{-10}	2.9373×10^{-3}	6.2494×10^{5}
	Std	0.0000	0.0000	3.5076×10^{-146}	5.4872×10^{-28}	1.3267×10^{-95}	2.7558×10^{-6}	1.2553×10^{-4}	1.6128×10^{-10}	3.0603×10^{-3}	9.8639×10^{4}
F8	Average	0.0000	0.0000	4.9884×10^{-283}	1.7473×10^{-89}	4.7178×10^{-184}	1.0944×10^{-10}	1.1862×10^{-6}	2.4482×10^{-19}	1.3297×10^{-4}	1.5343×10^{8}
	Std	0.0000	0.0000	0.0000	3.9071×10^{-89}	0.0000	2.0330×10^{-10}	2.2994×10^{-6}	3.0112×10^{-19}	2.1119×10^{-4}	2.5921×10^{7}
F9	Average	0.0000	0.0000	1.3983×10^{-289}	8.5594×10^{-120}	5.0159×10^{-202}	2.9436×10^{-10}	6.4955×10^{-7}	2.6418×10^{-17}	1.0147×10^{-3}	2.2796×10^{9}
	Std	0.0000	0.0000	0.0000	1.9139×10^{-119}	0.0000	3.4713×10^{-10}	9.5399×10^{-7}	5.8755×10^{-17}	1.0989×10^{-3}	3.6372×10^{8}
F10	Average	0.0000	0.0000	1.1662×10^{-2}	1.0000	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Std	0.0000	0.0000	2.6076×10^{-2}	1.3383×10^{-9}	0.0000	2.6928×10^{-11}	7.6557×10^{-10}	2.5327×10^{-11}	1.9230×10^{-16}	1.8846×10^{-8}
F11	Average	0.0000	0.0000	8.3800×10^{-294}	2.2833×10^{-131}	1.3353×10^{-295}	1.5095×10^{-10}	6.2053×10^{-9}	7.3865×10^{-24}	6.6536×10^{-6}	1.4321×10^{12}
	Std	0.0000	0.0000	0.0000	5.1056×10^{-131}	0.0000	2.7488×10^{-10}	9.7885×10^{-9}	1.6436×10^{-23}	7.2883×10^{-6}	3.1233×10^{11}
F12	Average	**6.5458×10^{-3}**	8.8714×10^{-2}	9.5595×10^{-3}	7.5066	9.7117×10^{-3}	7.3632	8.9502	7.3032	6.0514	9.4655×10^{8}
	Std	**6.5556×10^{-3}**	1.7098×10^{-1}	1.0124×10^{-2}	4.4457×10^{-1}	1.0480×10^{-2}	3.7827×10^{-1}	1.0031×10^{-1}	0.23831	2.9727	3.9936×10^{8}
F13	Average	**1.0427×10^{-123}**	1.0306×10^{-121}	5.0944×10^{-20}	2.7879×10^{-44}	8.2408×10^{-52}	7.3789×10^{-1}	3.4293×10^{-1}	9.3585×10^{-4}	6.9881	5.2362×10^{2}
	Std	**2.3314×10^{-123}**	2.1964×10^{-121}	1.1391×10^{-19}	5.7506×10^{-44}	1.5881×10^{-51}	1.5253	6.0003×10^{-1}	7.5831×10^{-4}	1.4919×10	4.1865×10

Continued on next page

Fun	Item	VAIAO	IAO	AO	AOA	HHO	STOA	CHIMP	SOA	PSO	DE
F14	Average	0.0000	0.0000	0.0000	0.0000	0.0000	6.0417×10^{-6}	4.5754×10^{-5}	2.5965×10^{-11}	6.9428×10^{-1}	4.6528×10^{4}
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	7.1292×10^{-6}	$9.8098 \times 10^{-6=}$	2.6781×10^{-11}	3.3943×10^{-1}	2.3086×10^{3}
F15	Average	0.0000	0.0000	0.0000	8.9007×10^{-9}	0.0000	3.6310×10^{-8}	6.0050×10^{-2}	1.8778×10^{-8}	6.4113×10^{-3}	12.866
	Std	0.0000	0.0000	0.0000	1.9903×10^{-8}	0.0000	4.4207×10^{-8}	7.9379×10^{-2}	4.1980×10^{-8}	9.7735×10^{-3}	9.4698×10^{-1}
F16	Average	$\mathbf{4.7865 \times 10^{-115}}$	1.2149×10^{-26}	9.0592×10^{-64}	9.9873×10^{-2}	4.6927×10^{-50}	2.1987×10^{-1}	1.7988×10^{-1}	1.7987×10^{-1}	4.8939×10^{-1}	21.397 ×
	Std	$\mathbf{1.0703 \times 10^{-114}}$	2.7166×10^{-26}	2.0257×10^{-63}	1.2063×10^{-8}	1.0476×10^{-49}	4.4721×10^{-2}	4.4723×10^{-2}	4.4722×10^{-2}	7.5154×10^{-2}	1.7984
F17	Average	0.0000	0.0000	3.3621×10^{-2}	0.0000	0.0000	12.486	20.821	4.1970	43.587	4.8691×10^{4}
	Std	0.0000	0.0000	7.5178×10^{-2}	0.0000	0.0000	10.150	27.344	2.8825	9.4970	3.4889×10^{3}
F18	Average	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	20.000	20.000	20.000	20.688	21.197
	Std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	6.1862×10^{-1}	1.7209×10^{-1}

5.6. Wilcoxon statistical sign-rank test and Friedman test on the 27 benchmark Functions

The Wilcoxon statistical tests are often introduced to verify the performance of the proposed algorithm. If the proposed algorithm value is smaller than another algorithm, it's proved the proposed algorithm is superior. This experiment is carried out 30 times, and the results of the Wilcoxon statistical test on Table 14. Table 14 shows that the Wilcoxon statistical test results for 27 functions are less than 0.05 in most cases. The results of the Wilcoxon sign-rank test and Friedman test on 27 functions are shown in Tables 15–18. That also means the VAIAO algorithm is better than the others.

Table 14. The results of the Wilcoxon statistical test on 27 functions.

