Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.024247

ARTICLE

A Double Adaptive Random Spare Reinforced Sine Cosine Algorithm

Abdelazim G. Hussien[1,2], **Guoxi Liang**[3], **Huiling Chen**[4,*] **and Haiping Lin**[5,*]

[1]Department of Computer and Information Science, Linköping University, Linköping, 4189, Sweden

[2]Faculty of Science, Fayoum University, Fayoum, 63514, Egypt

[3]Department of Information Technology, Wenzhou Polytechnic, Wenzhou, 325035, China

[4]Artificial Intelligence, Wenzhou University, Wenzhou, 325035, China

[5]College of Information Engineering, Hangzhou Vocational & Technical College, Hangzhou, 310018, China

*Corresponding Authors: Huiling Chen. Email: chenhuiling.jlu@gmail.com; Haiping Lin. Email: 2018010012@hzvtc.edu.cn

Received: 27 May 2022 Accepted: 23 November 2022

ABSTRACT

Many complex optimization problems in the real world can easily fall into local optimality and fail to find the optimal solution, so more new techniques and methods are needed to solve such challenges. Metaheuristic algorithms have received a lot of attention in recent years because of their efficient performance and simple structure. Sine Cosine Algorithm (SCA) is a recent Metaheuristic algorithm that is based on two trigonometric functions Sine & Cosine. However, like all other metaheuristic algorithms, SCA has a slow convergence and may fail in sub-optimal regions. In this study, an enhanced version of SCA named RDSCA is suggested that depends on two techniques: random spare/replacement and double adaptive weight. The first technique is employed in SCA to speed the convergence whereas the second method is used to enhance exploratory searching capabilities. To evaluate RDSCA, 30 functions from CEC 2017 and 4 real-world engineering problems are used. Moreover, a non-parametric test called Wilcoxon signed-rank is carried out at 5% level to evaluate the significance of the obtained results between RDSCA and the other 5 variants of SCA. The results show that RDSCA has competitive results with other metaheuristics algorithms.

KEYWORDS

Sine cosine algorithm; global optimization; swarm intelligence; meta-heuristic algorithms

1 Introduction

Recently, an enormous number of meta-heuristics algorithms have been proposed which simulate natural-based phenomena or biological behavior [1–6]. These meta-heuristics have been successfully applied to various domains of medical science [7–10], finance [11–13], energy [14–16], engineering [17–21], agriculture [22,23] and education [24–26]. Some examples of these algorithms are Artificial Bee Colony (ABC) [27], Genetic Algorithm (GA) [28], Ant Colony Optimization (ACO) [29], Particle Swarm Optimization (PSO) [30], Cuckoo Search (CS) [31,32], Grey Wolf Optimizer [33], Bat Algorithm (BA) [34], Runge Kutta Optimizer (RKN) [35], Colony Predation Algorithm (CPA) [36], Crow Search Algorithm (CSA) [37], Weighted Mean of Vectors [38], Snake Optimizer (SO) [39], Hunger

Games Search (HGS) [40], Whale Optimization Algorithm (WOA) [41], Symbiotic Organism Search (SOS) [42], Lightning Search Algorithm (LSA) [43], Moth-Flame Optimization (MFO) [44], Slime Mould Algorithm (SMA) [45], Ant Lion Optimizer (ALO) [46], Harris Hawks Optimization (HHO) [47,48], Remora Optimization Algorithm (ROA) [49], and Wild Horse Optimizer (WHO) [50].

Sine Cosine Algorithm (SCA) is a novel population-based algorithm that has been introduced to solve real-world problems. For example, Elaziz et al. [51] developed the use of SCA to optimize the optimal parameters of Random Vector Functional Chain (RVFC) networks and use the obtained optimal model for predicting ACE inhibitory activity. Li et al. [52] developed a novel SCA-SVR model using the SCA in order to select the penalty & parameters of the kernel in SVR to improve the performance of generalization to unknown data. The model was able to have the optimal values of the SVR parameters with good results. Nayak et al. [53] introduced an enhanced Extreme Learning Machine (ELM) approach for the sine cosine algorithm (MSCA-ELM) and designed a new automated Magnetic Resonance Imaging (MRI) brain pathology detection system. Wang et al. [54] introduced a hybrid wavelet neural network based on the proposed Multi-Objective Sine Cosine Algorithm (MOSCA), which was successfully applied in wind speed prediction. Dasgupta et al. [55] introduced a newly designed SCA for determining the optimal voltage regulator systems of proportional-integral differential controller parameters, which can effectively tune the PID controller parameters. Chen et al. [16] combined opposition-based learning and local search strategies with SCA to estimate photovoltaic models parameters. Also, Sahlol et al. [56] tried to improve fish liver Enzymes prediction by using SCA to train feedforward neural networks. Also, Sindhu et al. [57] used an elitism strategy with SCA to solve the feature selection problem. Likewise, SCA has been applied to other problems e.g., predicting cervical hyperextension injury [58], predicting master students' intention [26], hydrothermal scheduling [59], and image segmentation [60].

Many variants of SCA have been proposed such as a multi-objective SCA (MO-SCA) which was proposed by Tawhid and Savsani [61]. Also, Issa et al. [62] tried to design an adaptive SCA version that is combined with PSO called (ASCA-PSO). Likewise, Sine Cosine Crow Search Algorithm (SCSCA) [63], hybrid algorithms between sine cosine and particle swarm algorithm known as (H-PSO-SCAC) [64], hybrid algorithm between SCA and differential evolution (SCA-DE) [65], Opposition-Based SCA (OBSCA) [66], enhanced sine cosine algorithm called (MSCA) [67], and orthogonal-learning driven multi-swarm sine cosine called (OMGSCA) [68]. Abd Elaziz et al. [68] presented an improved version of SCA that incorporated backward learning to increase the exploration search space to generate more accurate solutions. Guo et al. [69] presented a Riesz derivative fractional sinusoidal algorithm abbreviated as that is based on Riesz fractional derivative variation mechanism and applyed it to the design of welded beams and pressure vessels. Gupta et al. [70] developed a novel and improved sine cosine algorithm using cross-exploitation techniques with the integration of individual optimal states of individual solutions, global search, and self-learning mechanisms and applied it effectively to multi-stage threshold segmentation in images. Gupta et al. [71] developed an improved SCA that is based on the logarithmic and adaptive components of the perturbation rate. Their algorithm was employed in order to solve five different engineering optimization problems and effectively solve the global optimization problem. Issa et al. [64] introduced a modified SCA version in which SCA was combined with PSO. This algorithm was used to solve the pairwise local alignment problem with the aim of looking for the longest continuous substring between two biological sequences, which has shown good performance in accuracy and calculation time. Long et al. [72] introduced inertial weights and a novel SCA version known as (ISCA) and applied it in solving high dimensional problems based on a Gaussian function with a nonlinear transformation parameter decreasing strategy. Compared to traditional SCA and other population-based algorithms, the ISCA algorithm has faster convergence and better ability to escape from local optimal solutions. Nenavath et al. [67] proposed a new differential evolutionary sine and cosine mixture optimization algorithm and confirmed its accuracy under various challenging conditions.

Although the success of all mentioned variants of SCA, it still has low convergence and may struggle in local optimal regions. This motivated us to propose a novel variant of SCA called RDSCA that employed two strategies: 1) double adaptive weight strategy in order to achieve a good balance between exploitation and exploration by using two weights w_1 in the first half of iterations and w_2 in the last half of iterations. 2) random spare strategy in which each individual in search space can replace the optimal solution in each dimension which helps in escaping from local optima. Moreover, 30 functions from CEC 2017 and 4 real-world constrained engineering problems are used to validate our algorithm. Also, a comparison with many SCA variants has been done with RDSCA. Chen et al. [73] has used the same strategies with WO and they argued that it has a very good results. So, we test the same mechansim on SCA.

To the best of our information & knowledge and according to the literature, this is the first time that these two strategies were introduced into SCA. The main contributions of this paper are shown below:

- A new metaheuristic algorithm, called RDSCA, is proposed based on a random replacement and double adaptive weight strategy, combined with SCA.
- In classical CEC2017 benchmark functions, experimental results with SCADE [74], OBSCA [68], CGSCA [75], ASCA_PSO [64] and CESCA [26] demonstrate that the RDSCA algorithm has efficient performance and competitive ability in optimization search.
- To verify the performance of the algorithm on real-world problems, RDSCA's experimental results on four engineering problems demonstrate that the optimizer can effectively solve complex real-world problems.

