International Journal of Information Technology and Applications
Volume 1, Issue 4, December 2024 ISSN 3007-7397 © P-Learn Press

Hybrid Strategy-Improved Dung Beetle Optimization Algorithm

Tien-Wen Sung, Binbin Wu*, Yuzhen Chen

College of Computer Science and Mathematics, Fujian University of Technology, China

tienwen.sung@gmail.com, bynnjuly.wu@gmail.com, chenyuzhen1177@gmail.com

Abstract. To address the issues of easy entrapment in local optima and insufficient convergence precision in the Dung Beetle Optimization (DBO) algorithm, an improved DBO algorithm with a hybrid strategy (SADBO) is proposed. Initially, the tent chaotic mapping strategy is used to initialize the population, making the initial positions of the dung beetles more evenly distributed and enhancing population diversity. Secondly, the Sine Algorithm (SA) is introduced to improve global exploration capabilities. Finally, adaptive t-distribution is applied to perturb individuals, which assists the algorithm in evading localized optimum. Comparative experiments with four other algorithms and the original DBO algorithm demonstrate that SADBO surpasses them on the basis of accuracy and convergence speed on multiple benchmark test functions, proving the proposed algorithm's superiority.

Keywords: DBO algorithm, Sine algorithm, Adaptive t-distribution.

1. Introduction

Optimization involves the process of adjusting and refining solutions to specific problems to achieve optimal or near-optimal outcomes. Utilizing optimization techniques allows for more rational and efficient use of resources in a sustainable manner. Traditional optimization algorithms typically employ specific mathematical models or solution methods to identify optimal solutions based on the problem at hand. However, these algorithms often have limitations, such as dependence on gradient information and difficulty in handling nonlinear complex problems [1]. Consequently, heuristic algorithms have emerged and are widely applied across various fields. Heuristic algorithms search for optimal solutions within the problem space by simulating natural evolution [2], physical phenomena [3], human behaviors [4], and more. They are frequently used in practical domains, such as fault diagnosis [5], path planning [6], wireless sensor networks [7], and various military-industrial applications [8], effectively addressing problems that traditional algorithms cannot solve in real-time execution.

In recent years, researchers have developed numerous population-based intelligence optimization algorithms, such as Differential Evolution (DE) [9], Particle Swarm Optimization (PSO) [10], Genetic Algorithm (GA) [11], Whale Optimization Algorithm (WOA) [12], Butterfly Optimization Algorithm (BOA) [13], and Sparrow Search Algorithm (SSA) [14]. Compared with traditional optimization techniques, these algorithms demonstrate superior performance in practical ap- plications. They offer high stability, easy

implementation, and effective solutions for complex optimization problems [15]. Despite the large variety of existing optimization algorithms, some cases still exist where these algorithms fail to adequately address the expected problems or produce counter-intuitive errors, highlighting the need for further algorithm development [16].

Due to the limitations of swarm intelligence algorithms, researchers have also proposed many improvement strategies to enhance algorithm performance. For example, Fu et al. [17] combined the Chicken Swarm Optimization (CSO) algorithm with the Sparrow Search Algorithm (SSA), ensuring population diversity while improving search efficiency. Jia et al. [18] proposed an Improved Dwarf Mongoose Optimization (IDMO) algorithm that incorporates reverse learning for lens imaging and elite pool strategies during the foraging process of the alpha group, improving convergence precision. Inspired by the unidimensional update strategy of the Artificial Bee Colony (ABC) algorithm, Sung et al. [19] proposed a novel Differential Evolution (DE) algorithm. This algorithm combines ranking behavior and adaptive dimensional strategies to achieve an appropriate equilibrium of exploitation and exploration.

