ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
https://doi.org/10.17559/TV-20240721001872
Original scientific paper

Adaptive Swarm Intelligence: A Time-Varying Fitness Guided Optimization Algorithm for Dynamic Search Strategies

Yali CHU, Xuming HAN*, Zhiquan LIU, Ting ZHOU

Abstract: Existing swarm intelligence algorithms have utilized fitness to select individuals and developed search strategies based on their locations. However, these algorithms overlook the dynamic variations in fitness over time, limiting the flexibility to adjust search step sizes and reducing optimization performance. To address this, a Time-varying Fitness Guided Optimization Algorithm (TFGOA) is proposed. TFGOA guides individuals to effectively adapt for global and local searches by quantifying time-varying fitness characteristics. Specifically, TFGOA proposes an individual time-varying fitness factor to detect real-time variations in performance. Based on this factor, a time-varying fitness guided search mechanism is developed. This mechanism flexibility adjusts the search behavior for each individual, thereby identifying potential optimal solutions more accurately. TFGOA is tested on the CEC2017, CEC2020, and CEC2022 benchmarks and is comprehensively compared with seven state-of-the-art algorithms. Experimental results demonstrate that TFGOA outperforms the comparison algorithms in terms of convergence and computational time. TFGOA is also tested on welded beam, gear train, and three-bar truss design problems, showing superiority over other comparison algorithms in terms of solution accuracy.

Keywords: fitness; search strategy; swarm intelligence optimization

1 INTRODUCTION

Swarm intelligence optimization algorithms are crucial for solving optimization problems due to their inherent randomness. Their advantages of easy implementation and high flexibility have been widely recognized, enabling successful applications in diverse fields [1-3].

Fitness is a primary criterion in optimization algorithms [4-6]. Most algorithms utilize fitness values to identify key individuals, such as those corresponding to global and local optima. Moreover, the population is segmented into distinct subpopulations based on fitness values. Each subpopulation adopts a tailored search strategy, enhancing overall search efficiency [4, 7]. This approach helps individuals escape local optima and find better solutions. Some algorithms further optimize search strategies by assessing fitness differences among individuals, thus improving search efficiency [8-11]. Although these algorithms are effective in improving search efficiency, they still face a limitation. Notably, these algorithms ignore the dynamic trend of fitness values over time, resulting in a lack of flexibility in adjusting individual search step sizes, thus affecting the accuracy of the algorithms in identifying better solutions.

To address this limitation, a time-varying fitness guided optimization algorithm (TFGOA) is proposed. TFGOA quantifies and utilizes time-varying fitness trends to guide exploration of the solution space while maintaining population diversity. Specifically, TFGOA introduces an individual time-varying fitness factor. This factor quantifies variations in fitness ranking for each individual and determines the search step size for the next iteration. Leveraging this factor, TFGOA employs a time-varying fitness guided search mechanism. This mechanism flexibly explores the solution space to identify potentially better solutions. It consists of two search strategies: a time-varying optimal center adaptation strategy and a time-varying individual center adaptation strategy. The former directs individuals to explore around the global optimum or their best-known solution in memory for improved solutions. The latter allows individuals to explore potentially better solutions in the current solution space location.

The main contributions of this paper are as follows:

- An individual time-varying fitness factor is proposed. This factor quantifies the variations in individual fitness over time, guiding the determination of the search step size for each iteration.
- A time-varying fitness guided search mechanism is proposed. This mechanism allows individuals to adjust their search step sizes according to their time-varying fitness factors. Consequently, the mechanism enables flexible searching around the optimal or current solution, improving solution accuracy.

The structure of the remainder of this paper is as follows: Section 2 reviews the latest advancements in fitness-based optimization algorithms. Section 3 provides detailed descriptions and the pseudo code of TFGOA. Section 4 validates and analyzes the performance of TFGOA. Three engineering design problems are presented in Section 5. Lastly, Section 6 presents conclusions and future work.

2 RELATED WORK

2.1 Specific Individual Selection Based on Fitness Value

Fitness values serve as crucial indicators of individual performance, enabling the selection of specific individuals within a population. Various algorithms update their positions and explore the solution space based on these selected individuals [6, 12, 13]. For example, Zhang [12] identified global and local optima by comparing fitness values, updating positions based on those optima and individual characteristics. Jayashree and Malarvizhi [6] determined the optimal particle position based on fitness values, with the particle having the maximum fitness value replacing the global leader in the entire solution space. Many optimization algorithms segment the population according to fitness values, assigning distinct search strategies to different subpopulations [4, 7, 14]. For instance, Minh et al. [14] determined the social hierarchy of ants in a termite colony by assessing their fitness values and assigning them roles as workers, soldiers, or reproductive termites, along with corresponding search tasks. However, static grouping based on fitness values may limit the algorithm's adaptability, leading to a decrease in population diversity and consequently causing premature convergence.

2.2 Search Strategy Selection Based on Fitness Difference

Some studies focus on fitness differences among individuals within a population to select appropriate search strategies [8-10, 15]. For example, Zamani et al. [15] assessed the quality of a bird flock population based on fitness. This algorithm compared quality differences between subpopulations and the overall average while adapting diving and rotation strategies to promote balanced exploration and development. Geng et al. [16] determined the number of duplicated individuals in the search pattern based on the magnitude of their fitness values. This approach increases the proportion of high-quality solutions in the population to balance the search and tracking modes. Jia et al. [8] determined the optimal solution by calculating the differences between the optimal solution and each individual's fitness value. They subsequently chose a more suitable feeding method during the foraging phase. Abdel-Basset et al. [9] calculated the probabilities of applying colored light scattering based on the current, best, and worst solutions, and selected different search strategies accordingly. These strategies may be overly sensitive to the initial distribution of the population, and minor fluctuations in initial fitness differences could lead to suboptimal search directions.

2.3 Optimization by Adaptive Search Strategies

Some important works on adaptive search strategies have been developed to balance exploration and exploitation capabilities during evolution [17-20]. For example, Li et al. [17] developed two competitive and cooperative strategies to update particles' information for winners and losers in the population, effectively improving a convergence rate and population diversity. Zeng et al. [18] designed an adaptive search strategy by comparing the success rate of the current and previous iterations, which is beneficial for balancing exploration and exploitation. Recently, Qaraad et al. [20] proposed an adaptive local search strategy to escape from local optima by using random operators, ensuring a balanced interplay between exploration and exploitation. However, these adaptive search strategies are dependent on the settings of parameters. Incorrect parameter values may lead to suboptimal performance, while finding the optimal parameter values often requires extensive trial-and-error, which may slow down the evolutionary process.

3 METHOD

TFGOA aims to comprehensively explore the solution space by quantifying and leveraging the time-varying characteristics of individual fitness values. To achieve this, an individual time-varying fitness factor is developed that quantifies the temporal trend of fitness values. Building on this factor, TFGOA proposes a time-varying fitness guided search mechanism, allowing for customized searches within the solution space. The flowchart of TFGOA is illustrated in Fig. 1.

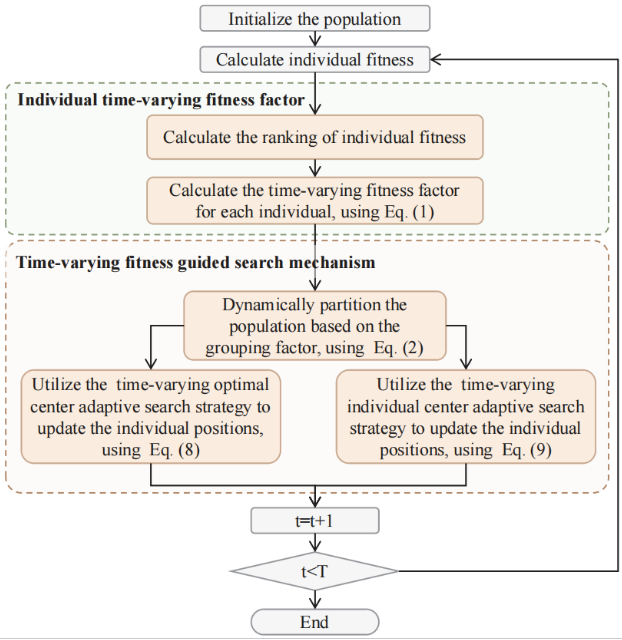


Figure 1 The flowchart of the proposed TFGOA

3.1 Individual Time-Varying Fitness Factor

An individual time-varying fitness factor is constructed to assess the variations in individual fitness over time. Consequently, different search steps are assigned to each individual during every iteration. The formula for calculating the fitness factor is as follows:

$$fitF(X_i^t) = \frac{fitRank(X_i^t)}{N} * u * \frac{t}{T}, \tag{1}$$

where $fitF(X_i^t)$ represents the fitness factor of the i-th individual, X_i, within the population at iteration t, with values ranging from –1 to 1. The notation $fitRank(X_i^t)$ signifies the fitness value ranking of individual X_i at iteration t. The individual possessing the smallest fitness value is assigned the highest rank, $fitRank(bestX^t) = 1$. Furthermore, the random variable μ is distributed uniformly within the interval [–1, 1]. Lastly, T indicates the maximum number of iterations permitted in the algorithm.

3.2 Time-Varying Fitness Guided Search Mechanism

A time-varying fitness guided search mechanism is proposed to enhance search flexibility. This mechanism dynamically adjusts the search step size based on a time-varying fitness factor, allowing it to focus on either the optimal or current individual. TFGOA algorithm employs a dynamic grouping factor to divide the population into two subpopulations. Elite-followers utilize a time-varying optimal center adaptation strategy, whereas self-actualizers employ a time-varying individual center adaptation strategy.

3.2.1 Grouping Factor

TFGOA categorizes a population of N individuals into elite-followers and self-actualizers. The grouping factor in the t-th iteration is defined by:

$$groupF^t = r_1 * \sin(\frac{t}{T} * \pi), \tag{2}$$

where $groupF^t \in [0,1]$. r_1 is a random number between zero and one. T indicates the maximum number of iterations. The grouping factor is dynamically adjusted with iteration, as shown in Fig. 2. During the early iterations ($0 \sim 200$) and late iterations ($800 \sim 1000$), the grouping factor is less than 0.5. This results in self-actualizers predominantly occupying the population, enabling the TFGOA algorithm to focus on enhancing the exploration capability of the population to search for potential optimal solutions in the solution space. In the mid-iterations ($200 \sim 800$), the proportion of self-actualizers and elite-followers varies randomly, facilitating a more comprehensive search.