The rest of this paper is organized as follows: the basic SCA and its mathematical equation are given in Section 2 whereas RDSCA is discussed comprehensively in Section 3. Section 4 shows both experiments & the results whereas the conclusion & possible future works are given in Section 5.

2 Sine Cosine Algorithm

Recently many new meta-heuristics algorithms have been developed and have shown great potential in tackling complex optimization problems such as image segmentation [76,77], PID optimization control [78], economic emission dispatch problem [79], expensive optimization problems [80], combination optimization problems [81], bankruptcy prediction [12,82], plant disease recognition [83], traveling salesman problem [84], feature selection [85,86], multi-objective problem [87], detection of foreign fiber in cotton [88,89], medical diagnosis [7,8], fault diagnosis [90,91], parameter optimization for machine learning models [92,93], prediction problems in educational field [24,25], constrained optimization problems [94,95], scheduling problem [96,97], and object tracking [98,99]. SCA can be considered as one of the most powerful and well-known meta-heuristics algorithms which was firstly proposed in 2016 [100]. The main idea in SCA is Sine & Cosine operator, which update agent movement according to Eqs. (1a), (1b):

$$x_i^{t+1} = x_i^t + r_1 \times \sin(r_2) \times |r_3 p_i^t - x_i^t| \quad \text{if } r_4 < 0.5 \tag{1a}$$

$$x_i^{t+1} = x_i^t + r_1 \times \cos(r_2) \times |r_3 p_i^t - x_i^t| \quad \text{if } r_4 \geq 0.5 \tag{1b}$$

where r_2, r_3, and r_4 are random numbers in the interval (0, 1], p_i^t is the best solution position obtained in the *i-th* iteration, and r_1 can be obtained from the following equation:

$$r_1 = a - t\frac{a}{T} \tag{2}$$

where a refers to a constant number, and T, t refer to the maximum iteration number and the current iteration, respectively. Also, flowchart of SCA is given in Fig. 1.

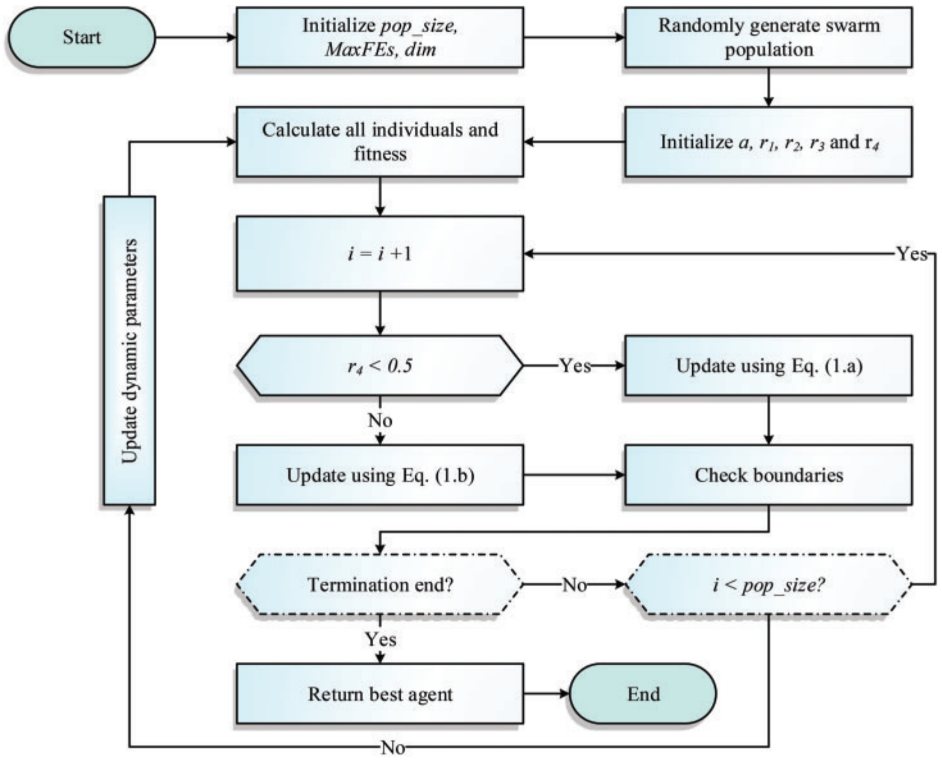

Start
Initialize pop_size, MaxFEs, dim
Randomly generate swarm population
Initialize a, r1, r2, r3 and r4
Calculate all individuals and fitness
Update dynamic parameters
i = i +1
Yes
r4 < 0.5
Yes
Update using Eq. (1.a)
No
Update using Eq. (1.b)
Check boundaries
Termination end?
No
i < pop_size?
Yes
Return best agent
End
No

Figure 1: Flowchart of SCA

3 Proposed Algorithm

3.1 Architecture of RDSCA

In this section, the structure of the suggested algorithm called RDSCA has been illustrated in detail. RDSCA is a modification of SCA with 2 strategies: 1) random spare/replacement. 2) double adaptive weight. The pseudo-code and flow chart are given in Algorithm 1 and Fig. 2. The integration of two mentioned strategies has a great impact on SCA in search capabilities and convergence curve.

Algorithm 1: RDSCA

```
1: Input: Individuals number in the population ($n$), Maximum number of function evaluation ($MaxFEs$), Number of dimensions ($dim$), Lower Bound ($lb$) & Upper Bound ($ub$)
2: Output: Best agent position
3: Positions = Initialize search agents positions $X$
4: Find $X^*$ = best search agent threads
5: $FEs = 0$
```

(Continued)

Algorithm 1: (Continued)

```
6: Calculate each individual fitness and update the current agent with best fitness
7: $X^* =$ best agent
8: **While** $FEs < MaxFEs$
9:    Check the boundary using $ub$ and $lb$
10:   **For** each search agent
11:     **If** $(tan\,(\pi * (rand - 0.5)) < (1 - FEs/MaxFEs))$
12:         Randomly select a dimension of the current individual: $j = randi([1 dim])$
13:         Use the dimension of the current optimal solution to replace the dimension of the current
            individual: $agent_i\,(j) = X^*\,(j)$
14:     **EndIf**
15:   **EndFor**
16:   Calculate each search agent fitness
17:   **For** ($agent_i$ belong to agent)
18:     **If** $(FEs/MaxFEs < 0.5)$
19:         Update search agent position using Eq. (5)
20:     **Else**
21:         Update search agent position using Eq. (6)
22:     **EndIf**
23:   **EndFor**
24:   Calculate each search agent fitness
25:   Update $X^*$
26: **EndWhile**
```

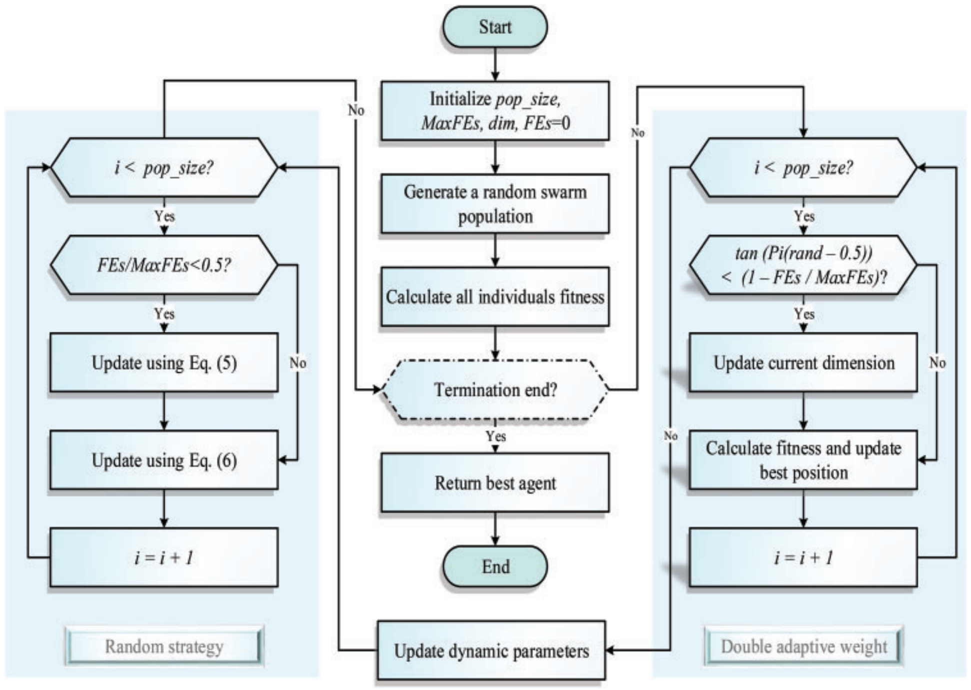


Figure 2: Flowchart of RDSCA

3.1.1 Random Spare/Replacement

In this strategy, the current individual with *n-th* dimension position will be replaced with the corresponding dimension of the optimal individual vector. A certain probability is used to implement a random spare strategy, since not any dimension value in the optimal individual is optimal. Cauchy random is used as a ratio between the current evaluation number to total evaluation number.

