mathematics

MDPI

Article

Individual Disturbance and Attraction Repulsion Strategy Enhanced Seagull Optimization for Engineering Design

Helong Yu [1], Shimeng Qiao [1], Ali Asghar Heidari [2], Chunguang Bi [1,*] and Huiling Chen [2,*]

[1] College of Information Technology, Jilin Agricultural University, Changchun 130118, China; yuhelong@jlau.edu.cn (H.Y.); qiao_shimeng@163.com (S.Q.)

[2] Department of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China; aliasghar68@gmail.com

* Correspondence: chunguangb@jlau.edu.cn (C.B.); chenhuiling.jlu@gmail.com (H.C.)

Abstract: The seagull optimization algorithm (SOA) is a novel swarm intelligence algorithm proposed in recent years. The algorithm has some defects in the search process. To overcome the problem of poor convergence accuracy and easy to fall into local optimality of seagull optimization algorithm, this paper proposed a new variant SOA based on individual disturbance (ID) and attraction-repulsion (AR) strategy, called IDARSOA, which employed ID to enhance the ability to jump out of local optimum and adopted AR to increase the diversity of population and make the exploration of solution space more efficient. The effectiveness of the IDARSOA has been verified using representative comprehensive benchmark functions and six practical engineering optimization problems. The experimental results show that the proposed IDARSOA has the advantages of better convergence accuracy and a strong optimization ability than the original SOA.

Keywords: seagull optimization algorithm; swarm intelligence; individual disturbance; attraction-repulsion strategy; engineering design

Citation: Yu, H.; Qiao, S.; Heidari, A.A.; Bi, C.; Chen, H. Individual Disturbance and Attraction Repulsion Strategy Enhanced Seagull Optimization for Engineering Design. *Mathematics* **2022**, *10*, 276. https://doi.org/10.3390/math10020276

Academic Editors: Fabio Caraffini and Oliviu Matei

Received: 10 November 2021
Accepted: 10 January 2022
Published: 16 January 2022

1. Introduction

With the emergence of the concept of swarm intelligence in 1989 [1], many scholars have proposed various swarm intelligence optimization algorithms in recent years, which show more efficient and stable effects in solving complex practical problems. Compared with traditional gradient descent algorithms, intelligent algorithms, such as novel whale optimization algorithm (WOA) [2], hunger games search (HGS) [3], colony predation algorithm (CPA) [4], slime mold algorithm (SMA) [5], Runge Kutta optimizer (RUN) [6], Harris hawks optimization (HHO) [7], bat algorithm (BA) [8], teaching-learning-based pathfinder algorithm (TLPFA) [9], wind-driven optimization algorithm(WDO) [10], salp swarm algorithm (SSA) [11,12], grey wolf optimizer (GWO) [13], and its variants I-GWO and Ex-GWO [14], usually have stronger optimization capabilities. These algorithms can effectively solve complex optimization problems and have strong flexibility, robustness, and self-organization. Furthermore, these algorithms have applications in many fields, such as neural network training [15], multi-attribute decision making [16–19], traveling salesman problem [20], object tracking [21,22], image segmentation [23,24], feature selection [25–29], engineering design problems [30–33], scheduling problem [34,35], medical data classification [36–39], bankruptcy prediction [40–42], parameter optimization [43–46], gate resource allocation [47,48], cloud workflow scheduling [49,50], fault diagnosis of rolling bearings [51,52], power electronic circuit design [53,54], detection of foreign fiber in cotton [55,56], and energy vehicle dispatch [57]. However, there are still common problems, such as slow convergence speed, easy to fall into local optimum, and poor convergence accuracy [23,58].

In 2019, Dhiman et al. [59] proposed a seagull optimization algorithm (SOA) based on seagull migration and attack behavior. The author verified the performance of the SOA on

44 well-known benchmark functions and applied SOA to optical buffers, pressure vessels, reducers, welded beams, tension/compression springs, 25 bar truss, and rolling circle problems. The results illustrate the effectiveness and practical value of SOA. However, like other swarm intelligence algorithms, the SOA also has the problems of slow convergence and low solution accuracy. Since the SOA was proposed recently, Lei et al. [60] introduced the Lévy flight strategy and singer function to improve the problem of slow convergence speed, and applied the improved SOA to find the lowest cost problem. To alleviate the problem of early convergence of the SOA, Cao et al. [61] proposed the balanced SOA, which was used to identify the best parameters of the exchange membrane fuel cell (PEMFC) chimney. In 2021, Dhiman et al. [62] introduced the concept of the dynamic archive to the SOA in the multi-objective problem. They then proposed the multi-objective SOA, relying on roulette selection to determine the effective archive solutions, and applied it to the six constraint problems of engineering design. Because SOA has been proposed in recent years, it does not have many variants like other swarm intelligence algorithms, which shows that SOA has a lot of room for improvement. There are not many cases where the SOA is applied to solve practical problems. There are more possibilities in the areas where the SOA can be applied. Since the realization of an SOA that requires fewer parameters and the characteristics of easy implementation, SOA has a larger optimization space and exploration prospects.

The idea of attraction and repulsion appeared in the attraction and repulsion particle swarm optimization (ARPSO) [63]. Through the alternation between the two stages of attraction and repulsion, it can enhance the ability of particle swarm to jump out of the local optimum, improve the diversity of search space, and prevent the problem of premature convergence to a great extent. Since ARPSO has a good ability to jump out of the local optimal solution, it has a strong ability to find the global optimal solution. On this basis, Pant et al. [64] proposed a diversity-guided particle swarm optimizer with three stages: attraction, repulsion, and attraction-repulsion. Mohamed et al. [65] proposed a modified multi-objective imperialist competitive algorithm for the shortcomings of a single-objective empire competition algorithm when used in high-dimensional or complex multimodal function problems. The algorithm introduced the concept of attraction and repulsion in the assimilation stage. It improved the algorithm's performance to achieve a better effect of finding the global optimal solution.

Inspired by predecessors, to solve the problems of poor optimization accuracy and easy to fall into local optimum in SOA, this paper proposes an improved SOA variant, called seagull optimization algorithm, based on individual disturbance and attraction-repulsion strategy (IDARSOA). It is easy to fall into local optimum in the original SOA when looking for the optimal forward direction. By adding the individual disturbance strategy in the process of looking for the forward direction of the seagull population, it can effectively increase the exploration and optimization ability of the algorithm and the ability to jump out of the local optimum. The attraction-repulsion strategy adopted in this paper makes the seagulls migrate in the optimal direction under the interaction of the global optimal seagull individual attraction, and the global worst seagull individual repulsion enhances the diversity and optimization ability of the algorithm population and makes the search solution space more comprehensive in the algorithm exploitation stage. To evaluate the performance of the IDARSOA, this paper uses 10 benchmark functions of IEEE CEC 2019 and 10 functions of IEEE CEC 2020 to effectively verify the effect of IDARSOA. The comparison experiment includes parameters sensitivity analyses, the comparison between the added mechanism and the original algorithm, the comparison with the widely used algorithm, and the comparison with the excellent variant algorithm. According to Wilcoxon signed-rank test and Friedman test, the performance of IDARSOA is better than the original algorithm.

The structure of the paper is as follows, and an overview of the SOA can be found in Section 2. Section 3 introduces the IDARSOA. The experimental results are described and discussed in Section 4. Section 5 applies IDARSOA to engineering problems and analyzes the experimental results. The sixth part includes the conclusion of the full text and summary of future work.

2. Overview of SOA

The SOA is a new meta-heuristic algorithm, first proposed in 2019 [59]. The SOA mainly simulates the critical characteristics of seagulls' social life migration behavior and attack behavior. During the migration of seagulls, the position of each seagull is different to avoid collisions. The entire population always migrates towards the optimal position, guiding the forward position of each seagull. During the migration process, seagulls will attack migratory birds in a spiral motion.

2.1. Population Initialization

Let the size of the population space be $N \times D$, where N represents the number of populations and the number of solutions, and D represents the dimension. The fitness is expressed as $F = [F_1\ F_2\ \ldots\ F_D]^T$, and the position of seagulls is represented as $F = [F_1\ F_2\ \ldots\ F_D]^T, n = 1, 2, \ldots, N$. The upper bound of the search range is $ub = [ub_1\ ub_2\ \ldots\ ub_D]$ and the lower bound is $lb = [lb_1\ lb_2\ \ldots\ lb_D]$. The initialization Equation (1) is shown below:

$$X_{N \times D} = rand(N, D) \times (ub - lb) + lb \tag{1}$$

2.2. Migration Behavior

During the migration process, the seagull moves to another new position through the position calculation equation at the current position while avoiding collisions with other seagulls. At the same time, the accessory variable A is introduced to calculate the new position of the seagull.

$$C_S(t) = A \times X(t) \tag{2}$$

where, $C_S(t)$ represents the new position of seagulls, and the new position of seagulls does not collide with the position of other seagulls. $X(t)$ denotes the initialized seagull position before updating, t represents the number of iterations, A is the seagull motion behavior in the search state, and the value range of A is $[0, f_C]$, and its equation is as follows:

$$A = f_C - \frac{t \times f_C}{Maxiteration} \tag{3}$$

where, $Maxiteration$ is the maximum number of iterations, the value of f_C is 2, and the value of A decreases linearly from 2 to 0.

In the process of migration, seagulls will move towards the optimal position, and the optimal direction expression is:

$$M_S = B \times (X_{best}(t) - X(t)) \tag{4}$$

where $X_{best}(t)$ is the optimal position of seagulls under the current iteration, and B is a randomly generated number that balances global search and local search. The equation is as follows:

$$B = 2 \times A^2 \times r_d \tag{5}$$

where r_d is a random number between [0, 1]. The seagull flies in the optimal direction to migrate to a better position. The updated position expression is as follows:

$$D_S(t) = |C_S(t) + M_S(t)| \tag{6}$$

2.3. Attack Behavior

When seagulls are migrating, they rely on their own wings and their own weight to maintain the corresponding height, and constantly change the angle and speed of flight according to the position of the prey, thereby launching an attack on the prey. When prey is found, the seagull attacks the prey in a spiral manner on a three-dimensional plane. The plane behavior of x, y, z is expressed as follows:

$$x = r \times \cos(\theta) \tag{7}$$

$$y = r \times \sin(\theta) \tag{8}$$

$$z = r \times \theta \tag{9}$$

$$r = u \times e^{\theta v} \tag{10}$$

where r represents the radius of the seagull in the circling process, and θ is a random angle value in the range of [0, 2π]. u and v are fixed values of the spiral state. e is the base of the natural logarithm. The equation for the position change of the seagull during the attack is as follows:

$$X(t) = D_S(t) \times x \times y \times z + X_{best}(t) \tag{11}$$

The pseudo-code of the traditional SOA is given as follows in Algorithm 1.

Algorithm 1. Pseudocode of SOA.

```
Set the size N, dim, maximum iterations, u, v, fc
Initialize seagulls' positions X
t = 0
while (t < Maxiteration) do
The default global optimal solution is the position of the first seagull
    for i = 1: size(X,1) do
        update additional variable A using Equation (3)
        Calculate Cs using Equation (2)
        rd takes a random value on (0, 1)
        Calculate Ms using Equation (4)
        Calculate Ds using Equation (6)
        Update r, x, y, z using Equations (7)–(10)
        Calculate new seagull position using Equation (11)
    end for
    for i = 1: size(X,1) do
        for j = 1: size(X,2) do
          Border control
      end for
    end for
    for i = 1: size(X,1) do
        Calculate the fitness value of the new seagull position
    end for
    Sort the fitness value and update the optimal position and fitness value of the seagull
    t ← t + 1
end while
return the best solution
```

3. Improvement Methods Based on SOA

The improved SOA has two effective strategies. Firstly, the individual disturbance strategy is added to improve the optimization ability of the algorithm. Then, embed the attraction-repulsion strategy into the original SOA to increase the possibility of the population approaching the optimal solution.

3.1. Individual Disturbance

In the process of searching for the optimal direction for seagulls, the original algorithm updates the optimal direction according to the seagull's own position and optimal position, which will cause the problem of falling into local optimality, causing the seagull population to lose its direction in the migration process, and misleading the seagull group to deviate from the optimal migration route. In this paper, in the process of seagulls looking for the migration direction, in addition to relying on their own position and optimal position, another seagull individual is also used, and a weight is added to coordinate the seagulls' exploration ability to find the optimal direction. The equation of updated seagull optimization direction is as follows:

$$M_S = X(t) - m \times B \times (X_{best}(t) - X_K(t)) \tag{12}$$

where $X(t)$ is the position of the seagull under the current iteration, $X_{best}(t)$ is the optimal position of the current seagull, and $X_K(t)$ is the position of the random seagull. The weight expression is as follows:

$$m = \frac{Maxiteration - t}{Maxiteration} \tag{13}$$

m is a linear weight, which decreases linearly with the increase in iteration times to balance global and local search.

3.2. Adopt an Attraction-Repulsion Strategy

The migration of the seagull population is often guided by the global optimal individual to move towards the optimal solution. Still, if the global optimal individual falls into the local optimal and cannot jump out, it is likely to stagnate the whole population. To solve this problem, this paper adopts the attraction-repulsion strategy. The idea of attraction-repulsion first appeared in the particle swarm optimization algorithm of attraction and repulsion in 2002 [63]. In this paper, a global best solution and a global worst solution are added in Equation (14) through the attraction-repulsion strategy, which allows the seagull population to move randomly under the effect of attraction and repulsion to find the optimal solution. As the iterative process of the algorithm enters the later stage, the diversity of the population will be significantly reduced, and the premature phenomenon will eventually occur. The global worst position introduced at this point can play a role in increasing the population diversity. It makes the population more comprehensive in the local search process and overcomes the problem of premature maturity of the algorithm. The search equation for the position of seagulls using the attraction-repulsion strategy is as follows:

$$newD_S = r \times D_S + (\omega_1 \times (1 - r)) \times (GBESTX - D_S) - (\omega_2 \times (1 - r)) \times (GWORSTX - D_S) \tag{14}$$

where $GWORSTX$ is the global worst position, $GBESTX$ is the global optimal position, $X(t)$ is the seagull position under the current iteration, and r is a random number between 0 and 1. Through the experiment, it is found that when the ω_1 and ω_2 are 0.5 and 0.4, respectively, the seagulls will be better affected by the interaction of attraction and repulsion in the process of migration, be close to the optimal seagull individual, enhance the diversity of seagull population, improve the optimization ability, and reduce the risk of falling into the local optimum. The pseudo-code of IDARSOA is shown in Algorithm 2.

To better understand the idea and algorithm flow of this optimization algorithm, the flow chart of IDARSOA is shown in Figure 1.

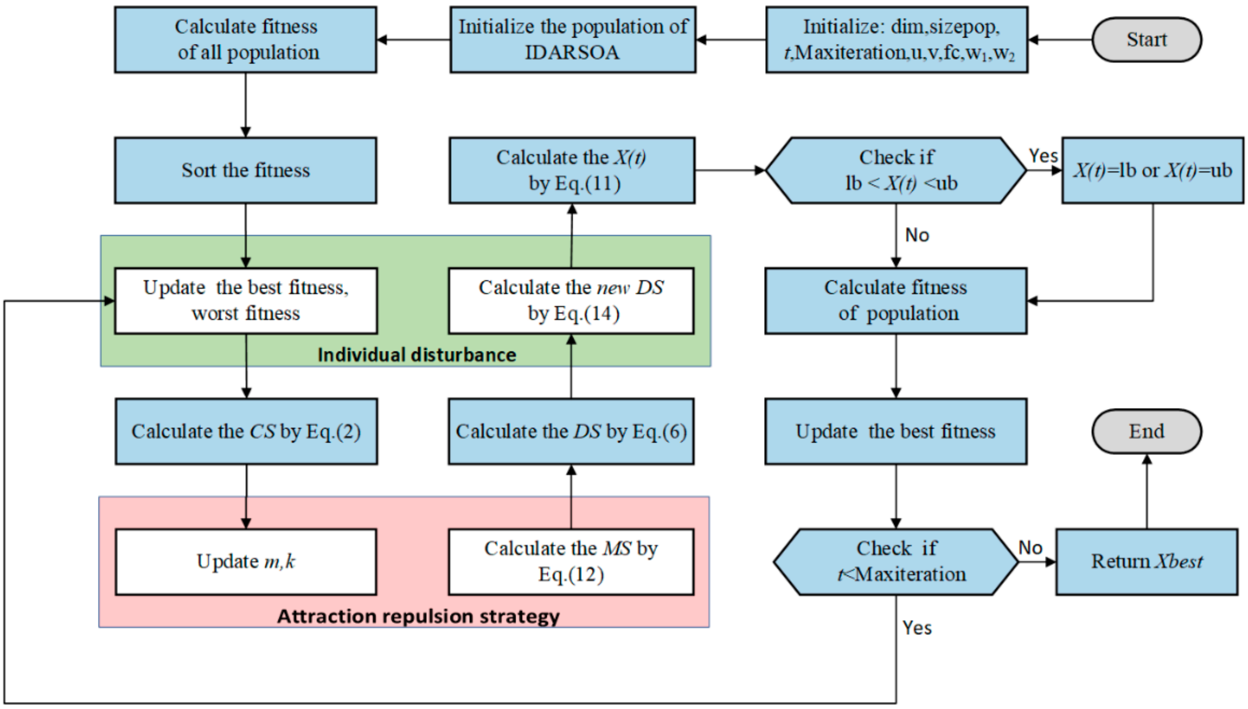

Figure 1. Flow chart of IDARSOA.

Algorithm 2. Pseudocode of IDARSOA.

Set the size N, dim, $maximum\ iterations$, u, v, fc, ω_1, ω_2
Initialize seagulls' positions X
$t = 0$
while (t < *Maxiteration*) **do**
    Calculate and rank the fitness value of the seagull population
    Get the best and worst positions in the population
    for i = 1: size(X,1) **do**
        Update additional variable A using Equation (3)
        Calculate Cs using Equation (2)
        Update m using Equation (13)
        Randomly generate an integer in (1, D) and assign it to K
        rd takes a random value on (0, 1)
        Calculate Ms using Equation (4)
        Calculate Ds using Equation (6)
        Generate a random number at (0, 1) and assign it to R
        Calculate *new Ds* according to the attraction and repulsion strategy using Equation (14)
        Update r, x, y, z using Equations (7)–(10)
        Calculate new seagull position using Equation (11)
    end for
    for i = 1: size(X,1) **do**
        for j = 1: size(X,2) **do**
            Border control
        end for
    end for
    for i = 1: size(X,1) **do**
        Calculate the fitness value of the new seagull position
    end for
    Sort the fitness value and update the optimal position and fitness value of the seagull
    $t \leftarrow t + 1$
end while
return *the best solution*

The time complexity of the improved IDARSOA depends on the number of iterations of the algorithm (S), the total number of seagulls (n), and the dimension of the case at hand (d). Through analysis, the overall time complexity is O(IDARSOA) = O(initialize) + O(calculate the fitness of initialize) + O(select the best fitness from the fitness) + S × (O(calculate the CS) + O(perform individual disturbance strategy) + O(perform attraction repulsion strategy) + O(attack) + O(boundary control) + O(update the positions of seagull)). Initialization time complexity is $O(n \times d)$, calculate the fitness of the initial value of $O(n)$, to find the best fitness by the fitness order of $O(n \times log2n)$, calculated CS time complexity of $O(n)$, the time complexity required to implement perform individual disturbance strategy is $O(n)$, the attack behavior and boundary control of the local search require $O(n \times d)$, the position of the updated seagull is $O(n)$. Therefore, the final time complexity of IDARSOA is as follows:

$$\begin{aligned} O(\text{IDARSOA}) &= O(n \times d) + O(n) + O(n \times log2n) + S \times (O(n) + O(n) + O(n) + O(n \times d) + O(n)) \\ &= O(n \times d) + O(n) + O(n \times log2n) + S \times \ (4O(n) + O(n \times d)). \end{aligned}$$

4. Experimental Results and Discussion

In this part, to verify the performance of the IDARSOA, 20 well-known functions are used to test the efficiency of the proposed optimizer. There are four experiments: The first is sensitivity analyses of the parameters in IDARSOA. The second is the comparison experiment between IDARSOA and IDSOA, ARSOA, and the original SOA, which proves that the SOA variant has an improved performance compared to the original algorithm, and the improvement strategy is effective. The third is a comparative experiment between IDARSOA and the novel swarm intelligence optimization algorithm to verify that IDARSOA is superior to those popular intelligent algorithms. The last is to compare IDARSOA with other algorithm variants. The results are used to verify the effects of IDARSOA. To ensure the fairness of the experiment, all methods should be tested under the same conditions [22]. All experiments in this paper use MATLAB2018 software; the dimension is determined to be 30, the number of running layers is 30, and the search agent is set to 30. The description of these 20 functions is shown in Table A1. F1–F10 are taken from CEC 2019 [66–71], F11–F20 are taken from CEC 2020. The bound is the search space range of the test function, and F(min) is the minimum value of the test function.