Dung Beetle Optimizer (DBO), introduced by Xue et al. in 2023 [20]. The algorithm simulates the division of labor among dung beetle populations and divides the population into ball-rolling, spawning, foraging, and stealing dung beetles. This algorithm achieves high precision and rapid convergence by maintaining a delicate equilibrium between global search and local exploitation. However, the DBO algorithm also has issues such as a propensity to become ensnared in local optima and an absence of population diversity. Therefore, this paper pro- poses a hybrid multi-strategy improved DBO algorithm named SADBO. First, the tent chaotic mapping method is employed to initialize the population, increasing diversity, expanding the solution space search range, and enhancing global optimization ability. Then, the Sine Algorithm (SA) is introduced to im- part the local exploitation and global exploration strengths of SA to the dung beetles. Finally, the algorithm's convergence accuracy is enhanced through an adaptive t-distribution perturbation, which helps in avoiding local optima.

2. Dung Beetle Optimizer (DBO)

2.1. Ball-Rolling Dung Beetles

Dung beetles use light to guide their navigation and ensure they move in a straight line when rolling dung to create dung balls in the natural environment. However, dung beetles occasionally encounter impediments. The probability of encountering an impediment is given by Eq. (1). The position update when rolling the dung ball without encountering an impediment is described by Eq. (2). The dung beetle reorients itself when it encounters an impediment by dancing over the dung ball to find a new rolling path, and it updates its position as indicated by Eq. (3).

$$a = \begin{cases} 1, & \eta > \lambda \\ -1, & \eta \leq \lambda \end{cases} \tag{1}$$

$$X_i^{iter+1} = X_i^{iter} + \alpha \times k \times X_i^{iter-1} + b \times \Delta X \tag{2}$$

$$X_i^{iter+1} = X_i^{iter} + \tan\theta \cdot \left| X_i^{iter} - X_i^{iter-1} \right| \tag{3}$$

The probability λ of the dung beetle encountering an impediment is denoted by Eq. (1), where η is a random integer within the range of 0 to 1, and $\lambda \in (0,1)$. In Eq. (2), k is the coefficient of deviation, with $k \in (0,0.02]$, b is a constant, with $b \in (0,1)$, and ΔX denotes the intensity of the light, where $\Delta X = X_i - X^w$, and X^w constitutes the most detrimental position within the population. In Eq. (3), $\theta \in [0,\pi]$ denotes the deviation coefficient, and when $\theta \in \{0,\pi/2,\pi\}$, the position is not updated, and tan θ is meaningless.

2.2. Spawning Dung Beetles (Brood Ball)

In order to guarantee an atmosphere of safety for female dung beetles to deposit their eggs and safeguard their offspring, they transport the dung ball to a particular location. The secure area is defined as follows:

$$\begin{cases} Lb^{sa} = \max\left(X^{lb} \times (1-R), Lb\right) \\ Ub^{sa} = \min\left(X^{lb} \times (1-R), Ub\right) \end{cases} \tag{4}$$

In this context, X^{lb} indicates the current local best position, R is calculated as t/T, Lb and Ub denote the lower and upper limits of the feasible region. Lower and upper boundaries of the secure area are denoted by the abbreviations Lb^{sa} and Ub^{sa}, respectively.

The secure area is recognized, and female dung beetles deposit its eggs there, resulting in the construction of a brood ball. The position on the brood ball changes in accordance with Eq. (5). If the brood ball's position exceeds the secure area, it is adjusted based on Eq. (6). In Eq. (5), the dimension of the optimization problem is D, and b_1 and b_2 are $1 \times D$ distinct random vectors.

$$B_i^{iter+1} = X^{lb} + b_1 \times (B_i^{iter} - Lb^{sa}) + b_1 \times (B_i^{iter} - Ub^{sa}) \tag{5}$$

$$B_i = \begin{cases} Lb^{sa}, & B_i < Lb^{sa} \\ Ub^{sa}, & B_i > Ub^{sa} \end{cases} \tag{6}$$

2.3. Foraging Dung Beetles

The brood ball matures through a little dung beetle, which then emerges from the earth to forage. The little dung beetle selects a foraging location that is strategically advantageous, as it prioritizes areas with a high probability of locating sustenance. The optimal foraging range is defined by Eq. (7). Once this optimal range is established, the little dung beetle's foraging position is updated according to Eq. (8).