Fun	VAIAO Vs. IAO	VAIAO Vs. AO	VAIAO Vs. AOA	VAIAO Vs. HHO	VAIAO Vs. STOA	VAIAO Vs. ChOA	VAIAO Vs. SOA
F1	3.1600×10^{-2}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F2	3.8811×10^{-4}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F3	2.5600×10^{-2}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F4	**NAN**	1.7300×10^{-6}	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F5	2.0000×10^{-3}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F6	1.4000×10^{-2}	1.7300×10^{-6}	3.7900×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F7	4.0400×10^{-2}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F8	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F9	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F10	1.7000×10^{-3}	7.8100×10^{-3}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F11	3.5152×10^{-6}	**NAN**	1.2300×10^{-5}	1.9200×10^{-6}	3.1800×10^{-6}	3.6100×10^{-3}	5.2200×10^{-6}
F12	2.3000×10^{-2}	1.4000×10^{-2}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F13	**NAN**	9.3200×10^{-6}	1.7300×10^{-6}	**NAN**	4.8800×10^{-4}	8.3000×10^{-6}	1.9500×10^{-3}
F14	**NAN**	**NAN**	**NAN**	**NAN**	7.8100×10^{-3}	1.8200×10^{-5}	3.9100×10^{-3}
F15	4.0700×10^{-2}	**NAN**	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F16	9.8000×10^{-3}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F17	1.9600×10^{-2}	**0.50000**	**0.25000**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F18	**NAN**	**NAN**	**NAN**	NAN	**NAN**	**NAN**	**NAN**
F19	**NAN**	1.7300×10^{-6}	6.1000×10^{-5}	NAN	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F20	**NAN**	3.8500×10^{-3}	1.7500×10^{-2}	1.8200×10^{-5}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F21	4.9500×10^{-2}	1.7300×10^{-6}	9.7700×10^{-4}	1.2900×10^{-3}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F22	2.3000×10^{-2}	**NAN**	**NAN**	1.4000×10^{-2}	1.7300×10^{-6}	1.7300×10^{-6}	3.1800×10^{-6}
F23	**NAN**	**NAN**	**NAN**	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F24	**NAN**	1.7300×10^{-6}	1.2200×10^{-4}	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F25	2.3000×10^{-3}	1.7300×10^{-6}	5.9600×10^{-5}	3.5900×10^{-4}	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F26	**NAN**	1.7300×10^{-6}	1.2200×10^{-4}	**NAN**	1.7300×10^{-6}	1.7300×10^{-6}	1.7300×10^{-6}
F27	**NAN**	**NAN**	**NAN**	**NAN**	6.0500×10^{-7}	4.3200×10^{-8}	1.4500×10^{-7}

Table 15. The results of the Wilcoxon sign-rank test between VAIAO and the compared algorithms (IAO, AO, AOA).

No.	VAIAO vs IAO			VAIAO vs AO			VAIAO vs AOA		
	R^+	R^-	sign	R^+	R^-	sign	R^+	R^-	sign
F1	465.0	0.0	+	0.0	465.0	−	0.0	465.0	−
F2	0.0	465.0	−	465.0	0.0	+	465.0	0.0	+
F3	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F4	232.5	232.5	≈	465.0	0.0	+	232.5	232.5	≈
F5	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F6	465.0	0.0	+	465.0	0.0	+	0.0	465.0	−
F7	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F8	232.5	232.5	≈	465.0	0.0	+	465.0	0.0	+
F9	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F10	232.5	232.5	≈	232.5	232.5	≈	465.0	0.0	+
F11	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F12	0.0	465.0	−	0.0	465.0	−	465.0	0.0	+
F13	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F14	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F15	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F16	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F17	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F18	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F19	232.5	232.5	≈	465.0	0.0	+	232.5	232.5	≈
F20	465.0	0.0	+	0.0	465.0	−	0.0	465.0	−
F21	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F22	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F23	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F24	0.0	465.0	−	465.0	0.0	+	0.0	465.0	−
F25	0.0	465.0	−	465.0	0.0	+	0.0	465.0	−
F26	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F27	232.5	232.5	≈	465.0	0.0	+	232.5	232.5	≈
+-≈	8/4/15			13/3/11			10/5/12		

Table 16. The results of the Wilcoxon sign-rank test between VAIAO and the compared algorithms (HHO, STOA, ChOA).

No.	VAIAO vs HHO			VAIAO vs STOA			VAIAO vs ChOA		
	R+	R-	sign	R+	R-	sign	R+	R-	sign
F1	0.0	465.0	−	0.0	465.0	−	0.0	465.0	−
F2	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F3	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F4	465.0	0.0	+	465.0	0.0	+	232.5	232.5	+
F5	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F6	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F7	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F8	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F9	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F10	232.5	232.5	≈	465.0	0.0	+	465.0	0.0	+
F11	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F12	0.0	465.0	−	465.0	0.0	+	465.0	0.0	+
F13	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F14	232.5	232.5	≈	465.0	0.0	+	465.0	0.0	+
F15	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F16	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F17	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F18	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F19	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F20	0.0	465.0	−	0.0	465.0	−	0.0	465.0	−
F21	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F22	232.5	232.5	≈	465.0	0.0	+	465.0	0.0	+
F23	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F24	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F25	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F26	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F27	232.5	232.5	≈	465.0	0.0	+	232.5	232.5	≈
+-≈	13/3/11			18/2/7			17/2/8		

Table 17. The results of the Wilcoxon sign-rank test between VAIAO and the compared algorithms (SOA, PSO, DE).

No.	VAIAO vs SOA			VAIAO vs PSO			VAIAO vs DE		
	R+	R-	sign	R+	R-	sign	R+	R-	sign
F1	0.0	465.0	−	0.0	465.0	−	0.0	465.0	−
F2	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F3	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F4	465.0	0.0	+	465.0	0.0	+	232.5	232.5	+
F5	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F6	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F7	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F8	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F9	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F10	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F11	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F12	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F13	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F14	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F15	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F16	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F17	0.0	465.0	−	0.0	465.0	−	0.0	465.0	−
F18	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F19	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F20	0.0	465.0	−	465.0	0.0	+	0.0	465.0	−
F21	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F22	465.0	0.0	+	465.0	0.0	+	232.5	232.5	≈
F23	465.0	0.0	+	465.0	0.0	+	232.5	232.5	≈
F24	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F25	465.0	0.0	+	465.0	0.0	+	465.0	0.0	+
F26	232.5	232.5	≈	232.5	232.5	≈	232.5	232.5	≈
F27	232.5	232.5	≈	465.0	0.0	+	232.5	232.5	≈
+-≈	18/3/6			20/2/5			16/3/9		

5.7. Ablation experiments

Since the proposed VAIAO introduced two improvement strategies, velocity-aided global search mechanism and adaptive opposition-based learning. Therefore, it is necessary to improve these strategies by ablation experiments.