4.1. IDARSOA's Parameters Sensitivity Analyses

A in Equation (3) in IDARSOA represents the motion behavior of seagulls in a specified space, which is mainly affected by the parameter fc. To explore the influence of the value of fc on the performance of the seagull optimization algorithm, we set the value of fc to 1, 2, 3, 5, 7, and 9, which are represented by IDARSOAfc1, IDARSOAfc2, IDARSOAfc3, IDARSOAfc5, IDARSOAfc7, and IDARSOAfc9, respectively. Table 1 shows how these algorithms find the optimal solution in 20 test functions. It can be seen from the data in the table that in the three functions F4, F19, and F20, the ability of IDARSOA with different parameters to find the optimal solution is the same. In F1, the average value of the optimal solution found by these algorithms is the same. However, through the comparison of STD, it is found that IDARSOAfc1 has the best stability. In other functions, the value of fc is different, and the optimization performance in functions is also different. Integrating 20 test functions, IDARSOAfc2 has the best effect. Therefore, this paper sets the value of fc in IDARSOA to two.

Table 1. Comparison of different fc of IDARSOA.

	F1		**F2**		**F3**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	1.000000×10^{0}	5.758899×10^{-14}	4.952258×10^{0}	1.817900×10^{-1}	2.433492×10^{0}	1.186997×10^{0}
IDARSOAfc2	1.000000×10^{0}	2.667535×10^{-13}	4.416804×10^{0}	2.696345×10^{-1}	2.125574×10^{0}	9.741309×10^{-1}
IDARSOAfc3	1.000000×10^{0}	1.427971×10^{-12}	4.620769×10^{0}	3.408565×10^{-1}	2.422004×10^{0}	1.022597×10^{0}
IDARSOAfc5	1.000000×10^{0}	7.603133×10^{-13}	4.468247×10^{0}	2.786393×10^{-1}	3.079520×10^{0}	1.460601×10^{0}
IDARSOAfc7	1.000000×10^{0}	3.477511×10^{-14}	4.630774×10^{0}	3.173943×10^{-1}	4.326896×10^{0}	1.815038×10^{0}
IDARSOAfc9	1.000000×10^{0}	2.433234×10^{-11}	4.729463×10^{0}	3.398487×10^{-1}	4.073180×10^{0}	1.644750×10^{0}
	F4		**F5**		**F6**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	3.688039×10^{1}	1.373013×10^{1}	5.821549×10^{0}	9.047358×10^{0}	5.491320×10^{0}	1.275813×10^{0}
IDARSOAfc2	2.463451×10^{1}	1.139996×10^{1}	2.194699×10^{0}	6.875180×10^{-1}	5.664204×10^{0}	1.692936×10^{0}
IDARSOAfc3	2.239873×10^{1}	8.125202×10^{0}	1.998788×10^{0}	3.113314×10^{-1}	4.818130×10^{0}	1.701835×10^{0}
IDARSOAfc5	2.424563×10^{1}	6.106529×10^{0}	2.061292×10^{0}	4.047150×10^{-1}	5.185845×10^{0}	1.352847×10^{0}
IDARSOAfc7	2.731944×10^{1}	1.041899×10^{0}	2.108973×10^{0}	6.882413×10^{-1}	5.632649×10^{0}	1.915015×10^{0}
IDARSOAfc9	2.859710×10^{1}	1.035596×10^{1}	2.131835×10^{0}	6.668301×10^{-1}	5.873998×10^{0}	2.092584×10^{0}
	F7		**F8**		**F9**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	1.263735×10^{3}	3.365559×10^{2}	4.074083×10^{0}	4.138044×10^{-1}	1.212927×10^{0}	7.951421×10^{-2}
IDARSOAfc2	1.406561×10^{3}	4.167857×10^{2}	3.953226×10^{0}	3.750406×10^{-1}	1.181665×10^{0}	7.032048×10^{-2}
IDARSOAfc3	1.509095×10^{3}	3.868018×10^{2}	3.920128×10^{0}	4.127622×10^{-1}	1.185820×10^{0}	8.267084×10^{-2}
IDARSOAfc5	1.468453×10^{3}	5.192703×10^{2}	4.002200×10^{0}	3.019612×10^{-1}	1.224425×10^{0}	7.752889×10^{-2}
IDARSOAfc7	1.619661×10^{3}	4.745775×10^{2}	4.158420×10^{0}	3.717104×10^{-1}	1.261381×10^{0}	8.587691×10^{-2}
IDARSOAfc9	1.646884×10^{3}	4.487442×10^{2}	4.139863×10^{0}	3.270712×10^{-1}	1.260103×10^{0}	9.578651×10^{-2}
	F10		**F11**		**F12**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	2.124024×10^{1}	1.382577×10^{-1}	8.891585×10^{9}	6.785941×10^{9}	6.663063×10^{3}	6.116660×10^{2}
IDARSOAfc2	2.098561×10^{1}	1.830573×10^{0}	4.385372×10^{9}	4.046630×10^{9}	7.024390×10^{3}	6.603009×10^{2}
IDARSOAfc3	2.125866×10^{1}	1.031388×10^{-1}	2.122388×10^{9}	1.018590×10^{9}	6.815012×10^{3}	6.312405×10^{2}
IDARSOAfc5	2.130665×10^{1}	1.963167×10^{-1}	2.727998×10^{9}	2.547515×10^{9}	7.080456×10^{3}	5.757104×10^{2}
IDARSOAfc7	2.142574×10^{1}	2.103051×10^{-1}	2.055958×10^{9}	1.768645×10^{9}	7.111943×10^{3}	6.878647×10^{2}
IDARSOAfc9	2.138810×10^{1}	2.077814×10^{-1}	2.791952×10^{9}	1.488388×10^{9}	7.212336×10^{3}	8.694651×10^{2}
	F13		**F14**		**F15**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	1.205610×10^{3}	7.625843×10^{1}	1.900000×10^{3}	0.000000×10^{0}	2.114255×10^{7}	5.741210×10^{7}
IDARSOAfc2	1.051132×10^{3}	8.194302×10^{1}	1.900000×10^{3}	0.000000×10^{0}	1.031563×10^{7}	2.481495×10^{7}
IDARSOAfc3	1.019698×10^{3}	5.671150×10^{1}	1.900000×10^{3}	0.000000×10^{0}	7.595739×10^{7}	1.634042×10^{8}
IDARSOAfc5	1.053406×10^{3}	6.639036×10^{1}	1.900000×10^{3}	0.000000×10^{0}	4.953248×10^{7}	1.150416×10^{8}
IDARSOAfc7	1.087131×10^{3}	8.449326×10^{1}	1.900000×10^{3}	0.000000×10^{0}	3.026353×10^{7}	1.108006×10^{8}
IDARSOAfc9	1.097117×10^{3}	1.210897×10^{2}	1.900000×10^{3}	0.000000×10^{0}	1.629272×10^{7}	2.692786×10^{7}
	F16		**F17**		**F18**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOAfc1	3.134750×10^{3}	3.749393×10^{2}	1.286831×10^{8}	2.581530×10^{8}	2.459859×10^{3}	2.249968×10^{1}
IDARSOAfc2	2.929809×10^{3}	4.989991×10^{2}	8.429436×10^{7}	1.946799×10^{8}	2.438595×10^{3}	2.656722×10^{1}
IDARSOAfc3	2.823617×10^{3}	3.599560×10^{2}	1.542068×10^{8}	2.822871×10^{8}	2.443144×10^{3}	2.455074×10^{1}
IDARSOAfc5	3.009103×10^{3}	4.150459×10^{2}	1.561803×10^{8}	2.815656×10^{8}	2.439564×10^{3}	3.676659×10^{1}
IDARSOAfc7	2.951470×10^{3}	2.636037×10^{2}	2.118624×10^{8}	2.940915×10^{8}	2.446642×10^{3}	3.351741×10^{1}
IDARSOAfc9	3.098547×10^{3}	3.829978×10^{2}	2.434406×10^{8}	4.983535×10^{8}	2.489955×10^{3}	6.573209×10^{1}
	F19		**F20**		**Mean Level**	**Rank**
	AVG	**STD**	**AVG**	**STD**		
IDARSOAfc1	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	3.35	4
IDARSOAfc2	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	2.1	1
IDARSOAfc3	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	2.2	2
IDARSOAfc5	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	2.9	3
IDARSOAfc7	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	3.75	5
IDARSOAfc9	2.600000×10^{3}	0.000000×10^{0}	2.700000×10^{3}	0.000000×10^{0}	4.45	6

In order to explore the best combination value of the attraction weight ω_1 of the best individual, and the repulsion weight ω_2 of the worst individual in the attraction-repulsion strategy, and considering that attraction-repulsion is a pair of interaction forces, this section selects another weight between 0.1–0.9 when ω_1 and ω_2 are 0.5 respectively, to obtain the most suitable weight. As shown in Table 2, after the combination of different weights, there are 17 combination forms, and the specific ω_1 and ω_2 values are shown in the table.

Table 2. Parameter settings of IDARSOA.

Algorithm	Parameters	Algorithm	Parameters
IDARSOA01	$\omega_1 = 0.5$; $\omega_1 = 0.1$	IDARSOA10	$\omega_1 = 0.1$; $\omega_1 = 0.5$
IDARSOA02	$\omega_1 = 0.5$; $\omega_1 = 0.2$	IDARSOA11	$\omega_1 = 0.2$; $\omega_1 = 0.5$
IDARSOA03	$\omega_1 = 0.5$; $\omega_1 = 0.3$	IDARSOA12	$\omega_1 = 0.3$; $\omega_1 = 0.5$
IDARSOA04	$\omega_1 = 0.5$; $\omega_1 = 0.4$	IDARSOA13	$\omega_1 = 0.4$; $\omega_1 = 0.5$
IDARSOA05	$\omega_1 = 0.5$; $\omega_1 = 0.5$	IDARSOA14	$\omega_1 = 0.6$; $\omega_1 = 0.5$
IDARSOA06	$\omega_1 = 0.5$; $\omega_1 = 0.6$	IDARSOA15	$\omega_1 = 0.7$; $\omega_1 = 0.5$
IDARSOA07	$\omega_1 = 0.5$; $\omega_1 = 0.7$	IDARSOA16	$\omega_1 = 0.8$; $\omega_1 = 0.5$
IDARSOA08	$\omega_1 = 0.5$; $\omega_1 = 0.8$	IDARSOA17	$\omega_1 = 0.9$; $\omega_1 = 0.5$
IDARSOA09	$\omega_1 = 0.5$; $\omega_1 = 0.9$		

The comparison of different weight values among the 20 tested functions is displayed in Table 3, where different combinations of weights have different effects in various functions. Mean level in the table indicates the average ranking value of the algorithm among the 20 functions, and rank is the final ranking obtained from mean level. The data in the table show that too much or too little attraction and too much or too little repulsion will affect the search capability. This is because when the attraction weight is too large, it will suppress the effect of repulsion. If the globally optimal individual falls into the local optimum, the weight given to the repulsion is not enough to get rid of the local optimal solution space. Only a larger weight is given to the repulsion, but this will lead to the current individual crossing the boundary, and the optimal solution is not true. When the attraction is too small, the present individual will approach the optimal solution. If the weight of the repulsion is small at this time, the effect of attraction and repulsion strategy will be weakened. However, if the weight given to the repulsion is too large, it will cause the individual to move away from the optimal solution. The average ranking value of IDARSOA04 is the best in all combinations, and the rank value is the first. This shows that when the attraction weight is 0.5 and the repulsion weight is 0.4, the performance of the attraction-repulsion strategy can play the best.

Table 3. Comparison of parameters settings.

	F1		F2		F3	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	1.0000×10^{0}	7.1098×10^{-13}	4.5221×10^{0}	3.0290×10^{-1}	2.2998×10^{0}	9.4177×10^{-1}
IDARSOA02	1.0000×10^{0}	3.8050×10^{-12}	4.6309×10^{0}	3.3643×10^{-1}	2.5054×10^{0}	8.1540×10^{-1}
IDARSOA03	1.0000×10^{0}	2.1336×10^{-13}	4.5643×10^{0}	3.4439×10^{-1}	2.7870×10^{0}	1.3151×10^{0}
IDARSOA04	1.0000×10^{0}	1.7623×10^{-12}	4.4649×10^{0}	2.7740×10^{-1}	2.1095×10^{0}	9.5977×10^{-1}
IDARSOA05	1.0000×10^{0}	6.2818×10^{-12}	4.4148×10^{0}	2.4449×10^{-1}	1.9846×10^{0}	8.1660×10^{-1}
IDARSOA06	1.0000×10^{0}	9.8255×10^{-13}	4.4005×10^{0}	2.4310×10^{-1}	2.8128×10^{0}	1.4789×10^{0}
IDARSOA07	1.0000×10^{0}	5.1951×10^{-12}	4.4686×10^{0}	3.0250×10^{-1}	2.7536×10^{0}	1.3484×10^{0}
IDARSOA08	1.0000×10^{0}	2.5142×10^{-12}	4.5113×10^{0}	3.2751×10^{-1}	3.1051×10^{0}	1.6605×10^{0}
IDARSOA09	1.0000×10^{0}	1.9889×10^{-13}	4.5311×10^{0}	3.4123×10^{-1}	3.1209×10^{0}	1.4098×10^{0}
IDARSOA10	1.0000×10^{0}	0.0000×10^{0}	4.6013×10^{0}	3.3527×10^{-1}	3.8233×10^{0}	1.4639×10^{0}
IDARSOA11	1.0000×10^{0}	1.3380×10^{-15}	4.5414×10^{0}	3.3407×10^{-1}	3.6497×10^{0}	1.5097×10^{0}
IDARSOA12	1.0000×10^{0}	4.1233×10^{-17}	4.5305×10^{0}	3.4018×10^{-1}	3.0959×10^{0}	1.2182×10^{0}
IDARSOA13	1.0000×10^{0}	6.6097×10^{-15}	4.4954×10^{0}	3.3859×10^{-1}	2.5881×10^{0}	1.3068×10^{0}
IDARSOA14	1.0000×10^{0}	9.3677×10^{-12}	4.5112×10^{0}	3.0444×10^{-1}	2.1808×10^{0}	8.4346×10^{-1}
IDARSOA15	1.0000×10^{0}	9.0943×10^{-13}	4.6305×10^{0}	3.1351×10^{-1}	2.8134×10^{0}	1.5691×10^{0}
IDARSOA16	1.0000×10^{0}	1.2898×10^{-11}	4.4263×10^{0}	2.4384×10^{-1}	2.9569×10^{0}	1.4764×10^{0}
IDARSOA17	1.0000×10^{0}	2.6900×10^{-12}	4.6626×10^{0}	3.1935×10^{-1}	2.6036×10^{0}	1.0676×10^{0}