$$\begin{cases} Lb^{fa} = \max\left(X^{gb} \times (1-R), Ub\right) \\ Ub^{fa} = \min\left(X^{gb} \times (1+R), Ub\right) \end{cases} \tag{7}$$

$$X_i^{iter+1} = X_i^{iter} + C_1 \times (X_i^{iter} - Lb^{fa}) + C_2 \times (X_i^{iter} - Ub^{fa}) \tag{8}$$

The global optimal position is denoted by X^{gb} in Eq. (7), while the optimal foraging area's lower and upper bounds are determined by Lb^{fa} and Ub^{fa}. In Eq. (8), C_1 is a $1 \times D$ vector of random numbers following a normal distribution, while C_2 is a random vector with a value range of (0, 1).

2.4. Stealing Dung Beetles

Stealing Dung Beetles engage in pilfering dung balls from their peers. They prefer not to steal from random locations but instead target areas near the optimal spot for theft. The position of the Stealing Dung Beetle has

been updated as follows:

$$X_i^{iter+1} = X^{gb} + S \times g \times (\left|X_i^{iter} - X^{lb}\right| + \left|X_i^{iter} - X^{gb}\right|) \tag{9}$$

In Eq. (9), g is a random variable of size $1 \times D$ from a distribution that is normal, while S is an invariant.

3. Proposed Improvement

3.1. Tent Chaotic Map

The initial dung beetle positions in the DBO algorithm are randomly distributed, resulting in an uneven distribution of individuals in the solution space. This irregular distribution reduces the algorithm's accuracy and rapidity of convergence. The paper adds the tent chaos mapping approach [21] to the DBO algorithm in order to overcome this problem. Unlike traditional random initialization methods, the tent chaos mapping generates a more even distribution of individual positions within the population, thereby enhancing population diversity. The tent mapping function is formulated as:

$$x_n^{m+1} = \begin{cases} 2x_n^m, & 0 \le x \le 0.5 \\ 2(1 - x_n^m), & 0.5 \le x \le 1 \end{cases} \tag{10}$$

where m represents the spatial dimension and n is the population size.

3.2. Improvements Using the Sine Algorithm

The Sine Algorithm (SA) [22] is a simplified variant derived from the theoretical framework of the Sine Cosine Algorithm (SCA) [23]. It employs the sine function to execute iterative optimization, demonstrating robust global exploration capabilities. Additionally, SA features a streamlined structure and operates with increased efficiency. The updated location of the Sine Algorithm is shown in Eq. (11).

$$x_i^{iter+1} = x_i^{iter} + r_1 \sin(r_2) \times \left|r_3 p_i^{iter} - x_i^{iter}\right| \tag{11}$$

In Eq. (11), p_i^{iter} represents the best individual position in the *iter*-th iteration. The variable r_1 denotes a nonlinear decreasing function, r_2 and r_3 represent random numbers, ranging from $[0, 2\pi]$ and $[-2, 2]$ respectively. The formula for r_1 is:

$$r_1 = a \times e^{-t/T} \tag{12}$$

To further cultivate the equilibrium between global exploration and local exploitation within the DBO algorithm, this paper integrates the SA guidance mechanism as a substitute for the conventional dung beetle tangent dance. During the rolling phase, the dung beetle's position is updated using a sinusoidal operation. The improved formula is presented below:

$$x_i^{iter+1} = \begin{cases} x_i^{iter} + \alpha \times k \times x_i^{iter-1} + b \times \Delta x, & r < ST \\ x_i^{iter} + r_1 \sin(r_2) \times \left|r_3 p_i^{iter} - x_i^{iter}\right|, & r \ge ST \end{cases} \tag{13}$$

where $ST \in (0.5, 1]$ and $r = rand(1)$. In the improved position update formula, when $r < ST$, it means that the dung beetle has a target to roll, and then a new solution is generated using the first formula in Eq. (13),

whereas when $r \geq ST$, it signifies that the dung beetle lacks a clear target and instead searches using sinusoidal function movements. The incorporation of the SA guidance mechanism significantly mitigates the randomness in the DBO algorithm's position update strategy. With the inclusion of the SA strategy, the current optimal individual p_i^{iter} will exchange information with the dung beetle individuals. This facilitates rapid information dissemination within the population and addresses the original algorithm's deficiency in inter-individual information exchange.