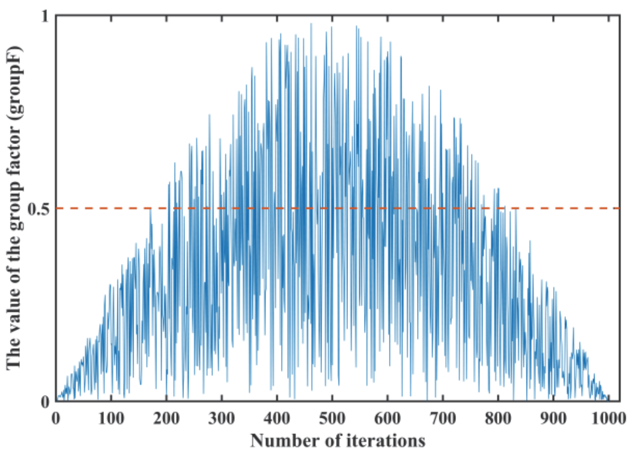


Figure 2 The trend of group factor with increasing number of iterations

Elite-follower. At t-th iteration, N^t_{Follower} individuals are randomly selected from population $X = \{X_1, \cdots, X_N\}$. The elite-followers as follows:

$$N^t_{\text{Follower}} = \lfloor groupF^t * N \rfloor, \tag{3}$$

$$\begin{aligned} FolX^t &= randSelect(X, N^t_{\text{Follower}}) \\ &= \left\{FX^t_1, \cdots, FX^t_s, \cdots, FX^t_{N^t_{\text{Follower}}}\right\}, \end{aligned} \tag{4}$$

where $FX^t_s = \{fx^t_{(s,1)}, \cdots, fx^t_{(s,d)}, \cdots, fx^t_{(s,D)}\}$ represents the sth elite-follower. $fx^t_{(s,d)}$ denotes a decision variable of the d-th dimension problem for FX^t_s. The variable Dim signifies the dimension of the optimization problem. $\lfloor \cdot \rfloor$ denotes the largest integer no larger than the real number within the symbols.

Self-actualizer. Excluding elite-followers, all remaining individuals in the population are self-actualizers. The self-actualizers are set as follows:

$$N^t_{\text{Actualizer}} = N - N^t_{\text{Follower}}, \tag{5}$$

$$\begin{aligned} ActX^t &= X - FolX^t \\ &= \left\{AX^t_1, \cdots, AX^t_g, \cdots, AX^t_{N^t_{\text{Actualizer}}}\right\}, \end{aligned} \tag{6}$$

where $AX^t_g = \{ax^t_{(g,1)}, \cdots, ax^t_{(g,d)}, \cdots, ax^t_{(g,D)}\}$ denotes the g-th self-actualizer. $N^t_{\text{Actualizer}}$ indicates the number of self-actualizers.

3.2.2 Time-Varying Optimal Center Adaptation Search

Elite-followers concentrate their search efforts around the best individual in the population, which can be either the individual's locally remembered best solution or the global best solution. They leverage the individual fitness factor to explore the nearby solution space. The time-varying optimal center adaptation search process is represented by the following formula:

$$fx^{t+1}_{(s,d)} = \begin{cases} fx^t_{(s,d),\text{mem}} + fitF(FX^t_s) \\ *(lb + r_2 * (ub - lb)), & \text{if } groupR < 0.5, \\ bestX^t_d + \cos(2\pi * fitF(FX^t_s)) & \text{others}, \\ *(bestX^t_d - fx^t_{(s,d)}), \end{cases} \tag{7}$$

where $fx^t_{(s,d),\text{mem}}$ represents the position of the d dimension of the best solution $FX^t_{s,\text{mem}}$ in the memory of the s-th elite-follower. $fitF(FX^t_s)$ represents the individual time-varying fitness factor of FX^t_s, as shown in Eq. (1). lb and ub represent the lower and upper bounds of the entire search space, respectively. r_2 is a random number in the range [0, 1]. $bestX^t_d$ denotes the position of the global best individual $bestX^t$ in the population at iteration t in the d dimension.

In the optimal-centered adaptive search strategy, individuals search for a better solution by exploring the area surrounding the optimal one. Elite-followers with higher fitness factors conduct a more focused search near the optimal solution, whereas those with lower fitness factors broaden their search area.

3.2.3 Time-Varying Individual Center Adaptation Search

Self-actualizers incorporate their fitness factor and positional information to determine the next position within the search space. The process by which the g-th self-actualizer utilizes individual-centered adaptive search is as follows:

$$ax_{(g,d)}^{t+1} = \begin{cases} ax_{(g,d)}^{t} + fitF(AX_{g}^{t}) \\ *(ax_{(g,d),\text{mem}}^{t} - ax_{(g,d)}^{t}), & \text{if } rand < 0.5, \\ \\ ax_{(g,d)}^{t} + fitF(AX_{g}^{t}) \\ *|rn_1|*(bestX_{d}^{t} - ax_{(g,d)}^{t}) & \text{others}, \\ +(1 - fitF(AX_{g}^{t})) \\ *|rn_2|*(randX_{d}^{t} - ax_{(g,d)}^{t}), \end{cases} \tag{8}$$

where $ax_{(g,d),\text{mem}}^{t}$ represents the excellent position in the d dimension in the memory of the self-actualizer. $bestX_{d}^{t}$ and $randX_{d}^{t}$ denote the positions of the best and random individuals in the population in the d dimension, respectively. rn_1 and rn_2 are two random numbers drawn from a normal distribution, controlling the extent to which $bestX^{t}$ and $randX^{t}$ influence the self-actualizer, respectively. Self-actualizers employ an individual-centered adaptive search strategy, which helps to avoid bias toward a single optimal individual and effectively escape local optima. The pseudocode of TFGOA is as shown in Algorithm 1.

Algorithm 1: Pseudo code of TFGOA
Require: The size of the population N, the maximum iterations T. Ensure: Optimal position $bestX^{t}$ and its fitness value.

Initialize each individual's position, $X_{i}^{t}, i = (1,2,...,N)$, $t = 1$;
Calculate the fitness of each individual X_{i}^{t} and find the global optimal individual $bestX$;
Obtain the optimal location of individual memory $X_{i,\text{mem}}^{t}$;
While $t \le T$
Calculate time-varying fitness factors using Eq. (1);
Obtain a group factor $groupF^{t}$ using Eq. (2);
Select elite-followers $FolX^{t}$. using Eqs. (3) and (4);
Obtain self-actualizers $ActX^{t}$ using Eqs. (5) and (6);
For $FX \in FolX^{t}$
Update FX position using Eq. (7);
EndFor
For $AX \in ActX^{t}$
Update AX position using Eq. (8);
End For
Calculate the fitness of each individual;
Find the global optima $bestX^{t}$ and $X_{i,\text{mem}}^{t}(i = 1,\cdots,N)$;
$t = t + 1$;
End While

3.3 Computational Complexity

TFGOA contains two search strategies: a time-varying optimal center adaptation search and a time-varying individual center adaptive search strategy. The computational complexity of the first search strategy is $O(N_{\text{Follower}} \cdot D \cdot T)$, and the computational complexity of the second search strategy is $O(N_{\text{Actualizer}} \cdot D \cdot T)$. Overall, the computational complexity of TFGOA is $O(N \cdot D \cdot T)$.

Furthermore, the computational complexities of the seven comparison algorithms are as follows: CPO [23] is $O(N \cdot D \cdot T)$, COA [26] is $O(N \cdot D \cdot T)$, SO [11] is $O(N \cdot D \cdot T)$, MGO [27] is $O(4 \cdot N \cdot D \cdot T)$, HO [24] is $O\left(N \cdot D \cdot (1 + \frac{5}{2}T)\right)$, GKSO [25] is $O\left(N \cdot D \cdot (1 + 4T)\right)$, and EAPSO [12] is $O\left((0.5ND + N^2 + 6.5N) \cdot T\right)$. Overall, compared with these seven algorithms, TFGOA has better computational complexity.

4 EXPERIMENTS

This section validates the effectiveness of TFGOA through comparative experiments. Section 4.1 outlines the experimental configuration, including benchmark functions, compared algorithms, parameter settings, and experimental platform. Section 4.2 presents and discusses detailed comparison results against seven state-of-the-art optimization algorithms. Section 4.3 analyzes the performance of TFGOA, including convergence analysis, robustness analysis, statistical tests analysis, and computational time. Finally, the ablation experiment of TFGOA is given in Section 4.4.

4.1 Experimental Configuration

Benchmark functions. The performance of TFGOA is assessed using three benchmark functions: CEC2017, CEC2020, and CEC2022. In the CEC2017 benchmark, 29 test functions are evaluated in 30-dimensional and 50-dimensional spaces, respectively. The F2 function is excluded due to its erratic behavior in CEC2017, as noted in other studies [10, 21-23]. For the CEC2020 and CEC2022 test suites, experiments are conducted on 10 and 12 functions, respectively, in a 20-dimensional space.

Compared algorithms. Extensive comparisons are conducted to demonstrate the effectiveness of TFGOA, comparing it with seven state-of-the-art optimization algorithms published in Q1 journals. These algorithms include CPO [23], HO [24], GKSO [25], COA [26], SO [11], MGO [27], EAPSO [12].

Table 1 Parameter settings of compared algorithms

Algorithm	Parameter settings		Year/Rank
	Meaning	Value	
CPO [23]	Minimum population size	20	2024/Q1
	Number of cycles	0.1	
	convergence rate	80	
	Percentage of the tradeoff	0.5	
HO [24]	-	-	2024/Q1
GKSO [25]	convergence speed factor	1.5	2023/Q1
COA [26]	-	-	2023/Q1
EAPSO [12]	-	-	2023/Q1
SO [11]	constant (c_1)	0.5	2022/Q1
	constant (c_2)	0.05	
	constant (c_3)	2	
	A threshold for finding food	0.25	
	Temperature threshold	0.6	
MGO [27]	-	-	2022/Q1

Parameter settings. Experiments are conducted using the parameter values suggested by the comparative algorithms, and Tab. 1 displays the specific parameter settings. A population size of 30 is set for all algorithms, with a maximum function evaluation of 30000 times. Each algorithm is run independently 30 times to ensure statistical reliability.

Experimental platform. All experiments are implemented in MATLAB R2020a software and run on an Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz 3.19 GHz and 8 GB RAM and x64-based processor.

4.2 Compare with State-Of-The-Art Optimization Algorithms

Tab. 2 and Tab. 3 show the experimental results of 30 and 50 dimensions of CEC2017 test functions respectively, and the optimal values are shown in bold. Tab. 2 data show TFGOA is superior in finding minimum fitness values. It outperforms other algorithms in 20 of the tested functions: F1, F3, F5, F7-F9, F11, F16-F18, F20-F27, F29, and F30. The solution accuracy of TFGOA on F4, F12, and F28 is lower than that of EAPSO, but significantly higher than that of HO, GKSP, COA, SO and MGO. TFGOA ranks first with an average of 1.3. In 50 dimensional search spaces, TFGOA performs even better, aiding in finding optimal solutions. Tab. 3 shows TFGOA ranks highest on 22 benchmark functions.

The experimental results on the CEC2020 benchmark, presented in Tab. 4, demonstrate that TFGOA outperforms other algorithms in mean fitness and standard deviation across seven functions. This performance, with an average ranking of 1.5, highlights the stability of TFGOA in solving complex optimization problems. Additionally, Tab. 5 presents the experimental results for the CEC2022 benchmark in 20-dimensional space, showing that TFGOA achieves a better mean fitness for nine out of twelve functions. This result further validates the excellent performance of TFGOA in solving optimization problems. As shown in Tabs. 2, 3, 4, and 5, TFGOA achieves excellent performance on the 20-dimensional, 30-dimensional, and 50-dimensional problems. This is attributed to the time-varying adaptive strategy in TFGOA, which effectively matches the dynamic characteristics of search behavior. By flexibly adjusting the search behavior of each individual, TFGOA is able to more accurately identify potential optimal solutions.