Table 3. *Cont.*

	F4		F5		F6	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	2.4581×10^{1}	1.0205×10^{1}	2.6664×10^{0}	1.0053×10^{0}	5.3757×10^{0}	1.3387×10^{0}
IDARSOA02	2.4620×10^{1}	9.5404×10^{0}	2.3394×10^{0}	7.2482×10^{-1}	5.6362×10^{0}	1.7517×10^{0}
IDARSOA03	2.6353×10^{1}	1.0136×10^{1}	2.3466×10^{0}	8.4829×10^{-1}	5.6669×10^{0}	1.6688×10^{0}
IDARSOA04	2.8527×10^{1}	1.1019×10^{1}	2.1741×10^{0}	9.0196×10^{-1}	5.6726×10^{0}	1.6060×10^{0}
IDARSOA05	2.8420×10^{1}	1.0021×10^{1}	2.1779×10^{0}	1.3784×10^{0}	5.4521×10^{0}	1.3099×10^{0}
IDARSOA06	2.6725×10^{1}	9.2999×10^{0}	3.2507×10^{0}	3.1355×10^{0}	5.6015×10^{0}	1.8411×10^{0}
IDARSOA07	2.6327×10^{1}	8.6022×10^{0}	3.2597×10^{0}	1.5464×10^{0}	5.9124×10^{0}	1.3527×10^{0}
IDARSOA08	2.5881×10^{1}	6.1845×10^{0}	3.2389×10^{0}	2.0344×10^{0}	5.6661×10^{0}	1.2747×10^{0}
IDARSOA09	2.6593×10^{1}	8.0807×10^{0}	3.4279×10^{0}	1.4143×10^{0}	6.0165×10^{0}	1.4066×10^{0}
IDARSOA10	3.8186×10^{1}	7.9266×10^{0}	6.1616×10^{0}	2.0080×10^{0}	6.6093×10^{0}	9.2928×10^{-1}
IDARSOA11	3.3962×10^{1}	7.3061×10^{0}	4.1841×10^{0}	1.4833×10^{0}	6.2598×10^{0}	1.1306×10^{0}
IDARSOA12	2.9104×10^{1}	6.2466×10^{0}	3.5271×10^{0}	1.2830×10^{0}	5.7546×10^{0}	1.4611×10^{0}
IDARSOA13	2.7372×10^{1}	7.7938×10^{0}	2.8897×10^{0}	1.1336×10^{0}	5.7283×10^{0}	1.9894×10^{0}
IDARSOA14	2.8092×10^{1}	8.7542×10^{0}	3.2801×10^{0}	3.5517×10^{0}	5.9541×10^{0}	1.8852×10^{0}
IDARSOA15	2.4409×10^{1}	8.0677×10^{0}	2.8339×10^{0}	1.6447×10^{0}	5.5501×10^{0}	1.5394×10^{0}
IDARSOA16	2.9955×10^{1}	1.1018×10^{1}	3.5394×10^{0}	2.3300×10^{0}	5.9731×10^{0}	1.9348×10^{0}
IDARSOA17	2.6693×10^{1}	1.0648×10^{1}	3.0238×10^{0}	1.2134×10^{0}	5.9119×10^{0}	2.0688×10^{0}
	F7		**F8**		**F9**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	1.4524×10^{3}	3.9054×10^{2}	4.0606×10^{0}	4.0185×10^{-1}	1.2206×10^{0}	7.9805×10^{-2}
IDARSOA02	1.3058×10^{3}	3.7027×10^{2}	3.9924×10^{0}	4.0475×10^{-1}	1.1941×10^{0}	6.3549×10^{-2}
IDARSOA03	1.4908×10^{3}	4.2485×10^{2}	4.0050×10^{0}	3.5385×10^{-1}	1.2189×10^{0}	1.0129×10^{-1}
IDARSOA04	1.3631×10^{3}	3.0353×10^{2}	4.0836×10^{0}	3.4300×10^{-1}	1.1955×10^{0}	7.6053×10^{-2}
IDARSOA05	1.4481×10^{3}	4.6512×10^{2}	3.9618×10^{0}	4.4447×10^{-1}	1.2555×10^{0}	1.1981×10^{-1}
IDARSOA06	1.5382×10^{3}	4.0746×10^{2}	4.0030×10^{0}	3.6576×10^{-1}	1.2387×10^{0}	1.0696×10^{-1}
IDARSOA07	1.5988×10^{3}	4.3166×10^{2}	4.2008×10^{0}	2.4802×10^{-1}	1.2737×10^{0}	7.0067×10^{-2}
IDARSOA08	1.6605×10^{3}	3.6934×10^{2}	4.2143×10^{0}	3.4708×10^{-1}	1.3011×10^{0}	6.8430×10^{-2}
IDARSOA09	1.6913×10^{3}	4.4211×10^{2}	4.2161×10^{0}	3.0293×10^{-1}	1.3086×10^{0}	7.1227×10^{-2}
IDARSOA10	1.5330×10^{3}	3.8835×10^{2}	4.4975×10^{0}	4.4491×10^{-1}	1.4773×10^{0}	2.6855×10^{-1}
IDARSOA11	1.5015×10^{3}	3.5299×10^{2}	4.3507×10^{0}	3.2739×10^{-1}	1.3384×10^{0}	7.9786×10^{-2}
IDARSOA12	1.5489×10^{3}	4.5487×10^{2}	4.2247×10^{0}	2.2252×10^{-1}	1.3136×10^{0}	6.0874×10^{-2}
IDARSOA13	1.5232×10^{3}	4.0966×10^{2}	4.0739×10^{0}	3.4958×10^{-1}	1.2415×10^{0}	6.6822×10^{-2}
IDARSOA14	1.4081×10^{3}	4.2644×10^{2}	3.9631×10^{0}	3.8229×10^{-1}	1.2280×10^{0}	1.0433×10^{-1}
IDARSOA15	1.4451×10^{3}	4.3183×10^{2}	4.0201×10^{0}	3.7615×10^{-1}	1.2287×10^{0}	1.0141×10^{-1}
IDARSOA16	1.5349×10^{3}	4.1635×10^{2}	3.9816×10^{0}	2.6288×10^{-1}	1.2260×10^{0}	1.2272×10^{-1}
IDARSOA17	1.6059×10^{3}	4.9431×10^{2}	4.0404×10^{0}	3.5546×10^{-1}	1.2206×10^{0}	1.1546×10^{-1}
	F10		**F11**		**F12**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	2.1280×10^{1}	1.7507×10^{-1}	2.1382×10^{8}	3.4416×10^{8}	2.0200×10^{3}	2.8335×10^{2}
IDARSOA02	2.1286×10^{1}	1.6340×10^{-1}	1.2274×10^{8}	1.4623×10^{8}	2.1510×10^{3}	3.2795×10^{2}
IDARSOA03	2.1269×10^{1}	1.5574×10^{-1}	4.2864×10^{8}	1.6445×10^{9}	2.0298×10^{3}	3.3485×10^{2}
IDARSOA04	2.1279×10^{1}	1.2501×10^{-1}	1.1905×10^{8}	3.1222×10^{8}	1.9318×10^{3}	3.0480×10^{2}
IDARSOA05	2.1321×10^{1}	1.7899×10^{-1}	1.8711×10^{8}	6.2487×10^{8}	1.9360×10^{3}	3.2684×10^{2}
IDARSOA06	2.1322×10^{1}	1.8987×10^{-1}	9.8061×10^{8}	2.2807×10^{9}	2.1234×10^{3}	3.8321×10^{2}
IDARSOA07	2.1420×10^{1}	2.1176×10^{-1}	3.6423×10^{8}	1.2382×10^{9}	2.1887×10^{3}	3.9374×10^{2}
IDARSOA08	2.1464×10^{1}	2.0693×10^{-1}	4.6418×10^{8}	1.6441×10^{9}	2.1188×10^{3}	3.4011×10^{2}
IDARSOA09	2.1392×10^{1}	1.9553×10^{-1}	8.3583×10^{8}	2.2192×10^{9}	2.3045×10^{3}	3.2662×10^{2}
IDARSOA10	2.1570×10^{1}	1.5581×10^{-1}	9.3577×10^{8}	2.2582×10^{9}	2.1975×10^{3}	2.7247×10^{2}
IDARSOA11	2.1482×10^{1}	2.0087×10^{-1}	5.0671×10^{8}	1.6407×10^{9}	2.2381×10^{3}	2.6707×10^{2}
IDARSOA12	2.1438×10^{1}	1.8690×10^{-1}	2.2028×10^{8}	2.1073×10^{8}	2.1718×10^{3}	2.1766×10^{2}
IDARSOA13	2.1385×10^{1}	2.0300×10^{-1}	4.1066×10^{8}	1.6480×10^{9}	2.1200×10^{3}	3.3548×10^{2}
IDARSOA14	2.1291×10^{1}	1.8167×10^{-1}	3.9643×10^{8}	1.6447×10^{9}	2.2573×10^{3}	4.5010×10^{2}
IDARSOA15	2.1225×10^{1}	1.1628×10^{-1}	2.4735×10^{8}	5.3249×10^{8}	2.1028×10^{3}	4.2773×10^{2}
IDARSOA16	2.1260×10^{1}	1.4831×10^{-1}	1.4818×10^{8}	1.4395×10^{8}	2.1507×10^{3}	3.4058×10^{2}
IDARSOA17	2.1298×10^{1}	1.6992×10^{-1}	2.2210×10^{8}	4.2484×10^{8}	2.1251×10^{3}	3.6144×10^{2}

Table 3. *Cont.*

	F13		F14		F15	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	7.3352×10^{2}	7.8137×10^{0}	1.9000×10^{3}	0.0000×10^{0}	1.9923×10^{6}	9.7803×10^{6}
IDARSOA02	7.3522×10^{2}	8.2661×10^{0}	1.9000×10^{3}	0.0000×10^{0}	1.9171×10^{6}	9.7928×10^{6}
IDARSOA03	7.3240×10^{2}	9.9243×10^{0}	1.9000×10^{3}	0.0000×10^{0}	5.2395×10^{4}	1.8689×10^{5}
IDARSOA04	7.3718×10^{2}	1.2167×10^{1}	1.9000×10^{3}	0.0000×10^{0}	1.6897×10^{5}	3.0976×10^{5}
IDARSOA05	7.3812×10^{2}	1.4718×10^{1}	1.9000×10^{3}	0.0000×10^{0}	3.7445×10^{6}	1.3596×10^{7}
IDARSOA06	7.3422×10^{2}	9.1288×10^{0}	1.9000×10^{3}	0.0000×10^{0}	2.1241×10^{5}	3.2841×10^{5}
IDARSOA07	7.3583×10^{2}	7.0423×10^{0}	1.9000×10^{3}	0.0000×10^{0}	1.9924×10^{6}	9.7802×10^{6}
IDARSOA08	7.3996×10^{2}	1.0126×10^{1}	1.9000×10^{3}	0.0000×10^{0}	2.7562×10^{6}	1.0101×10^{7}
IDARSOA09	7.3726×10^{2}	6.7043×10^{0}	1.9000×10^{3}	0.0000×10^{0}	9.6037×10^{6}	2.0197×10^{7}
IDARSOA10	7.4737×10^{2}	6.7846×10^{0}	1.9000×10^{3}	0.0000×10^{0}	2.4328×10^{6}	9.9372×10^{6}
IDARSOA11	7.4906×10^{2}	6.9347×10^{0}	1.9000×10^{3}	0.0000×10^{0}	3.0081×10^{5}	3.3392×10^{5}
IDARSOA12	7.4096×10^{2}	7.4914×10^{0}	1.9000×10^{3}	0.0000×10^{0}	2.0581×10^{6}	9.7678×10^{6}
IDARSOA13	7.3780×10^{2}	7.5538×10^{0}	1.9000×10^{3}	0.0000×10^{0}	3.7639×10^{6}	1.3590×10^{7}
IDARSOA14	7.3852×10^{2}	1.0807×10^{1}	1.9000×10^{3}	0.0000×10^{0}	7.5295×10^{4}	2.2042×10^{5}
IDARSOA15	7.3825×10^{2}	1.1380×10^{1}	1.9000×10^{3}	0.0000×10^{0}	9.3968×10^{4}	2.3985×10^{5}
IDARSOA16	7.3843×10^{2}	1.2874×10^{1}	1.9000×10^{3}	0.0000×10^{0}	2.2068×10^{5}	3.2102×10^{5}
IDARSOA17	7.3505×10^{2}	8.0908×10^{0}	1.9000×10^{3}	0.0000×10^{0}	1.3849×10^{5}	2.5673×10^{5}
	F16		**F17**		**F18**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA01	1.6346×10^{3}	2.6021×10^{1}	2.4613×10^{5}	2.0420×10^{5}	2.2977×10^{3}	1.4230×10^{1}
IDARSOA02	1.6415×10^{3}	3.7938×10^{1}	2.2126×10^{6}	$1.0799 \times 10^{+7}$	2.3003×10^{3}	1.3598×10^{-1}
IDARSOA03	1.6415×10^{3}	4.2627×10^{1}	1.6688×10^{5}	1.9092×10^{5}	2.2925×10^{3}	2.3775×10^{1}
IDARSOA04	1.6335×10^{3}	4.4810×10^{1}	2.3329×10^{5}	4.5377×10^{5}	2.2949×10^{3}	2.0431×10^{1}
IDARSOA05	1.6336×10^{3}	4.0635×10^{1}	2.6040×10^{5}	2.0104×10^{5}	2.2937×10^{3}	2.0205×10^{1}
IDARSOA06	1.6291×10^{3}	2.2961×10^{1}	2.8097×10^{5}	4.5706×10^{5}	2.2820×10^{3}	3.1238×10^{1}
IDARSOA07	1.6339×10^{3}	3.2790×10^{1}	4.3703×10^{5}	7.0714×10^{5}	2.2840×10^{3}	3.0200×10^{1}
IDARSOA08	1.6304×10^{3}	2.3174×10^{1}	2.8207×10^{5}	4.5525×10^{5}	2.2801×10^{3}	3.2190×10^{1}
IDARSOA09	1.6467×10^{3}	4.0193×10^{1}	4.2740×10^{5}	5.7028×10^{5}	2.2827×10^{3}	3.0705×10^{1}
IDARSOA10	1.6450×10^{3}	3.1481×10^{1}	3.2174×10^{5}	1.8418×10^{5}	2.2830×10^{3}	2.6971×10^{1}
IDARSOA11	1.6380×10^{3}	2.8113×10^{1}	2.7357×10^{5}	2.1122×10^{5}	2.2817×10^{3}	3.0474×10^{1}
IDARSOA12	1.6298×10^{3}	2.2318×10^{1}	3.8072×10^{5}	5.8818×10^{5}	2.2811×10^{3}	3.1022×10^{1}
IDARSOA13	1.6515×10^{3}	1.2246×10^{2}	2.0171×10^{5}	2.0005×10^{5}	2.2817×10^{3}	3.2621×10^{1}
IDARSOA14	1.6362×10^{3}	3.8043×10^{1}	1.3530×10^{5}	1.8946×10^{5}	2.2983×10^{3}	1.0829×10^{1}
IDARSOA15	1.6569×10^{3}	4.6309×10^{1}	2.5786×10^{5}	4.5928×10^{5}	2.2980×10^{3}	1.2827×10^{1}
IDARSOA16	1.6468×10^{3}	4.3076×10^{1}	2.9995×10^{5}	4.4376×10^{5}	2.3003×10^{3}	1.4027×10^{-1}
IDARSOA17	1.6395×10^{3}	4.2421×10^{1}	3.4032×10^{5}	4.3474×10^{5}	2.2979×10^{3}	1.3421×10^{1}
	F19		**F20**		**Mean Level**	**Rank**
	AVG	**STD**	**AVG**	**STD**		
IDARSOA01	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	5	2
IDARSOA02	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	6.6	6
IDARSOA03	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	5.3	3
IDARSOA04	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	4.65	1
IDARSOA05	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	5.8	4
IDARSOA06	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	6.5	5
IDARSOA07	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	8.85	13
IDARSOA08	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	8.65	11
IDARSOA09	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	10.95	16
IDARSOA10	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	11.9	17
IDARSOA11	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	10.65	15
IDARSOA12	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	9.25	14
IDARSOA13	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	7.5	9
IDARSOA14	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	7.45	8
IDARSOA15	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	6.6	6
IDARSOA16	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	8.75	12
IDARSOA17	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}	8.15	10

4.2. Study of the Proposed Method

This section describes the effects of two optimization mechanisms added to SOA: individual disturbance and attraction-repulsion strategy. Four different SOA effects were compared to examine the impact of all combinations of each mechanism on SOA. As shown in Table 4 below, "ID" and "AR" represent "individual disturbance" and "attraction-repulsion strategy", respectively. In Table 4, "1" indicates that SOA uses this mechanism, and "0" indicates the opposite; that is, it does not use this optimization mechanism. For example, the IDSOA representation combines the "individual disturbance" rather than the "attraction-repulsion strategy". The combination of the two strategies is shown in Table 4.

Table 4. The performance of the two strategies on SOA.

	ID	AR
SOA	0	0
ARSOA	0	1
IDSOA	1	0
IDARSOA	1	1

Based on the 20 functions in the test functions table, four SOAs were applied to these functions for testing. Four kinds of SOA results are shown in Table 5 below. This paper uses a non-parametric Wilcoxon signed-rank test at 5% significance level to prove the difference between IDARSOA and the other three algorithms. The "+", "−", and "=" in the table indicate superior to IDARSOA, inferior to IDARSOA, and equal to IDARSOA, respectively. According to the average ranking ARV in Table 5, IDARSOA outperforms the other three algorithms with a score of 1.4. This shows that IDARSOA performs better than other algorithms in the 20 test functions, reflecting that IDARSOA has better advantages than the other three algorithms. In addition, IDSOA and ARSOA are better than SOA in average ranking. This is because the individual disturbance strategy in this paper will use different random agent positions to perturb each time SOA looks for the optimal direction, to enhance the ability of the algorithm to jump out of the local optimization. The attraction-repulsion strategy makes SOA more comprehensive in the process of searching solution space through the interaction of attraction and repulsion between the optimal solution and the worst solution.

Table 5. Comparison of Wilcoxon signed-rank test results of different SOAs.

	F1	F2	F3	F4	F5	F6
IDARSOA	N/A	N/A	N/A	N/A	N/A	N/A
IDSOA	9.7656×10^{-4}	1.0201×10^{-1}	3.5152×10^{-6}	1.9729×10^{-5}	3.7243×10^{-5}	1.2506×10^{-4}
ARSOA	9.7656×10^{-4}	2.7016×10^{-5}	1.3820×10^{-3}	1.3601×10^{-5}	1.0246×10^{-5}	4.9916×10^{-3}
SOA	9.7656×10^{-4}	2.7016×10^{-5}	1.9209×10^{-6}	1.9209×10^{-6}	1.7344×10^{-6}	3.5152×10^{-6}
	F7	**F8**	**F9**	**F10**	**F11**	**F12**
IDARSOA	N/A	N/A	N/A	N/A	N/A	N/A
IDSOA	1.9209×10^{-6}	1.7344×10^{-6}	4.0715×10^{-5}	2.6033×10^{-6}	1.6046×10^{-4}	2.6033×10^{-6}
ARSOA	7.1889×10^{-1}	2.7116×10^{-1}	2.5967×10^{-5}	3.6826×10^{-2}	7.5137×10^{-5}	5.4463×10^{-2}
SOA	5.7517×10^{-6}	1.7344×10^{-6}	2.1266×10^{-6}	2.6033×10^{-6}	1.9729×10^{-5}	1.7344×10^{-6}
	F13	**F14**	**F15**	**F16**	**F17**	**F18**
IDARSOA	N/A	N/A	N/A	N/A	N/A	N/A
IDSOA	7.8126×10^{-1}	1.0000×10^{0}	9.2710×10^{-3}	1.7344×10^{-6}	2.2102×10^{-1}	2.5967×10^{-5}
ARSOA	6.3391×10^{-6}	1.0000×10^{0}	1.9569×10^{-2}	1.5658×10^{-2}	8.5896×10^{-2}	2.1827×10^{-2}
SOA	5.2165×10^{-6}	1.0000×10^{0}	1.6046×10^{-4}	1.7344×10^{-6}	1.5286×10^{-1}	1.7344×10^{-6}

Table 5. *Cont.*

	F19	F20	+/−/=	ARV	RANK
IDARSOA	N/A	N/A		1.4	1
IDSOA	9.7656×10^{-4}	4.3778×10^{-4}	13/3/4	2.5	3
ARSOA	1.0000×10^{0}	1.0000×10^{0}	11/2/7	2.05	2
SOA	4.8828×10^{-4}	4.3778×10^{-4}	17/1/2	3.45	4

Figure 2 shows IDARSOA and its two strategies used in SOA and compares the original SOA. It can be seen from Figure 2 that in F3, F6, F8, and F12, the convergence speed of IDARSOA is not as fast as SOA. Still, the best solution found by this algorithm in these functions is closer to the theoretical value of each function. It performs better in terms of optimality finding accuracy, indicating the strong exploration performance of IDARSOA. Overall, IDARSOA has a better optimization effect than IDSOA, ARSOA, and SOA, which shows that adding "individual disturbance" and "attraction-repulsion strategy" is very helpful to the search of algorithms and improves SOA performance. IDARSOA is the best way to deal with these different types of functions.