On the other hand, as seen from Eq. (11), r1 controls the search distance of the dung beetles. The linear decreasing strategy of r_1 enables the algorithm to perform extensive searches in the early stages, providing strong global search capabilities. In the latter phases, the algorithm's local exploitation capabilities are enhanced by the relatively smaller r_1.

3.3. Adaptive t-distribution Perturbation

To address the problem of inadequate search capability and sensitivity to local optima to in subsequent iterations of the DBO algorithm, this paper employs an adaptive t-distribution perturbation. The specific position update method is detailed in Eq. (14).

$$X_{new} = X^{gb}(1 + t(iter)) \tag{14}$$

where $t(iter)$ denotes a t-distribution and $iter$ is the number of iterations. Early in the iteration process, the value of $iter$ is small, the t-distribution behaves resemble a Cauchy distribution, which grants the algorithm robust global exploration capabilities. Conversely, as iterations increase, the t-distribution becomes resemble to a Gaussian distribution in the later stages, which helps improve the local exploitation capabilities of the algorithm.

Although the adaptive t-distribution perturbation enhances the algorithm's global search capabilities and helps escape local optima, it cannot guarantee that the new solutions acquired through perturbation is better than the original solution. Thus, the greedy rule is introduced: X^{gb} will be updated only when the new solution of the individual through perturbation is superior to that before perturbation. The greedy rule is as shown in Eq. (15):

$$X^{gb} = \begin{cases} X_{new}, & fit(X_{new}) < fit(X^{gb}) \\ X^{gb}, & \text{otherwise} \end{cases} \tag{15}$$

3.4. SADBO Algorithm Implementation Process

Figure 1 presents the flowchart for the proposed SADBO algorithm.

4. Discussion and Analysis of Experimental Results

4.1. Experiment Preparation

The simulation experiments presented in this paper were performed on a Windows 10 operating system. The hardware configuration includes a 12th generation Intel® Core™ i5-12600KF CPU and 32 GB of RAM (3600 MHz). The programming software used is MATLAB R2022a.

The convergence and equilibrium of the improved DBO algorithm were verified through the comparison of

the proposed SADBO algorithm to the DBO, SSA, PSO, WOA, and GWO algorithms. For fairness in the experiments, a uniform population size of 30 was assigned to each algorithm, with a maximum of 500 iterations. Table 1 displays the parameter settings for the algorithms that are being compared.