Table 2 Results of CEC2017 (Dim = 30) test function with comparative algorithms

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F1	avg	**2.4446E+03**	2.4720E+03	2.0187E+09	4.4164E+08	5.6641E+10	2.0195E+05	5.3844E+06	4.6030E+03
F1	std	**2.7259E+03**	2.8516E+03	9.1226E+08	1.4639E+08	6.2251E+09	2.1624E+05	1.6294E+06	5.0099E+03
F3	avg	**4.5265E+03**	4.0031E+04	6.5157E+04	4.2147E+04	8.3631E+04	6.6094E+04	5.1997E+04	2.5722E+04
F3	std	**2.4107E+03**	1.0308E+04	5.4466E+03	8.6288E+03	6.0914E+03	8.5673E+03	6.9943E+03	2.1211E+04
F4	avg	4.9301E+02	4.9959E+02	7.9827E+02	6.1215E+02	1.6662E+04	5.0758E+02	5.2214E+02	**4.6423E+02**
F4	std	2.6074E+01	2.1965E+01	9.7926E+01	4.6426E+01	2.4777E+03	2.8843E+01	**1.4663E+01**	3.0252E+01
F5	avg	**5.5645E+02**	6.4727E+02	7.5361E+02	7.4430E+02	9.2311E+02	5.9781E+02	6.1043E+02	5.9795E+02
F5	std	1.5042E+01	1.4979E+01	3.4975E+01	3.9581E+01	3.5146E+01	**1.4461E+01**	2.4326E+01	2.7116E+01
F6	avg	6.0003E+02	**6.0003E+02**	6.6390E+02	6.5520E+02	6.9136E+02	6.1491E+02	6.1700E+02	6.1114E+02
F6	std	1.7235E–02	**1.1972E–02**	8.3047E+00	1.0661E+01	6.2007E+00	5.7898E+00	5.8559E+00	7.4654E+00
F7	avg	**7.8190E+02**	8.9308E+02	1.1891E+03	1.0728E+03	1.4352E+03	8.5712E+02	9.3104E+02	8.7063E+02
F7	std	**1.2221E+01**	1.8558E+01	6.2390E+01	5.6428E+01	5.6401E+01	2.6190E+01	4.3335E+01	3.5959E+01
F8	avg	**8.5326E+02**	9.4208E+02	9.7846E+02	9.9687E+02	1.1434E+03	8.9936E+02	9.0297E+02	9.1078E+02
F8	std	**1.3294E+01**	2.4028E+01	3.1435E+01	3.2205E+01	2.4324E+01	1.8601E+01	1.9940E+01	3.2237E+01
F9	avg	**9.1881E+02**	9.8194E+02	6.2819E+03	6.1659E+03	1.1307E+04	2.0042E+03	3.3716E+03	2.4518E+03
F9	std	**2.9622E+01**	9.0570E+01	7.4114E+02	2.0153E+03	8.3999E+02	6.4904E+02	1.2914E+03	1.0662E+03
F10	avg	4.3186E+03	6.8060E+03	5.9383E+03	6.5198E+03	8.9494E+03	**3.5671E+03**	7.2337E+03	4.6899E+03
F10	std	5.8140E+02	**3.6362E+02**	6.4439E+02	7.8427E+02	4.0822E+02	4.0631E+02	1.3540E+03	6.1943E+02
F11	avg	**1.1759E+03**	1.2046E+03	2.0120E+03	1.5188E+03	8.8598E+03	1.3060E+03	1.2604E+03	1.2110E+03
F11	std	3.4525E+01	**2.5193E+01**	2.6340E+02	9.9803E+01	2.2698E+03	8.1318E+01	4.6277E+01	6.1522E+01
F12	avg	7.9462E+04	2.6018E+05	2.3818E+08	9.9563E+07	1.4427E+10	2.6585E+06	5.8970E+06	**4.3687E+04**
F12	std	8.4734E+04	2.3684E+05	1.9792E+08	6.3663E+07	3.4638E+09	2.6085E+06	4.7104E+06	**2.1042E+04**
F13	avg	1.1316E+04	**8.5517E+03**	1.9013E+05	1.0410E+06	1.0847E+10	3.0339E+04	1.9176E+04	2.2532E+04
F13	std	**5.7151E+03**	7.9480E+03	2.0364E+05	6.5834E+05	4.5152E+09	2.1250E+04	1.4651E+04	2.1169E+04
F14	avg	1.6828E+03	**1.5297E+03**	8.8261E+05	2.9183E+04	3.6529E+06	6.0421E+04	6.4185E+04	1.3464E+04
F14	std	3.0487E+02	**4.4695E+01**	1.0882E+06	2.9850E+04	3.4592E+06	6.2661E+04	6.1309E+04	7.6723E+03
F15	avg	2.6099E+03	**2.2994E+03**	4.5004E+04	9.3681E+04	6.2712E+08	1.2188E+04	5.2247E+03	1.8674E+04
F15	std	1.5036E+03	**1.4012E+03**	3.9146E+04	6.6687E+04	3.7398E+08	9.4594E+03	5.0608E+03	1.3100E+04
F16	avg	**2.2531E+03**	2.5257E+03	3.5597E+03	3.1678E+03	6.3508E+03	2.5659E+03	2.6348E+03	2.5668E+03
F16	std	**1.9227E+02**	2.6811E+02	3.3430E+02	3.8926E+02	1.1734E+03	2.7611E+02	2.9016E+02	2.9506E+02
F17	avg	**1.8275E+03**	1.8959E+03	2.5293E+03	2.3782E+03	4.3986E+03	2.2553E+03	2.1698E+03	2.2632E+03
F17	std	**6.1962E+01**	1.2298E+02	2.4320E+02	2.1750E+02	1.6262E+03	1.4604E+02	2.2261E+02	1.8080E+02
F18	avg	**4.2124E+04**	5.2110E+04	2.2894E+06	3.7415E+05	3.8620E+07	5.5197E+05	1.0157E+06	1.4719E+05
F18	std	**1.9040E+04**	3.3731E+04	2.8264E+06	3.9226E+05	3.5760E+07	4.9608E+05	1.0056E+06	8.8108E+04
F19	avg	4.2601E+03	**2.2209E+03**	4.3198E+06	2.0011E+06	7.2483E+08	1.2193E+04	6.4975E+03	1.2659E+04
F19	std	3.0666E+03	**7.6513E+02**	3.6304E+06	1.8751E+06	4.5120E+08	9.3595E+03	3.6945E+03	1.0858E+04
F20	avg	**2.2270E+03**	2.2447E+03	2.5924E+03	2.5843E+03	3.0125E+03	2.4897E+03	2.4363E+03	2.5987E+03
F20	std	**8.5729E+01**	1.2353E+02	1.1336E+02	1.6413E+02	2.1272E+02	1.4541E+02	1.8451E+02	2.3824E+02
F21	avg	**2.3528E+03**	2.4455E+03	2.5413E+03	2.5075E+03	2.7541E+03	2.3963E+03	2.3926E+03	2.3951E+03
F21	std	**1.2653E+01**	2.2034E+01	3.4896E+01	4.0074E+01	4.8387E+01	1.6687E+01	2.4286E+01	2.2323E+01
F22	avg	**2.3000E+03**	2.3000E+03	5.3833E+03	2.8334E+03	9.2945E+03	3.9568E+03	2.9950E+03	4.8068E+03
F22	std	**2.1784E–03**	6.9644E–03	2.2217E+03	1.3351E+03	9.1856E+02	1.4784E+03	1.7663E+03	2.0348E+03
F23	avg	**2.7080E+03**	2.7966E+03	3.0078E+03	2.9324E+03	3.6433E+03	2.8151E+03	2.7780E+03	2.7529E+03
F23	std	1.6568E+01	**1.4401E+01**	9.0249E+01	4.7528E+01	1.9638E+02	4.3370E+01	2.8459E+01	3.7465E+01
F24	avg	**2.8694E+03**	2.9527E+03	3.2304E+03	3.0928E+03	3.7729E+03	2.9685E+03	2.9267E+03	2.9243E+03

Table 2 Results of CEC2017 (Dim = 30) test function with comparative algorithms (continuation)

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F24	std	**1.3108E+01**	3.1965E+01	1.3836E+02	6.8888E+01	1.6559E+02	3.1825E+01	2.4231E+01	3.9525E+01
F25	avg	**2.8889E+03**	2.8927E+03	3.0822E+03	3.0229E+03	5.2306E+03	2.9046E+03	2.9187E+03	2.8889E+03
F25	std	**2.9045E+00**	9.8152E+00	3.7572E+01	4.6152E+01	5.0156E+02	1.7742E+01	2.3580E+01	6.4044E+00
F26	avg	**4.2824E+03**	4.5118E+03	7.0112E+03	6.4414E+03	1.1596E+04	5.6006E+03	4.9173E+03	4.5146E+03
F26	std	5.9172E+02	1.1227E+03	1.8006E+03	1.2858E+03	8.0956E+02	**4.3968E+02**	7.3885E+02	7.1916E+02
F27	avg	**3.2286E+03**	3.2368E+03	3.5183E+03	3.3021E+03	4.5840E+03	3.2912E+03	3.2509E+03	3.2370E+03
F27	std	**6.9238E+00**	1.2240E+01	1.3613E+02	4.1677E+01	4.4949E+02	3.0210E+01	2.1386E+01	2.1567E+01
F28	avg	3.2157E+03	3.2251E+03	3.5287E+03	3.3816E+03	7.6533E+03	3.2831E+03	3.2671E+03	**3.1866E+03**
F28	std	**1.6068E+01**	2.2024E+01	1.0724E+02	3.4298E+01	7.6633E+02	2.5824E+01	2.1305E+01	6.2156E+01
F29	avg	**3.5549E+03**	3.6277E+03	5.0478E+03	4.4693E+03	7.9620E+03	4.0740E+03	3.9016E+03	3.9887E+03
F29	std	**9.7786E+01**	1.0217E+02	6.2481E+02	2.8699E+02	1.7788E+03	2.6363E+02	2.2071E+02	1.9987E+02
F30	avg	**8.0719E+03**	1.5822E+04	4.9735E+07	6.5618E+06	1.7189E+09	1.1732E+05	1.3573E+05	1.0827E+04
F30	std	**2.2375E+03**	5.3644E+03	6.8222E+07	4.0269E+06	7.8365E+08	2.0023E+05	1.8941E+05	3.8099E+03
Average Rank		**1.31**	2.69	6.72	5.72	8.00	4.10	4.21	3.24