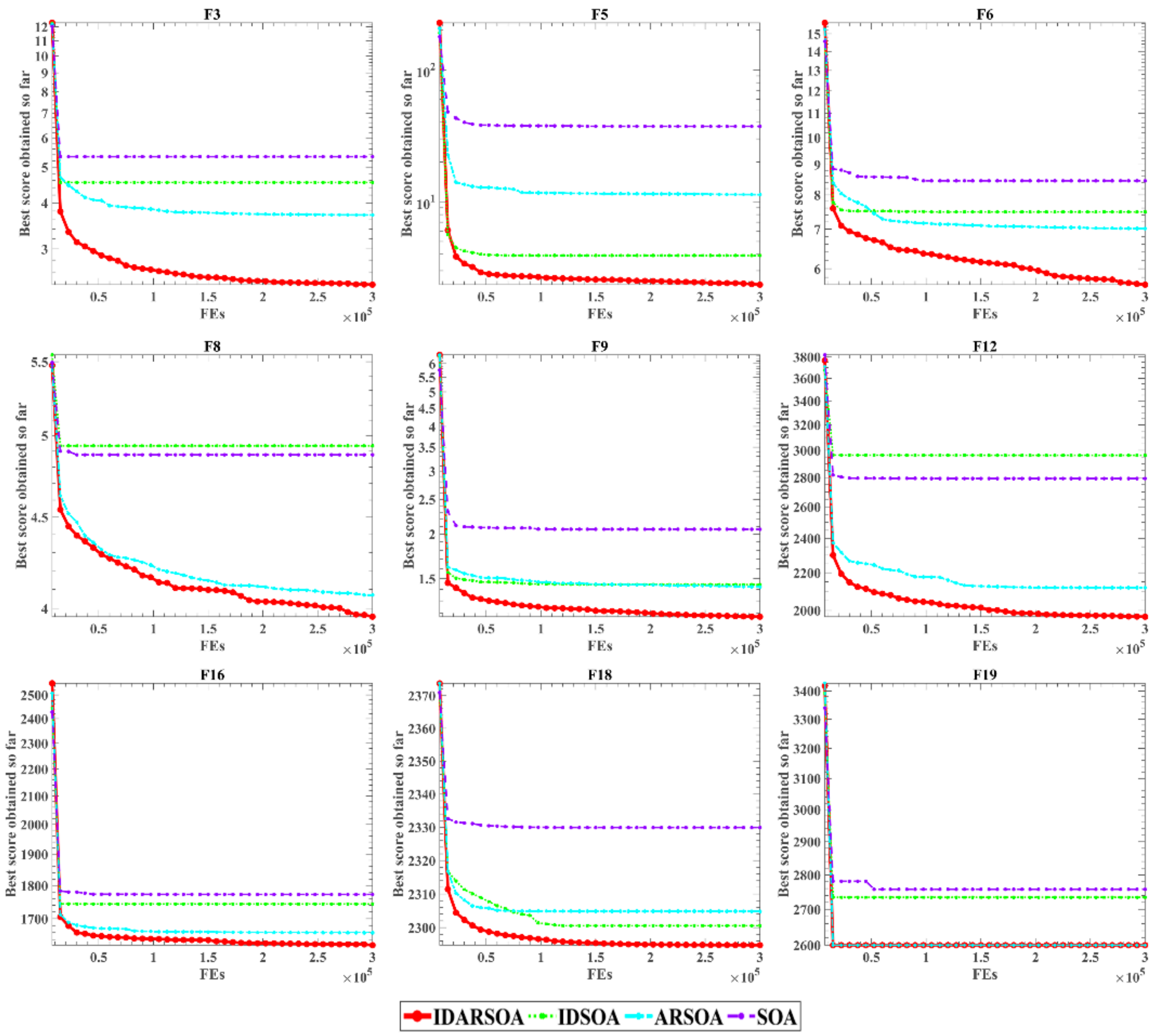

Figure 2. Comparison and convergence curve of mechanism.

To explore the changes in the performance of IDARSOA with the increase in data size and to ensure the reliability of the experiments, this section uses the univariate principle for the experiments. Under the same operating environment, we set the dim in the experiment to 50 and 100, the number of evaluations in the experiment to 300,000, and the number of trials to 30. Because the test function of CEC2019 has a fixed dimension, this part uses the CEC2020 test functions for validation. The Wilcoxon signed-rank test data for SOA with different mechanisms in different dimensions are shown in the following Table 6. When dim is 50, IDARSOA shows better performance than SOA in seven test functions compared to SOA, while the other three test functions, IDARSOA and SOA, obtain the same optimal solution. SOA with both ID and DD strategies outperformed SOA in terms of average ranking. When dim is set to 100, IDARSOA still ranks first among these algorithms with an ARV of 1.3. Still, the optimal value obtained among the seven functions is better than SOA. Combined with Table 5 above, the increase in data size does not affect the performance improvement of the ID and AR strategies for SOA, as IDARSOA is sufficient proof.

Table 6. Comparison of Wilcoxon signed rank test results of different SOAs in high dimension.

			dim = 50				
	F11	**F12**	**F13**	**F14**	**F15**	**F16**	
IDARSOA	N/A	N/A	N/A	N/A	N/A	N/A	
IDSOA	1.7344×10^{-6}	1.7344×10^{-6}	1.7344×10^{-6}	1.0000×10^{0}	4.0715×10^{-5}	4.7292×10^{-6}	
ARSOA	1.7344×10^{-6}	2.4308×10^{-2}	1.7344×10^{-6}	1.0000×10^{0}	3.1817×10^{-6}	1.7344×10^{-6}	
SOA	1.7344×10^{-6}	1.7344×10^{-6}	1.7344×10^{-6}	1.0000×10^{0}	1.7344×10^{-6}	1.7344×10^{-6}	
	F17	**F18**	**F19**	**F20**	**+/−/=**	**ARV**	**RANK**
IDARSOA	N/A	N/A	N/A	N/A		1.1	1
IDSOA	1.4936×10^{-5}	1.7344×10^{-6}	1.0000×10^{0}	1.0000×10^{0}	6/1/3	1.9	2
ARSOA	4.4493×10^{-5}	2.8786×10^{-6}	1.0000×10^{0}	1.0000×10^{0}	7/0/3	2.2	3
SOA	1.7344×10^{-6}	1.7344×10^{-6}	1.0000×10^{0}	1.0000×10^{0}	7/0/3	3	4
			dim = 100				
	F11	**F12**	**F13**	**F14**	**F15**	**F16**	
IDARSOA	N/A	N/A	N/A	N/A	N/A	N/A	
IDSOA	1.7344×10^{-6}	1.7344×10^{-6}	2.7653×10^{-3}	1.0000×10^{0}	3.3173×10^{-4}	4.1955×10^{-4}	
ARSOA	1.7344×10^{-6}	4.2843×10^{-1}	1.9209×10^{-6}	1.0000×10^{0}	8.1878×10^{-5}	4.8603×10^{-5}	
SOA	1.7344×10^{-6}	1.7344×10^{-6}	1.7344×10^{-6}	1.0000×10^{0}	1.7344×10^{-6}	1.7344×10^{-6}	
	F17	**F18**	**F19**	**F20**	**+/−/=**	**ARV**	**RANK**
IDARSOA	N/A	N/A	N/A	N/A		1.3	1
IDSOA	2.7653×10^{-3}	4.1955×10^{-4}	1.0000×10^{0}	1.0000×10^{0}	5/2/3	2.1	2
ARSOA	9.2710×10^{-3}	2.1266×10^{-6}	1.0000×10^{0}	1.0000×10^{0}	6/0/4	2.1	2
SOA	3.3173×10^{-4}	1.7344×10^{-6}	1.0000×10^{0}	1.0000×10^{0}	7/0/3	3	4

To explore the impact of the two mechanisms used in this paper on SOA performance in high dimensions, this section uses box plots to reflect the data distribution characteristics of the different algorithms. As shown in Figure 3 below, when dim = 50, the median of IDARSOA in F11 is smaller than the other three algorithms; the ranges of upper and lower edges are also very small, indicating the stable performance of the optimal value found by IDARSOA. In F14, from the data distribution of the four algorithms for function finding, all four algorithms find the theoretical optimal value. When dim = 100, the range between the upper and lower edges and the range between the upper and lower quartiles of IDARSOA in F15 and F17 are smaller than those of any of the algorithms, proving the stable performance of the search for the optimum. As a whole, the original SOA is not very stable in finding the optimal solution, and the optimal solution found is rather scattered. In contrast, the performance of IDARSOA, IDSOA, and ARSOA is more stable.

To explore the impact of the two strategies adopted in this paper on SOA, this section analyzes the balance and diversity of IDARSOA and SOA. As shown in Figure 4 below, this paper selects F1, F2, F14, and F18 from 20 test functions for discussion. The first column in Figure 4 is the balance diagram of IDARSOA, the second column shows the balance diagram of SOA, and the third column is the diversity analysis diagram. The balance diagrams contain three curves: exploration, development, and incremental decline. It can be seen from the figure that the exploration ability of the original algorithm SOA is weak, and the mining ability accounts for a large proportion of the whole search process. Due to its early entry into the development stage and long local development process, SOA has a weak global search ability and cannot get a good optimal solution. As can be seen from the balance analysis diagram of IDARSOA, its global search ability has been significantly improved.

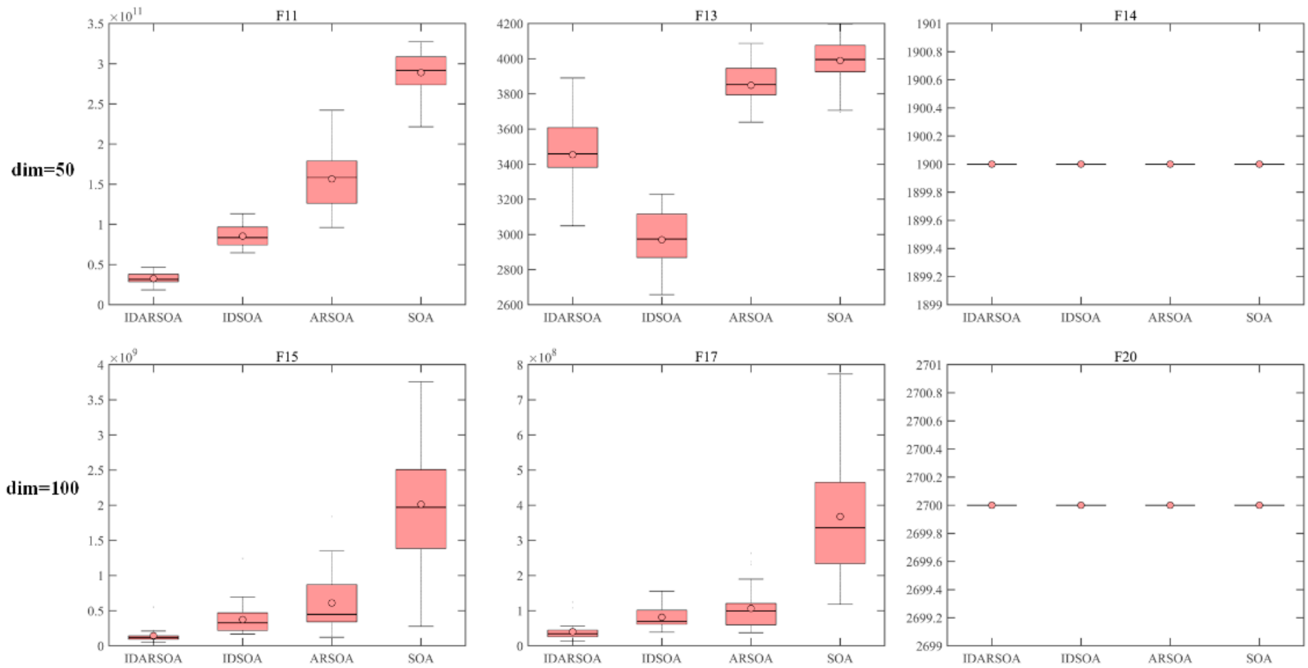

Figure 3. Box plots of SOAs.

By comparing the population diversity of IDARSOA and SOA, it can be seen that the two mechanisms used in this paper significantly increase the population diversity. Furthermore, the oscillation of IDARSOA diversity is much larger than that of SOA, which indicates that IDARSOA has more solutions to search in the solution space, effectively reducing the problem of stagnation occurring in the algorithm. This is because the diversity of the population is increased by the perturbation of random individuals when seagulls are searching for the optimal direction. In the process of local search time, the influence of the attraction-repulsion strategy used makes the search space more comprehensive. Still, at the same time, the IDARSOA population diversity decreases seriously slow, and the state of particles is scattered, which affects the convergence speed of IDARSOA. This phenomenon arises because we try to introduce other individuals for perturbation in the process of finding the optimal migration direction of the seagull population. Although the perturbation by individuals can reduce the risk of falling into the local optimum, the disadvantage exists that it leads to a slow decline in diversity and does not perfectly achieve a rapid decrease in population diversity with the increase in the number of iterations.

4.3. Comparative Study with Swarm Intelligence Algorithm

This part selects five popular metaheuristic algorithms: sine cosine algorithm (SCA) [72], firefly algorithm (FA) [73], whale optimization algorithm (WOA), bat algorithm (BA) [74] moth-flame optimization, and (MFO) [75] to compare with IDARSOA on 20 functions. The main parameter settings of these algorithms are shown in Table 7 below. In the previous part, it has been proved that the variant IDARSOA has better performance than the original SOA, so the next comparative experiment will not add SOA for comparison.

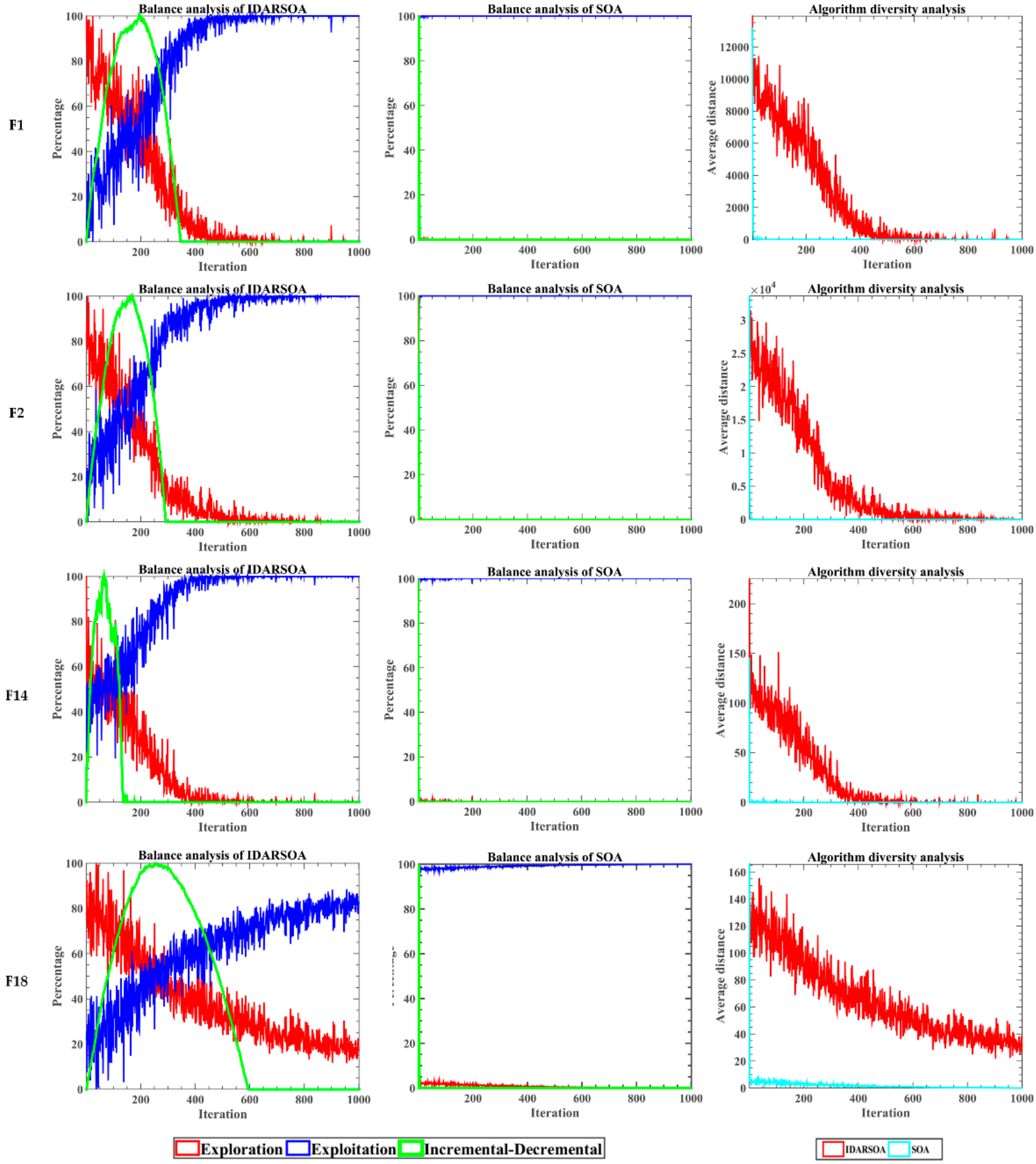

Figure 4. Balance and diversity analysis diagram of IDARSOA and SOA.

Table 7. Parameter settings of original algorithms.

Algorithm	Population Size	Maximum Evaluation Times	Other Parameters
IDARSOA	30	300,000	$fc = 2$; $k \in [1, 30]$; $rd = [0, 1]$; $u = 1$; $v = 1$; $R \in [0, 1]$; $\theta \in [0, 2\pi]$; $\omega_1 = 0.5$; $\omega_2 = 0.4$
SCA	30	300,000	$a = 2$; $r_1 \in [0, 2]$; $r_2 \in [0, 2\pi]$; $r_3 \in [0, 2]$; $r_4 \in [0, 1]$
FA	30	300,000	$alpha = 0.5$; $beta = 0.2$; $gamma = 1$
WOA	30	300,000	$a_1 \in [0, 2]$; $a_2 \in [-2, -1]$; $b = 1$; $p \in [0, 1]$; $r_1 \in [0, 1]$; $r_2 \in [0, 1]$
BA	30	300,000	$A = 0.5$; $r = 0.5$
MFO	30	300,000	$b = 1$; $a \in [-2, -1]$; $t \in [-1, 1]$

To prove the optimized performance of IDARSOA, the following Table 8 shows the average value and standard deviation of the six algorithms, including IDARSOA in F1 to F20. In most functions, the standard deviation of IDARSOA is reasonable and small overall, reflecting the stability and superiority of IDARSOA. In comparison with the five algorithms, IDARSOA ranks first among the six algorithms with ARV = 2.55, which shows the superiority of IDARSOA.

Table 8. Comparison of IDARSOA and original algorithms.