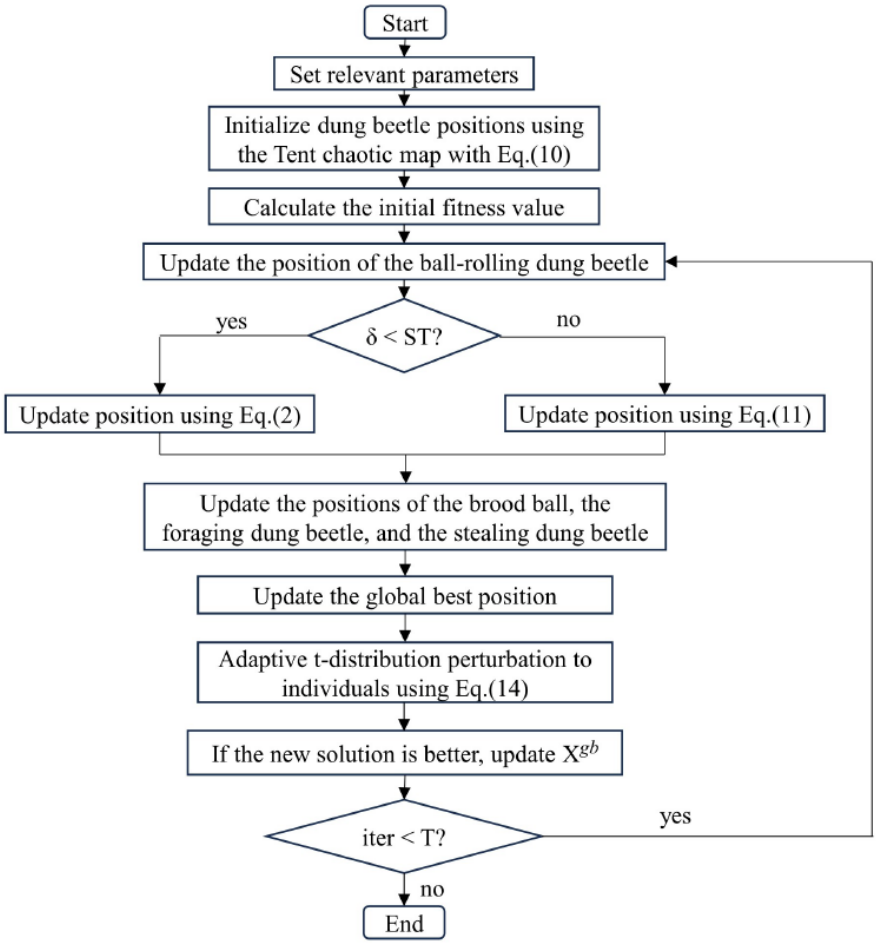

Figure 1. Flowchart of SADBO algorithm.

Table 1. Parameter values of algorithms.

Algorithm	Parameters
SADBO	$a = 1$, $k = 0.1$, $\lambda = 0.1$, $S = 0.5$, $b = 0.3$
DBO	$k = 0.1$, $\lambda = 0.1$, $S = 0.5$, $b = 0.3$
SSA	$PD = 0.2$, $SD = 0.1$, $ST = 0.8$
PSO	$\omega = 0.9 - t \times (0.9 - 0.2)/T$, c_1, $c_2 = 2$
WOA	$a = 2 - 2t/T$
GWO	$a = 2 - 2t/T$

4.2. Test Functions

In this paper, 10 benchmark test functions via distinct characteristics have been selected over comparative experiments in function optimization. Table 2 displays the specific function information.

In the selected benchmark functions, f1 to f6 are unimodal test functions, while the others are multimodal test functions. Unimodal test functions have only one extreme point within the feasible domain and are typically utilized to evaluate the algorithm's convergence capabilities, verifying whether the algorithm can find the global optimum with fewer iterations. Multimodal test functions usually contain multiple local minima, which can easily cause algorithms to become stuck in local optima. They are utilized to evaluate the algorithm's local search capability, determining whether the algorithm can circumvent the current local optimum to determine the global optimum.

Table 2. Benchmark functions.

Function	D	Search space	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^D$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]^D$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30	$[-100, 100]^D$	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \leq i \leq n \}$	30	$[-100, 100]^D$	0
$f_5(x) = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^D$	0
$f_6(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1]$	30	$[-1.28, 1.28]^D$	0
$f_7(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]^D$	-12569.5
$f_8(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]^D$	0
$f_9(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$	30	$[-32, 32]^D$	0
$f_{10}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^D$	0

4.3. Results and Analysis

The five selected comparison algorithms and the proposed SADBO were independently executed on 10 basic test functions for a total of 30 runs. The specific results are presented in Table 3. Fig. 2 presents the average fitness convergence curves of each algorithm.

From the data in the table and the figures, it is clear that SADBO achieves the theoretical optimal value in most test functions. It demonstrates superior accuracy in overall optimization results compared to other algorithms, reflecting the excellent local development performance of SADBO. As shown in Fig. 2, on most multimodal test functions, SADBO not only achieves the highest convergence accuracy but also converges the fastest, typically finding the optimal value within 30 iterations. Thus, its convergence speed is significantly faster than that of other algorithms. In a few test functions (f_5, f_7), although the optimal value was not found during the optimization process, the precision of optimization improved compared to the original DBO. Overall, compared to other algorithms, SADBO achieves a good balance between development capability and exploration capability.