Table 3 Results of CEC2017 (Dim = 50) test function with comparative algorithms

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F1	avg	1.1290E+04	2.5785E+05	1.2225E+10	3.7199E+09	1.1353E+11	1.0811E+07	8.5057E+07	**7.7666E+03**
F1	std	1.0971E+04	1.6772E+05	3.1332E+09	8.2725E+08	9.4699E+09	5.8581E+06	1.9388E+07	**6.9799E+03**
F3	avg	**3.9668E+04**	1.4471E+05	1.4411E+05	1.3465E+05	2.0442E+05	1.5071E+05	1.4705E+05	1.7287E+05
F3	std	**9.5167E+03**	2.0719E+04	1.3893E+04	2.2675E+04	2.6648E+04	1.4942E+04	2.0423E+04	5.2533E+04
F4	avg	5.4654E+02	5.8706E+02	2.6799E+03	1.2536E+03	3.9834E+04	6.6238E+02	6.9773E+02	**5.0429E+02**
F4	std	5.1250E+01	4.1734E+01	6.1377E+02	1.8147E+02	4.5787E+03	4.1428E+01	5.6477E+01	**3.6248E+01**
F5	avg	**6.1834E+02**	8.6442E+02	9.1974E+02	9.9813E+02	1.2020E+03	7.0931E+02	7.8084E+02	7.2512E+02
F5	std	**2.0972E+01**	2.7236E+01	2.9595E+01	3.8047E+01	3.9849E+01	3.0469E+01	4.4348E+01	5.5722E+01
F6	avg	**6.0064E+02**	6.0125E+02	6.7510E+02	6.7604E+02	7.0220E+02	6.2684E+02	6.3932E+02	6.2444E+02
F6	std	**3.0902E–01**	5.8992E–01	4.8232E+00	1.0068E+01	4.2180E+00	6.4257E+00	8.9346E+00	1.0447E+01
F7	avg	**8.6126E+02**	1.1697E+03	1.7019E+03	1.5893E+03	2.0448E+03	1.0693E+03	1.2737E+03	1.1486E+03
F7	std	**2.4945E+01**	3.2186E+01	7.2014E+01	1.2213E+02	5.4057E+01	6.2319E+01	9.1390E+01	1.0720E+02
F8	avg	**9.1826E+02**	1.1586E+03	1.2021E+03	1.2690E+03	1.4890E+03	1.0174E+03	1.0677E+03	1.0175E+03
F8	std	**2.8950E+01**	3.4245E+01	3.8441E+01	4.3067E+01	3.9013E+01	3.5077E+01	5.2764E+01	5.1650E+01
F9	avg	**1.5177E+03**	5.0620E+03	2.2409E+04	2.7047E+04	3.8283E+04	6.2515E+03	1.5807E+04	8.7701E+03
F9	std	**3.8173E+02**	2.1612E+03	2.7364E+03	5.2146E+03	3.0874E+03	2.6128E+03	5.8338E+03	3.1858E+03
F10	avg	7.0156E+03	1.2641E+04	9.9362E+03	1.1524E+04	1.5480E+04	**6.1145E+03**	1.2571E+04	7.5385E+03
F10	std	7.4846E+02	4.3473E+02	1.2267E+03	8.4105E+02	**4.1790E+02**	8.5897E+02	2.6495E+03	1.0699E+03
F11	avg	**1.2918E+03**	1.4519E+03	5.4909E+03	3.5730E+03	2.7247E+04	1.9160E+03	1.6160E+03	1.3078E+03
F11	std	**3.9051E+01**	6.4629E+01	1.0321E+03	6.3751E+02	1.9265E+03	2.6079E+02	8.0454E+01	7.2029E+01
F12	avg	1.8430E+06	2.8736E+06	1.9950E+09	6.9781E+08	8.5249E+10	2.4216E+07	8.1067E+07	**9.0842E+05**
F12	std	7.4077E+05	1.4604E+06	8.0371E+08	3.8513E+08	1.5289E+10	1.2785E+07	4.3289E+07	**6.3002E+05**
F13	avg	**2.9944E+03**	3.7326E+03	4.0704E+07	2.3018E+07	5.1069E+10	7.2607E+04	2.8778E+05	6.2688E+03
F13	std	**1.3257E+03**	2.2113E+03	6.0168E+07	1.2391E+07	1.4783E+10	4.6325E+04	1.4650E+05	5.0951E+03
F14	avg	**3.4782E+04**	3.7176E+04	3.2910E+06	5.3149E+05	1.4108E+08	2.8364E+05	5.5249E+05	6.2273E+04
F14	std	3.2335E+04	**3.1596E+04**	3.6172E+06	4.4313E+05	1.1360E+08	2.2818E+05	3.6111E+05	4.1109E+04
F15	avg	**7.4025E+03**	8.0978E+03	1.7230E+05	2.4994E+06	9.2791E+09	1.9031E+04	4.8926E+04	8.3369E+03
F15	std	**3.1907E+03**	3.4938E+03	6.6546E+04	1.5667E+06	3.2616E+09	9.3780E+03	2.3049E+04	7.5518E+03
F16	avg	**2.8745E+03**	3.6542E+03	5.1704E+03	4.8030E+03	9.8670E+03	3.2330E+03	3.7362E+03	3.3320E+03
F16	std	**2.7741E+02**	4.9069E+02	8.3708E+02	6.7974E+02	1.4340E+03	3.1250E+02	4.6901E+02	5.1486E+02
F17	avg	**2.6949E+03**	2.8995E+03	3.8347E+03	3.5968E+03	1.0355E+04	3.2302E+03	3.0985E+03	3.2357E+03
F17	std	**2.2791E+02**	2.4812E+02	4.6180E+02	3.6680E+02	5.5571E+03	3.4036E+02	2.8608E+02	3.0756E+02
F18	avg	3.5485E+05	8.8823E+05	1.0173E+07	2.9831E+06	2.2625E+08	2.6319E+06	3.0327E+06	**2.9543E+05**
F18	std	5.4478E+05	8.6666E+05	6.5039E+06	1.9965E+06	1.1387E+08	1.7148E+06	2.0828E+06	**1.4842E+05**
F19	avg	**1.6121E+04**	1.8706E+04	7.4100E+06	5.1532E+06	4.4911E+09	2.4271E+04	3.5048E+04	1.6711E+04
F19	std	**4.5578E+03**	7.5225E+03	1.1370E+07	6.1011E+06	1.9265E+09	1.5173E+04	1.4768E+04	1.3672E+04
F20	avg	**2.8005E+03**	3.0760E+03	3.3438E+03	3.4474E+03	4.2158E+03	3.0931E+03	3.5571E+03	3.3860E+03
F20	std	**1.7615E+02**	3.3895E+02	1.9556E+02	2.6486E+02	2.2433E+02	2.9198E+02	4.5895E+02	3.9316E+02
F21	avg	**2.4129E+03**	2.6328E+03	2.8245E+03	2.8070E+03	3.2598E+03	2.5183E+03	2.5442E+03	2.5031E+03
F21	std	**2.1885E+01**	2.9014E+01	8.0278E+01	5.4504E+01	1.0720E+02	3.0085E+01	5.3616E+01	5.1355E+01
F22	avg	**6.9727E+03**	1.2027E+04	1.2059E+04	1.2881E+04	1.7107E+04	7.9521E+03	1.3841E+04	9.2476E+03
F22	std	3.1947E+03	4.9081E+03	1.7175E+03	2.7575E+03	**6.2354E+02**	1.0473E+03	4.0267E+03	9.7600E+02
F23	avg	**2.8658E+03**	3.1041E+03	3.5587E+03	3.4067E+03	4.5991E+03	3.0786E+03	3.0657E+03	2.9454E+03
F23	std	**2.8151E+01**	3.0097E+01	1.2881E+02	1.0439E+02	1.7889E+02	5.7292E+01	6.9519E+01	4.3960E+01
F24	avg	**3.0248E+03**	3.2647E+03	3.9631E+03	3.5576E+03	4.9056E+03	3.2072E+03	3.1698E+03	3.1188E+03
F24	std	**2.5304E+01**	3.3466E+01	2.2174E+02	1.3110E+02	2.8188E+02	5.8178E+01	6.2486E+01	5.7556E+01
F25	avg	3.0969E+03	3.1341E+03	4.2607E+03	3.6532E+03	1.5809E+04	3.1440E+03	3.1719E+03	**3.0389E+03**
F25	std	**1.9417E+01**	3.5334E+01	3.2448E+02	1.9452E+02	1.1301E+03	4.1459E+01	3.5044E+01	2.7929E+01
F26	avg	**5.5130E+03**	6.9108E+03	1.2778E+04	9.5812E+03	1.7603E+04	7.6999E+03	9.3462E+03	6.5425E+03
F26	std	**3.4706E+02**	2.0508E+03	8.9691E+02	2.8786E+03	6.8935E+02	7.2307E+02	1.6348E+03	7.8201E+02
F27	avg	**3.4501E+03**	3.4714E+03	4.6157E+03	3.8308E+03	7.2025E+03	3.7810E+03	3.6499E+03	3.4801E+03
F27	std	**5.2600E+01**	7.1750E+01	3.4420E+02	1.9084E+02	7.4432E+02	1.0850E+02	1.3966E+02	8.9664E+01
F28	avg	3.3745E+03	3.4308E+03	5.3087E+03	4.2531E+03	1.3773E+04	3.6043E+03	3.4941E+03	**3.2943E+03**
F28	std	3.2000E+01	4.0142E+01	4.9423E+02	2.7808E+02	1.2021E+03	1.5686E+02	5.5772E+01	**2.9394E+01**
F29	avg	**3.9722E+03**	4.3692E+03	8.8647E+03	6.6341E+03	1.3210E+05	4.9953E+03	4.8597E+03	4.4836E+03

Table 3 Results of CEC2017 (Dim = 50) test function with comparative algorithms (continuation)

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F29	std	**2.4051E+02**	2.9023E+02	1.1844E+03	7.5440E+02	1.4623E+05	3.8969E+02	3.5122E+02	4.0867E+02
F30	avg	**1.1807E+06**	1.8592E+06	3.0673E+08	1.8587E+08	8.7033E+09	6.0216E+06	9.4104E+06	1.2260E+06
F30	std	**1.3306E+05**	6.3689E+05	1.2506E+08	7.2639E+07	2.6295E+09	2.5097E+06	3.0486E+06	3.4585E+05
Average Rank		**1.24**	3.31	6.41	5.93	8.00	3.66	4.79	2.66

Table 4 Results of CEC2020 (Dim = 20) test function with comparative algorithms

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F1	avg	**1.0555E+02**	4.5175E+02	2.3963E+08	4.8492E+07	3.0961E+10	1.0676E+04	3.5683E+05	3.4856E+03
F1	std	**7.8261E+00**	4.2268E+02	2.3990E+08	1.4101E+07	5.6926E+09	1.5145E+04	1.6416E+05	3.2062E+03
F2	avg	**1.8643E+03**	2.2883E+03	3.4357E+03	3.6213E+03	5.6148E+03	2.1001E+03	3.7118E+03	2.4624E+03
F2	std	**3.1303E+02**	3.1397E+02	4.7993E+02	5.0807E+02	3.3678E+02	3.3041E+02	1.2366E+03	5.1994E+02
F3	avg	**7.4230E+02**	7.5233E+02	9.0147E+02	8.8544E+02	1.0223E+03	7.9258E+02	7.8528E+02	7.7624E+02
F3	std	**6.0062E+00**	8.1120E+00	3.3537E+01	2.7878E+01	3.1825E+01	2.1570E+01	1.7186E+01	2.2499E+01
F4	avg	**1.9024E+03**	1.9054E+03	1.9719E+03	1.9141E+03	6.4924E+05	1.9061E+03	1.9101E+03	1.9040E+03
F4	std	**5.9195E–01**	9.2993E–01	5.0772E+01	2.6872E+00	5.7657E+05	2.2559E+00	2.2581E+00	1.7337E+00
F5	avg	**8.2586E+03**	9.4760E+03	6.5473E+05	9.9451E+04	4.4243E+06	1.8012E+05	2.3714E+05	6.5579E+04
F5	std	**5.0671E+03**	9.8874E+03	5.6474E+05	6.8300E+04	2.6460E+06	1.9063E+05	1.6818E+05	5.6801E+04
F6	avg	**1.6062E+03**	1.6119E+03	2.4607E+03	2.2108E+03	3.3089E+03	1.8711E+03	1.8719E+03	1.9388E+03
F6	std	**2.1602E+01**	2.9974E+01	2.7831E+02	2.6682E+02	2.4641E+02	1.8768E+02	1.2804E+02	1.9817E+02
F7	avg	4.0687E+03	**2.8742E+03**	7.2831E+04	2.2327E+04	2.1728E+06	1.0712E+05	9.8134E+04	4.1889E+04
F7	std	2.4473E+03	**3.6797E+02**	5.9614E+04	2.4647E+04	1.8542E+06	1.1192E+05	8.7896E+04	4.0377E+04
F8	avg	2.3000E+03	**2.3000E+03**	2.3688E+03	2.3256E+03	5.9858E+03	2.5525E+03	2.3073E+03	3.1324E+03
F8	std	2.0554E–01	**4.5776E–05**	2.0582E+02	4.6990E+00	6.2235E+02	6.5784E+02	1.9345E+00	1.2339E+03
F9	avg	**2.8228E+03**	2.8477E+03	3.0484E+03	2.9161E+03	3.3595E+03	2.8814E+03	2.8463E+03	2.8479E+03
F9	std	**1.1113E+01**	1.9975E+01	9.0986E+01	3.0135E+01	1.4780E+02	2.3954E+01	1.7558E+01	1.9992E+01
F10	avg	2.9640E+03	2.9497E+03	3.0338E+03	2.9977E+03	5.5025E+03	2.9361E+03	2.9652E+03	**2.9282E+03**
F10	std	3.0984E+01	3.3482E+01	4.4680E+01	**2.4394E+01**	8.3061E+02	2.6544E+01	3.2909E+01	2.6170E+01
Average Rank		**1.50**	2.20	6.40	5.30	8.00	4.30	4.70	3.60