3.1.2 Double Adaptive Weight

In this strategy, similar to PSO [30], a double adaptive weight was used to improve the ability of local & global search w_1, w_2 can be calculated from Eqs. (3) and (4). Furthermore, Fig. 1 in the supplement analyzes the balance of RDSCA and SCA.

$$w_1 = \left(1 - \frac{FEs}{MaxFES}\right)^{1-\tan\left(\pi\times(rand-0.5)\times\frac{s}{MaxFEs}\right)} \tag{3}$$

$$w_2 = \left(2 - 2\frac{FEs}{MaxFES}\right)^{1-\tan\left(\pi\times(rand-0.5)\times\frac{s}{MaxFEs}\right)} \tag{4}$$

where FEs is the current number of iterations, $MaxFES$ is the maximum number of iterations of the algorithm, $\tan(\pi \times (rand - 0.5)$ denotes the Cauchy random number, s changes with the local optimum, and s automatically increases by 1 when an individual is not updated, otherwise the value of s is subtracted by half when an individual is updated. w_1 and w_2 are not linearly decreasing, but fluctuate in frequency as the algorithm falls into the local optimum, and the fluctuation range of w_1 is [0, 1], w_2 takes values in the range [0.5, 1]. Therefore, w_1 is mainly used in the first half of the algorithm iteration to improve the global search ability of the algorithm, and w_2 is used in the second half of the algorithm iteration to improve the ability of the algorithm to jump out of the local optimum. Eq. (5) is used in the first half of iterations whereas, Eq. (6) is used in the last half of iterations.

$$x_i^{t+1} = \begin{cases} x_i^t + w_1 \times r_1 \times \sin(r_2) \times |r_3 p_i^t - x_i^t| & \text{if } r_4 < 0.5 \\ x_i^t + w_1 \times r_1 \times \cos(r_2) \times |r_3 p_i^t - x_i^t| & \text{if } r_4 \geq 0.5 \end{cases} \tag{5}$$

$$x_i^{t+1} = \begin{cases} x_i^t + w_2 \times r_1 \times \sin(r_2) \times |r_3 p_i^t - x_i^t| & \text{if } r_4 < 0.5 \\ x_i^t + w_2 \times r_1 \times \cos(r_2) \times |r_3 p_i^t - x_i^t| & \text{if } r_4 \geq 0.5 \end{cases} \tag{6}$$

3.2 The Proposed RDSCA

In the algorithmic optimization search process, certain dimensional vectors of the current individual may be responsible for the superior quality of the individual, and in contrast, certain dimensions may also lead to the poor quality of the individual. This is explained by the fact that the result of the mapping from the decision space to the objective space is determined jointly by multiple dimensions of the decision variables. Therefore, we go to adopt the values of some dimensions of the current optimal individual to replace the values of the corresponding dimensions of the current individual with a certain probability in the optimization process of the algorithm. In addition, the timing of random replacement is also very important, and we use the mathematical description in lines 10 to 13 of Algorithm 1 to define a random replacement probability. The mathematical description is mainly used to determine whether the individual adopts the random replacement strategy through the Cauchy random number and the current number of iterations. At the beginning of the algorithm iteration, the Cauchy random number can improve the randomness, so we have a higher probability to replace some dimensions of the individual, thus speeding up the convergence of the algorithm; at the later stage of the algorithm iteration, the mathematical description in the third line of the pseudo-code will have a smaller probability to replace the dimensions of the individual, resulting in Individuals have more vectors of random dimensions, thus reducing the possibility of the algorithm falling into a local optimum.

In addition, the proposed RDSCA algorithm in which we introduce an double adaptive weight strategy is shown in lines 16 to 22 of Algorithm 1. w_1 and w_2 are affected by the change of individuals into local optimum, which can increase the ability of the algorithm to jump out of local optimum when individuals are detected to be in local optimum. The specific principle is that when an individual is not updated for a long time, the individual is more likely to fall into a local optimum, and then the values of weights w_1 and w_2 increase, increasing the range of random fluctuations in Eqs. (1a), (1b) and enhancing the ability of the algorithm to jump out of the local optimum. As a result, the algorithm can jump out of the local optimum faster and increase the speed of convergence. The random replacement strategy and the double adaptive weight strategy cooperate with each other in the iterative process of the algorithm, which well balances the algorithm between search and exploitation to improve the convergence ability and the ability to jump out of the local optimum.

In addition, it should be especially noted that RDSCA and SCA use the same algorithm framework, and SCA mainly accomplishes the balance of algorithm exploration and exploitation through Eqs. (1a), (1b), while RDSCA mainly balances algorithm exploration and exploitation through Eqs. (5) and (6). Eqs. (5) and (6) expand the perturbation frequency and amplitude of RDSCA by adding adaptive dual weights w_1 and w_2 without changing the conversion mechanism of the algorithm from exploration to exploitation.

3.3 Time Complexity of RDSCA

The complexity of any meta-heuristic algorithm is very crucial as it computes its needed time. The time complexity of RDSCA is is depended on the number of individuals (N), maximum iteration number (T), and number of dimensions (Dim). Therefore, RDSCA complexity is as follows:

$$\begin{aligned} O(RDSCA) &= O(\text{Initialization}) + O(\text{Evaluating }) + O(\text{Updating individual}) \\ &= O(N \times D) + O(N \times D \times T) + O(N \times D \times T) = O(N \times D \times T) \end{aligned}$$

4 Experiment

In order to verify the effectiveness of the algorithm, a fair experiment is necessary, and the whole process must be controlled and have a verifiable data set [101]. In this section, a discussion on the performance of RDSCA is introduced based on some experiments using CEC 2017, they have been internationally recognized test data in the field of metaheuristic algorithms. References [102,103] are a benchmark and a number of compared algorithms. All experimental results are the average of the results of 30 independent runs of the algorithm on each benchmark function. All experiments of the algorithms were coded and run on the MAT-LAB R2014b software platform.

4.1 Benchmark Test Functions

In order to evaluate the performance of the proposed RDSCA, 30 functions from CEC 2017 benchmark were used. These functions have really complex parameters and a variety of real optimization difficulties that are close to reality. They can be classified into unimodal (F1–F3), multimodal (F4–F10), hybrid (F11–F20), and composite (F21–F30) as given in Table S1 in the supplement, where ''Types'' denotes the type of function; unimodal indicates that there is only one global optimum solution and no local optimum, which can test the convergence speed of the algorithm; multi-modal contains multiple local optima with one and only one global optimum; hybrid and composite functions are more complex, which often have multi-modal nature and have more difficult to jump out of the local optimum, which requires a better balance between exploration and exploitation of the algorithm. "Name" denotes the name of the function; "Opt" denotes the optimal objective value of the function. In all experiments, the number of population (n) is set to 30, dimension numbers (dim) is equal to 30, and the number of maximum evaluation of fitness function ($MaxFEs$) has been set to $dim * 10000 = 300000$ as it is shown in Table 1.

Table 1: The parameter settings

No.	Parameter name	Value
1	Population size	30
2	No. of dim	30
3	MaxFEs	$dim * 10000$

4.2 *Comparative Algorithms*

To evaluate the performance of RDSCA, a comparison against many other variants of SCA has been done using 30 functions from CEC 2017. Modified SCA versions are chosen for comparison purpose, which are mentioned as bellow:

- SCADE [74]: differential evolution (DE) operators have been employed in SCA as local search operators.
- OBSCA [68]: opposition-based learning (OBL) is used in the initialization and updating phases.
- CGSCA [75]: exponential decreasing conversion parameter and linear decreasing inertia weight methods are employed within SCA to achieve a good balance between exploration and exploitation. Moreover, a greedy Levy mutation technique is also applied.
- ASCA_PSO [62]: ASCA-PSO is an algorithm that hybridizes PSO with SCA in order to have more exploitation properties.
- CESCA [26]: chaos theory concepts have been employed in SCA.