Table 3. Experimental results.

		SADBO	DBO	SSA	PSO	WOA	GWO
	best	0	9.92E-163	0	1.12E-05	1.95E-86	3.85E-29
f_1	mean	0	3.35E-103	3.25E-52	2.37E-04	1.44E-74	1.57E-27
	std	0	1.83E-102	1.77E-51	2.31E-04	3.94E-74	3.10E-27
	best	3.34E-307	4.66E-81	9.70E-142	4.01E-03	8.53E-58	2.35E-17
f_2	mean	4.01E-236	3.10E-57	2.78E-30	4.72E+00	1.79E-50	1.24E-16
	std	0	1.41E-56	1.21E-29	7.29E+00	5.96E-50	9.12E-17
	best	0	1.84E-129	1.61E-136	2.32E+01	1.98E+04	3.50E-09
f_3	mean	0	2.45E-84	1.11E-25	8.74E+01	4.25E+04	2.96E-05
	std	0	1.32E-83	6.04E-25	3.80E+01	1.12E+04	1.21E-04
	best	8.26E-293	1.94E-80	2.03E-302	6.96E-01	2.23E-01	5.06E-08
f_4	mean	3.63E-221	7.19E-52	2.92E-26	1.10E+00	4.39E+01	5.82E-07
	std	0	3.93E-51	1.57E-25	2.58E-01	2.92E+01	5.05E-07
	best	2.50E+01	2.54E+01	1.50E-09	2.18E+01	2.68E+01	2.59E+01
f_5	mean	2.54E+01	2.58E+01	9.84E-06	9.73E+01	2.79E+01	2.71E+01
	std	3.34E-01	2.07E-01	1.71E-05	1.19E+02	4.47E-01	7.50E-01
	best	1.31E-06	2.43E-04	5.81E-05	9.96E-02	3.93E-05	5.79E-04
f_6	mean	1.72E-04	1.20E-03	1.73E-03	5.97E+00	3.17E-03	1.71E-03
	std	1.44E-04	6.80E-04	1.26E-03	7.40E+00	4.22E-03	8.77E-04
	best	-1.23E+04	-1.03E+04	-9.30E+03	-6.88E+03	-1.26E+04	-8.08E+03
f_7	mean	-8.74E+03	-8.84E+03	-8.41E+03	-4.72E+03	-1.03E+04	-6.02E+03
	std	1.04E+03	1.59E+03	6.82E+02	1.23E+03	1.75E+03	9.88E+02
	best	0	0	0	4.89E+01	0	5.68E-14
f_8	mean	0	0	0	1.00E+02	0	3.44E+00
	std	0	0	0	2.50E+01	0	4.38E+00
	best	4.44E-16	4.44E-16	4.44E-16	2.02E-03	4.44E-16	6.79E-14
f_9	mean	4.44E-16	5.63E-16	4.44E-16	2.21E-01	4.35E-15	1.02E-13
	std	0	6.49E-16	0	5.54E-01	2.35E-15	1.67E-14
	best	0	0	0	9.12E-07	0	0
f_{10}	mean	0	3.74E-03	0	6.67E-03	1.16E-02	3.80E-03
	std	0	2.05E-02	0	9.11E-03	4.47E-02	7.53E-03

5. Conclusion

Given that the DBO algorithm is capable of becoming ensnared in local optima and may suffer from insufficient convergence accuracy, this paper presents a hybrid strategy-improved algorithm, SADBO, which substantially improves the algorithm's search performance and achieves a suitable equilibrium among the capacity for exploration and exploitation. The DBO, SSA, PSO, WOA, and GWO algorithms were compared to the results of experiments conducted on 10 benchmark test functions. The results suggest that the SADBO algorithm exhibits a higher degree of convergence accuracy and a faster convergence speed when contrasted with a variety of metaheuristic optimization algorithms. Therefore, the SADBO algorithm offers a new

perspective and methodology for addressing complex optimization problems and is expected to find widespread application in various fields, including task scheduling, path planning, and other combinatorial optimization problems.