Table 5 Results of CEC2022 (Dim = 20) test function with comparative algorithms

Fun	Criterion	Ours	CPO	HO	GKSO	COA	SO	MGO	EAPSO
F1	avg	**3.0758E+02**	3.0849E+02	4.3235E+03	2.9203E+04	7.6956E+03	4.8216E+04	1.1182E+04	7.2333E+03
F1	std	**8.9965E+00**	1.4102E+01	2.0201E+03	9.7788E+03	3.1089E+03	1.1914E+04	4.0649E+03	3.0594E+03
F2	avg	4.5279E+02	4.5326E+02	4.5608E+02	5.3702E+02	4.8155E+02	2.8136E+03	**4.4983E+02**	4.5848E+02
F2	std	**8.5698E+00**	9.5212E+00	1.1906E+01	6.0822E+01	2.7555E+01	7.9663E+02	1.5505E+01	1.6425E+01
F3	avg	**6.0000E+02**	6.0000E+02	6.0000E+02	6.5724E+02	6.3692E+02	6.8042E+02	6.0617E+02	6.0451E+02
F3	std	2.5606E–03	3.2907E–03	**4.4224E–04**	9.2576E+00	1.0509E+01	9.4502E+00	5.4151E+00	3.8189E+00
F4	avg	8.2842E+02	**8.2806E+02**	8.6257E+02	8.8134E+02	8.8748E+02	9.7987E+02	8.3680E+02	8.4289E+02
F4	std	1.0564E+01	**8.3832E+00**	1.2670E+01	9.5830E+00	2.0397E+01	1.6377E+01	9.5664E+00	1.4553E+01
F5	avg	**9.0037E+02**	9.0056E+02	9.0047E+02	2.3876E+03	1.8674E+03	3.5198E+03	1.2952E+03	1.2447E+03
F5	std	**3.9210E–01**	8.5997E–01	8.0760E–01	3.5243E+02	6.2935E+02	4.6304E+02	2.5526E+02	3.5208E+02
F6	avg	**2.8582E+03**	3.3672E+03	3.2321E+03	1.3118E+04	1.9422E+05	2.1795E+09	7.2021E+03	5.6714E+03
F6	std	**1.1517E+03**	1.5077E+03	1.8181E+03	7.7338E+03	1.1393E+05	1.1157E+09	4.7545E+03	3.8923E+03
F7	avg	**2.0338E+03**	2.0352E+03	2.0410E+03	2.1550E+03	2.1065E+03	2.2143E+03	2.0875E+03	2.0671E+03
F7	std	**6.9373E+00**	9.1341E+00	7.7257E+00	2.5289E+01	3.0776E+01	4.9823E+01	4.9859E+01	2.9300E+01
F8	avg	**2.2224E+03**	2.2225E+03	2.2268E+03	2.2415E+03	2.2625E+03	2.4813E+03	2.2377E+03	2.2322E+03
F8	std	**1.0704E+00**	1.9454E+00	1.4314E+00	7.4333E+00	4.6702E+01	1.7143E+02	2.3861E+01	7.6642E+00
F9	avg	2.4808E+03	**2.4808E+03**	2.4808E+03	2.5784E+03	2.5077E+03	3.5005E+03	2.4808E+03	2.4817E+03
F9	std	5.0580E–04	**3.6273E–04**	2.2365E–03	5.5337E+01	1.7232E+01	3.9484E+02	3.5618E–02	5.6699E–01
F10	avg	2.5160E+03	**2.5096E+03**	2.5147E+03	4.2795E+03	2.6370E+03	6.1981E+03	3.0637E+03	3.4222E+03
F10	std	4.3657E+01	**3.4419E+01**	4.3145E+01	1.1389E+03	4.9560E+02	1.4480E+03	4.3178E+02	1.4511E+03
F11	avg	**2.9000E+03**	2.9000E+03	2.9100E+03	3.1435E+03	3.4939E+03	8.8204E+03	2.9122E+03	2.9294E+03
F11	std	3.1181E–07	**2.5442E–07**	3.0513E+01	1.9182E+02	1.2395E+02	8.2704E+02	1.1111E+01	2.7667E+01
F12	avg	**2.9516E+03**	2.9549E+03	2.9545E+03	3.0968E+03	2.9803E+03	3.6946E+03	3.0044E+03	2.9765E+03
F12	std	**7.1115E+00**	7.3826E+00	9.0303E+00	9.3290E+01	2.4712E+01	2.5104E+02	2.5187E+01	2.5152E+01
Average Rank		**1.50**	2.17	2.75	6.67	6.00	8.00	4.50	4.42

4.3 Performance Analysis
4.3.1 Convergence Analysis

Fig. 3 compares the convergence performance of the TFGOA algorithm with seven algorithms on various test functions from CEC2017 (F20–F24), CEC2020 (F1-F5), and CEC2022 (F3–F7). TFGOA outperforms the others in convergence on most of these functions. For example, on the F21 function, TFGOA escapes local optima after about 5000 function evaluations, while algorithms such as CPO, HMO, and EAPSO converge prematurely. On the F1 function from CEC2020, though TFGOA's initial convergence is slightly slower, its ability to consistently escape local optima becomes evident with more iterations, eventually surpassing the other algorithms. TFGO has a strong search ability for problems with multiple local optima.

TFGOA achieves these results mainly because of its unique search mechanism. By randomly choosing between optimal-center adaptive search and individual - center adaptive search, the algorithm effectively balances exploration and exploitation. This enables TFGOA to leverage individuals' capabilities, break free from local optima, and converge to more accurate solutions.

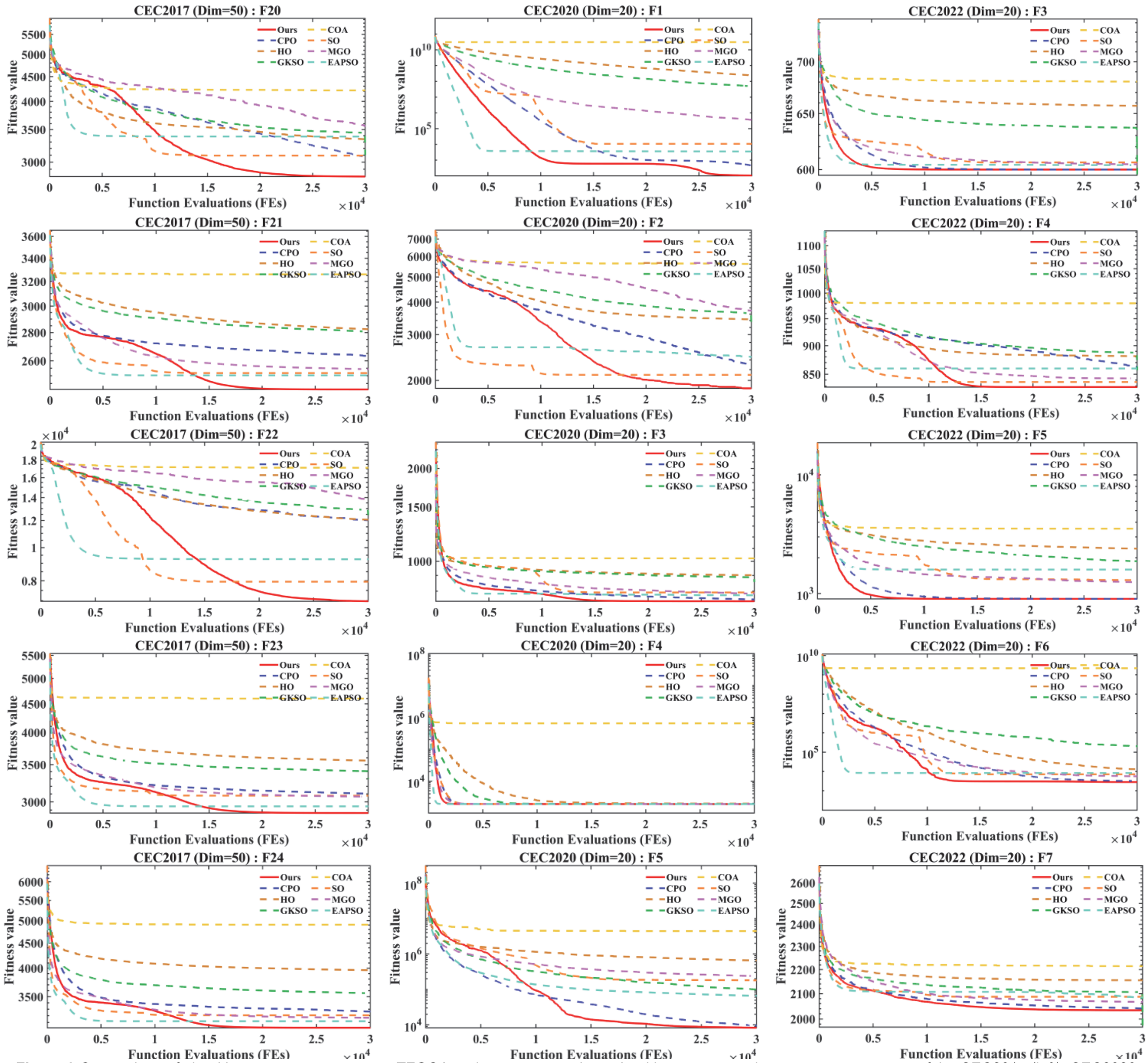


Figure 3 Comparison of algorithm convergence curves. TFGOA and seven comparison algorithms compare the convergence curves of the CEC2017 (left), CEC2020 (middle) and CEC2022 (right) test functions. TFGOA is a solid red line, and other algorithms are dashed lines

4.3.2 Robustness Analysis

Fig. 4 depicts the distribution of optimal fitness values obtained by all algorithms after 30 runs on the CEC2017, CEC2020, and CEC2022 benchmarks, used to evaluate algorithm stability. The narrow width of its box plot indicates that TFGOA's results are more concentrated, with consistently lower optimal fitness values across most test functions. Notably, TFGOA exhibits significantly fewer "+" symbols (outliers) beyond the upper edge compared to other algorithms, further demonstrating its high search precision. For the F22 function in CEC2017, the box plot of TFGOA is slightly wider, yet its median remains lower than those of the other seven algorithms. This indicates that TFGOA achieves better search results despite higher dispersion. For benchmark functions such as F3, F5, and F7 in CEC2022, TFGOA shows a smaller box plot area, which reflects lower dispersion and a lower mean in execution outcomes. These characteristics confirm the robust stability of TFGOA.