	F1		F2		F3	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.0000×10^{0}	9.9190×10^{-14}	4.4624×10^{0}	2.6850×10^{-1}	2.4698×10^{0}	1.3612×10^{0}
SCA	1.6438×10^{5}	4.7520×10^{5}	1.5949×10^{3}	9.1850×10^{2}	7.3779×10^{0}	1.5579×10^{0}
FA	1.9599×10^{7}	7.4561×10^{6}	4.8321×10^{3}	5.5803×10^{2}	8.7553×10^{0}	3.7347×10^{-1}
WOA	5.8467×10^{5}	1.0555×10^{6}	7.1961×10^{3}	2.7167×10^{3}	2.2456×10^{0}	1.0804×10^{0}
BA	1.7570×10^{8}	1.8006×10^{8}	1.2794×10^{4}	7.2381×10^{3}	9.0701×10^{0}	9.9474×10^{-1}
MFO	7.5475×10^{6}	7.7712×10^{6}	1.8202×10^{3}	2.7692×10^{3}	6.9546×10^{0}	2.1818×10^{0}
	F4		**F5**		**F6**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	2.8324×10^{1}	1.1700×10^{1}	2.0808×10^{0}	7.2777×10^{-1}	5.5185×10^{0}	1.3272×10^{0}
SCA	3.5327×10^{1}	6.6120×10^{0}	5.5044×10^{0}	2.1222×10^{0}	6.2583×10^{0}	1.0812×10^{0}
FA	3.6286×10^{1}	4.4263×10^{0}	9.4106×10^{0}	1.6023×10^{0}	7.4503×10^{0}	4.8768×10^{-1}
WOA	4.8000×10^{1}	1.8752×10^{1}	1.7457×10^{0}	3.2824×10^{-1}	7.0760×10^{0}	1.9836×10^{0}
BA	7.1763×10^{1}	2.2951×10^{1}	1.4952×10^{0}	8.8400×10^{-2}	9.4079×10^{0}	2.0098×10^{0}
MFO	2.8907×10^{1}	9.5872×10^{0}	2.3353×10^{0}	3.9566×10^{0}	4.4310×10^{0}	1.7569×10^{0}
	F7		**F8**		**F9**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.3741×10^{3}	3.6634×10^{2}	3.9526×10^{0}	3.2795×10^{-1}	1.2210×10^{0}	9.9084×10^{-2}
SCA	1.1619×10^{3}	2.1455×10^{2}	3.9556×10^{0}	2.8269×10^{-1}	1.3891×10^{0}	8.2047×10^{-2}
FA	1.1666×10^{3}	1.5428×10^{2}	4.2398×10^{0}	1.4230×10^{-1}	1.6918×10^{0}	9.4065×10^{-2}
WOA	1.1236×10^{3}	3.8056×10^{2}	4.2289×10^{0}	3.5462×10^{-1}	1.3197×10^{0}	1.7161×10^{-1}
BA	1.4699×10^{3}	3.0922×10^{2}	4.5483×10^{0}	2.8625×10^{-1}	1.3975×10^{0}	1.9923×10^{-1}
MFO	1.0700×10^{3}	3.8785×10^{2}	4.4744×10^{0}	2.9561×10^{-1}	1.3434×10^{0}	1.5116×10^{-1}

Table 8. *Cont.*

	F10		F11		F12	
	AVG	STD	AVG	STD	AVG	STD
IDARSOA	2.1250×10^{1}	1.3100×10^{-1}	5.5253×10^{7}	7.9504×10^{7}	2.2202×10^{3}	4.4254×10^{2}
SCA	2.1102×10^{1}	1.0959×10^{0}	4.9127×10^{8}	2.4689×10^{8}	2.1567×10^{3}	1.5194×10^{2}
FA	2.1023×10^{1}	6.0846×10^{-1}	5.6923×10^{8}	1.5375×10^{8}	2.1972×10^{3}	1.5409×10^{2}
WOA	2.1065×10^{1}	8.9335×10^{-2}	2.2785×10^{3}	1.1634×10^{3}	2.0847×10^{3}	3.3080×10^{2}
BA	2.1306×10^{1}	7.8357×10^{-2}	1.0418×10^{5}	5.1233×10^{4}	2.4523×10^{3}	3.1571×10^{2}
MFO	2.1168×10^{1}	1.7784×10^{-1}	7.9222×10^{7}	3.0168×10^{8}	2.0807×10^{3}	3.3741×10^{2}
	F13		**F14**		**F15**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	7.3703×10^{2}	1.2169×10^{1}	1.9000×10^{3}	0.0000×10^{0}	1.8669×10^{6}	9.8014×10^{6}
SCA	7.5499×10^{2}	7.4892×10^{0}	1.9001×10^{3}	5.8440×10^{-1}	8.8925×10^{4}	1.2521×10^{5}
FA	7.9477×10^{2}	8.1099×10^{0}	1.9104×10^{3}	2.1422×10^{0}	1.8894×10^{4}	8.4283×10^{3}
WOA	7.7068×10^{2}	2.3497×10^{1}	1.9000×10^{3}	6.1056×10^{-2}	5.3733×10^{3}	3.3812×10^{3}
BA	8.5547×10^{2}	5.4071×10^{1}	1.9025×10^{3}	1.1437×10^{0}	3.5577×10^{3}	1.1950×10^{3}
MFO	7.3963×10^{2}	1.7116×10^{1}	1.9016×10^{3}	1.6222×10^{0}	8.3721×10^{4}	1.4693×10^{5}
	F16		**F17**		**F18**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.6286×10^{3}	2.4444×10^{1}	2.6228×10^{5}	2.0391×10^{5}	2.2955×10^{3}	1.8224×10^{1}
SCA	1.6233×10^{3}	1.4381×10^{1}	4.4090×10^{3}	1.1815×10^{3}	2.2840×10^{3}	2.4760×10^{1}
FA	1.6511×10^{3}	1.5534×10^{1}	4.5973×10^{3}	1.0688×10^{3}	2.2893×10^{3}	1.0275×10^{1}
WOA	1.7174×10^{3}	6.8250×10^{1}	4.7778×10^{3}	2.2965×10^{3}	2.2982×10^{3}	1.0879×10^{1}
BA	1.8946×10^{3}	1.3756×10^{2}	2.8174×10^{3}	3.1854×10^{2}	2.3172×10^{3}	1.3160×10^{1}
MFO	1.7649×10^{3}	1.1782×10^{2}	3.4778×10^{4}	8.8401×10^{4}	2.2960×10^{3}	1.5197×10^{1}

	F19		F20		+/−/=	ARV	Rank
	AVG	STD	AVG	STD			
IDARSOA	2.6000×10^{3}	0.0000×10^{0}	2.7000×10^{3}	0.0000×10^{0}		2.55	1
SCA	2.8366×10^{3}	6.1747×10^{0}	2.9578×10^{3}	2.2701×10^{1}	11/4/5	3.25	3
FA	2.8317×10^{3}	6.3671×10^{0}	2.9833×10^{3}	1.1435×10^{1}	14/4/2	4.35	5
WOA	2.7221×10^{3}	1.3152×10^{2}	2.9248×10^{3}	7.9355×10^{1}	11/5/4	2.85	2
BA	2.7663×10^{3}	1.1570×10^{2}	2.9257×10^{3}	7.9028×10^{1}	15/3/2	4.65	6
MFO	2.8201×10^{3}	7.2237×10^{0}	2.9526×10^{3}	3.9734×10^{1}	12/5/3	3.35	4

To more clearly show the change of convergence curve of IDARSOA and the other five algorithms under the same experimental conditions, 9 of the 20 functions are selected as follows. These functions are F1, F2, F4, F8, F9, F13, F16, F19, and F20, respectively. It can be seen from Figure 5 that in F1 and F2, IDARSOA converges rapidly and is closer to the optimal value in optimization accuracy than the other five algorithms, which also reflects the advantages of IDARSOA in exploration performance. In F4, F9, and F13, although IDARSOA is not as good as MFO in finding the optimal solution initially, IDARSOA can also find a good optimal value through its continuous exploration. In F19 and F20, IDARSOA is as good as other algorithms in convergence speed, but it is better in finding the optimal value. Overall, IDARSOA shows its advantages in finding the optimal value of the function.

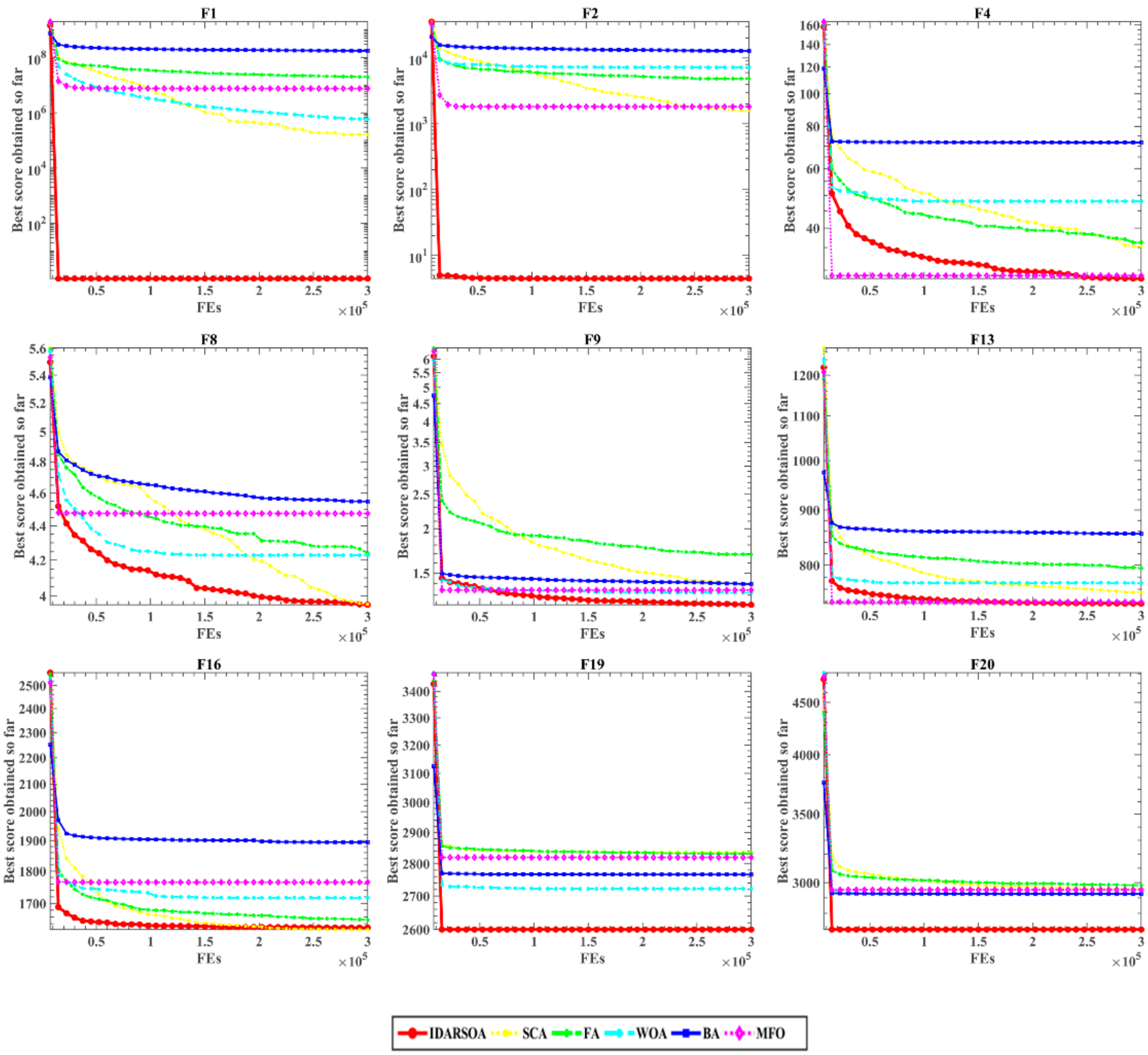

Figure 5. Convergence curve of IDARSOA and original algorithms.

4.4. Comparative Study with Variants of Novel Intelligent Algorithms

In order to verify the effectiveness of IDARSOA, this paper selects CBA [76], FSTPSO [77], CDLOBA [78], PPPSO [79], CESCA [80], CMFO [81], SCAPSO [82], CCMWOA [83], and BSSFOA [84] to compare with IDARSOA. The specific parameter settings in these algorithms are shown in Table 9 below.

Table 10 shows the average value and standard deviation of the optimal solution obtained by IDARSOA and the advanced algorithm in 20 test functions. Among these 10 algorithms, IDARSOA ranks first with an ARV of 3.05. Compared with the PSO variant algorithm with good performance, it is stronger than FSTPSO in 15 functions, PPPSO in 12 functions, and SCAPSO in 7 functions. As a typical algorithm of the WOA variant, CCMWOA ranks third among the 10 algorithms, but it is only stronger than IDARSOA in the four test functions. Among the three functions F14, F19, and F20, IDARSOA, BSSFOA, SCAPSO, and CCMWOA achieved the same optimal value. This shows that IDARSOA, like these three advanced algorithms, can effectively find the best value.

Table 9. Parameter settings of advanced algorithms.

Algorithm	Population Size	Maximum Evaluation Times	Other Parameters
IDARSOA	30	300,000	$fc = 2$; $k \in [1, 30]$; $rd = [0, 1]$; $u = 1$; $v = 1$; $R \in [0, 1]$; $\theta \in [0, 2\pi]$; $\omega_1 = 0.5$; $\omega_2 = 0.4$
CBA	30	300,000	$Q_{min} = 0$; $Q_{max} = 2$
FSTPSO	30	300,000	$V_{max} = 6$; $V_{min} = -6$; $c_1 = 2$; $c_2 = 2$; $\omega = 0.9$
CDLOBA	30	300,000	$Q_{min} = 0$; $Q_{max} = 2$
BSSFOA	30	300,000	$delta = 0.9$; $F_{min} = 0$; $F_{max} = 2$
PPPSO	30	300,000	$V_{max} = 6$; $V_{min} = 0$; $c_1 = 2$; $c_2 = 2$; $k_1 = 0.5$; $k_2 = 0.3$; $kp = 0.45$; $ki = 0.3$; $X = 0.7298$
CESCA	30	300,000	$a = 2$; beta $= 1.5$; ChaosVec$(1) = 0.7$
CMFO	30	300,000	$CC(1) = 0.7$; $b = 1$
SCAPSO	30	300,000	$V_{max} = 4$; $\omega_{max} = 0.9$; $\omega_{min} = 0.4$; $c_1 = 2$; $c_2 = 2$; $a = 2$
CCMWOA	30	300,000	$m = 1500$; $b = 1$

Table 10. Comparison of IDARSOA and other advanced algorithms.

	F1		F2		F3	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.000000×10^{0}	9.503525×10^{-13}	4.484137×10^{0}	3.007712×10^{-1}	2.467507×10^{0}	1.378922×10^{0}
CBA	2.045314×10^{5}	3.169591×10^{5}	7.063138×10^{3}	4.005939×10^{3}	9.473201×10^{0}	1.441603×10^{0}
FSTPSO	5.416835×10^{6}	7.040553×10^{6}	2.823097×10^{3}	1.665675×10^{3}	8.854926×10^{0}	1.712659×10^{0}
CDLOBA	5.366078×10^{8}	3.447411×10^{8}	2.126215×10^{4}	5.968429×10^{3}	8.539816×10^{0}	1.479686×10^{0}
BSSFOA	1.000000×10^{0}	3.955114×10^{-11}	5.000000×10^{0}	2.366243×10^{-6}	5.313317×10^{16}	2.834720×10^{17}
PPPSO	4.063737×10^{7}	3.657041×10^{7}	6.747908×10^{3}	3.576540×10^{3}	4.287855×10^{0}	2.220982×10^{0}
CESCA	1.000000×10^{0}	0.000000×10^{0}	1.209563×10^{3}	7.315292×10^{2}	9.812875×10^{0}	6.840541×10^{-1}
CMFO	2.495589×10^{7}	1.852789×10^{7}	8.484967×10^{3}	3.251125×10^{3}	2.102516×10^{0}	8.270586×10^{-1}
SCAPSO	1.000003×10^{0}	1.283621×10^{-5}	5.000000×10^{0}	0.000000×10^{0}	8.806864×10^{0}	5.061346×10^{-1}
CCMWOA	1.000000×10^{0}	0.000000×10^{0}	5.000000×10^{0}	0.000000×10^{0}	3.963592×10^{0}	1.168292×10^{0}
	F4		**F5**		**F6**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	2.717251×10^{1}	1.254797×10^{1}	2.139273×10^{0}	7.451321×10^{-1}	5.271715×10^{0}	1.476269×10^{0}
CBA	6.694629×10^{1}	2.317729×10^{1}	1.490043×10^{0}	5.325698×10^{-1}	1.073698×10^{1}	1.808086×10^{0}
FSTPSO	5.043442×10^{1}	1.343329×10^{1}	6.034827×10^{0}	3.838452×10^{0}	6.986759×10^{0}	1.597126×10^{0}
CDLOBA	5.746809×10^{1}	2.240174×10^{1}	1.242134×10^{0}	1.954810×10^{-1}	1.043361×10^{1}	1.157089×10^{0}
BSSFOA	1.442075×10^{2}	4.588835×10^{0}	1.668396×10^{2}	2.200394×10^{1}	1.697462×10^{1}	5.541694×10^{-1}
PPPSO	3.839310×10^{1}	1.074413×10^{1}	1.283875×10^{0}	1.518439×10^{-1}	6.462171×10^{0}	1.531601×10^{0}
CESCA	9.448928×10^{1}	9.699659×10^{0}	8.953531×10^{1}	1.596622×10^{1}	1.108156×10^{1}	8.666638×10^{-1}
CMFO	3.749406×10^{1}	1.589558×10^{1}	2.574459×10^{0}	3.660344×10^{0}	7.706524×10^{0}	1.621081×10^{0}
SCAPSO	5.057120×10^{1}	1.642438×10^{1}	1.565520×10^{0}	8.810254×10^{-2}	6.913048×10^{0}	1.777494×10^{0}
CCMWOA	4.986932×10^{1}	1.008877×10^{1}	3.479725×10^{0}	1.161293×10^{0}	7.444709×10^{0}	1.245598×10^{0}