In VAIAO, there are two variants. The first variant is VAIAO-2 which is with only velocity-aided global search mechanism. The second variant is VAIAO-3 which with adaptive opposition-based learning. Three single-peak functions (F1, F2, F3), three two-dimensional multi-channel functions (F12, F13, F16), and 3 multidimensional multi-peak functions (F24, F25, F26) are tested. The results of the ablation experiments are shown in Table 19.

Table 18. The results of the Friedman test on 27 Functions.

Algorithm	Final-Rank
VAIAO	3.48
IAO	3.64
AO	3.75
AOA	4.31
HHO	3.75
STOA	6.57
ChOA	6.53
SOA	5.57
PSO	9.87
DE	7.48

Table 19. The results of the ablation experiments on 27 Functions.

Functions	Item	VAIAO	VAIAO-2	VAIAO-3
F1	Average	$\mathbf{1.6451 \times 10^{-293}}$	5.9584×10^{-275}	1.1838×10^{-245}
	Std	0.0000	0.0000	0.0000
F2	Average	$\mathbf{2.5965 \times 10^{-148}}$	5.0994×10^{-134}	1.1056×10^{-132}
	Std	$\mathbf{4.5876 \times 10^{-148}}$	1.1403×10^{-133}	2.4723×10^{-132}
F3	Average	$\mathbf{3.9676 \times 10^{-276}}$	3.2373×10^{-259}	3.2917×10^{-289}
	Std	0.0000	0.0000	0.0000
F12	Average	$\mathbf{3.6658 \times 10^{-3}}$	0.11316	3.1808×10^{-2}
	Std	$\mathbf{1.5875 \times 10^{-3}}$	0.24508	3.6911×10^{-2}
F13	Average	$\mathbf{2.6766 \times 10^{-129}}$	4.3596×10^{-123}	1.5940×10^{-124}
	Std	$\mathbf{5.9850 \times 10^{-129}}$	9.7483×10^{-123}	2.9863×10^{-124}
F16	Average	$\mathbf{3.0590 \times 10^{-146}}$	6.4337×10^{-134}	6.9369×10^{-122}
	Std	0.0000	0.0000	0.0000
F24	Average	$\mathbf{8.5225 \times 10^{-138}}$	2.5222×10^{-135}	1.1379×10^{-128}
	Std	$\mathbf{1.9057 \times 10^{-137}}$	5.6294×10^{-135}	2.5445×10^{-128}
F25	Average	$\mathbf{7.4699 \times 10^{-3}}$	3.5276×10^{-2}	1.9608×10^{-2}
	Std	0.0000	0.0000	0.0000
F26	Average	$\mathbf{1.4884 \times 10^{-301}}$	4.3692×10^{-293}	1.4242×10^{-272}
	Std	0.0000	0.0000	0.0000

6. Engineering design problem

Engineering design problems are used to test the performance of algorithm, the results of the engineering design problem in real world can reflect the advantages and disadvantages of an algorithms. It is important to use the traditional engineering design problem to test proposed VAIAO algorithm.

6.1. Three-bar truss design problem

The three-bar truss is a classic optimization problem which is an engineering design problem. The schematic model of three-bar truss is shown in Figure 6. The pressure of each Truss member is σ. the cross-sectional area is $A_1(= x_1)$ and $A_2(= x_2)$. The two parameters are optimized to minimize the total weight in the case of satisfying three limiting conditions. The three constraints are as follows:

$$\vec{x} = [x_1 x_2] = [A_1 A_2]$$

Objective:

$$f(\vec{x}) = \left(2\sqrt{2}x_1 + x_2\right) * l$$

Subject to:

$$g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} p - \sigma \le 0$$

$$g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_2^2 + 2x_1x_2} p - \sigma \le 0$$

$$g_1(\vec{x}) = \frac{1}{\sqrt{2}x_2 + x_1} p - \sigma \le 0$$

Variable range:

$$0 \le x_1, x_2 \le 1$$

Wheel $l = 100cm, P = 2KN/cm^2, \sigma = 2KN/cm^2$, Table 20 shows the results of three bar truss design problem. T It can be seen from the data results in the table 20 that the best value of VAIAO smaller than that of other algorithms. From this reason, it can be concluded that VAIAO can better solve the engineering problem.

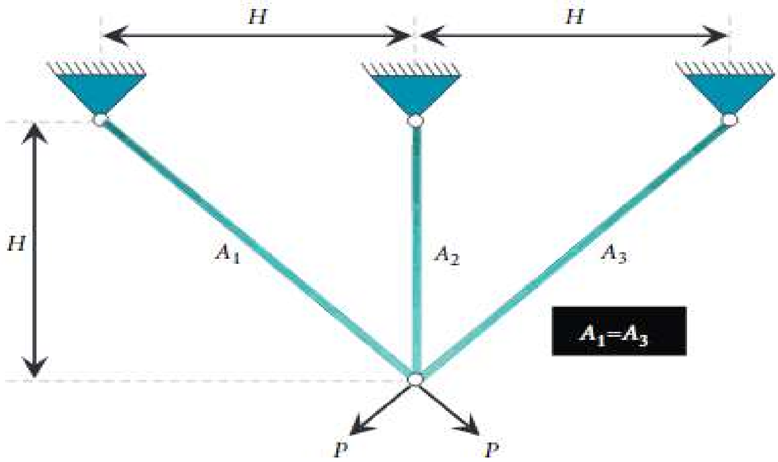

Figure 6. The schematic model of three-bar truss [72].

Table 20. The results of three bar truss design problem.