4.3.3 Statistical Test Analysis

A Wilcoxon rank-sum test at a significance level of 5% assesses the significant differences between TFGOA and the other comparative algorithms. TFGOA serves as the control algorithm and is paired with each comparative algorithm to determine significant differences. Tab. 6 showcases the Wilcoxon test results for the CEC2017, CEC2020, and CEC2022 test suites, denoted by "+/=/–". A "+" implies that TFGOA outperforms its counterparts, a "–" indicates inferior performance, and "=" means no significant statistical difference.

Table 6 Wilcoxon rank-sum test statistical results.

Benchmark	CPO	HO	GKSO	COA	SO	MGO	EAPSO
CEC2017 (Dim = 30)	18/8/3	29/0/0	29/0/0	29/0/0	28/0/1	28/1/0	21/7/1
CEC2017 (Dim = 50)	24/5/0	29/0/0	29/0/0	29/0/0	26/2/1	29/0/0	17/8/4
CEC2020 (Dim = 20)	6/3/1	10/0/0	10/0/0	10/0/0	9/0/1	9/1/0	9/0/1
CEC2022 (Dim = 20)	8/3/1	12/0/0	12/0/0	12/0/0	12/0/0	12/0/0	8/1/3

From the experimental results in Tab. 6 that the TFGOA algorithm demonstrates significant advantages over the other seven comparison algorithms. Specifically, TFGOA outperforms HO, GKSO, and COA on three test suites. As the dimension increases from 30 to 50 in CEC2017, the number of functions for which TFGOA is superior to CPO rises from 18 to 24, indicating that TFGOA has stronger adaptability and search ability in high-dimensional problems. Overall, TFGOA performs excellently in most test functions.

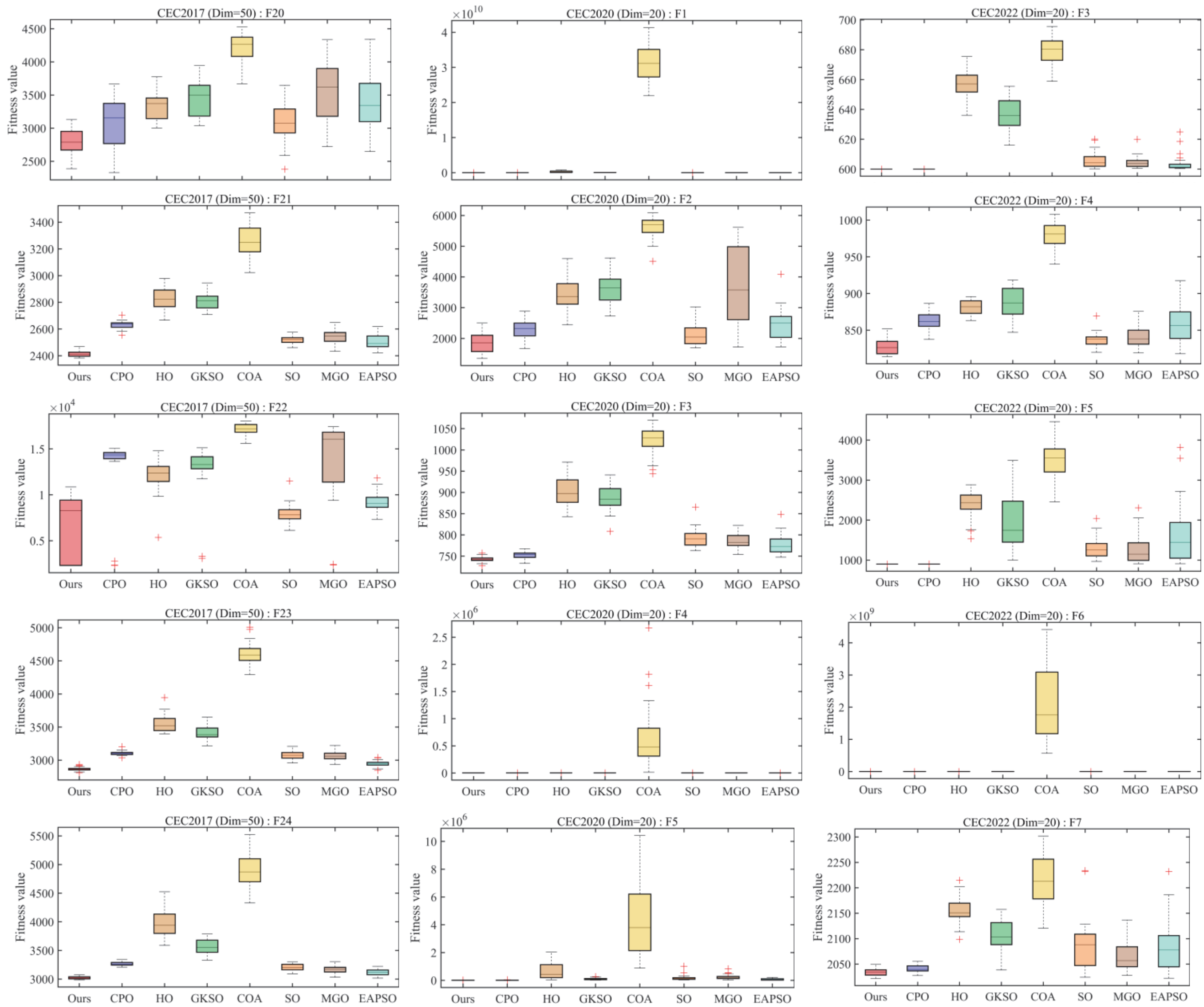


Figure 4 Comparison of algorithm robustness. TFGOA and seven comparison algorithms solve the box plot comparison of the CEC2017 (left), CEC2020 (middle) and CEC2022 (right) test functions. TFGOA is the leftmost red box in the box plot

4.3.4 Computational Time

Fig. 5 shows the average computational time of TFGOA and seven comparison algorithms on CEC2017 and CEC2020 based on the results of 30 independent algorithm runs. As can be seen from the figure, TFGOA algorithm is significantly superior to most of the compared algorithms, and its average computational time ranks third. In CEC2020, TFGOA's computational time outperformed the latest HO algorithm by 25.34 seconds (6.78 vs. 32.12). In the CEC2017 tests, the computational time of the TFGOA algorithm is slightly higher than that of CPO. These results further highlight TFGOA’s superior computational efficiency.

4.4 Ablation Experiment

Search strategy effectiveness. Two TFGOA variants are designed: (1) Our-Optimal, using only the time-varying optimal center adaptation search strategy; (2) Our -Individual, employing only the time-varying individual center adaptive strategy. The CEC2020 benchmark evaluates these variants' performance. Fig. 5 shows that for CEC2020 functions F1, F2, and F3, Our-Individual converges earlier than Our-Optimal and TFGOA. This may be because it tends to search locally, making it more likely to be trapped in local optima and converge prematurely. Our-Optimal performs poorly during the search, probably as all individuals focus on searching near the global optimum, limiting the algorithm's exploration ability and making it hard to find the global optimum. TFGOA, which combines the two search strategies, has a better ability to avoid local optima. In the boxplot analysis in Fig. 7, TFGOA shows better stability than its two variants, indicating it can explore the search space more effectively and achieve better optimization results for complex problems.

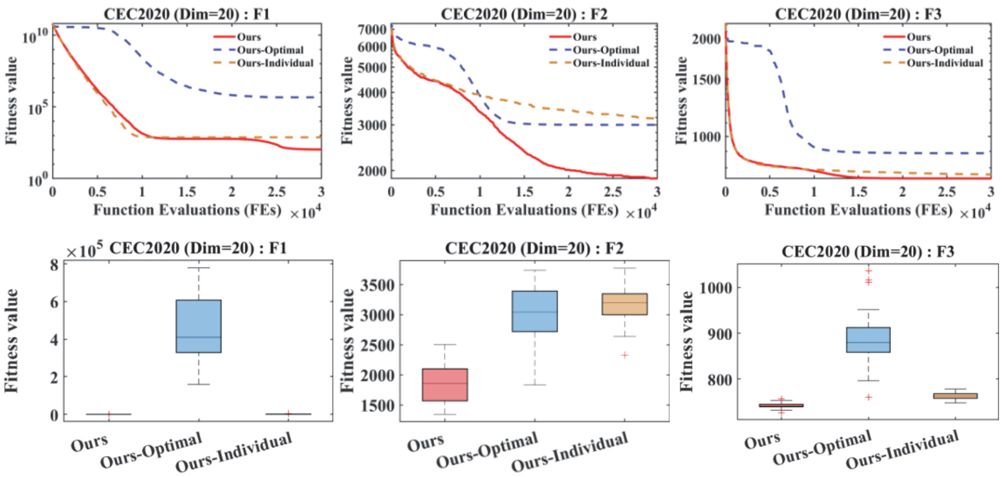
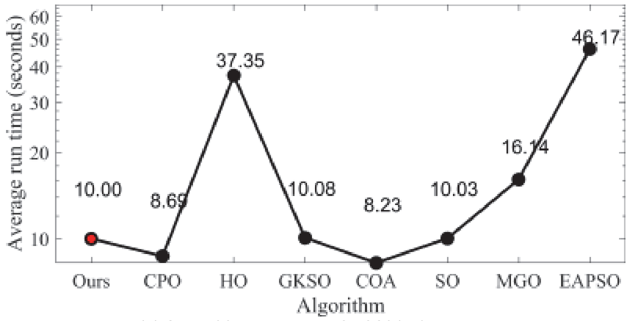
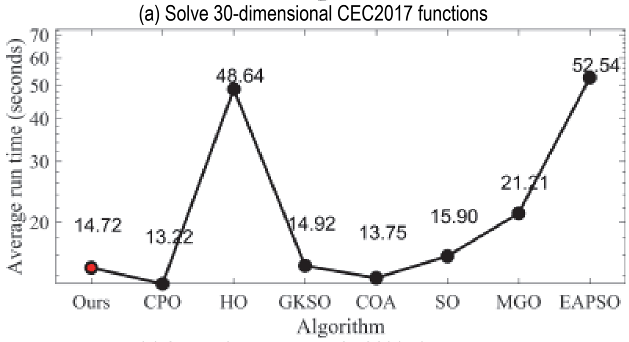


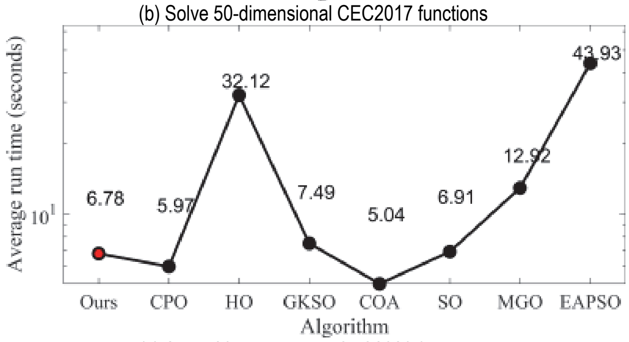
Figure 7 Convergence curves and box plots for TFGOA and the two variants



(a) Solve 30-dimensional CEC2017 functions



(b) Solve 50-dimensional CEC2017 functions



(c) Solve 20-dimensional CEC2020 functions

Figure 5 Comparison of average run time

Grouping factor sensitivity. The grouping factor, mainly a hyperparameter of TFGOA, determines the proportion of elite-followers and self-actualizers in the population. Fig. 6 shows the performance rankings of TFGOA with grouping factor values of 0.3, 0.5, and 0.7. TFGOA achieved the best performance with a grouping factor of 0.5, ranking first in average fitness for functions F1-F7 and F10.