4.3 *Measures of the Performance*

Here, a number of statistical metrics are used to compute the RDSCA performance.

- Average (mean):

$$Avg = \frac{1}{N}\sum_{i=1}^{N} g_*^i \tag{7}$$

- Std:

$$SD = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(g_*^i - Avg)^2} \tag{8}$$

where N refers the number of total evaluated times, g_*^i is the value of the fitness function at i-th iteration.

- Min: the best solution obtained over all running.

$$best = min_{i=1}^{N} g_*^i \tag{9}$$

where M refers to the iteration number to perform the experiments and g_*^i refers to the result obtained from the fitness function at i-th iteration and Avg refers to the average of all running.

- Max: the worst solution obtained over all running.

$$Worst = max_{i=1}^{N} g_*^i \tag{10}$$

where N refers to iteration number to perform the experiments and g_*^i is the result obtain fron the fitness function at i-th iteration.

4.4 Discussion

4.4.1 Comparison between RDSCA and Other Algorithms

The comparison results between RDSCA and other SCA variants mentioned in Section 4.2 are shown in Table 2. From this table, it is notable that the suggested algorithm achieved better fitness value. RDSCA achieved the best average in 24 functions from 30 functions, second best in 4 other functions (F2, F11, F18, and F24) and the third best in F14. Furthermore, RDSCA ranked first in term of standard deviation in 19 functions and the second best in the other 11 functions.

Table 2: Results of the RDSCA compared with other peers over 30 functions

	F1		F2		F3	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**6.2219E + 08**	**1.8712E + 08**	**1.1238E + 22**	**2.0717E + 22**	1.3508E + 04	3.2852E + 03
SCADE	1.9458E + 10	2.9400E + 09	6.5175E + 36	1.9188E + 37	5.9178E + 04	8.8280E + 03
OBSCA	1.6934E + 10	2.7440E + 09	4.8814E + 35	1.5962E + 36	6.1150E + 04	8.0092E + 03
CGSCA	1.4120E + 10	2.5116E + 09	7.4460E + 34	1.6640E + 35	4.2958E + 04	6.9959E + 03
ASCA_PSO	7.7378E + 08	1.5252E + 09	1.2520E + 27	4.6678E + 27	**1.3321E + 03**	**4.9046E + 02**
CESCA	5.7287E + 10	6.1611E + 09	3.7953E + 45	5.0936E + 45	1.0539E + 05	1.6255E + 04
	F4		F5		F6	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	5.5428E + 02	**2.4583E + 01**	**6.7389E + 02**	2.2956E + 01	**6.1130E + 02**	**2.2483E + 00**
SCADE	4.0202E + 03	1.3020E + 03	8.3253E + 02	2.2439E + 01	6.6306E + 02	8.3519E + 00
OBSCA	2.7684E + 03	9.3773E + 02	8.0663E + 02	2.5466E + 01	6.5491E + 02	6.3008E + 00
CGSCA	1.6880E + 03	3.4037E + 02	7.8427E + 02	2.0481E + 01	6.5665E + 02	5.1924E + 00
ASCA_PSO	**5.3237E + 02**	6.6006E + 01	7.2477E + 02	4.3301E + 01	6.3756E + 02	8.4264E + 00
CESCA	1.6195E + 04	2.5128E + 03	9.6710E + 02	**1.8414E + 01**	7.0422E + 02	5.7106E + 00
	F7		F8		F9	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**9.5297E + 02**	**2.7136E + 01**	**9.6254E + 02**	2.5753E + 01	**63.1188E + 03**	**1.1288E + 03**
SCADE	1.1704E + 03	2.9579E + 01	1.0879E + 03	**1.3911E + 01**	7.8497E + 03	1.1822E + 03
OBSCA	1.1782E + 03	3.5411E + 01	1.0723E + 03	2.3461E + 01	6.7385E + 03	1.1729E + 03
CGSCA	1.1400E + 03	2.8765E + 01	1.0611E + 03	1.7267E + 01	6.7102E + 03	1.2599E + 03
ASCA_PSO	9.8674E + 02	3.9556E + 01	9.9564E + 02	3.8894E + 01	4.9985E + 03	1.9772E + 03
CESCA	1.5571E + 03	4.7498E + 01	1.1812E + 03	2.1949E + 01	1.4828E + 04	1.4044E + 03
	F10		F11		F12	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**5.5289E + 03**	5.1937E + 02	1.3329E + 03	5.9536E + 01	**1.3777E + 07**	**5.6877E + 06**
SCADE	8.1219E + 03	3.7608E + 02	3.3038E + 03	5.6592E + 02	1.9612E + 09	4.6716E + 08
OBSCA	7.5473E + 03	3.2308E + 02	2.7073E + 03	5.1518E + 02	2.0054E + 09	5.5859E + 08
CGSCA	8.1201E + 03	3.3866E + 02	2.2159E + 03	3.5319E + 02	1.4299E + 09	3.4403E + 08
ASCA_PSO	5.9528E + 03	7.9130E + 02	**1.3065E + 03**	**5.1754E + 01**	7.2270E + 07	1.1435E + 08
CESCA	8.7428E + 03	**2.5306E + 02**	1.0656E + 04	1.8880E + 03	1.5387E + 10	1.9571E + 09

(Continued)

Table 2 (continued)

	F13		F14		F15	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**1.6006E + 05**	**2.8831E + 05**	1.2088E + 05	1.0403E + 05	**1.4060E + + 04**	**1.5482E + 04**
SCADE	6.4792E + 08	2.4197E + 08	3.4666E + 05	2.2972E + 05	7.0301E + 06	3.6304E + 06
OBSCA	6.7626E + 08	3.0294E + 08	2.6070E + 05	1.3005E + 05	9.5494E + 06	1.2684E + 07
CGSCA	4.8767E + 08	2.1961E + 08	1.6833E + 05	7.8688E + 04	7.2397E + 06	6.0676E + 06
ASCA_PSO	1.0763E + 07	1.2694E + 07	**3.0588E + 04**	**2.3903E + 04**	1.1501E + 06	4.5750E + 05
CESCA	1.2639E + 10	3.4271E + 09	5.5891E + 06	2.5718E + 06	5.2247E + 08	1.7039E + 08

	F16		F17		F18	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**2.6320E + 03**	2.9551E + 02	**2.1111E + 03**	1.9596E + 02	9.5726E + 05	1.1475E + 06
SCADE	3.8803E + 03	2.4641E + 02	2.4752E + 03	1.5444E + 02	3.6425E + 06	2.4851E + 06
OBSCA	3.7950E + 03	2.0856E + 02	2.5064E + 03	1.8704E + 02	3.8241E + 06	1.8391E + 06
CGSCA	3.6968E + 03	**1.8349E + 02**	2.5356E + 03	**1.4308E + 02**	3.4098E + 06	2.0000E + 06
ASCA_PSO	2.8850E + 03	3.5832E + 02	2.3200E + 03	1.9910E + 02	**5.0653E + 05**	**2.8667E + 05**
CESCA	5.9314E + 03	5.1787E + 02	4.5860E + 03	7.4977E + 02	5.4109E + 07	2.3017E + 07

	F19		F20		F21	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	**1.7003E + 04**	**1.6999E + 04**	**2.4398E + 03**	1.6119E + 02	**2.4592E + 03**	2.8768E + 01
SCADE	2.6168E + 07	1.7777E + 07	2.7560E + 03	**8.7286E + 01**	2.5820E + 03	2.4448E + 01
OBSCA	4.2022E + 07	2.2274E + 07	2.6801E + 03	1.1593E + 02	2.4642E + 03	8.0519E + 01
CGSCA	2.6169E + 07	1.3637E + 07	2.6328E + 03	1.2153E + 02	2.5624E + 03	**2.0162E + 01**
ASCA_PSO	3.1399E + 06	2.0703E + 06	2.4800E + 03	1.2768E + 02	2.4967E + 03	3.9199E + 01
CESCA	1.0812E + 09	3.6928E + 08	3.1630E + 03	1.5016E + 02	2.7655E + 03	3.3188E + 01