Algorithm	X_1	X_2	Best
VAIAO	**0.78975**	**0.40516**	**263.8926**
IAO	0.80211	0.38269	265.1395
AO	0.77612	0.44565	264.0855
AOA	0.809	0.35403	264.2239
HHO	0.76304	0.48595	264.4203
STOA	0.79306	0.39626	263.9367
ChOA	0.77768	0.44029	263.9912
SOA	0.79261	0.39721	263.9045
PSO	0.78867	0.40818	264.5890
DE	0.84867	0.71405	318.1953

6.2. Compression spring design problem

The compression spring design problem is one of the classic engineering optimization problems in mechanical engineering. The schematic model of compression spring design problem is shown in Figure 7. The essence of the compression spring design problem is to minimize the weight of the tension and compression spring by optimizing three parameters (wire diameter (d), average coil diameter (D) and effective coil number (N)).

Consider:

$$\overrightarrow{x_1} = [x_1, x_2, x_3] = [d \;\; D \;\; N]$$

Objective:

$$f(\vec{x}) = (x_3 + 2)x_2x_1^2$$

Subject to:

$$g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \leq 0$$

$$g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0$$

$$g_3(\vec{x}) = 1 - \frac{140.45x_3}{x_2^2 x_3} \leq 0$$

$$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$$

Variable ranges:

$$0.05 \leq x_1 \leq 2.00$$

$$0.25 \leq x_2 \leq 1.30$$

$$2.00 \leq x_3 \leq 15.00$$

The experimental results are shown in Table 21. The schematic model of the Compression Spring Design Problem is shown in Figure 7. Table 21 shows the final results of VAIAO and comparative algorithms after solving this engineering problem. As shown in the Table 16, VAIAO achieves the minimum weight of the compression spring. IAO and AO are in all cases applied to this engineering problem the algorithm presents the worst objective value.

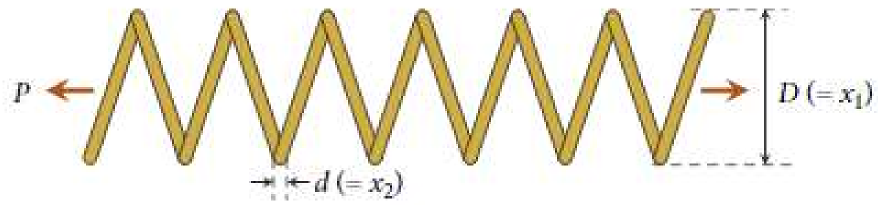

Figure 7. The schematic model of compression spring design problem [72].

Table 21. The results of compression spring design problem.

Algorithm	X_1	X_2	X_3	Best
VAIAO	**0.05**	**0.316923**	**14.1159**	**0.012769**
IAO	0.075746	0.592546	11.6982	0.04657
AO	0.07433	1.0468	2.4959	0.026002
AOA	0.05	0.311514	15	0.013239
HHO	0.057138	0.50254	6.0288	0.013173
STOA	0.051846	0.359471	11.3453	0.012895
ChOA	0.05	0.315274	14.4887	0.012996
SOA	0.05014	0.320512	13.997	0.01289
PSO	0.05886	0.55504	5.039	0.014581
DE	0.05000	0.32349	14.0266	0.0129611

6.3. Pressure vessel design problem

The pressure vessel design is also an engineering problem which needs to optimize cost, including material cost, molding cost and welding cost, to minimize the total cost. This problem has four issues that need to be optimized (the thickness of the shell (X1), the thickness of the head (X2), the inner radius (X3), the length of the vessel section (X4). There are 4 constraints in this problem.

Consider:

$$\vec{x} = [x_1, x_2, x_3, x_4]$$

Objective:

$$f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

Subject to:

$$g_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0$$

$$g_2(\vec{x}) = -x_3 + 0.0954x_3 \leq 0$$

$$g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0$$

$$g_4(\vec{x}) = x_4 - 240 \leq 0$$

Variable ranges:

$$0 \leq x_1 \leq 99$$

$$0 \leq x_2 \leq 99$$

$$10 \leq x_3 \leq 200$$

$$10 \leq x_4 \leq 200$$

The schematic model of pressure vessel is shown in Figure 8. Table 22 shows that the best value of VAIAO is the smallest, which means that the optimized solution is obtained by VAIAO.

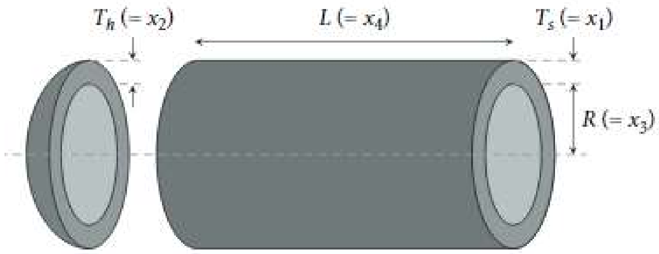

Figure 8. The schematic model of pressure vessel [72].

Table 22. The results of pressure vessel.

Algorithm	X_1	X_2	X_3	X_4	Best
VAIAO	**1.00954**	**0.496241**	**52.2862**	**90.309**	**6.73 × 10³**
IAO	1.33753	1.42724	53.4433	86.9307	1.35 × 10⁴
AO	1.04199	0.558116	54.073	70.3731	6.78 × 10³
AOA	1.03106	1.867994	44.41967	200	1.39 × 10⁴
HHO	1.1552	0.5725	58.5983	42.0081	6.99 × 10³
STOA	1.23586	0.598644	63.3739	18.6267	7.23 × 10³
ChOA	1.27118	0.658186	65.6949	10	7.73 × 10³
SOA	0.779165	0	40.32663	200	1.53 × 10⁵
PSO	0.983267	0.486026	51.0843	89.9709	6.34 × 10³
DE	76.03506	27.32045	159.8693	18.81918	3.4 × 10⁵

6.4. Speeds reducer design problem

The speeds reducer design problem needs to optimize 7 variables to get the minimize the weight of the reducer. Face width parameter is b (=x_1), Tooth modulus is m (=x_2), the number of teeth in the pinion is z (=x_3). Between bearings, the length of the first shaft is l1 (=x_4) and the length of the second shaft l2 (=x_5). the diameter of first shafts is d1 (=x_6), and the diameter of second shafts is d2 (=x_7). At the same time, the speed reducer design problem has 11 constraints:

Consider:

$$\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$$

Objective:

$$f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$$

Subject to:

$$g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$$

$$g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0$$

$$g_3(\vec{x}) = \frac{1.934x_4^3}{x_2x_3x_6^4} - 1 \leq 0$$

$$g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0$$

$$g_5(\vec{x}) = \frac{\sqrt{(\frac{745x_4}{x_2x_3})^2 + 16.9 \times 10^6}}{110x_6^3} - 1 \leq 0$$

$$g_6(\vec{x}) = \frac{\sqrt{(\frac{745x_4}{x_2x_3})^2 + 157.5 \times 10^6}}{85x_7^3} - 1 \leq 0$$

$$g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \leq 0$$

$$g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \leq 0$$

$$g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \leq 0$$

$$g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0$$

$$g_{11}(\vec{x}) = \frac{1.1x_1 + 1.9}{x_5} - 1 \leq 0$$

Variable ranges:

$$2.6 \leq x_1 \leq 3.6$$

$$0.7 \leq x_2 \leq 0.8$$

$$17 \leq x_3 \leq 28$$

$$7.3 \leq x_4 \leq 8.3$$

$$7.8 \leq x_5 \leq 8.3$$

$$2.9 \leq x_6 \leq 3.9$$

$$5.0 \leq x_7 \leq 5.5$$

The schematic model of speeds reducer design problem is shown in Figure 9. The experimental results are shown in Table 23. Table 23 shows the final results of the VAIAO and comparison algorithms after solving the speed reducer design problem. As shown in Table 21, VAAIO has the smallest best value compared to another algorithm.

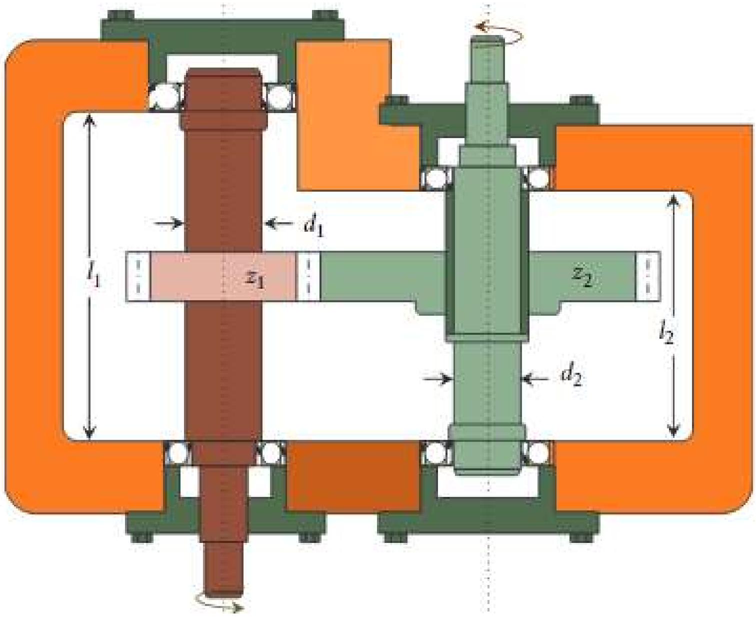

Figure 9. The schematic model of speeds reducer design problem [72].

Table 23. The results of speeds reducer design problem.

Algorithm	X_1	X_2	X_3	X_4	X_5	X_6	X_7	Best
VAIAO	**3.48616**	**0.7**	**17**	**7.42066**	**8.01342**	**3.35348**	**5.31633**	**3032.273**
IAO	3.6	0.7	17	8.3	8.3	3.9	5.5	3363.876
AO	3.49415	0.7	17.0532	8.26393	7.80686	3.43739	5.32147	3060.208
AOA	3.6	0.7	17	7.3	8.3	3.48962	5.29573	3089.259
HHO	3.49436	0.7	17	7.95063	7.8	3.76166	5.28586	3121.865
STOA	3.55498	0.7	17	7.98411	7.89628	3.38054	5.29571	3039.726
ChOA	3.54913	0.7	17	8.3	8.3	3.51904	5.32825	3108.260
SOA	3.49757	0.7	17	8.3	7.8	3.47918	5.28433	3039.816
PSO	3.6	0.7	17	8.3	7.8	3.35006	5.2857	3035.2958
DE	2.70401	0.752413	21.7262	7.61006	8.3	3.89253	5.43619	7120.8812

6.5. Gear design problem

The gear design problem is also a classic engineering optimization problem. The problem is to achieve the minimum transmission ratio cost by optimizing the number of teeth. The four gears are A(x_1), B(x_2), C(x_3), A(x_4). This engineering optimization problems has no constraints.

Consider:

$$\vec{x} = [x_1, x_2, x_3, x_4]$$

Objective:

$$f(\vec{x}) = (\frac{1}{6.913} - \frac{x_2 x_3}{x_1 x_4})^2$$

Variable ranges:

$$12 \le x_1, x_2, x_3, x_4 \le 60$$

The schematic model of gear design problem is shown in Figure 10. The results of gear design problem are shown in Table 24 The results of VAIAO algorithm are also the best compared with other algorithms. As shown in Table 24, Compared with IAO, AO and other objective algorithms, the most competitive results.

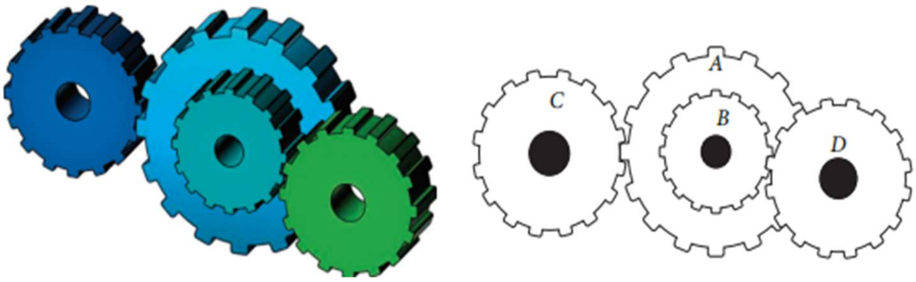

Figure 10. The schematic model of gear design problem [72].