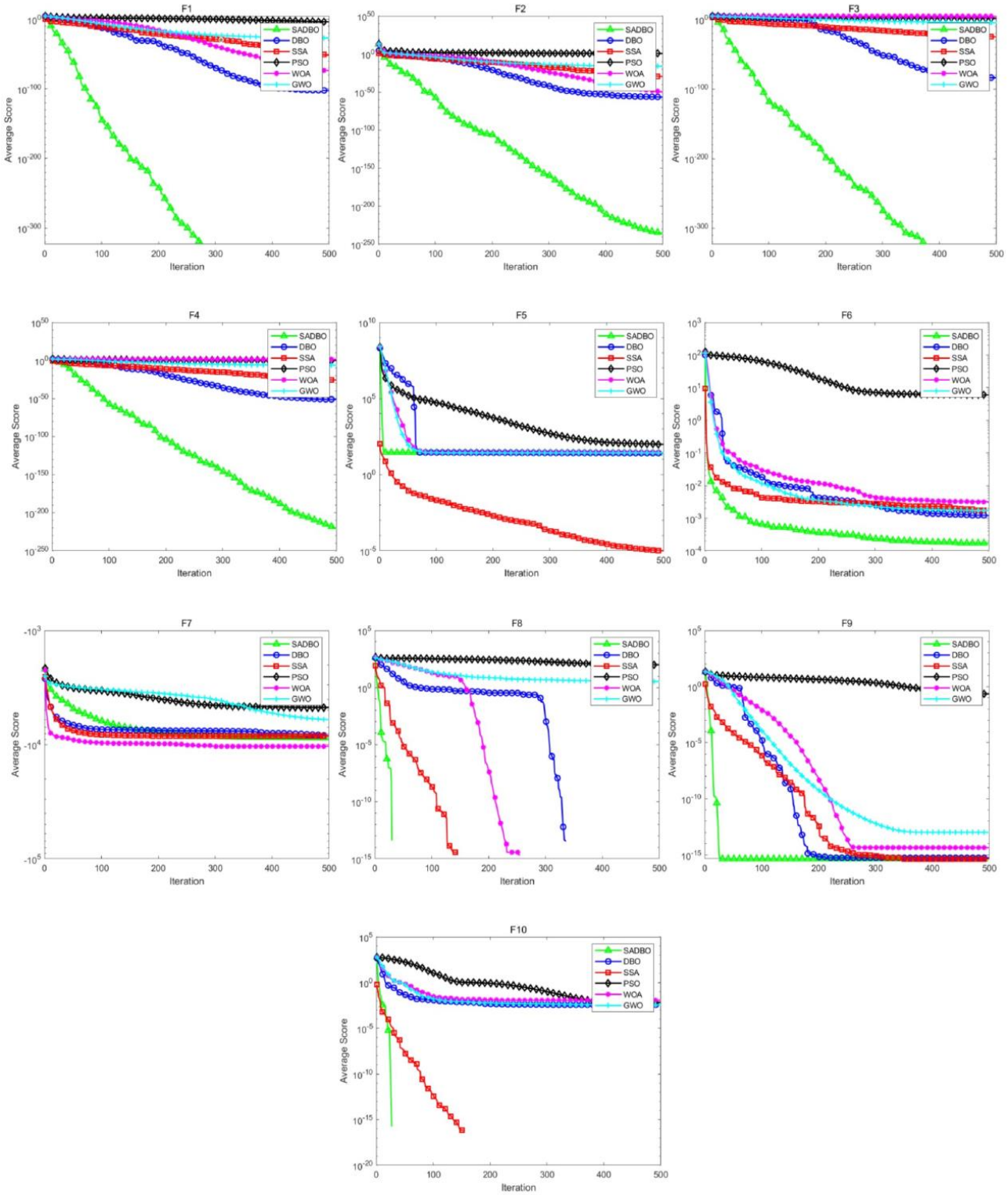

Figure 1. Convergence curves of different algorithms.