	F22		F23		F24	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	6.7629E + 03	1.5406E + 03	**2.8289E + 03**	2.5385E + 01	3.0600E + 03	3.4885E + 01
SCADE	4.5688E + 03	4.3701E + 02	3.0113E + 03	**2.4627E + 01**	3.1721E + 03	**2.8027E + 01**
OBSCA	4.1686E + 03	2.9004E + 02	3.0123E + 03	3.3441E + 01	3.1861E + 03	3.0541E + 01
CGSCA	**3.8925E + 03**	**2.8385E + 02**	2.9985E + 03	2.6455E + 01	3.1473E + 03	3.0094E + 01
ASCA_PSO	5.7603E + 03	2.1633E + 03	2.8827E + 03	2.9855E + 01	**3.0428E + 03**	3.1956E + 01
CESCA	9.3554E + 03	5.1942E + 02	3.4619E + 03	6.2651E + 01	3.4831E + 03	4.1613E + 01

	F25		F26		F27	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	2.9410E + 03	**3.1895E + 01**	**5.2515E + 03**	4.6596E + 02	**3.2311E + 03**	**1.2686E + 01**
SCADE	3.4387E + 03	7.9046E + 01	7.5227E + 03	**3.8399E + 02**	3.4437E + 03	5.6021E + 01
OBSCA	3.3592E + 03	1.1763E + 02	7.1190E + 03	5.7717E + 02	3.4644E + 03	5.8403E + 01
CGSCA	3.2545E + 03	8.2526E + 01	7.0669E + 03	5.1308E + 02	3.3868E + 03	3.9520E + 01
ASCA_PSO	**2.9334E + 03**	4.6595E + 01	6.1373E + 03	9.3932E + 02	3.3062E + 03	5.6614E + 01
CESCA	5.6851E + 03	4.8949E + 02	1.1209E + 04	4.9538E + 02	3.7079E + 03	8.6987E + 01

	F28		F29		F30	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*
RDSCA	3.3476E + 03	**3.9886E + 01**	**3.8596E + 03**	**1.8331E + 02**	**2.2287E + 05**	**3.7272E + 05**
SCADE	4.3131E + 03	2.9379E + 02	5.0399E + 03	2.9513E + 02	1.1438E + 08	3.7507E + 07
OBSCA	4.2012E + 03	2.2539E + 02	5.0085E + 03	2.1728E + 02	1.0531E + 08	3.9825E + 072
CGSCA	3.9233E + 03	1.6877E + 02	4.7242E + 03	2.0969E + 02	9.0569E + 07	3.2672E + 072
ASCA_PSO	**3.3253E + 03**	8.8893E + 01	4.4363E + 03	2.0425E + 02	9.2961E + 06	4.8182E + 06
CESCA	7.0690E + 03	4.6778E + 02	6.0375E + 03	1.9535E + 02	2.3883E + 09	7.2958E + 08

Fig. 3 shows the convergence curve for some functions from all types (unimodal, multimodal, hybrid, and composite). It can be observed from this figure that RDSCA has a better speed convergence compared to other algorithms.

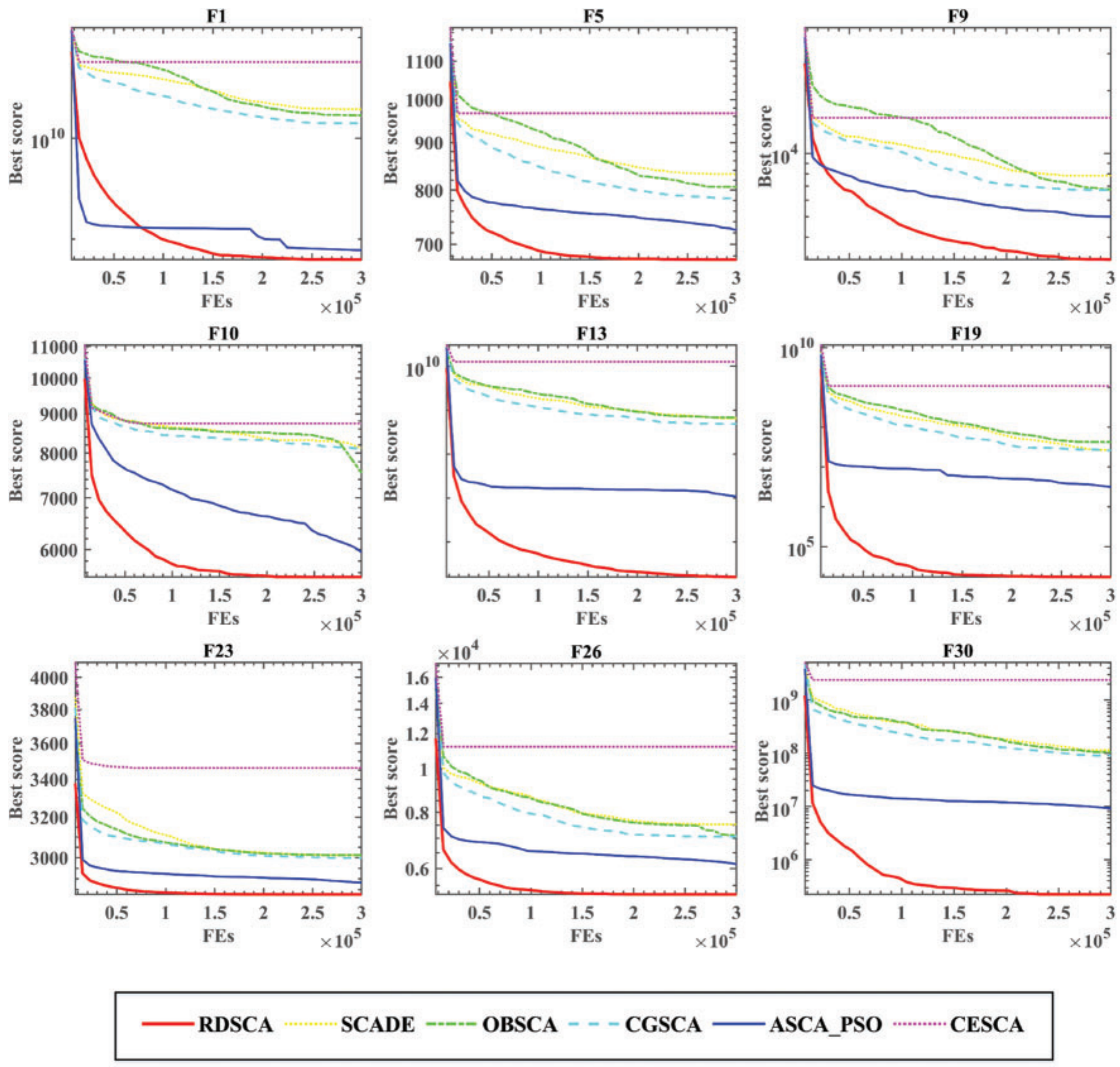
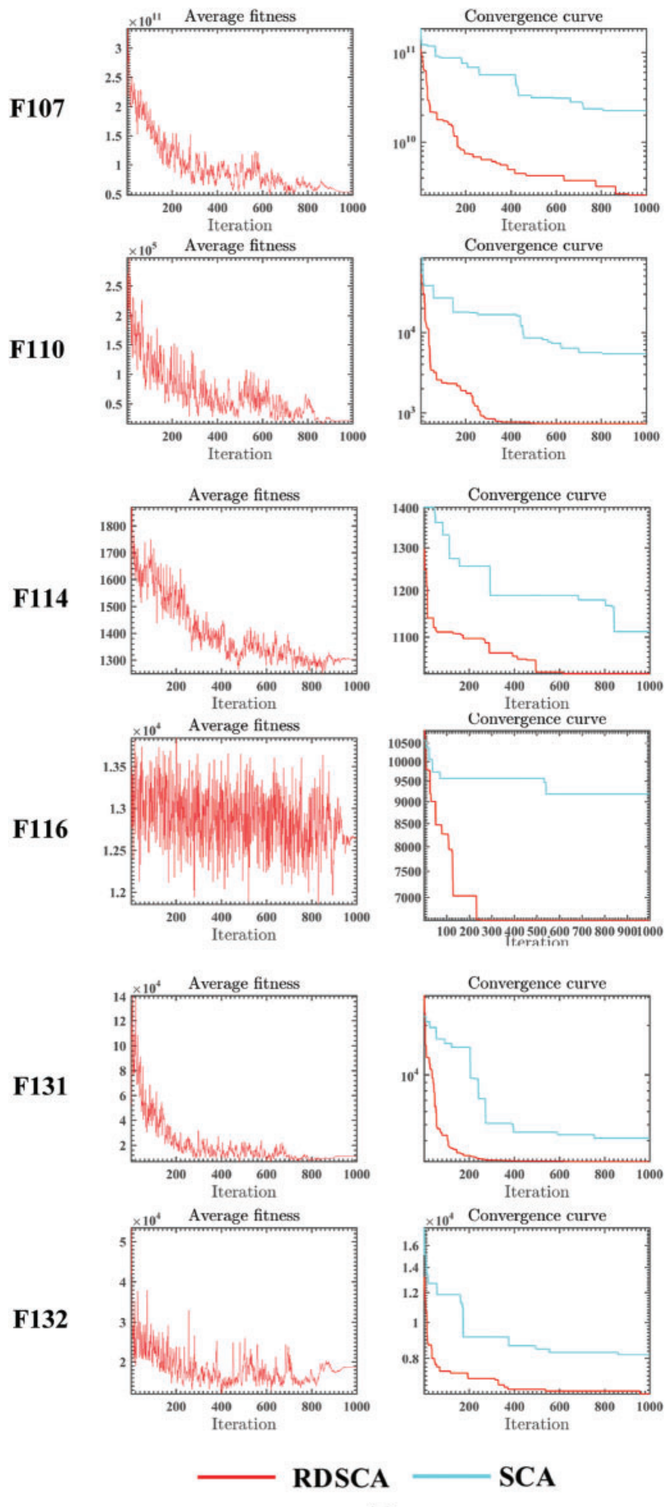