Table 24. The results of gear design problem.

Algorithm	X_1	X_2	X_3	X_4	Best
VAIAO	**44**	**20**	**16**	**49**	$\mathbf{2.70 \times 10^{-12}}$
IAO	52	14	14	27	4.78×10^{-7}
AO	49	23	18	55	1.17×10^{-10}
AOA	60	13	12	19	2.73×10^{-8}
HHO	44	17	19	50	2.70×10^{-12}
STOA	47	12	14	24	9.92×10^{-10}
ChOA	48	13	12	23	1.36×10^{-9}
SOA	50	14	17	34	2.36×10^{-9}
PSO	23	12	13	47	9.92×10^{-10}
DE	24	23	18	60	6.74×10^{-3}

7. Conclusions and future works

In this paper, an enhanced AO algorithm is proposed to improve the exploration ability and convergence speed of IAO. Inspired by the velocity-Aided Global Search Mechanism, the velocity parameters and acceleration parameters are introduced into the AO algorithm to help the search agent update the position and prevent a number of good positions from being missed during the optimization process. In addition, introduced the adaptive opposition-based learning rule is introduced to improve the local optimum. The performance proposed VAIAO algorithm and the comparison algorithms are tested by 27 classical benchmark functions, and five engineering optimization problems. The results of the experiment show that the VAIAO algorithm is easier to obtain better global exploration and exploitation capabilities, faster convergence speed and higher convergence accuracy than any other algorithm. Because the effect of the proposed VAIAO algorithm is better than the original algorithm in terms of optimization. In future work, the proposed VAIAO will be used to solve complex multidisciplinary problems and real-meaning engineering problems, such as Photovoltaic module model parameter extraction, image segmentation and engineering problems of multi-objective optimization.

Acknowledgments

This work was supported by the Innovation Fund for Industry-University-Research in Chinese Universities with grant number 2021ZYA11012, the open project of Hubei Engineering Research Center for Specialty Flowers Biological Breeding with grant number 2022ZD006, the educational teaching research project of Jingchu university of technology with grant number JX 2022-016,the science research project of Jingchu university of technology with grant number YY202203 and the general funded science and technology research project of Jingmen with grant number 2022YFYB152.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
2. M. Mernik, S. H. Liu, D. Karaboga, M. Črepinšek, On clarifying misconceptions when comparing variants of the artificial bee colony algorithm by offering a new implementation, *Inf. Sci.*, **291** (2016), 115–127. https://doi.org/10.1016/j.ins.2014.08.040.
3. R. V. Rao, V. J. Savsani, D. P. Vakharia, Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems, *Inf. Sci.*, **183** (2012), 1–15. https://doi.org/10.1016/j.ins.2011.08.006
4. Y. Tan, Y. Zhu, Fireworks algorithm for optimization, in *International conference in swarm intelligence*, (2010), 355–364. https://doi.org/10.1007/978-3-642-13495-1_44
5. C. Armin, H. K. Mostafa, P. M. Mahdi, Tree Growth Algorithm (TGA), *Eng. Appl. Artif. Intell.*, **72** (2018), 393–414. https://doi.org/10.1016/j.engappai.2018.04.021

6. L. Abualigah, A. Diabatb, S. Mirjalilid, M. A. Elazizf, A. H. Gandomih, The arithmetic optimization algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/j.cma.2020.113609
7. J. F. Frenzel, Genetic algorithms, *IEEE Potentials*, **12** (1993), 21–24. https://doi.org/10.1109/45.282292
8. R. A. Sarker, S. M. Elsayed, R. Tapabrata, Differential evolution with dynamic parameters selection for optimization problems, *IEEE Trans. Evol. Comput.*, **18** (2014), 689–707. https://doi.org/10.1109/TEVC.2013.2281528
9. J. R. Koza, J. P. Rice, Automatic programming of robots using genetic programming, in *Proceedings of the Tenth 20 Computational Intelligence and Neuroscience National Conference on Artificial Intelligence*, (1992).
10. H. G. Beyer, H. P. Schwefel, Evolution strategies–A comprehensive introduction, *Nat. Comput.*, **1** (2002), 3–52. https://doi.org/10.1023/A:1015059928466
11. Z. W. Geem, J. H. Kim, G. V. Loganathan, A new heuristic optimization algorithm: harmony search, *Simulation*, **76** (2001), 60–68. https://doi.org/10.1177/003754970107600201
12 E. Atashpaz-Gargari, C. Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, in *2007 IEEE Congress on Evolutionary Computation,* (2007), 4661–4667. https://doi.org/10.1109/CEC.2007.4425083
13. Q. Zhang, R. Wang, K. D. Juan Yang, Y. Li, J. Hu, Collective decision optimization algorithm: A new heuristic optimization method, *Neurocomputing*, **221** (2017), 123–137. https://doi.org/10.1016/j.neucom.2016.09.068
14. M. Kumar, A. J. Kulkarni, S. C. Satapathy, Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology, *Future Gener. Comput. Syst.*, **81** (2018), 252–272. https://doi.org/10.1016/j.future.2017.10.052
15. A. Qamar, Y. Irfan, S. Mehreen, Political Optimizer: A novel socio-inspired meta-heuristic for global optimization, *Knowl. Based Syst.*, **195** (2020), 105709. https://doi.org/10.1016/j.knosys.2020.105709
16. F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, *Future Gener. Comput. Syst.*, **101** (2019), 646–667. https://doi.org/10.1016/j.future.2019.07.015
17. O. K. Erol, I. Eksin, A new optimization method: big bang–big crunch, *Adv. Eng. Software*, **37** (2006), 106–111. https://doi.org/10.1016/j.advengsoft.2005.04.005
18. S. Mirjalili, S. M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, *Neural Comput. Appl.*, **27** (2016), 495–513. https://doi.org/10.1007/s00521-015-1870-7
19. H. Abedinpourshotorban, S. M. Shamsuddin, Z. Beheshti, D. N. A. Jawawi, Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm, *Swarm Evol. Comput.*, **26** (2016), 8–22. https://doi.org/10.1016/j.swevo.2015.07.002
20. E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, *Inf. Sci.*, **179** (2009), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
21. A. Kaveh, A. Dadras, A novel meta-heuristic optimization algorithm: thermal exchange optimization, *Adv. Eng. Software*, **110** (2017), 69–84. https://doi.org/10.1016/j.advengsoft.2017.03.014