Table 10. *Cont.*

	F7		F8		F9	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.528968×10^{3}	4.727661×10^{2}	3.940307×10^{0}	4.320032×10^{-1}	1.235618×10^{0}	1.113516×10^{-1}
CBA	1.374383×10^{3}	3.420663×10^{2}	4.853054×10^{0}	2.295465×10^{-1}	1.405100×10^{0}	1.721617×10^{-1}
FSTPSO	1.203606×10^{3}	3.716102×10^{2}	4.538657×10^{0}	4.262319×10^{-1}	1.340619×10^{0}	1.231064×10^{-1}
CDLOBA	1.380348×10^{3}	3.537207×10^{2}	4.896797×10^{0}	1.828565×10^{-1}	1.479007×10^{0}	2.249296×10^{-1}
BSSFOA	3.192738×10^{3}	2.412846×10^{2}	5.611770×10^{0}	9.256059×10^{-2}	4.769840×10^{0}	7.678918×10^{-1}
PPPSO	1.318088×10^{3}	2.810489×10^{2}	4.498047×10^{0}	3.964417×10^{-1}	1.249141×10^{0}	3.765369×10^{0}
CESCA	1.993929×10^{3}	1.805498×10^{2}	5.028661×10^{0}	1.188186×10^{-1}	9.121988×10^{-2}	4.013362×10^{-1}
CMFO	1.336276×10^{3}	3.394988×10^{2}	4.722025×10^{0}	2.486458×10^{-1}	1.249141×10^{0}	3.765369×10^{0}
SCAPSO	1.166070×10^{3}	2.627874×10^{2}	4.100872×10^{0}	3.769670×10^{-1}	9.121988×10^{-2}	4.013362×10^{-1}
CCMWOA	1.141992×10^{3}	3.629329×10^{2}	4.476690×10^{0}	3.195163×10^{-1}	1.249141×10^{0}	3.765369×10^{0}
	F10		**F11**		**F12**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	2.128730×10^{1}	1.542929×10^{-1}	2.055593×10^{8}	8.387840×10^{8}	2.167846×10^{3}	5.225523×10^{2}
CBA	2.104854×10^{1}	9.277244×10^{-2}	1.393178×10^{3}	7.324373×10^{2}	2.402761×10^{3}	3.703850×10^{2}
FSTPSO	2.103147×10^{1}	3.766262×10^{-2}	6.699923×10^{8}	5.783327×10^{8}	2.213364×10^{3}	2.199824×10^{2}
CDLOBA	2.128063×10^{1}	7.271713×10^{-2}	2.264579×10^{3}	9.065271×10^{2}	2.415362×10^{3}	2.767366×10^{2}
BSSFOA	2.152992×10^{1}	1.151529×10^{-2}	3.246366×10^{10}	4.869552×10^{9}	4.034769×10^{3}	1.913518×10^{2}
PPPSO	2.109848×10^{1}	6.506809×10^{-2}	4.246755×10^{5}	1.279981×10^{10}	2.158117×10^{3}	3.566239×10^{2}
CESCA	2.151104×10^{1}	1.304537×10^{-1}	2.316693×10^{6}	2.045188×10^{9}	2.885617×10^{3}	1.487243×10^{2}
CMFO	2.129575×10^{1}	2.345907×10^{-1}	4.246755×10^{5}	1.279981×10^{10}	2.325639×10^{3}	3.842820×10^{2}
SCAPSO	2.129200×10^{1}	8.592595×10^{-2}	2.316693×10^{6}	2.045188×10^{9}	2.188654×10^{3}	2.606792×10^{2}
CCMWOA	2.074818×10^{1}	1.984757×10^{0}	4.246755×10^{5}	1.279981×10^{10}	2.066235×10^{3}	2.230539×10^{2}
	F13		**F14**		**F15**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	7.365436×10^{2}	1.047151×10^{1}	1.900000×10^{3}	0.000000×10^{0}	1.905660×10^{6}	9.794928×10^{6}
CBA	9.157041×10^{2}	8.172201×10^{1}	1.909141×10^{3}	3.830691×10^{0}	4.250929×10^{3}	2.427470×10^{3}
FSTPSO	7.607505×10^{2}	1.897089×10^{1}	1.903777×10^{3}	2.134699×10^{0}	1.443403×10^{4}	5.269551×10^{4}
CDLOBA	9.592444×10^{2}	8.807963×10^{1}	1.908928×10^{3}	5.950454×10^{0}	4.725657×10^{3}	2.539698×10^{3}
BSSFOA	8.771444×10^{2}	1.008231×10^{1}	1.900000×10^{3}	0.000000×10^{0}	1.373129×10^{7}	2.834251×10^{7}
PPPSO	7.532335×10^{2}	1.710264×10^{1}	1.901063×10^{3}	5.062910×10^{-1}	1.011323×10^{4}	1.127936×10^{4}
CESCA	8.456961×10^{2}	1.331773×10^{1}	1.900604×10^{3}	7.846227×10^{-1}	1.154661×10^{6}	6.668748×10^{5}
CMFO	7.614705×10^{2}	2.633592×10^{1}	1.903336×10^{3}	3.169298×10^{0}	8.447643×10^{4}	4.428584×10^{5}
SCAPSO	7.526187×10^{2}	9.735354×10^{0}	1.900000×10^{3}	0.000000×10^{0}	4.110251×10^{3}	2.153134×10^{3}
CCMWOA	7.666253×10^{2}	2.147281×10^{1}	1.900000×10^{3}	0.000000×10^{0}	3.088870×10^{4}	6.547435×10^{4}
	F16		**F17**		**F18**	
	AVG	**STD**	**AVG**	**STD**	**AVG**	**STD**
IDARSOA	1.625281×10^{3}	1.773744×10^{1}	2.026475×10^{5}	2.060346×10^{5}	2.297973×10^{3}	1.264591×10^{1}
CBA	1.870947×10^{3}	1.636097×10^{2}	3.128701×10^{3}	5.301465×10^{2}	2.300030×10^{3}	4.910720×10^{-2}
FSTPSO	1.807712×10^{3}	1.274306×10^{2}	4.046043×10^{3}	2.391059×10^{3}	2.347481×10^{3}	1.181476×10^{0}
CDLOBA	1.883110×10^{3}	1.966674×10^{2}	3.480247×10^{3}	1.252334×10^{3}	2.301347×10^{3}	1.185150×10^{0}
BSSFOA	2.498101×10^{3}	1.056882×10^{1}	3.893858×10^{7}	1.048266×10^{8}	2.335064×10^{3}	4.808892×10^{-2}
PPPSO	1.782462×10^{3}	1.091577×10^{2}	3.608611×10^{3}	1.151662×10^{3}	2.302283×10^{3}	1.167129×10^{0}
CESCA	1.811594×10^{3}	1.063348×10^{2}	3.837579×10^{5}	2.882279×10^{5}	2.341942×10^{3}	4.403841×10^{0}
CMFO	1.759178×10^{3}	1.097689×10^{2}	3.918881×10^{3}	2.766490×10^{3}	2.305250×10^{3}	2.638451×10^{1}
SCAPSO	1.743337×10^{3}	8.170653×10^{1}	2.972308×10^{3}	3.940947×10^{2}	2.338823×10^{3}	2.257253×10^{0}
CCMWOA	1.736167×10^{3}	1.335177×10^{2}	5.400884×10^{3}	2.830385×10^{3}	2.301539×10^{3}	4.067334×10^{-1}

Table 10. *Cont.*

	F19		F20		+/−/=	ARV	RANK
	AVG	STD	AVG	STD			
IDARSOA	2.600000×10^3	0.000000×10^0	2.700000×10^3	0.000000×10^0		3.05	1
CBA	2.737212×10^3	1.384116×10^2	2.986157×10^3	5.742234×10^1	13/4/3	6.15	7
FSTPSO	2.734298×10^3	9.769952×10^1	2.971398×10^3	2.729669×10^1	15/3/2	5.95	6
CDLOBA	2.805355×10^3	9.421156×10^1	2.980184×10^3	6.246217×10^1	14/3/3	6.7	8
BSSFOA	2.600000×10^3	0.000000×10^0	2.700000×10^3	0.000000×10^0	17/0/3	7.8	10
PPPSO	2.645155×10^3	1.051386×10^2	2.926862×10^3	5.154461×10^1	12/4/4	4.3	4
CESCA	2.613790×10^3	7.941062×10^0	2.750519×10^3	3.357604×10^1	18/2/0	7.55	9
CMFO	2.804255×10^3	1.237585×10^2	2.962768×10^3	3.912437×10^1	13/1/6	5.9	5
SCAPSO	2.600000×10^3	0.000000×10^0	2.700000×10^3	0.000000×10^0	7/4/9	3.2	2
CCMWOA	2.600000×10^3	0.000000×10^0	2.700000×10^3	0.000000×10^0	11/4/5	3.4	3

In order to clearly and intuitively understand the convergence of IDARSOA with the advanced algorithm, the following Figure 6 shows the convergence effect plots compared with the advanced algorithm. The convergence plots of nine test functions are selected in the figure, namely F2, F4, F6, F8, F13, F16, F18, F19, and F20. In F4, F6, and F8, the advantages of IDARSOA's optimization ability in these three functions are obviously displayed. IDARSOA gradually enters the state of convergence only in the late iteration, which is due to the addition of the individual perturbation strategy, the search solution is influenced by random individuals, which reduces the risk of falling into local optimum and enhances the exploration ability, but this also leads to the problem that IDARSOA converges slower than other algorithms. In F9 and F20, as the data in the above table show, IDARSOA, BSSFOA, SCAPSO, and CCMWOA obtain the same optimal values, so the curves of these algorithms overlap together in the figure. Owing to the great potential of the proposed method, in the future, it can also be extended to tackle other practical problems, such as medical diagnosis [85–88], microgrid planning [89], engineering optimization problems [31,33], energy storage planning and scheduling [90], active surveillance [91], kayak cycle phase segmentation [92], location-based services [93,94], image dehazing [95], information retrieval services [96–98], human motion capture [99], and video deblurring [100].

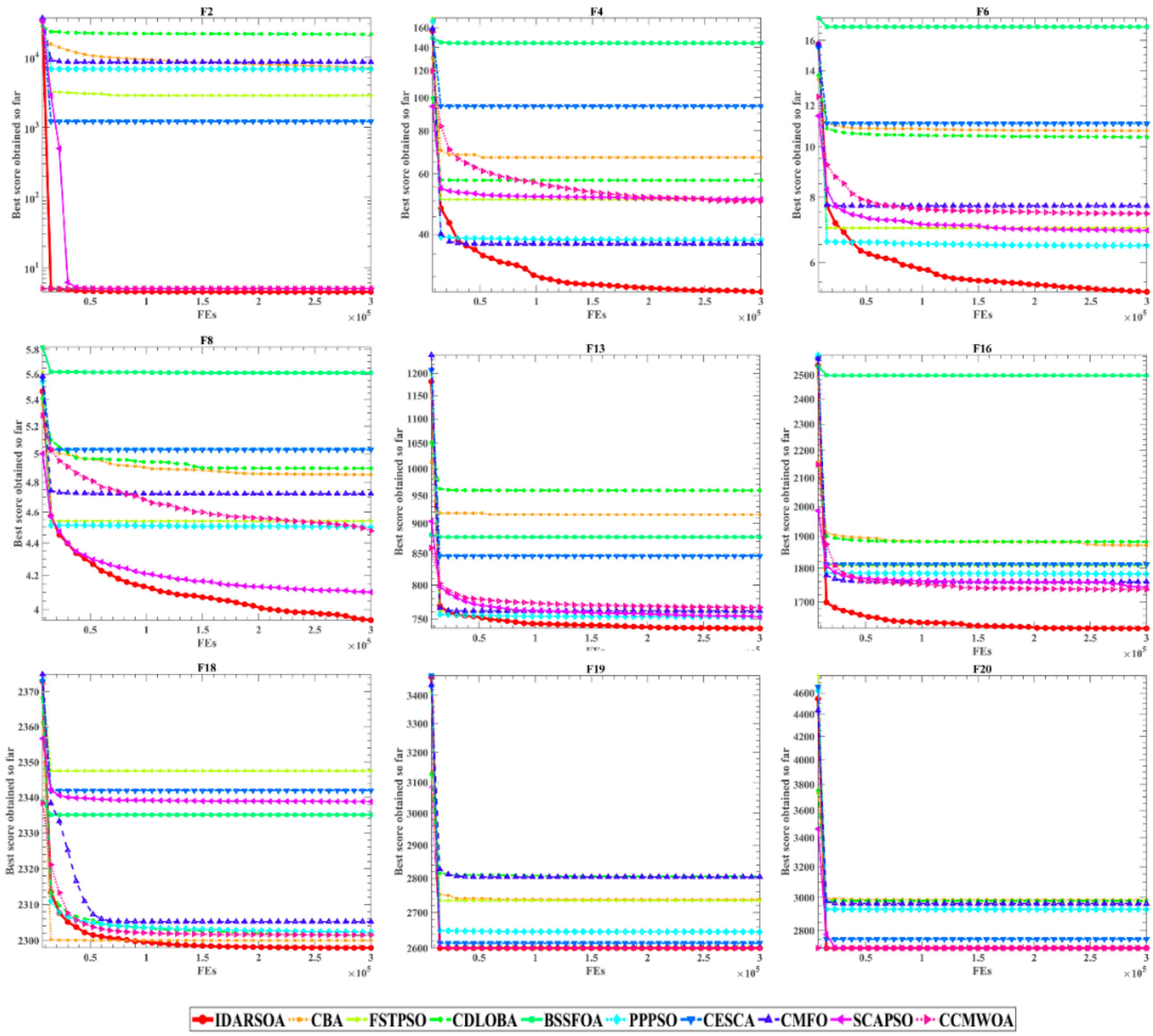

Figure 6. Convergence curve of IDARSOA and advanced algorithms.

5. Engineering Design Issues

In this section, the performance of IDARSOA is verified on six well-known engineering design optimization problems, including tension/compression spring, pressure vessels, I-beam, speed reducer, welded beam, and three-bar truss design problems. It is worth noting that the optimal solution to be obtained has many constraints that should not be violated [62].

5.1. Tension-Compression String Problem

This problem aims to design a tension/compression spring with the smallest weight while satisfying the constraints. In this model, the design parameters are wire diameter (*d*), average coil diameter (*D*), and effective coil number (*N*). The specific model is as follows:

Consider

$$\vec{x} = [x_1\ x_2\ x_3\ ] = [d\ D\ N]$$

Minimize

$$f\left(\vec{x}\right) = x_1^2 x_2 (x_3 + 2)$$

Subject to

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

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

$$g_3\left(\vec{x}\right) = 1 - \frac{140.45x_1}{x_2^2 x_3} \leq 0$$

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

Variable range:

$$0.05 \leq x_1 \leq 2,\ 0.25 \leq x_2 \leq 1.3,\ 2 \leq x_3 \leq 15$$

The IDARSOA and other algorithms were applied to optimize the tension/compression spring design problem, and the results are shown in Table 11. IDARSOA and the other 10 algorithms are applied to the same problem; IDARSOA and DE get the lowest optimization cost at 0.012670, which shows the enhancement effect of the proposed IDARSOA in practical engineering applications.

Table 11. Comparison results of the tension-compression string problem.

Algorithm	Optimal Values for Variables			Optimum Cost
	d	***D***	***N***	
IDARSOA	0.051960	0.363240	10.91947	0.012670
DE	0.051609	0.354714	11.41083	0.012670
Improved HS [101]	0.051154	0.349871	12.07643	0.012671
PSO [102]	0.051728	0.357644	11.24454	0.012675
WOA [2]	0.051207	0.345215	12.00430	0.012676
RO [103]	0.051370	0.349096	11.76279	0.012679
ES [104]	0.051989	0.363965	10.89052	0.012681
GSA [105]	0.050276	0.323680	13.52541	0.012702
GA [106]	0.051480	0.351661	11.63220	0.012705
Mathematical optimization	0.053396	0.399180	9.185400	0.012730
Constraint correction	0.050000	0.315900	14.25000	0.012833

5.2. Pressure Vessel Design Problem

For the design of cylindrical pressure vessels, the main difficulty is to reduce the manufacturing cost while meeting the four parameters of the pressure vessel, namely, the thickness of the head (T_h), the inner radius (R), the thickness of the shell (T_s), and the cross-sectional range minus the head (L). The model can be described as:

Consider

$$\vec{x} = [x_1\ x_2\ x_3\ x_4\,] = [T_s\ T_h\ R\ L]$$

Objective:

$$f\left(\vec{x}\right)_{min} = 0.6224x_1 x_3 x_4 + 1.7781x_3 x_1^2 + 3.1661x_4 x_1^2 + 19.84x_3 x_1^2$$

Subject to

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

$$g_2\left(\vec{x}\right) = -x_3 + 0.00954x_3 \leq 0$$

$$g_3\left(\vec{x}\right) = -\pi x_4 x_3^2 - \frac{4}{3}\pi x_3^3 + 1{,}296{,}000 \leq 0$$

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

Variable ranges:

$$0 \leq x_1 \leq 99, 0 \leq x_2 \leq 99,\ 10 \leq x_3 \leq 200,\ 10 \leq x_4 \leq 200$$

Applying IDARSOA and several other algorithms to this engineering problem, the results obtained are shown in Table 12. It can be seen from the data that IDARSOA ranks second among these algorithms at the cost of 6072.4301, which indicates that IDARSOA has a good effect in optimizing the design of pressure vessels.

Table 12. Comparison results of pressure vessel design issues.

Algorithm	Optimal Values for Variables				Optimum Cost
	T_s	T_h	R	L	
PSO (He et al.)	0.812500	0.437500	42.091266	176.746500	6061.0777
IDARSOA	0.812500	0.4375	42.09711	177.1901	6072.4301
GA [106]	0.93750	0.500000	48.32900	112.6790	6410.381
Lagrangian multiplier [107]	1.12500	0.625000	58.29100	43.69000	7198.043
BA [74]	98.80150	98.10897	10.98606	200.0000	7258.564
Branch-and-bound [108]	1.12500	0.625000	47.70000	117.7100	8129.104
GSA [105]	1.125000	0.625000	55.988659	84.4542025	8538.8359

5.3. I-Beam Design Problem

The goal of the structural design problem of the I-steel is to minimize vertical deflection. The problem involves four structural parameters: two thicknesses, one length, and one height. The specific problem model is as follows:

Consider:

$$\vec{x} = [x_1\ x_2\ x_3\ x_4] = \left[b\ h\ t_w\ t_f\right]$$

The value range of the four parameters:

$$10 \leq x_1 \leq 50$$

$$10 \leq x_2 \leq 80$$

$$0.9 \leq x_3 \leq 5$$

$$0.9 \leq x_4 \leq 5$$

Minimize:

$$f\left(\vec{x}\right) = \frac{5000}{\frac{t_w\left(h-2t_f\right)^3}{12} + \frac{bt_f^3}{6} + 2bt_f\left(\frac{h-t_f}{2}\right)^2}$$

Subject to:

$$g\left(\vec{x}\right) = 2bt_f + t_w\left(h - 2t_f\right) \leq 0$$

$$g_1\left(\vec{x}\right) = \frac{18h \times 10^4}{t_w\left(h - 2t_f\right)^3 + 2bt_f\left(4t_f + 3h\left(h - 2t_f\right)\right)} + \frac{15b \times 10^3}{\left(h - 2t_f\right)t_w{}^3 + 2t_f b^3} - 6 \leq 0$$

The results of the IDARSOA and other six algorithms to the I-beam design problem are shown in the following Table 13. It can be seen from the data in the table that IDARSOA and SOS can effectively solve this problem at the same time.

Table 13. Comparison results of I-beam problem.

Algorithm	Optimum Variables				Optimum Cost
	b	h	t_w	t_f	
IDARSOA	50.0000	80.0000	0.9000	2.321769	0.013074
SOS [109]	50.0000	80.0000	0.9000	2.3218	0.013074
CS [110]	50.0000	80.0000	0.9000	2.3217	0.013075
AGOA [111]	43.12663	79.91247	0.932602	2.671865	0.013295
ARSM [112]	37.0500	80.0000	1.7100	2.3100	0.015700
IARSM [112]	48.4200	79.9900	0.9000	2.4000	0.131000

5.4. Speed Reducer Design Problem

The premise of the problem is to minimize the weight of the speed reducer while satisfying each parameter in the engineering design model within the valid range. The parameters involved: x_1 is the face width (b), x_2 is the tooth mode (m), x_3 is the number of gear teeth (z), x_4 is the length of the first shaft between bearings (l_1), x_5 is the length of the second shaft between bearings (l_2), x_6 is the diameter first (d_1), and x_7 is the second shaft (d_2). The specific mathematical model is shown below.

Consider

$$\vec{z} = [x_1 \ \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7\ ] = [b \ m \ z \ l_1 \ l_2 \ d_1 \ d_2],$$

Minimize $f\left(\vec{x}\right) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^2 + x_7^2\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right).$

Subject to:

$$g_1\left(\vec{x}\right) = \frac{27}{x_1x_3x_2^2} - 1 \leq 0$$

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

$$g_3\left(\vec{x}\right) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \leq 0$$

$$g_4\left(\vec{x}\right) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \leq 0$$

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

$$g_6\left(\vec{x}\right) = \frac{\left[(745(x_5/x_2x_3))^2 + 157.5 \times 10^6\right]^{1/2}}{85x_7^3} - 1 \leq 0$$

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

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

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

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

$$g_{11}\left(\vec{x}\right) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0$$

where

$$2.6 \leq x_1 \leq 3.6,\ 0.7 \leq x_2 \leq 0.8,\ 17 \leq x_3 \leq 28,\ 7.3 \leq x_4 \leq 28,\ 7.3 \leq x_5 \leq 8.3,\ 2.9 \leq x_6 \leq 3.9,\ 5.0 \leq x_7 \leq 5.5$$

As shown from the data in Table 14 below, IDARSOA performs well in this problem, proving its advantage in solving constrained problems. The advantage is outstanding compared to other hHHO-SCA, SCA, and GSA.