Figure 3: The average of convergence curves of unimodal & multi-modal functions

Also, Fig. 4 shows the 2-D appearance of the function in the first column whereas the search history is shown in the second one where we can observe that RDSCA's search agent tends to exploit the promising regions of search space more extensively. The third column shows the first particle trajectory where we can see the first agent changes in the first dimension and the fourth and fifth columns show RDSCA's average fitness and convergence curve, respectively. Furthermore, the result of banlance analysis of RDSCA and SCA is shown in Figure S1 in the supplement, which also shows the superiority of RDSCA.



(a)

Figure 4: (Continued)

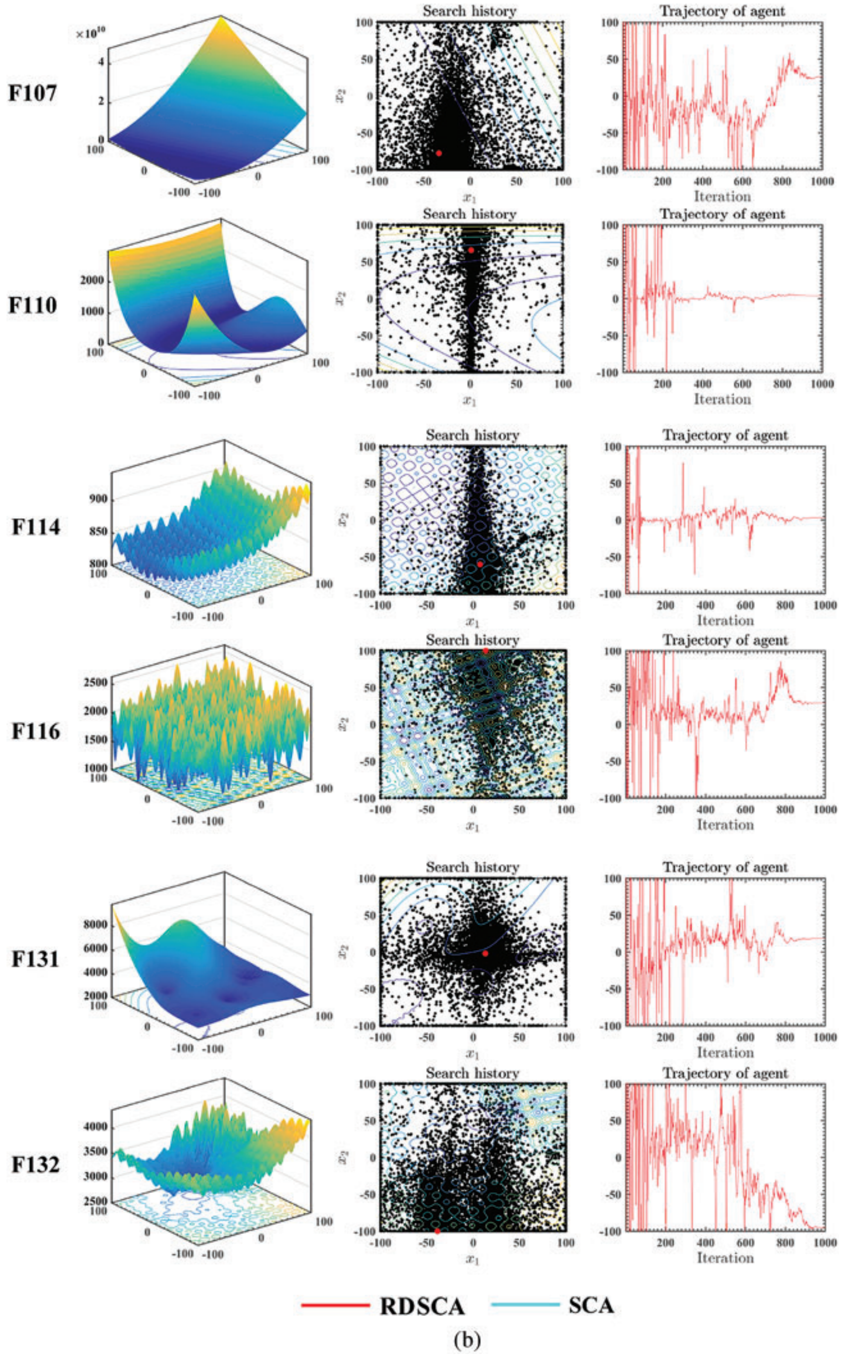


Figure 4: (a) The average of convergence curves of unimodal & multi-modal functions 2; (b) The average of convergence curves of unimodal & multi-modal functions 2

4.4.2 Statistical Analysis

Besides the performance measured mentioned in Section 4.3, a non-parametric test called Wilcoxon signed-rank (WSR) [104] is used to be able to prove the significant superiority of the RDSCA. WRS considers all results as one group and then ranks are assigned to all scores in each group then, WRS test sums each group rank.

All algorithms are tested in all test functions for WSR. Each algorithm is performed 30 times independently on each test function, and the resulting solutions are available as data samples. WSR can test whether the two algorithms are significantly different on that test function based on these samples. First, we start with the null hypothesis, which is the assumption that there is no significant difference in the performance of the two algorithms. When $p < 0.05$, the null hypothesis is rejected, and when $p > 0.05$, the null hypothesis is accepted. Therefore, the two algorithms are significantly different when the p-value is greater than 0.05. Table S2 in the supplement represents the p-values for RDSCA and each comparison algorithm on each test function. From the table, we can find that most of the p-values are less than 0.05, therefore, we can conclude that RDSCA and other comparison algorithms are significantly different on the CEC2017 test functions and that RDSCA outperforms the comparison algorithms from the results in Table 2.

4.5 Engineering Problems

To validate our proposed meta-heuristic which known as RDSCA, we apply it to solve real engineering constrained problems which have various inequality and equality constraints. Furthermore, best values of such parameter of these problems remain unknown. Here, four different engineering optimization problems are used in order to validate RDSCA namely: tension/compression spring design (TCSD), welded beam design (WBD), pressure vessel design (PVD), and I-beam design (IBD).

4.5.1 TCSD Problem

The first engineering problem is called TCSD [105] (see Appendix A for the mathematical formulation). This problem aims to find the optimal and minimum weight and includes 3 design variables: mean coil diameter (D), active coils number (N), and diameter of wire (d).

Table S3 in the supplement shows the results of RDSCA against other peers. It can be noticed that the proposed algorithm achieved the best results.

4.5.2 WBD Problem

WBD problem is a well-known metal problem [106] which aims to obtain welded beam that has the minimum cost of fabrication. The mathematical modeling of WBD can be found in Appendix B. Here a comparison of our proposed approach has been done against other peers as shown in Table S4 in the supplement. It is obviously noted from the above mentioned table that RDSCA has got the minimum optimal solution, i.e., minimum fabrication cost.

4.5.3 PVD Problem

The purpose of this unconstrained engineering problem is to find the minimum cost of the cylindrical pressure vessel [107]. The PVD has 4 variables: head thickness (T_h), shell thickness (T_s), inner radius (R), and cylindrical section length (L). Appendix C shows mathematical equations of PVD.

Table S5 in the supplement shows the comparison results among RDSCA, other peers. It is obviously noticed that RDSCA ranked first.