22. R. A. Formato, Central force optimization, *Progress Electromagn. Res*., **77** (2007), 425–491. https://doi.org/10.2528/PIER07082403
23. D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, *J. Global Optim*., **39** (2007), 459–471. https://doi.org/10.1007/s10898-007-9149-x
24. R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, (1995), 39–43.
25. S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowl. Based Syst.,* **89** (2015), 228–249. https://doi.org/10.1016/j.knosys.2015.07.006
26. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man Cybern. Part B*, **26** (1996), 29–41. https://doi.org/10.1109/3477.484436
27. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H. Gandomic, Marine predators algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl*., **152** (2020), 113377. https://doi.org/10.1016/j.eswa.2020.113377
28. D. Gaurav, K. Vijay, Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems, *Knowl. Based Syst*., **165** (2019), 169–196. https://doi.org/10.1016/j.knosys.2018.11.024
29. D. Gaurav, K. Amandeep, STOA: A bio-inspired based optimization algorithm for industrial engineering problems, *Eng. Appl. Artif. Intell*., **82** (2019), 148–174. https://doi.org/10.1016/j.engappai.2019.03.021
30. L. Abualigah, D. Yousri, M. A. Elaziz, A. A.Ewees, M. A. A. Al-qaness, A. H. Gandomi, Aquila Optimizer: a novel meta-heuristic optimization algorithm, *Comput. Ind. Eng*., **157** (2021), 107250. https://doi.org/10.1016/j.cie.2021.107250
31. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
32. S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, *Adv. Eng. Software*, **114** (2017), 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
33. A. A. Heidari, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, *Future Gener. Comput. Syst*., **97** (2019), 849–872. https://doi.org/10.1016/j.future.2019.02.028
34. Y. Feng, S. Deb, G. G. Wang, A. H. Alavi, Monarch butterfly optimization: A comprehensive review, *Expert Syst. Appl*., **168** (2020), 114418. https://doi.org/10.1016/j.eswa.2020.114418
35. S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst*., **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055
36. A. Luque-Chang, E. Cuevas, M. Pérez-Cisneros, F. Fausto, R. Sarkar, Moth swarm algorithm for image contrast enhancement, *Knowl. Based Syst*., **212** (2021), 106607. https://doi.org/10.1016/j.knosys.2020.106607
37. Y. Yang, H. Chen, A. A. Heidari, A. H. Gandomi, Open source MATLAB software of hunger games search (HGS) optimization algorithm, 2021. http://dx.doi.org/10.13140/RG.2.2.10702.18241
38. D. Aniszewska, Multiplicative Runge–Kutta methods, *Nonlinear Dyn*., **50** (2007), 265–272. https://doi.org/10.1007/s11071-006-9156-3

39. R. S. Parpinelli, H. S. Lopes, A. A. Freitas, Data mining with an ant colony optimization algorithm, *Evol. Comput. IEEE Trans*., **6** (2002), 321–332. https://doi.org/10.1109/TEVC.2002.802452
40. I. Ahmadianfar, A. A. Heidari, S. Noshadian, H. Chen, A. H. Gandomi, INFO: An efficient optimization algorithm based on weighted mean of vectors, *Expert Syst. Appl*., **195** (2022). https://doi.org/10.1016/j.eswa.2022.116516
41. F. A. Hashim, R. R. Mostafa, A. G. Hussien, S. Mirjalili, K. M. Sallam, Fick's Law Algorithm: A physical law-based algorithm for numerical optimization, *Knowl. Based Syst*., **260** (2023) 110146. https://doi.org/10.1016/j.knosys.2022.110146
42. A. S. Assiri, A. G. Hussien, M. Amin, Ant lion optimization: Variants, hybrids, and applications, *IEEE Access*, 8 (2020), 77746–77764. https://doi.org/10.1109/ACCESS.2020.2990338
43. Z. M. Gao, J. Zhao, Y. R. Hu, H. F. Chen, The challenge for the nature-inspired global optimization algorithms: Non-symmetric benchmark functions, *IEEE Access*, **9** (2021), 106317–106339. https://doi.org/10.1109/ACCESS.2021.3100365
44. S. Wang, H. Jia, L. Abualigah, Q. Liu, R. Zheng, An improved hybrid aquila optimizer and harris hawks algorithm for solving industrial engineering optimization problems, *Processes*, **9** (2021), 1551. https://doi.org/10.3390/pr9091551
45. M. Ahmadein, Boosting COVID-19 image classification using MobileNetV3 and aquila optimizer algorithm, *Entropy*, **23** (2021), https://doi.org/10.3390/e23111383
46. Y. J. Zhang, Y. X. Yan, J. Zhao, Z. M. Gao, AOAAO: The hybrid algorithm of arithmetic optimization algorithm with aquila optimizer, *IEEE Access*, **10** (2022), 10907–10933. https://doi.org/10.1109/ACCESS.2022.3144431
47. J. Zhao, Y. Zhang, S. Li, Y. Wang, Y. Yan, Z. Gao, A chaotic self-adaptive JAYA algorithm for parameter extraction of photovoltaic models, *Math. Biosci. Eng*., **19** (2022), 5638–5670. https://doi.org/10.3934/mbe.2022264
48. Y. Zhang, Y. Wang, S. Li, F. Yao, L. Tao, Y. Yan, An enhanced adaptive comprehensive learning hybrid algorithm of Rao-1 and JAYA algorithm for parameter extraction of photovoltaic models, *Math. Biosci. Eng*., **19** (2022), 5610–5637. https://doi.org/10.3934/mbe.2022263
49. W. Zhou, P. Wang, A. A. Heidari, X. Zhao, H. Turabieh, M. Mafarja, Metaphor-free dynamic spherical evolution for parameter estimation of photovoltaic modules, *Energy Rep*., **7** (2021), 5175–5202. https://doi.org/10.1016/j.egyr.2021.07.041
50. S. Singh, H. Singh, N. Mittal, H. Singh, A. G. Hussien, F. Sroubek, A feature level image fusion for Night-Vision context enhancement using Arithmetic optimization algorithm based image segmentation, *Expert Syst. Appl*., **209** (2022), 118272. https://doi.org/10.1016/j.eswa.2022.118272.
51. A. G. Hussien, L. Abualigah, R. A. Zitar, F. A. Hashim, M. Amin, A. Saber, et al., Recent advances in harris hawks optimization: A comparative study and applications, *Electronics*, **11** (2022), 1919. https://doi.org/10.3390/electronics11121919
52. S. Wang, A. G. Hussien, H. Jia, L. Abualigah, R. Zheng, Enhanced remora optimization algorithm for solving constrained engineering optimization problems, *Mathematics*, **10** (2022), 1696. https://doi.org/10.3390/math10101696
53. F. A. Hashim, A. G. Hussien, Snake optimizer: A novel meta-heuristic optimization algorithm, *Knowl. Based Syst.*, **242** (2022), 108320. https://doi.org/10.1016/j.knosys.2022.108320