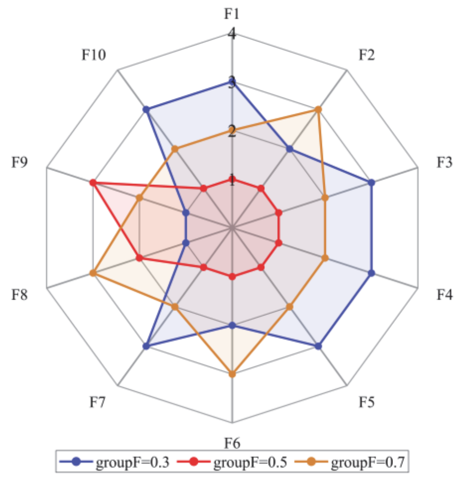


Figure 6 Sensitivity analysis of grouping factor

Time-varying fitness factor necessity. A comparison is made between TFGOA and its variant, TFGOA-Random, where Eq. (1) is replaced by a random number. Experiments are conducted on the CEC2020 test suite, and the average fitness values are presented in Tab. 7. For the F5 function, TFGOA exhibits significantly better performance than TFGOA-Random, with average fitness values of 8.2586E+03 and 1.2583E+04, respectively. For the F4 and F8 functions, the differences in average fitness between TFGOA and TFGOA-Random are minimal (0.6 and 0.3). On F10 function, TFGOA-Random is superior to TFGOA. This may be due to the fact that F10 is composed of five base functions. Compared with the time-varying fitness factor, the randomness of the random factor is stronger, which enhances the exploration of the solution space, and makes TFGOA-Random have an advantage in this function.

Table 7 Results of the necessity of the time-varying fitness factor

Fun	TFGOA	TFGOA-Random
F1	**1.0555E+02**	7.9020E+02
F2	**1.8643E+03**	2.6642E+03
F3	**7.4230E+02**	7.5804E+02
F4	**1.9024E+03**	1.9030E+03
F5	**8.2586E+03**	1.2583E+04
F6	**1.6062E+03**	1.6133E+03
F7	**4.0687E+03**	4.0919E+03
F8	**2.3000E+03**	2.3003E+03
F9	**2.8228E+03**	2.8278E+03
F10	2.9640E+03	**2.9388E+03**

5 VERIFICATION ON ENGINEERING PROBLEMS

To further verify the effectiveness of TFGOA, we employed three benchmark design problems that have been widely utilized in top-tier journals, such as Knowledge-Based Systems [23, 28] and Expert Systems with Applications [21, 29]. Specifically, these problems include the welded beam design problem, the gear train design problem, and the three-bar truss design problem. In addressing these engineering design problems, TFGOA demonstrates superior solution accuracy compared to other comparison algorithms.

5.1 Welded Beam Design Problem

The primary objective of this problem is to identify the optimal solution for minimizing the manufacturing cost of a welded beam design [30]. This optimization involves four crucial design variables: weld thickness (x_1), clamped bar length (x_2), bar height (x_3), and bar thickness (x_4). Ensuring cost optimization is crucial, but it must also be balanced with compliance to four constraint conditions. The complete structure of the welded beam is shown in Fig. 8, and its formula is shown in Eq. (10).

$$
\begin{aligned}
&\text{Minimize} \quad f(X) = 1.10471 x_1^2 x_2 \\
&\qquad\qquad +0.04811 x_3 x_4 \left(14.0 + x_2\right), \\
&\text{Subject to} \quad g_1(X) = \tau(X) - \tau_{\max} \le 0, \\
&\qquad g_2(X) = \sigma(X) - \sigma_{\max} \le 0, \\
&\qquad g_3(X) = \delta(X) - \delta_{\max} \le 0, \\
&\qquad g_4(X) = x_1 - x_4 \le 0, \\
&\qquad g_5(X) = P - Pc(X) \le 0, \\
&\qquad g_6(X) = 0.125 - x_1 \le 0, \\
&\qquad g_7(X) = 1.10471 x_1^2 - 5.0 \\
&\qquad\quad +0.04811 x_3 x_4 \left(14.0 + x_2\right) \\
&\qquad\quad \le 0 \\
&\text{Variable range} \quad 0 \le x_1, x_4 \le 2, 0 \le x_2, x_3 \le 10. \\
&\quad \text{where} \quad \tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau'' \tfrac{x_2}{2R} + (\tau'')^2}, \\
&\qquad \tau' = \frac{p}{\sqrt{2} x_1 x_2}, \tau'' = \frac{MR}{J}, \\
&\qquad \sigma(X) = \frac{6PL}{x_3^2 x_4}, \delta(X) = \frac{6PL^3}{E x_3^3 x_4}, \\
&\qquad Pc(X) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \\
&\qquad M = P(L + \tfrac{x_2}{2}), R = \sqrt{\tfrac{x_2^2}{4} + (\tfrac{x_1+x_3}{2})^2}, \\
&\qquad J = 2\left\{\sqrt{2} x_1 x_2 \left[\tfrac{x_2^2}{4} + (\tfrac{x_1+x_3}{2})^2\right]\right\}, \\
&\qquad P = 6000 \text{ lb}, L = 14 \text{ in}, \\
&\qquad E = 30 \times 10^6 \text{ psi}, G = 12 \times 10^6 \text{ psi}.
\end{aligned}
\tag{10}
$$

Various algorithms are applied to the welded beam design problem, and experimental results are summarized in Tab.8. Notably, a manufacturing cost function value of 1.6928 is achieved by the TFGOA algorithm, outperforming GKSO, COA, SO, MGO, and EAPSO.

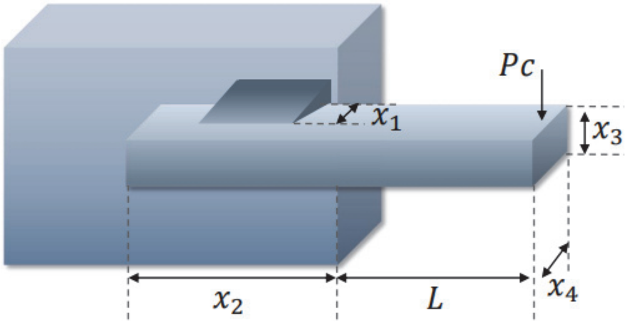


Figure 8 Welded beam design problem

Table 8 Optimal results on welded beam design problem

Algorithms	x_1	x_2	x_3	x_4	f_{best}
CPO [23]	0.2057	3.2349	9.0366	0.2057	1.6928
HO [24]	0.2143	3.1478	8.8522	0.2144	1.6929
GKSO [25]	0.2049	3.2948	9.0247	0.2065	1.7255
COA [26]	0.1818	5.2958	8.1966	0.2501	1.7031
SO [11]	0.9646	2.0000	2.0000	2.0000	2.0965
MGO [27]	0.2053	3.2449	9.0380	0.2057	1.6938
EAPSO [12]	0.2057	3.2349	9.0366	0.2057	1.6938
Ours	0.2057	3.2349	9.0366	0.2057	1.6928

5.2 Gear Train Design Problem

The gear train design problem aims to minimize overall cost by optimizing the number of teeth on the gears [31]. Four critical discrete integer variables, x_1, x_2, x_3 and x_4, correspond to the number of teeth on four different gears. The selection of these variables directly impacts both the gear ratio and the associated manufacturing cost. The complete structure of the gear train is shown in Fig. 9. The mathematical expression is described as follows:

$$
\begin{aligned}
&\text{Minimize} \quad f(X) = \left(\frac{1}{6.931} - \frac{x_2 x_3}{x_1 x_4}\right), \\
&\text{Variable range} \quad x_1, x_2, x_1, x_2 \in \{12, 13, \cdots, 60\}.
\end{aligned}
\tag{11}
$$

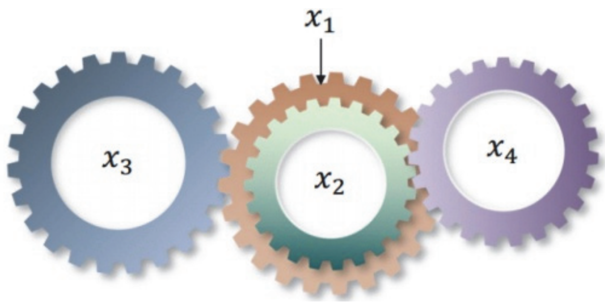


Figure 9 Gear train design problem

The optimization results are summarized in Tab. 9. Through comparative analysis, it is evident that TFGOA achieves the best value of 2.7009E–12 compared to its competing algorithms. This demonstrates that TFGOA exhibits excellent search capabilities when dealing with complex optimization problems involving multiple discrete integer design variables.

Table 9 Optimal results on gear train design problem

Algorithms	x_1	x_2	x_3	x_4	f_{best}
CPO [23]	43	16	19	49	2.7009E–12
HO [24]	49	16	19	43	2.7009E–12
GKSO [25]	49	19	16	43	2.7009E–12
COA [26]	51	30	13	53	2.7009E–12
SO [11]	43	16	19	49	2.3078E–11
MGO [27]	49	19	16	43	2.7009E–12
EAPSO [12]	43	19	16	49	2.7009E–12
Ours	43	19	16	49	2.7009E–12

5.3 Three-Bar Truss Design Problem

The core objective of the three-bar truss design problem is to minimize the total weight of the truss [32]. This is achieved by determining the cross-sectional areas of the truss bars, which involve two key design variables, x_1 and x_2. The design process must adhere to three crucial constraint conditions to ensure the structural safety and stability of the truss. Fig. 10 shows the three-bar truss and its parameters. It is mathematically stated as follows:

$$\begin{aligned}
&\text{Minimize} \quad f(X) = \left(2\sqrt{2}x_1 + x_2\right) \times L,\\
&\text{Subject to} \quad g_1(X) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} \times P - \sigma \le 0,\\
&\qquad g_2(X) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} \times P - \sigma \le 0,\\
&\qquad g_3(X) = \frac{1}{\sqrt{2}x_2 + 2x_1} \times P - \sigma \le 0,\\
&\text{Variable range} \quad 0 \le x_1, x_2 \le 1.\\
&\qquad \text{where} \quad L = 100 \text{ cm}, P = \frac{2\text{KN}}{\text{cm}^2}, \sigma = \frac{2\text{KN}}{\text{cm}^2}.
\end{aligned} \tag{12}$$

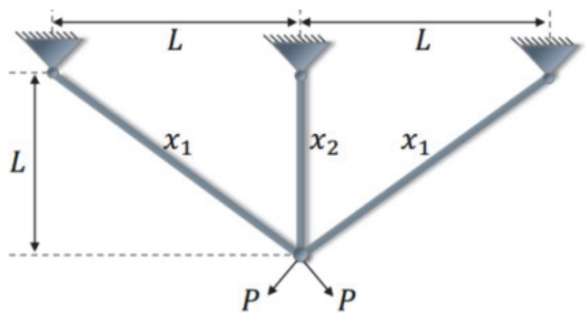


Figure 10 Three-bar truss design problem

According to the results summarized in Tab. 10, TFGOA demonstrates superior performance compared to seven other optimization algorithms in solving the three-bar truss design problem. TFGOA achieves higher solution accuracy, further validating its effectiveness in practical engineering optimization problems.