4.5.4 IBD Problem

In this section, our proposed algorithm is used to find the optimal design of I-beam problem [108] in order to have minimum deflection. This problem has four design parameters: height, length, and 2 thickness. The mathematical formulation is founded in section Appendix D. The results of RDSCA against other algorithms are shown in Table S6 in the supplement where we can observe that RDSCA ranked first.

5 Conclusion & Possible Future Works

In this study, a new metaheuristic algorithm, called RDSCA, is proposed by adding double adaptive weight and random replacement strategy to the algorithmic framework of SCA, where random replacement are used in the process of iterative optimization of the algorithm, and each individual has a certain probability to select a dimensional value and the corresponding dimension of the current optimal value for replacement. The basic principle of this strategy is that the cause of poor individual fitness values may be the value of a single dimension of an individual. And this method can effectively increase the convergence speed of the algorithm because the individuals are using as much information as possible from the current optimal solution. The double adaptive weight strategy mainly uses two adaptive weights w_1 and w_2 to enhance the ability of the algorithm to jump out of the local optimum, where the updates of w_1 and w_2 are related to the frequency of individuals falling into the local optimum and the Corsi random number. Once an individual falls into a local optimum, the algorithm can jump out faster, thus improving the convergence speed. Based on this, we experimentally compare RDSCA with other mainstream metaheuristic algorithms on CEC2017 and four restricted real-world engineering problems, and the experimental results prove that RDSCA has stronger performance and strong competitiveness.

However, RDSCA and SCA's use the same algorithm framework, which also means that the algorithm is explored and developed in the same way of conversion as the original SCA, which may not be able to better balance the exploration and development of the algorithm when optimizing complex composite functions. In the future, we will use new techniques to improve the ability of algorithms to balance exploration and development. In the future, we will apply this version to many applications in different fields such as community service [109], multicore systems [110], etc. We can also try to propose a multi-objective version and binary version of RDSCA for other interesting scenarios. Also, it is possible in future to apply these combination of operators to other metaheurstics and comparing them.

Funding Statement: This study was supported in part by the Hangzhou Science and Technology Development Plan Project (Grant No. 20191203B30).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Feng, Y., Wang, G., Deb, S., Lu, M., Zhao, X. (2019). Monarch butterfly optimization. *Neural Computing and Applications, 31(7),* 1995–2014. https://doi.org/10.1007/s00521-015-1923-y

2. Hussien, A. G., Hassanien, A. E., Houssein, E. H., Bhattacharyya, S., Amin, M. (2019). S-shaped binary whale optimization algorithm for feature selection. In: *Recent trends in signal and image processing*, pp. 79–87. Singapore: Springer.
3. Wang, G., Coelho, L., Gao, X. Z., Deb, S. (2016). A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. *International Journal of Bio-Inspired Computation, 8(6),* 394–409. https://doi.org/10.1504/IJBIC.2016.081335
4. Wang, G. G. (2016). Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Computing, 10,* 151–164. https://doi.org/10.1007/s12293-016-0212-3
5. Hussien, A. G., Houssein, E. H., Hassanien, A. E. (2017). A binary whale optimization algorithm with hyperbolic tangent fitness function for feature selection. *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pp. 166–172. Cairo, Egypt, IEEE.
6. Hussien, A. G., Hassanien, A. E., Houssein, E. H. (2017). Swarming behaviour of salps algorithm for predicting chemical compound activities. *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pp. 315–320. Cairo, Egypt, IEEE.
7. Xia, J., Wang, Z., Yang, D., Li, R., Liang, H. et al. (2022). Performance optimization of support vector machine with oppositional grasshopper optimization for acute appendicitis diagnosis. *Computers in Biology and Medicine, 143,* 105206.
8. Xia, J., Yang, D., Zhou, H., Chen, Y., Zhang, H. et al. (2022). Evolving kernel extreme learning machine for medical diagnosis via a disperse foraging sine cosine algorithm. *Computers in Biology and Medicine, 141,* 105137. https://doi.org/10.1016/j.compbiomed.2021.105137
9. Fathi, H., AlSalman, H., Gumaei, A., Manhrawy, I. I., Hussien, A. G. et al. (2021). An efficient cancer classification model using microarray and high-dimensional data. *Computational Intelligence and Neuroscience, 2021*. https://doi.org/10.1155/2021/7231126
10. Hu, J., Zhengyuan, H., Heidari, A. A., Shou, Y., Ye, H. et al. (2022). Detection of COVID-19 severity using blood gas analysis parameters and harris hawks optimized extreme learning machine. *Computers in Biology and Medicine, 142,* 105166. https://doi.org/10.1016/j.compbiomed.2021.105166
11. Yu, C., Chen, M., Cheng, K., Zhao, X., Ma, C. et al. (2022). SGOA: Annealing-behaved grasshopper optimizer for global tasks. *Engineering with Computers, 38(5),* 3761–3788.
12. Yza, B., Rl, A., Aahc, D., Xin, W. E., Ying, C. F. et al. (2021). Towards augmented kernel extreme learning models for bankruptcy prediction: Algorithmic behavior and comprehensive analysis-sciencedirect. *Neurocomputing, 430,* 185–212. https://doi.org/10.1016/j.neucom.2020.10.038
13. Wang, M., Chen, H., Li, H., Cai, Z., Zhao, X. et al. (2017). Grey wolf optimization evolving kernel extreme learning machine: Application to bankruptcy prediction. *Engineering Applications of Artificial Intelligence, 63,* 54–68. https://doi.org/10.1016/j.engappai.2017.05.003
14. Jiao, S., Chong, G., Huang, C., Hu, H., Wang, M. et al. (2020). Orthogonally adapted harris hawk optimization for parameter estimation of photovoltaic models. *Energy, 203,* 117804. https://doi.org/10.1016/j.energy.2020.117804
15. Zhang, H., Heidari, A. A., Wang, M., Zhang, L., Chen, H. et al. (2020). Orthogonal nelder-mead moth flame method for parameters identification of photovoltaic modules. *Energy Conversion and Management, 211,* 112764. https://doi.org/10.1016/j.enconman.2020.112764
16. Chen, H., Jiao, S., Heidari, A. A., Wang, M., Chen, X. et al. (2019). An opposition-based sine cosine approach with local search for parameter estimation of photovoltaic models. *Energy Conversion and Management, 195,* 927–942. https://doi.org/10.1016/j.enconman.2019.05.057
17. Zhang, H., Liu, T., Ye, X., Heidari, A. A., Liang, G. et al. (2022). Differential evolution-assisted salp swarm algorithm with chaotic structure for real-world problems. *Engineering with Computers, 38,* 1. https://doi.org/10.1007/s00366-022-01609-6