54. R. Zheng, A. G. Hussien, H. M. Jia, L. Abualigah, S. Wang, D. Wu, An improved wild horse optimizer for solving optimization problems, *Mathematics*, **10** (2022), 8. https://doi.org/10.3390/math10081311
55. A. Hussien, R. Mostafa, M. Khan, S. Kadry, F. A. Hashim, Enhanced COOT optimization algorithm for dimensionality reduction, in *2022 Fifth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU)*, (2022). https://doi.org/10.1109/WiDS-PSU54548.2022.00020
56. H. Yu, H. Jia, J. Zhou, A. G. Hussien, Enhanced Aquila optimizer algorithm for global optimization and constrained engineering problems, *Math. Biosci. Eng.,* **19** (2022), 14173–14211. https://doi.org/10.3934/mbe.2022660
57. Y. Yang, C. Qian, H. Li, Y. Gao, J. Wu, C. Liu, et al., An efficient DBSCAN optimized by arithmetic optimization algorithm with opposition-based learning, *J. Supercomput*., **78** (2022), 19566–19604. https://doi.org/10.1007/s11227-022-04634-w
58. Z. Cui, X. Hou, H. Zhou, W. Lian, J. Wu, Modified slime mould algorithm via levy flight, in *13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, (2020).
59. Y. Yang, Y. Gao, S. Tan, S. Zhao, J. Wu, S. Gao, et al., An opposition learning and spiral modelling based arithmetic optimization algorithm for global continuous optimization problems, *Eng. Appl. Artif. Intell.*, **113** (2022), 104981. https://doi.org/10.1016/j.engappai.2022.104981
60. M. Abd Elaziz, D. Oliva, Parameter estimation of solar cells diode models by an improved opposition-based whale optimization algorithm, *Energy Convers. Manage.,* **171** (2018), 1843–1859. https://doi.org/10.1016/j.enconman.2018.05.062
61. A. G. Hussien, M. Amin, M. Abd El Aziz, A comprehensive review of moth-flame optimisation: variants, hybrids, and applications, *J. Exp. Theor. Artif. Intell.,* **32** (2020), 705–725. https://doi.org/10.1080/0952813X.2020.1737246
62. H. Yu, S. Qiao, A. A. Heidari, A. A. El-Saleh, C. Bi, M. Mafarja, et al., Laplace crossover and random replacement strategy boosted Harris hawks optimization: performance optimization and analysis, *J. Comput. Design Eng.,* **9** (2022), 1879–1916. https://doi.org/10.1093/jcde/qwac085
63. A. Qi, D. Zhao, F. Yu, A. A. Heidari, H. Chen, L. Xiao, Directional mutation and crossover for immature performance of whale algorithm with application to engineering optimization, *J. Comput. Design Eng.,* **9** (2022), 519–563. https://doi.org/10.1093/jcde/qwac014
64. D. Zhao, L. Liu, F. Yu, A. A. Heidari, M. Wang, H. Chen, et al., Opposition-based ant colony optimization with all-dimension neighborhood search for engineering design, *J. Comput. Design Eng.*, **9** (2022), 1007–1044. https://doi.org/10.1093/jcde/qwac038
65. X. Zhou, W. Gui, A. A. Heidari, Z. Cai, H. Elmannai, M. Hamdi, et al., Advanced orthogonal learning and Gaussian barebone hunger games for engineering design, *J. Comput. Design Eng.,* **9** (2022), 1699–1736. https://doi.org/10.1093/jcde/qwac075
66. F. Rezaei, H. R. Safavi, M. Abd Elaziz, S. H. A. El-Sappagh, M. A. Al-Betar, T. Abuhmed, An enhanced grey wolf optimizer with a velocity-aided global search mechanism, *Mathematics,* **10** (2022), 351. https://doi.org/10.3390/math10030351
67. J. Zhao, Z. M. Gao, H. F. Chen, The simplified aquila optimization algorithm, *IEEE Access,* **10** (2022), 22487–22515. https://doi.org/10.1109/ACCESS.2022.3153727
68. M. Khishe, M. R. Mosavi, Chimp optimization algorithm, *Expert Syst. Appl.,* **149** (2020), 113338. https://doi.org/10.1016/j.eswa.2020.113338

69. L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. A. Al-qaness, A. H. Gandomi, Aquila Optimizer: A novel meta-heuristic optimization algorithm, *Comput. Ind. Eng.,* **157** (2021), 107250. https://doi.org/10.1016/j.cie.2021.107250
70. A. G. Hussien, M. Amin, A self-adaptive Harris Hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection, *Int. J. Mach. Learn. Cybern.,* **13** (2022), 309–336. https://doi.org/10.1007/s13042-021-01326-4
71. A. G. Hussien, An enhanced opposition-based Salp Swarm Algorithm for global optimization and engineering problems, *J. Ambient Intell. Humanized Comput.,* **13** (2022), 129–150. https://doi.org/10.1007/s12652-021-02892-9
72. H. Bayzidi, S. Talatahari, M. Saraee, C. P. Lamarche, Social network search for solving engineering optimization problems, *Comput. Intell. Neurosci.,* **2021** (2021), 8548639. https://doi.org/10.1155/2021/8548639