Table 10 Optimal results on three-bar truss design problem

Algorithms	x_1	x_2	f_{best}
CPO [23]	0.7887	0.4082	263.8958
HO [24]	0.7888	0.4078	263.8958
GKSO [25]	0.7893	0.4063	263.8959
COA [26]	0.7900	0.4047	263.8964
SO [11]	0.7887	0.4081	263.9060
MGO [27]	0.7892	0.4069	263.8958
EAPSO [12]	0.7889	0.4075	263.8960
Ours	0.7887	0.4082	263.8957

6 CONCLUSIONS

This paper proposes a time-varying fitness guided optimization algorithm (TFGOA). It monitors individual fitness changes over time to dynamically adjust search step sizes during iterations. TFGOA uses an individual time-varying fitness factor to quantify these changes and integrates it into the search strategy for intelligent step - size adjustment. The algorithm introduces a time-varying adaptive search mechanism to explore neighborhoods of optimal or current solutions, effectively searching the solution space. Comprehensive experiments on benchmark functions from CEC2017, CEC2020, and CEC2022 show that TFGOA significantly outperforms seven other state-of-the-art algorithms in solution accuracy. The results confirm that tracking the fitness trend over time helps precisely identify potential optimal solutions. In engineering design problems, TFGOA also shows better optimization performance than other algorithms.

In future studies, we will further improve TFGOA algorithm to better solve complex optimization problems. Moreover, we have carried out relevant research on combining optimization algorithms with neural networks and reinforcement learning, and will apply TFGOA to solve optimization problems in these fields to obtain more excellent research outcomes.

Acknowledgements

This work is supported by the National Social Science Foundation of China (24BTJ041).

7 REFERENCES

[1] Zhou, T., Han, X., Wang, L., Gan, W., Chu, Y., & Gao, M. (2023). A multiobjective differential evolution algorithm with subpopulation region solution selection for global and local Pareto optimal sets. *Swarm and Evolutionary Computation*, *83*, 101423. https://doi.org/10.1016/j.swevo.2023.101423

[2] Zhu, D., Wang, S., Zhou, C., & Yan, S. (2023). Manta ray foraging optimization based on mechanics game and progressive learning for multiple optimization problems. *Applied Soft Computing*, *145*, 110561. https://doi.org/10.1016/j.asoc.2023.110561

[3] Luo, T., Sun, J., Zhang, G., Li, Z., & Li, C. (2023). Analysis of influencing factors of green building energy consumption based on genetic algorithm. *Tehnički vjesnik/Technical Gazette*, *30*(5), 1486-1495. https://doi.org/10.17559/TV-20230307121954

[4] Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. *Neurocomputing*, *532*, 183-214. https://doi.org/10.1016/j.neucom.2023.02.073

[5] Liu, L., Fei, T., Zhu, Z., Wu, K., & Zhang, Y. (2023). A survey of evolutionary algorithms. *2023 4th International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, 22-27. https://doi.org/10.1109/ICBAIE57202.2023.00015

[6] Jayashree, S. & Malarvizhi, K. (2023). Spider Monkey Optimization Based Optimal Sizing of Battery Energy Storage for Micro-Grid. *Tehnički vjesnik/Technical Gazette*, *30*(5). https://doi.org/10.17559/TV-20230512102058

[7] Matoušová, I., Trojovský, P., Dehghani, M., Trojovská, E., & Kostra, J. (2023). Mother optimization algorithm: A new human-based metaheuristic approach for solving engineering optimization. *Scientific Reports*, *13*(1), 10312. https://doi.org/10.1038/s41598-023-37464-5

[8] Jia, H., Rao, H., Wen, C., & Mirjalili, S. (2023). Crayfish optimization algorithm. *Artificial Intelligence Review, 56*, 1919-1979. https://doi.org/10.1007/s10462-023-10372-6
[9] Abdel-Basset, M., Mohamed, R., Sallam, K. M., & Chakrabortty, R. K. (2022). Light spectrum optimizer: A novel physics-inspired metaheuristic optimization algorithm. *Mathematics*, *10*(19), 3466. https://doi.org/10.3390/math10193466
[10] Abdel-Basset, M., Mohamed, R., Azeem, S. A. A., Jameel, M., & Abouhawwash, M. (2023). Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. *Knowledge-Based Systems*, *268*, 110454. https://doi.org/10.1016/j.knosys.2023.110454
[11] Hashim, F. A. & Hussien, A. G. (2022). Snake Optimizer: A novel meta-heuristic optimization algorithm. *Knowledge-Based Systems*, *242*, 108320. https://doi.org/10.1016/j.knosys.2022.108320
[12] Zhang, Y. (2023). Elite archives-driven particle swarm optimization for large scale numerical optimization and its engineering applications. *Swarm and Evolutionary Computation*, *76*, 101212. https://doi.org/10.1016/j.swevo.2022.101212
[13] Abdel-Basset, M., Mohamed, R., Jameel, M., & Abouhawwash, M. (2023). Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. *Knowledge-Based Systems*, *262*, 110248. https://doi.org/10.1016/j.knosys.2023.110248
[14] Minh, H. L., Sang-To, T., Theraulaz, G., Wahab, M. A., & Cuong-Le, T. (2023). Termite life cycle optimizer. *Expert Systems with Applications*, *213*, 119211. https://doi.org/10.1016/j.eswa.2022.119211
[15] Zamani, H., Nadimi-Shahraki, M. H., & Gandomi, A. H. (2022). Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization. *Computer Methods in Applied Mechanics and Engineering*, *392*, 114616. https://doi.org/10.1016/j.cma.2022.114616
[16] Geng, Q., Sun, D., & Jia, Y. (2023). Planning of electric vehicle charging facilities on highways based on chaos cat swarm simulated annealing algorithm. *Tehnički vjesnik/Technical Gazette*, *30*(5), 1554-1566. https://doi.org/10.17559/TV-20230407115945
[17] Li, T., Shi, J., Deng, W., & Hu, Z. (2022). Pyramid particle swarm optimization with novel strategies of competition and cooperation. *Applied Soft Computing*, *121*, 108731. https://doi.org/10.1016/j.asoc.2022.108731
[18] Zeng, T., Wang, W., Wang, H., Cui, Z., Wang, F., Wang, Y., & Zhao, J. (2022). Artificial bee colony based on adaptive search strategy and random grouping mechanism. *Expert Systems with Applications*, *192*, 116332. https://doi.org/10.1016/j.eswa.2022.116332
[19] Zhou, T., Hu, Z., Su, Q., & Xiong, W. (2023). A clustering differential evolution algorithm with neighborhood-based dual mutation operator for multimodal multiobjective optimization. *Expert Systems with Applications*, *216*, 119438. https://doi.org/10.1016/j.eswa.2023.119438
[20] Qaraad, M., Amjad, S., Hussein, N. K., Farag, M. A., Mirjalili, S., & Elhosseini, M. A. (2024). Quadratic interpolation and a new local search approach to improve particle swarm optimization: Solar photovoltaic parameter estimation. *Expert Systems with Applications*, *236*, 121417. https://doi.org/10.1016/j.eswa.2023.121417
[21] Zhao, S., Zhang, T., Cai, L., & Yang, R. (2024). Triangulation topology aggregation optimizer: A novel mathematics-based meta-heuristic algorithm for continuous optimization and engineering applications. *Expert Systems with Applications*, *238*, 121744. https://doi.org/10.1016/j.eswa.2023.121744
[22] Taheri, A., RahimiZadeh, K., Beheshti, A., Baumbach, J., Rao, R. V., Mirjalili, S., & Gandomi, A. H. (2024). Partial reinforcement optimizer: An evolutionary optimization algorithm. *Expert Systems with Applications*, *238*, 122070. https://doi.org/10.1016/j.eswa.2023.122070
[23] Abdel-Basset, M., Mohamed, R., & Abouhawwash, M. (2024). Crested porcupine optimizer: A new nature-inspired metaheuristic. *Knowledge-Based Systems*, *284*, 111257. https://doi.org/10.1016/j.knosys.2024.111257
[24] Amiri, M. H., Mehrabi Hashjin, N., Montazeri, M., Mirjalili, S., & Khodadadi, N. (2024). Hippopotamus optimization algorithm: A novel nature-inspired optimization algorithm. *Scientific Reports*, *14*(1), 5032. https://doi.org/10.1038/s41598-024-41987-6
[25] Hu, G., Guo, Y., Wei, G., & Abualigah, L. (2023). Genghis Khan shark optimizer: A novel nature-inspired algorithm for engineering optimization. *Advanced Engineering Informatics*, *58*, 102210. https://doi.org/10.1016/j.aei.2023.102210
[26] Dehghani, M., Montazeri, Z., Trojovská, E., & Trojovský, P. (2023). Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. *Knowledge-Based Systems*, *259*, 110011. https://doi.org/10.1016/j.knosys.2023.110011
[27] Abdollahzadeh, B., Gharehchopogh, F. S., Khodadadi, N., & Mirjalili, S. (2022). Mountain gazelle optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. *Advances in Engineering Software*, *174*, 103282. https://doi.org/10.1016/j.advengsoft.2022.103282
[28] Han, M., Du, Z., Yuen, K. F., Zhu, H., Li, Y., & Yuan, Q. (2024). Walrus optimizer: A novel nature-inspired metaheuristic algorithm. *Expert Systems with Applications*, *239*, 122413. https://doi.org/10.1016/j.eswa.2024.122413
[29] Zhang, H., San, H., Sun, H., Ding, L., & Wu, X. (2024). A novel optimization method: Wave search algorithm. *The Journal of Supercomputing*, 1-36. https://doi.org/10.1007/s11227-024-05591-9
[30] Ghasemi, M., Golalipour, K., Zare, M., Mirjalili, S., Trojovský, P., Abualigah, L., & Hemmati, R. (2024). Flood algorithm (FLA): An efficient inspired meta-heuristic for engineering optimization. *The Journal of Supercomputing*, *80*(15), 22913-23017. https://doi.org/10.1007/s11227-024-05632-3
[31] Zhang, W., Zhao, J., Liu, H., & Tu, L. (2024). Cleaner fish optimization algorithm: A new bio-inspired meta-heuristic optimization algorithm. *The Journal of Supercomputing*, 1-39. https://doi.org/10.1007/s11227-024-05620-7

Contact information:

Yali CHU, Dr.
Changchun University of Technology, School of Mathematics and Statistics,
Changchun 130000, China
E-mail: yalilichu@gmail.com

Xuming HAN, PhD, Full Professor
(Corresponding author)
Jinan University, College of Information Science and Technology,
Guangzhou 510632, China
E-mail: hanxuming@jnu.edu.cn

Zhiquan LIU, PhD, Full Professor
Jinan University, College of Cyber Security,
Guangzhou 510632, China
E-mail: zqliu@vip.qq.com

Ting ZHOU, Dr.
Jinan University, College of Information Science and Technology,
Guangzhou 510632, China
E-mail: tingzhou1208@163.com