Table 14. Comparison results of speed reducer design problem.

Algorithm	Optimal Values for Variables							Optimum Cost
	b	m	z	l_1	l_2	d_1	d_2	
IDARSOA	3.50608	0.7	17	7.3	7.719262	3.353154	5.288364	2998.7797
PSO [102]	3.50001	0.7	17	8.3	7.8	3.352412	5.286715	3005.7630
hHHO-SCA [113]	3.56061	0.7	17	7.3	7.991410	3.452569	5.286749	3029.8731
SCA [72]	3.50875	0.7	17	7.3	7.8	3.461020	5.289213	3030.5630
GSA [105]	3.6	0.7	17	8.3	7.8	3.369658	5.289224	3051.1200

5.5. Welded Beam Design Problem

The objective of this engineering problem is to reduce the manufacturing cost of a welded beam, where the variables involved are: welding seam thickness (h), welding joint length (l), beam width (t), beam thickness (b). A detailed model is shown below.

Consider

$$\vec{x} = [x_1, x_2, x_3, x_4] = [\,h\ l\ t\ b]$$

Minimize

$$f\left(\vec{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_4)$$

Subject to

$$g_1\left(\vec{x}\right) = \tau\left(\vec{x}\right) - \tau_{\max} \leq 0$$

$$g_2\left(\vec{x}\right) = \sigma\left(\vec{x}\right) - \sigma_{\max} \leq 0$$

$$g_3\left(\vec{x}\right) = \delta\left(\vec{x}\right) - \delta_{\max} \leq 0$$

$$g_4\left(\vec{x}\right) = x_1 - x_4 \leq 0$$

$$g_5\left(\vec{x}\right) = P - P_C\left(\vec{x}\right) \leq 0$$

$$g_6\left(\vec{x}\right) = 0.125 - x_1 \leq 0$$

$$g_7\left(\vec{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0$$

Variable range $0.1 \leq x_1 \leq 2$, $0.1 \leq x_2 \leq 10$, $0.1 \leq x_3 \leq 10$, $0.1 \leq x_4 \leq 2$

where

$$\tau\left(\vec{x}\right) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{\mathrm{MR}}{J}, M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\sigma\left(\vec{x}\right) = \frac{6PL}{x_4 x_3{}^2},\ \delta\left(\vec{x}\right) = \frac{6PL^3}{Ex_3^2 x_4}$$

$$P_C\left(\vec{x}\right) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$P = 60,001b,\ L = 14,\ \delta_{max} = 0.25$$

$$E = 30 \times 1^6 \text{ psi}, G = 12 \times 10^6 \text{ psi}$$

$$\tau_{max} = 13,600 \text{ psi},\ \sigma_{max} = 30,000 \text{ psi}$$

IDARSOA for this problem has an inferior performance to EO and RO methods when solving the same problem with other algorithms. However, it has advantages compared with HS, FSA, SCA, and SBM (see Table 15).

Table 15. Comparison results of the welded beam design problem.

Algorithm	Optimal Values for Variables				Optimum Cost
	h	***l***	***t***	***b***	
EO [114]	0.2057	3.4705	9.03664	0.2057	1.7249
RO [103]	0.203687	3.528467	9.004233	0.207241	1.735344
IDARSOA	0.2275	5.8045	8.261455	0.247557	2.280517
HS [115]	0.244200	6.223100	8.291500	0.243300	2.380700
FSA [116]	0.244356	6.125792	8.293905	0.244356	2.38119
SCA [117]	0.244438	6.237967	8.288576	0.244566	2.385435
SBM [118]	0.2407	6.4851	8.2399	0.2497	2.4426

5.6. Three-Bar Truss Design Problem

The three-bar truss design problem is a typical constrained engineering problem that requires obtaining a smaller weight while satisfying two parameters x_1, x_2. The specific mathematical model is as follows.

Objective function:

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

Subject to:

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

$$g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1 x_2}P - \sigma \le 0$$

$$g_2(x) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \le 0$$

where

$$0 \le x_i \le 1 \ \ i = 1,\ 2$$

$$L = 100 \text{ cm}, P = 2 \text{ kN/cm}^2, \sigma = 2 \text{ kN/cm}^2$$

IDARSOA is compared with the four metaheuristic algorithms, and its minimum weight obtained for solving this problem is 263.8960. As shown in Table 16, IDARSOA has an advantage in solving the problem and can handle the three-bar truss design problem well.

Table 16. Comparison results of the three-bar truss design problem.

Algorithm	Optimal Values for Variables		Optimum Cost
	x_1	x_2	
IDARSOA	0.788906	0.40760	263.8960
GWO [119]	0.79477	0.39192	263.987
SCADE [120]	0.73942	0.5691	266.0501
RCBA [79]	0.56544	0.64079	266.6156
ALCPSO [121]	0.999924	0.000108	282.8427

6. Conclusions and Future Works

The IDARSOA proposed in this paper is designed to overcome the lack of search ability of the original SOA. When seagulls look for the optimal migration direction, the individual disturbance mechanism is added to enhance the ability to jump out of the local optimum through the disturbance of seagulls in different individual positions. At the same time, the attraction-repulsion strategy is introduced to guide the seagulls to move towards the position of the optimal solution. The combination of these two mechanisms improves the optimization accuracy of the algorithm, makes up for the lack of search ability of the original algorithm, enhances the diversity of the population, and makes the process of exploring the solution space more comprehensive. Data results of 20 representative benchmark functions show that the performance of this optimization algorithm is significantly improved compared with the original SOA, and it can effectively solve the function optimization problem. In the application of IDARSOA to six engineering examples, there are sound effects which can be a good solution to the actual engineering problems, and shows that IDARSOA can improve the accuracy of the calculation results and has a certain practical value.

Although our proposed method effectively improves the optimization performance of SOA, IDARSOA takes more time to complete in dealing with complex and large-scale problems. Therefore, we will consider combining IDARSOA with distributed platforms, such as Hadoop, to improve its parallel performance and speed up the time to solve real industrial environment problems. In addition, there are still many problems worthy of further study. On the one hand, IDARSOA suffers from the problem of slow convergence. In the next stage of research, we consider balancing the relationship between population diversity and the number of iterations by adding complementary strategies to speed up the convergence trend of IDARSOA while ensuring that it has an affluent population. At the same time, under the core idea of SOA, how to enrich the algorithm model and improve the algorithm performance so that the improved SOA has the same superior performance as SASS [122], COLSHADE [123], and CMA-ES [124] algorithms are also the critical research contents in our subsequent work. On the other hand, our goal is to better integrate optimized SOA into real-life problems and make full use of the advantages of SOA. Due to the good performance of IDARSOA in functions, we plan to combine IDARSOA with machine learning to solve more complex real-world problems. Then IDARSOA will be applied to other scenarios, such as for image enhancement optimization, image segmentation and classification, and handling dynamic landscapes. Moreover, learning techniques can be used to further boost the proposed method [5,125,126], and the proposed method can also be extended to the multi/many-objective optimization algorithms [127–131].

Author Contributions: Funding acquisition, H.C.; Investigation, C.B.; Methodology, A.A.H.; Writing—original draft, H.Y.; Writing—review & editing, S.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Development Program of Jilin Province (20190301024NY).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Description of the benchmark functions.

NO.	Functions	Dim	F (min)
CEC2019 benchmark functions			
F1	Storn's Chebyshev Polynomial Fitting Problem	9	1
F2	Inverse Hilbert Matrix Problem	16	1
F3	Lennard-Joes Minimum Energy Cluster	18	1
F4	Rastrigin's Function	10	1
F5	Griewangk's Function	10	1
F6	Weierstrass Function	10	1
F7	Modified Schwefel's Function	10	1
F8	Expand Schaffer's F6 function	10	1
F9	Happy Cat Function	10	1
F10	Ackley Function	10	1
CEC2020 benchmark functions			
F11	Shifted and Rotated Bent Cigar Function (CEC2017 F1)	30	100
F12	Shifted and Rotated Schwefel's Function (CEC2014 F11)	30	1100
F13	Shifted and Rotated Lunacek bi-Rastrigin Function (CEC2017 F7)	30	700
F14	Expanded Rosenbrock's plus Griewangk's Function (CEC2017 F19)	30	1900
F15	Hybrid Function1 (n = 3) (CEC2014 F17)	30	1700
F16	Hybrid Function2 (n = 4) (CEC2017 F16)	30	1600
F17	Hybrid Function3 (n = 5) (CEC2014 F21)	30	2100
F18	Composition Function1 (n = 3) (CEC2017 F22)	30	2200
F19	Composition Function2 (n = 4) (CEC2017 F24)	30	2400
F20	Composition Function3 (n = 5) (CEC2017 F25)	30	2500

References

1. Beni, G.; Wang, J. Swarm intelligence in cellular robotic systems. In *Robots and Biological Systems: Towards a New Bionics?* Springer: Cham, Switzerland, 1993; pp. 703–712.
2. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
3. Yang, Y.; Chen, H.; Heidari, A.A.; Gandomi, A.H. Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Syst. Appl.* **2021**, *177*, 114864. [CrossRef]
4. Tu, J.; Chen, H.; Wang, M.; Gandomi, A.H. The Colony Predation Algorithm. *J. Bionic Eng.* **2021**, *18*, 674–710. [CrossRef]
5. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Gener. Comput. Syst.* **2020**, *111*, 300–323. [CrossRef]
6. Ahmadianfar, I.; Asghar Heidari, A.; Gandomi, A.H.; Chu, X.; Chen, H. RUN Beyond the Metaphor: An Efficient Optimization Algorithm Based on Runge Kutta Method. *Expert Syst. Appl.* **2021**, *181*, 115079. [CrossRef]
7. Piri, J.; Mohapatra, P. An analytical study of modified multi-objective Harris Hawk Optimizer towards medical data feature selection. *Comput. Biol. Med.* **2021**, *135*, 104558. [CrossRef] [PubMed]
8. Chen, M.-R.; Huang, Y.-Y.; Zeng, G.-Q.; Lu, K.-D.; Yang, L.-Q. An improved bat algorithm hybridized with extremal optimization and Boltzmann selection. *Expert Syst. Appl.* **2021**, *175*, 114812. [CrossRef]
9. Tang, C.; Zhou, Y.; Tang, Z.; Luo, Q. Teaching-learning-based pathfinder algorithm for function and engineering optimization problems. *Appl. Intell.* **2021**, *51*, 5040–5066. [CrossRef]
10. Zhong, L.; Zhou, Y.; Luo, Q.; Zhong, K. Wind driven dragonfly algorithm for global optimization. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6054. [CrossRef]
11. Salgotra, R.; Singh, U.; Singh, S.; Singh, G.; Mittal, N. Self-adaptive salp swarm algorithm for engineering optimization problems. *Appl. Math. Model.* **2021**, *89*, 188–207. [CrossRef]
12. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
13. Kapoor, S.; Zeya, I.; Singhal, C.; Nanda, S.J. A Grey Wolf Optimizer Based Automatic Clustering Algorithm for Satellite Image Segmentation. *Procedia Comput. Sci.* **2017**, *115*, 415–422. [CrossRef]

14. Seyyedabbasi, A.; Kiani, F. I-GWO and Ex-GWO: Improved algorithms of the Grey Wolf Optimizer to solve global optimization problems. *Eng. Comput.* **2021**, *37*, 509–532. [CrossRef]
15. Yang, Z.; Li, K.; Guo, Y.; Ma, H.; Zheng, M. Compact real-valued teaching-learning based optimization with the applications to neural network training. *Knowl.-Based Syst.* **2018**, *159*, 51–62. [CrossRef]
16. Fan, C.; Hu, K.; Feng, S.; Ye, J.; Fan, E. Heronian mean operators of linguistic neutrosophic multisets and their multiple attribute decision-making methods. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719843059. [CrossRef]
17. Cui, W.-H.; Ye, J. Logarithmic similarity measure of dynamic neutrosophic cubic sets and its application in medical diagnosis. *Comput. Ind.* **2019**, *111*, 198–206. [CrossRef]
18. Fan, C.; Fan, E.; Hu, K. New form of single valued neutrosophic uncertain linguistic variables aggregation operators for decision-making. *Cogn. Syst. Res.* **2018**, *52*, 1045–1055. [CrossRef]
19. Ye, J.; Cui, W. Modeling and stability analysis methods of neutrosophic transfer functions. *Soft Comput.* **2020**, *24*, 9039–9048. [CrossRef]
20. Lai, X.; Zhou, Y. Analysis of multiobjective evolutionary algorithms on the biobjective traveling salesman problem (1, 2). *Multimed. Tools Appl.* **2020**, *79*, 30839–30860. [CrossRef]
21. Hu, K.; Ye, J.; Fan, E.; Shen, S.; Huang, L.; Pi, J. A novel object tracking algorithm by fusing color and depth information based on single valued neutrosophic cross-entropy. *J. Intell. Fuzzy Syst.* **2017**, *32*, 1775–1786. [CrossRef]
22. Hu, K.; He, W.; Ye, J.; Zhao, L.; Peng, H.; Pi, J. Online Visual Tracking of Weighted Multiple Instance Learning via Neutrosophic Similarity-Based Objectness Estimation. *Symmetry* **2019**, *11*, 832. [CrossRef]
23. Zhao, D.; Liu, L.; Yu, F.; Heidari, A.A.; Wang, M.; Liang, G.; Muhammad, K.; Chen, H. Chaotic random spare ant colony optimization for multi-threshold image segmentation of 2D Kapur entropy. *Knowl.-Based Syst.* **2020**, *216*, 106510. [CrossRef]
24. Zhao, D.; Liu, L.; Yu, F.; Asghar Heidari, A.; Wang, M.; Oliva, D.; Muhammad, K.; Chen, H. Ant Colony Optimization with Horizontal and Vertical Crossover Search: Fundamental Visions for Multi-threshold Image Segmentation. *Expert Syst. Appl.* **2020**, *167*, 114122. [CrossRef]
25. Zhang, Y.; Liu, R.; Wang, X.; Chen, H.; Li, C. Boosted binary Harris hawks optimizer and feature selection. *Eng. Comput.* **2020**, *37*, 3741–3770. [CrossRef]
26. Hu, J.; Chen, H.; Heidari, A.A.; Wang, M.; Zhang, X.; Chen, Y.; Pan, Z. Orthogonal learning covariance matrix for defects of grey wolf optimizer: Insights, balance, diversity, and feature selection. *Knowl.-Based Syst.* **2021**, *213*, 106684. [CrossRef]
27. Zhang, X.; Xu, Y.; Yu, C.; Heidari, A.A.; Li, S.; Chen, H.; Li, C. Gaussian mutational chaotic fruit fly-built optimization and feature selection. *Expert Syst. Appl.* **2020**, *141*, 112976. [CrossRef]
28. Li, Q.; Chen, H.; Huang, H.; Zhao, X.; Cai, Z.; Tong, C.; Liu, W.; Tian, X. An Enhanced Grey Wolf Optimization Based Feature Selection Wrapped Kernel Extreme Learning Machine for Medical Diagnosis. *Comput. Math. Methods Med.* **2017**, *2017*, 9512741. [CrossRef] [PubMed]
29. Liu, T.; Hu, L.; Ma, C.; Wang, Z.-Y.; Chen, H.-L. A fast approach for detection of erythemato-squamous diseases based on extreme learning machine with maximum relevance minimum redundancy feature selection. *Int. J. Syst. Sci.* **2015**, *46*, 919–931. [CrossRef]
30. Gupta, S.; Deep, K.; Heidari, A.A.; Moayedi, H.; Chen, H. Harmonized salp chain-built optimization. *Eng. Comput.* **2019**, *37*, 1049–1079. [CrossRef]
31. Ba, A.F.; Huang, H.; Wang, M.; Ye, X.; Gu, Z.; Chen, H.; Cai, X. Levy-based antlion-inspired optimizers with orthogonal learning scheme. *Eng. Comput.* **2020**, 1–22. [CrossRef]
32. Zhang, H.; Cai, Z.; Ye, X.; Wang, M.; Kuang, F.; Chen, H.; Li, C.; Li, Y. A multi-strategy enhanced salp swarm algorithm for global optimization. *Eng. Comput.* **2020**, 1–27. [CrossRef]
33. Liang, X.; Cai, Z.; Wang, M.; Zhao, X.; Chen, H.; Li, C. Chaotic oppositional sine–cosine method for solving global optimization problems. *Eng. Comput.* **2020**, 1–17. [CrossRef]
34. Pang, J.; Zhou, H.; Tsai, Y.-C.; Chou, F.-D. A scatter simulated annealing algorithm for the bi-objective scheduling problem for the wet station of semiconductor manufacturing. *Comput. Ind. Eng.* **2018**, *123*, 54–66. [CrossRef]
35. Zhou, H.; Pang, J.; Chen, P.-K.; Chou, F.-D. A modified particle swarm optimization algorithm for a batch-processing machine scheduling problem with arbitrary release times and non-identical job sizes. *Comput. Ind. Eng.* **2018**, *123*, 67–81. [CrossRef]
36. Hu, L.; Li, H.; Cai, Z.; Lin, F.; Hong, G.; Chen, H.; Lu, Z. A new machine-learning method to prognosticate paraquat poisoned patients by combining coagulation, liver, and kidney indices. *PLoS ONE* **2017**, *12*, e0186427. [CrossRef] [PubMed]
37. Li, C.; Hou, L.; Sharma, B.Y.; Li, H.; Chen, C.; Li, Y.; Zhao, X.; Huang, H.; Cai, Z.; Chen, H. Developing a new intelligent system for the diagnosis of tuberculous pleural effusion. *Comput. Methods Programs Biomed.* **2018**, *153*, 211–225. [CrossRef]
38. Zhao, X.; Zhang, X.; Cai, Z.; Tian, X.; Wang, X.; Huang, Y.; Chen, H.; Hu, L. Chaos enhanced grey wolf optimization wrapped ELM for diagnosis of paraquat-poisoned patients. *Comput. Biol. Chem.* **2019**, *78*, 481–490. [CrossRef]
39. Huang, H.; Zhou, S.; Jiang, J.; Chen, H.; Li, Y.; Li, C. A new fruit fly optimization algorithm enhanced support vector machine for diagnosis of breast cancer based on high-level features. *BMC Bioinform.* **2019**, *20*, 290. [CrossRef]
40. Zhang, Y.; Liu, R.; Heidari, A.A.; Wang, X.; Chen, Y.; Wang, M.; Chen, H. Towards augmented kernel extreme learning models for bankruptcy prediction: Algorithmic behavior and comprehensive analysis. *Neurocomputing* **2020**, *430*, 185–212. [CrossRef]
41. Yu, C.; Chen, M.; Cheng, K.; Zhao, X.; Ma, C.; Kuang, F.; Chen, H. SGOA: Annealing-behaved grasshopper optimizer for global tasks. *Eng. Comput.* **2021**. [CrossRef]