Acknowledgements

This work was partially supported by the Fujian University of Technology under Grant GY-Z20016 and GY-Z18183.

References

[1] Yang, X.S.: Nature-inspired optimization algorithms: Challenges and open problems. Journal of Computational Science 46, 101104 (2020).

[2] Holland, J.H.: Genetic algorithms. Scientific American 267(1), 66–73 (1992).

[3] Hashim, F.A., Hussain, K., Houssein, E.H., Mabrouk, M.S., Al-Atabany, W.: Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. Applied intelligence 51, 1531–1551 (2021).

[4] Mousavirad, S.J., Ebrahimpour-Komleh, H.: Human mental search: a new population-based metaheuristic optimization algorithm. Applied Intelligence 47, 850–887 (2017).

[5] Qin, Y., Jin, L., Zhang, A., He, B.: Rolling bearing fault diagnosis with adaptive harmonic kurtosis and improved bat algorithm. IEEE Transactions on Instrumentation and Measurement 70, 1–12 (2020).

[6] Karur, K., Sharma, N., Dharmatti, C., Siegel, J.E.: A survey of path planning algorithms for mobile robots. Vehicles 3(3), 448–468 (2021).

[7] Sung, T.W., Tsai, P.W., Gaber, T., Lee, C.Y.: Artificial intelligence of things (AIoT) technologies and applications. Wireless Communications and Mobile Computing 2021 (2021).

[8] Dou, R., Duan, H.: L´evy flight based pigeon-inspired optimization for control parameters optimization in automatic carrier landing system. Aerospace Science and Technology 61, 11–20 (2017).

[9] Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization 11, 341–359 (1997).

[10] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks. vol. 4, pp. 1942–1948. IEEE (1995).

[11] Grefenstette, J.J.: Genetic algorithms and machine learning. In: Proceedings of the sixth annual conference on Computational Learning Theory. pp. 3–4 (1993).

[12] Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in engineering software 95, 51–67 (2016).

[13] Arora, S., Singh, S.: Butterfly optimization algorithm: a novel approach for global optimization. Soft computing 23, 715–734 (2019).

[14] Xue, J., Shen, B.: A novel swarm intelligence optimization approach: sparrow search algorithm. Systems science & control engineering 8(1), 22–34 (2020).

[15] Li, D.: Optimization method of marine logistics distribution path based on improved swarm intelligence optimization algorithm. Ship Science and Technology 42(16), 184–186 (2020).

[16] Onwubolu, G.C., Babu, B.: New optimization techniques in engineering, vol. 141. Springer (2013).

[17] Fu, H., Liu, H.: Improved sparrow search algorithm with multi-strategy integration and its application. Control and Decision 37(1), 87–96 (2022).

[18] JIA, H., CHEN, L., et al.: Optimization algorithm of elite pool dwarf mongoose based on lens imaging reverse learning. Computer Engineering and Applications 59(24), 131–139 (2023).

[19] Sung, T.W., Zhao, B., Zhang, X.: An adaptive dimension differential evolution algorithm based on ranking scheme

for global optimization. PeerJ Computer Science 8, e1007 (2022).

[20] Xue, J., Shen, B.: Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. The Journal of Supercomputing 79(7), 7305–7336 (2023).

[21] Fan, J., Li, Y., Wang, T.: An improved African vultures optimization algorithm based on tent chaotic mapping and time-varying mechanism. Plos one 16(11), e0260725 (2021).

[22] Qu, L., He, D.: Simplified sine cosine algorithm: sine algorithm. Comput Appl Res 35(012), 3694–3696 (2018).

[23] Mirjalili, S.: Sca: a sine cosine algorithm for solving optimization problems. Knowledge-based systems 96, 120–133 (2016).