42. Cai, Z.; Gu, J.; Luo, J.; Zhang, Q.; Chen, H.; Pan, Z.; Li, Y.; Li, C. Evolving an optimal kernel extreme learning machine by using an enhanced grey wolf optimization strategy. *Expert Syst. Appl.* **2019**, *138*, 112814. [CrossRef]
43. Heidari, A.A.; Abbaspour, R.A.; Chen, H. Efficient boosted grey wolf optimizers for global search and kernel extreme learning machine training. *Appl. Soft Comput.* **2019**, *81*, 105521. [CrossRef]
44. Shen, L.; Chen, H.; Yu, Z.; Kang, W.; Zhang, B.; Li, H.; Yang, B.; Liu, D. Evolving support vector machines using fruit fly optimization for medical data classification. *Knowl.-Based Syst.* **2016**, *96*, 61–75. [CrossRef]
45. Wang, M.; Chen, H.; Yang, B.; Zhao, X.; Hu, L.; Cai, Z.; Huang, H.; Tong, C. Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. *Neurocomputing* **2017**, *267*, 69–84. [CrossRef]
46. Wang, M.; Chen, H. Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis. *Appl. Soft Comput.* **2020**, *88*, 105946. [CrossRef]
47. Deng, W.; Xu, J.; Zhao, H.; Song, Y. A Novel Gate Resource Allocation Method Using Improved PSO-Based QEA. *IEEE Trans. Intell. Transp. Syst.* **2020**. [CrossRef]
48. Deng, W.; Xu, J.; Song, Y.; Zhao, H. An Effective Improved Co-evolution Ant Colony Optimization Algorithm with Multi-Strategies and Its Application. *Int. J. Bio-Inspired Comput.* **2020**, *16*, 158–170. [CrossRef]
49. Chen, Z.-G.; Zhan, Z.-H.; Lin, Y.; Gong, Y.-J.; Gu, T.-L.; Zhao, F.; Yuan, H.-Q.; Chen, X.; Li, Q.; Zhang, J. Multiobjective cloud workflow scheduling: A multiple populations ant colony system approach. *IEEE Trans. Cybern.* **2018**, *49*, 2912–2926. [CrossRef]
50. Wang, Z.-J.; Zhan, Z.-H.; Yu, W.-J.; Lin, Y.; Zhang, J.; Gu, T.-L.; Zhang, J. Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling. *IEEE Trans. Cybern.* **2019**, *50*, 2715–2729. [CrossRef]
51. Deng, W.; Liu, H.; Xu, J.; Zhao, H.; Song, Y. An improved quantum-inspired differential evolution algorithm for deep belief network. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 7319–7327. [CrossRef]
52. Zhao, H.; Liu, H.; Xu, J.; Deng, W. Performance prediction using high-order differential mathematical morphology gradient spectrum entropy and extreme learning machine. *IEEE Trans. Instrum. Meas.* **2019**, *69*, 4165–4172. [CrossRef]
53. Liu, X.-F.; Zhan, Z.-H.; Zhang, J. Resource-Aware Distributed Differential Evolution for Training Expensive Neural-Network-Based Controller in Power Electronic Circuit. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**. [CrossRef]
54. Zhan, Z.-H.; Liu, X.-F.; Zhang, H.; Yu, Z.; Weng, J.; Li, Y.; Gu, T.; Zhang, J. Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *28*, 704–716. [CrossRef]
55. Zhao, X.; Li, D.; Yang, B.; Ma, C.; Zhu, Y.; Chen, H. Feature selection based on improved ant colony optimization for online detection of foreign fiber in cotton. *Appl. Soft Comput.* **2014**, *24*, 585–596. [CrossRef]
56. Zhao, X.; Li, D.; Yang, B.; Chen, H.; Yang, X.; Yu, C.; Liu, S. A two-stage feature selection method with its application. *Comput. Electr. Eng.* **2015**, *47*, 114–125. [CrossRef]
57. Liang, D.; Zhan, Z.-H.; Zhang, Y.; Zhang, J. An efficient ant colony system approach for new energy vehicle dispatch problem. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 4784–4797. [CrossRef]
58. Ridha, H.M.; Heidari, A.A.; Wang, M.; Chen, H. Boosted mutation-based Harris hawks optimizer for parameters identification of single-diode solar cell models. *Energy Convers. Manag.* **2020**, *209*, 112660. [CrossRef]
59. Dhiman, G.; Kumar, V. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowl.-Based Syst.* **2019**, *165*, 169–196. [CrossRef]
60. Lei, G.; Song, H.; Rodriguez, D. Power generation cost minimization of the grid-connected hybrid renewable energy system through optimal sizing using the modified seagull optimization technique. *Energy Rep.* **2020**, *6*, 3365–3376. [CrossRef]
61. Cao, Y.; Li, Y.; Zhang, G.; Jermsittiparsert, K.; Razmjooy, N. Experimental modeling of PEM fuel cells using a new improved seagull optimization algorithm. *Energy Rep.* **2019**, *5*, 1616–1625. [CrossRef]
62. Dhiman, G.; Singh, K.K.; Soni, M.; Nagar, A.; Dehghani, M.; Slowik, A.; Kaur, A.; Sharma, A.; Houssein, E.H.; Cengiz, K. MOSOA: A new multi-objective seagull optimization algorithm. *Expert Syst. Appl.* **2021**, *167*, 114150. [CrossRef]
63. Riget, J.; Vesterstrøm, J.S. A diversity-guided particle swarm optimizer-the ARPSO. *Dept. Comput. Sci. Univ. of Aarhus Aarhus Denmark Tech. Rep.* **2002**, *2*, 2002.
64. Pant, M.; Radha, T.; Singh, V.P. A simple diversity guided particle swarm optimization. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 3294–3299.
65. Mohamed, N.; Bilel, N.; Alsagri, A.S. A multi-objective methodology for multi-criteria engineering design. *Appl. Soft Comput.* **2020**, *91*, 106204. [CrossRef]
66. Price, K.; Awad, N.; Ali, M.; Suganthan, P. Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. In *Technical Report*; Nanyang Technological University: Singapore, 2018.
67. Ahmed, A.M.; Rashid, T.A.; Saeed, S.A.M. Cat swarm optimization algorithm: A survey and performance evaluation. *Comput. Intell. Neurosci.* **2020**, *2020*, 4854895. [CrossRef]
68. Rahman, C.M.; Rashid, T.A. A new evolutionary algorithm: Learner performance based behavior algorithm. *Egypt. Inform. J.* **2021**, *22*, 213–223. [CrossRef]
69. Abdullah, J.M.; Ahmed, T. Fitness dependent optimizer: Inspired by the bee swarming reproductive process. *IEEE Access* **2019**, *7*, 43473–43486. [CrossRef]

70. Rahman, C.M.; Rashid, T.A. Dragonfly algorithm and its applications in applied science survey. *Comput. Intell. Neurosci.* **2019**, *2019*, 9293617. [CrossRef]
71. Li, Z.; Tam, V. A novel meta-heuristic optimization algorithm inspired by the spread of viruses. *arXiv* **2020**, arXiv:2006.06282.
72. Mirjalili, S. SCA: A sine cosine algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
73. Yang, X.-S. Firefly algorithms for multimodal optimization. In *International Symposium on Stochastic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 169–178.
74. Yang, X.-S. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74.
75. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
76. Adarsh, B.; Raghunathan, T.; Jayabarathi, T.; Yang, X.-S. Economic dispatch using chaotic bat algorithm. *Energy* **2016**, *96*, 666–675. [CrossRef]
77. Nobile, M.S.; Cazzaniga, P.; Besozzi, D.; Colombo, R.; Mauri, G.; Pasi, G. Fuzzy Self-Tuning PSO: A settings-free algorithm for global optimization. *Swarm Evol. Comput.* **2018**, *39*, 70–85. [CrossRef]
78. Yong, J.; He, F.; Li, H.; Zhou, W. A novel bat algorithm based on collaborative and dynamic learning of opposite population. In Proceedings of the 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD), Nanjing, China, 9–11 May 2018; pp. 541–546.
79. Zhang, H.; Yuan, M.; Liang, Y.; Liao, Q. A novel particle swarm optimization based on prey–predator relationship. *Appl. Soft Comput.* **2018**, *68*, 202–218. [CrossRef]
80. Lin, A.; Wu, Q.; Heidari, A.A.; Xu, Y.; Chen, H.; Geng, W.; Li, C. Predicting intentions of students for master programs using a chaos-induced sine cosine-based fuzzy K-nearest neighbor classifier. *IEEE Access* **2019**, *7*, 67235–67248. [CrossRef]
81. Hongwei, L.; Jianyong, L.; Liang, C.; Jingbo, B.; Yangyang, S.; Kai, L.J. Chaos-enhanced moth-flame optimization algorithm for global optimization. *J. Syst. Eng. Electron.* **2019**, *30*, 1144–1159.
82. Nenavath, H.; Jatoth, R.K.; Das, S. A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. *Swarm Evol. Comput.* **2018**, *43*, 1–30. [CrossRef]
83. Luo, J.; Chen, H.; Heidari, A.A.; Xu, Y.; Zhang, Q.; Li, C. Multi-strategy boosted mutative whale-inspired optimization approaches. *Appl. Math. Model.* **2019**, *73*, 109–123. [CrossRef]
84. Fan, Y.; Wang, P.; Mafarja, M.; Wang, M.; Zhao, X.; Chen, H. A bioinformatic variant fruit fly optimizer for tackling optimization problems. *Knowl.-Based Syst.* **2021**, *213*, 106704. [CrossRef]
85. Hu, Z.; Wang, J.; Zhang, C.; Luo, Z.; Luo, X.; Xiao, L.; Shi, J. Uncertainty Modeling for Multi center Autism Spectrum Disorder Classification Using Takagi-Sugeno-Kang Fuzzy Systems. *IEEE Trans. Cogn. Dev. Syst.* **2021**. [CrossRef]
86. Chen, C.; Wu, Q.; Li, Z.; Xiao, L.; Hu, Z.Y. Diagnosis of Alzheimer's disease based on Deeply-Fused Nets. *Comb. Chem. High Throughput Screen.* **2020**, *24*, 781–789. [CrossRef]
87. Fei, X.; Wang, J.; Ying, S.; Hu, Z.; Shi, J. Projective parameter transfer based sparse multiple empirical kernel learning Machine for diagnosis of brain disease. *Neurocomputing* **2020**, *413*, 271–283. [CrossRef]
88. Saber, A.; Sakr, M.; Abo-Seida, O.M.; Keshk, A.; Chen, H. A Novel Deep-Learning Model for Automatic Detection and Classification of Breast Cancer Using the Transfer-Learning Technique. *IEEE Access* **2021**, *9*, 71194–71209. [CrossRef]
89. Cao, X.Y.; Wang, J.X.; Wang, J.H.; Zeng, B. A Risk-Averse Conic Model for Networked Microgrids Planning With Reconfiguration and Reorganizations. *IEEE Trans. Smart Grid* **2020**, *11*, 696–709. [CrossRef]
90. Cao, X.; Cao, T.; Gao, F.; Guan, X. Risk-Averse Storage Planning for Improving RES Hosting Capacity under Uncertain Siting Choice. *IEEE Trans. Sustain. Energy* **2021**. [CrossRef]
91. Pei, H.; Yang, B.; Liu, J.; Chang, K. Active Surveillance via Group Sparse Bayesian Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**. [CrossRef] [PubMed]
92. Qiu, S.; Hao, Z.; Wang, Z.; Liu, L.; Liu, J.; Zhao, H.; Fortino, G. Sensor Combination Selection Strategy for Kayak Cycle Phase Segmentation Based on Body Sensor Networks. *IEEE Internet Things J.* **2021**. [CrossRef]
93. Wu, Z.; Li, G.; Shen, S.; Cui, Z.; Lian, X.; Xu, G. Constructing dummy query sequences to protect location privacy and query privacy in location-based services. *World Wide Web* **2021**, *24*, 25–49. [CrossRef]
94. Wu, Z.; Wang, R.; Li, Q.; Lian, X.; Xu, G. A location privacy-preserving system based on query range cover-up for location-based services. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5244–5254. [CrossRef]
95. Zhang, X.; Wang, T.; Wang, J.; Tang, G.; Zhao, L. Pyramid channel-based feature attention network for image dehazing. *Comput. Vis. Image Underst.* **2020**, *197*, 103003. [CrossRef]
96. Wu, Z.; Li, R.; Xie, J.; Zhou, Z.; Guo, J.; Xu, X. A user sensitive subject protection approach for book search service. *J. Assoc. Inf. Sci. Technol.* **2020**, *71*, 183–195. [CrossRef]
97. Wu, Z.; Shen, S.; Lian, X.; Su, X.; Chen, E. A dummy-based user privacy protection approach for text information retrieval. *Knowl.-Based Syst.* **2020**, *195*, 105679. [CrossRef]
98. Wu, Z.; Shen, S.; Zhou, H.; Li, H.; Lu, C.; Zou, D. An effective approach for the protection of user commodity viewing privacy in e-commerce website. *Knowl.-Based Syst.* **2021**, *220*, 106952. [CrossRef]
99. Qiu, S.; Zhao, H.; Jiang, N.; Wu, D.; Song, G.; Zhao, H.; Wang, Z. Sensor network oriented human motion capture via wearable intelligent system. *Int. J. Intell. Syst.* **2022**, *137*, 1646–1673. [CrossRef]

100. Zhang, X.; Jiang, R.; Wang, T.; Wang, J. Recursive Neural Network for Video Deblurring. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 3025–3036. [CrossRef]
101. Mahdavi, M.; Fesanghary, M.; Damangir, E. An improved harmony search algorithm for solving optimization problems. *Appl. Math. Comput.* **2007**, *188*, 1567–1579. [CrossRef]
102. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948.
103. Kaveh, A.; Khayatazad, M. A new meta-heuristic method: Ray optimization. *Comput. Struct.* **2012**, *112*, 283–294. [CrossRef]
104. Mezura-Montes, E.; Coello, C.A.C. An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *Int. J. Gen. Syst.* **2008**, *37*, 443–473. [CrossRef]
105. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
106. Coello, C.A.C. Use of a self-adaptive penalty approach for engineering optimization problems. *Comput. Ind.* **2000**, *41*, 113–127. [CrossRef]
107. Kannan, B.; Kramer, S.N. An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*; American Society of Mechanical Engineers: New York, NY, USA, 1994; pp. 103–112.
108. Sandgren, E. Nonlinear Integer and Discrete Programming in Mechanical Design Optimization. *J. Mech. Des.* **1990**, *112*, 223–229. [CrossRef]
109. Cheng, M.-Y.; Prayogo, D. Symbiotic organisms search: A new metaheuristic optimization algorithm. *Comput. Struct.* **2014**, *139*, 98–112. [CrossRef]
110. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Eng. Comput.* **2013**, *29*, 17–35. [CrossRef]
111. Wang, G.; Heidari, A.A.; Wang, M.; Kuang, F.; Zhu, W.; Chen, H. Chaotic arc adaptive grasshopper optimization. *IEEE Access* **2021**, *9*, 17672–17706. [CrossRef]
112. Wang, G.G. Adaptive response surface method using inherited latin hypercube design points. *J. Mech. Des.* **2003**, *125*, 210–220. [CrossRef]
113. Kamboj, V.K.; Nandi, A.; Bhadoria, A.; Sehgal, S. An intensify Harris Hawks optimizer for numerical and engineering optimization problems. *Appl. Soft Comput.* **2020**, *89*, 106018. [CrossRef]
114. Faramarzi, A.; Heidarinejad, M.; Stephens, B.; Mirjalili, S. Equilibrium optimizer: A novel optimization algorithm. *Knowl.-Based Syst.* **2020**, *191*, 105190. [CrossRef]
115. Lee, K.S.; Geem, Z.W. A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Comput. Methods Appl. Mech. Eng.* **2005**, *194*, 3902–3933. [CrossRef]
116. Hedar, A.-R.; Fukushima, M. Derivative-free filter simulated annealing method for constrained continuous global optimization. *J. Glob. Optim.* **2006**, *35*, 521–549. [CrossRef]
117. Ray, T.; Liew, K.-M. Society and civilization: An optimization algorithm based on the simulation of social behavior. *IEEE Trans. Evol. Comput.* **2003**, *7*, 386–396. [CrossRef]
118. Akhtar, S.; Tai, K.; Ray, T. A socio-behavioural simulation model for engineering design optimization. *Eng. Optim.* **2002**, *34*, 341–354. [CrossRef]
119. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
120. Nenavath, H.; Jatoth, R.K. Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking. *Appl. Soft Comput.* **2018**, *62*, 1019–1043. [CrossRef]
121. Chen, W.-N.; Zhang, J.; Lin, Y.; Chen, N.; Zhan, Z.-H.; Chung, H.S.-H.; Li, Y.; Shi, Y.-H. Particle swarm optimization with an aging leader and challengers. *IEEE Trans. Evol. Comput.* **2012**, *17*, 241–258. [CrossRef]
122. Kumar, A.; Das, S.; Zelinka, I. A self-adaptive spherical search algorithm for real-world constrained optimization problems. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, Cancún, Mexico, 8–12 July 2020; pp. 13–14.
123. Gurrola-Ramos, J.; Hernàndez-Aguirre, A.; Dalmau-Cedeño, O. COLSHADE for real-world single-objective constrained optimization problems. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020; pp. 1–8.
124. Kumar, A.; Das, S.; Zelinka, I. A modified covariance matrix adaptation evolution strategy for real-world constrained optimization problems. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, Cancún, Mexico, 8–12 July 2020; pp. 11–12.
125. Li, J.; Xiao, D.-D.; Zhang, T.; Liu, C.; Li, Y.-X.; Wang, G.-G. Multi-swarm cuckoo search algorithm with Q-learning model. *Comput. J.* **2020**, *64*, 108–131. [CrossRef]
126. Nan, X.; Bao, L.; Zhao, X.; Zhao, X.; Sangaiah, A.K.; Wang, G.-G.; Ma, Z. EPuL: An enhanced positive-unlabeled learning algorithm for the prediction of pupylation sites. *Molecules* **2017**, *22*, 1463. [CrossRef] [PubMed]
127. Liu, D.; Fan, Z.; Fu, Q.; Li, M.; Faiz, M.A.; Ali, S.; Li, T.; Zhang, L.; Khan, M.I. Random forest regression evaluation model of regional flood disaster resilience based on the whale optimization algorithm. *J. Clean. Prod.* **2020**, *250*, 119468. [CrossRef]
128. Gu, Z.-M.; Wang, G.-G. Improving NSGA-III algorithms with information feedback models for large-scale many-objective optimization. *Future Gener. Comput. Syst.* **2020**, *107*, 49–69. [CrossRef]

129. Yi, J.-H.; Deb, S.; Dong, J.; Alavi, A.H.; Wang, G.-G. An improved NSGA-III Algorithm with adaptive mutation operator for big data optimization problems. *Future Gener. Comput. Syst.* **2018**, *88*, 571–585. [CrossRef]
130. Wang, G.-G.; Cai, X.; Cui, Z.; Min, G.; Chen, J. High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm. *IEEE Trans. Emerg. Top. Comput.* **2020**, *8*, 20–30. [CrossRef]
131. Sun, J.; Miao, Z.; Gong, D.; Zeng, X.-J.; Li, J.; Wang, G.-G. Interval multi-objective optimization with memetic algorithms. *IEEE Trans. Cybern.* **2020**, *50*, 3444–3457. [CrossRef]