18. Shan, W., Qiao, Z., Heidari, A. A., Chen, H., Turabieh, H. et al. (2021). Double adaptive weights for stabilization of moth flame optimizer: Balance analysis, engineering cases, and medical diagnosis. *Knowledge-Based Systems, 214,* 106728. https://doi.org/10.1016/j.knosys.2020.106728
19. Tu, J., Chen, H., Liu, J., Heidari, A. A., Zhang, X. et al. (2021). Evolutionary biogeography-based whale optimization methods with communication structure: Towards measuring the balance. *Knowledge-Based Systems, 212,* 106642. https://doi.org/10.1016/j.knosys.2020.106642
20. Hussien, A. G., Oliva, D., Houssein, E. H., Juan, A. A., Yu, X. (2020). Binary whale optimization algorithm for dimensionality reduction. *Mathematics, 8(10),* 1821. https://doi.org/10.3390/math8101821
21. Hussien, A. G. (2021). An enhanced opposition-based salp swarm algorithm for global optimization and engineering problems. *Journal of Ambient Intelligence and Humanized Computing, 13,* 129–150. https://doi.org/10.1007/s12652-021-02892-9
22. Yu, H., Song, J., Chen, C., Heidari, A. A., Liu, J. et al. (2022). Image segmentation of leaf spot diseases on maize using multi-stage Cauchy-enabled grey wolf algorithm. *Engineering Applications of Artificial Intelligence, 109,* 104653. https://doi.org/10.1016/j.engappai.2021.104653
23. Chen, C., Wang, X., Heidari, A. A., Yu, H., Chen, H. (2021). Multi-threshold image segmentation of maize diseases based on elite comprehensive particle swarm optimization and otsu. *Frontiers in Plant-Science, 12,* 789911–789911. https://doi.org/10.3389/fpls.2021.789911
24. Wei, Y., Lv, H., Chen, M., Wang, M., Heidari, A. A. et al. (2020). Predicting entrepreneurial intention of students: An extreme learning machine with Gaussian barebone harris hawks optimizer. *IEEE Access, 8,* 76841–76855. https://doi.org/10.1109/ACCESS.2020.2982796
25. Zhu, W., Ma, C., Zhao, X., Wang, M., Heidari, A. A. et al. (2020). Evaluation of sino foreign cooperative education project using orthogonal sine cosine optimized kernel extreme learning machine. *IEEE Access, 8,* 61107–61123. https://doi.org/10.1109/Access.6287639
26. Lin, A., Wu, Q., Heidari, A. A., Xu, Y., Chen, H. et al. (2019). Predicting intentions of students for master programs using a chaos-induced sine cosine-based fuzzy k-nearest neighbor classifier. *IEEE Access, 7,* 67235–67248. https://doi.org/10.1109/Access.6287639
27. Karaboga, D., Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization, 39(3),* 459–471. https://doi.org/10.1007/s10898-007-9149-x
28. Holland, J. H. (1992). Genetic algorithms. *Scientific American, 267(1),* 66–73. https://doi.org/10.1038/scientificamerican0792-66
29. Dorigo, M., Birattari, M., Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine, 1(4),* 28–39. https://doi.org/10.1109/MCI.2006.329691
30. Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95–International Conference on Neural Networks*, Perth, WA, Australia.
31. Yang, X. S., Deb, S. (2009). Cuckoo search via lévy flights. *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, Coimbatore, India.
32. Abualigah, L., Gandomi, A. H., Elaziz, M. A., Hussien, A. G., Khasawneh, A. M. et al. (2020). Nature-inspired optimization algorithms for text document clustering a comprehensive analysis. *Algorithms, 13(12),* 345. https://doi.org/10.3390/a13120345
33. Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69,* 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
34. Yang, X. S., Gandomi, A. H. (2012). Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations, 29,* 464–483. https://doi.org/10.1108/02644401211235834
35. Ahmadianfar, I., Heidari, A. A., Gandomi, A. H., Chu, X., Chen, H. (2021). Run beyond the metaphor: An efficient optimization algorithm based on runge kutta method. *Expert Systems with Applications, 181,* 115079. https://doi.org/10.1016/j.eswa.2021.115079

36. Tu, J., Chen, H., Wang, M., Gandomi, A. H. (2021). The colony predation algorithm. *Journal of Bionic Engineering, 18(3),* 674–710. https://doi.org/10.1007/s42235-021-0050-y
37. Hussien, A. G., Amin, M., Wang, M., Liang, G., Alsanad, A. et al. (2020). Crow search algorithm: Theory, recent advances, and applications. *IEEE Access, 8,* 173548–173565. https://doi.org/10.1109/ACCESS.2020.3024108
38. Ahmadianfar, I., Heidari, A. A., Noshadian, S., Chen, H., Gandomi, A. H. (2022). Info: An efficient optimization algorithm based on weighted mean of vectors. *Expert Systems with Applications, 195,* 116516. https://doi.org/10.1016/j.eswa.2022.116516
39. Hashim, F. A., Hussien, A. G. (2022). Snake optimizer: A novel meta-heuristic optimization algorithm. *Knowledge-Based Systems, 242,* 108320. https://doi.org/10.1016/j.knosys.2022.108320
40. Yang, Y., Chen, H., Heidari, A. A., Gandomi, A. H. (2021). Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Systems with Applications,* 114864. https://doi.org/10.1016/j.eswa.2021.114864
41. Hussien, A. G., Hassanien, A. E., Houssein, E. H., Amin, M., Azar, A. T. (2019). New binary whale optimization algorithm for discrete optimization problems. *Engineering Optimization, 52(6),* 945–959. https://doi.org/10.1080/0305215X.2019.1624740
42. Abdullahi, M., Ngadi, M. A., Abdulhamid, S. I. M. (2016). Symbiotic organism search optimization based task scheduling in cloud computing environment. *Future Generation Computer Systems, 56,* 640–650. https://doi.org/10.1016/j.future.2015.08.006
43. Abualigah, L., Abd Elaziz, M., Hussien, A. G., Alsalibi, B., Jalali, S. M. J. et al. (2020). Lightning search algorithm: A comprehensive survey. *Applied Intelligence, 51,* 2353–2376. https://doi.org/10.1007/s10489-020-01947-2
44. Hussien, A. G., Amin, M., Abd El Aziz, M. (2020). A comprehensive review of moth-flame optimisation: Variants, hybrids, and applications. *Journal of Experimental & Theoretical Artificial Intelligence, 32(4),* 705–725. https://doi.org/10.1080/0952813X.2020.1737246
45. Li, S., Chen, H., Wang, M., Heidari, A. A., Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems, 111,* 300–323. https://doi.org/10.1016/j.future.2020.03.055
46. Assiri, A. S., Hussien, A. G., Amin, M. (2020). Ant lion optimization: Variants, hybrids, and applications. *IEEE Access, 8,* 77746–77764. https://doi.org/10.1109/Access.6287639
47. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. et al. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97,* 849–872. https://doi.org/10.1016/j.future.2019.02.028
48. Hussien, A. G., Amin, M. (2022). A self-adaptive harris hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection. *International Journal of Machine Learning and Cybernetics, 13,* 309–336. https://doi.org/10.1007/s13042-021-01326-4
49. Wang, S., Hussien, A. G., Jia, H., Abualigah, L., Zheng, R. (2022). Enhanced remora optimization algorithm for solving constrained engineering optimization problems. *Mathematics, 10(10),* 1696. https://doi.org/10.3390/math10101696
50. Zheng, R., Hussien, A. G., Jia, H. M., Abualigah, L., Wang, S. et al. (2022). An improved wild horse optimizer for solving optimization problems. *Mathematics, 10(8),* 1311. https://doi.org/10.3390/math10081311
51. Elaziz, M. A., Hemedan, A. A., Ostaszweski, M., Schneider, R., Lu, S. (2019). Optimization ace inhibition activity in hypertension based on random vector functional link and sine-cosine algorithm. *Chemometrics and Intelligent Laboratory Systems, 190,* 69–77. https://doi.org/10.1016/j.chemolab.2019.05.009
52. Li, S., Fang, H., Liu, X. (2018). Parameter optimization of support vector regression based on sine cosine algorithm. *Expert Systems with Applications, 91,* 63–77. https://doi.org/10.1016/j.eswa.2017.08.038

53. Nayak, D. R., Dash, R., Majhi, B., Wang, S. (2018). Combining extreme learning machine with modified sine cosine algorithm for detection of pathological brain. *Computers & Electrical Engineering, 68,* 366–380. https://doi.org/10.1016/j.compeleceng.2018.04.009

54. Wang, J., Yang, W., Du, P., Niu, T. (2018). A novel hybrid forecasting system of wind speed based on a newly developed multi-objective sine cosine algorithm. *Energy Conversion and Management, 163,* 134–150. https://doi.org/10.1016/j.enconman.2018.02.012

55. Dasgupta, K., Roy, P. K., Mukherjee, V. (2020). Power flow based hydro-thermal-wind scheduling of hybrid power system using sine cosine algorithm. *Electric Power Systems Research, 178,* 106018. https://doi.org/10.1016/j.epsr.2019.106018

56. Sahlol, A. T., Ewees, A. A., Hemdan, A. M., Hassanien, A. E. (2016). Training feedforward neural networks using sine-cosine algorithm to improve the prediction of liver enzymes on fish farmed on nano-selenite. *2016 12th International Computer Engineering Conference (ICENCO)*, pp. 35–40. Cairo, Egypt, IEEE.

57. Sindhu, R., Ngadiran, R., Yacob, Y. M., Zahri, N. A. H., Hariharan, M. (2017). Sine–cosine algorithm for feature selection with elitism strategy and new updating mechanism. *Neural Computing and Applications, 28(10),* 2947–2958. https://doi.org/10.1007/s00521-017-2837-7

58. Liu, G., Jia, W., Wang, M., Heidari, A. A., Chen, H. et al. (2020). Predicting cervical hyperextension injury: A covariance guided sine cosine support vector machine. *IEEE Access, 8,* 46895–46908. https://doi.org/10.1109/Access.6287639

59. Das, S., Bhattacharya, A., Chakraborty, A. K. (2018). Solution of short-term hydrothermal scheduling using sine cosine algorithm. *Soft Computing, 22(19),* 6409–6427. https://doi.org/10.1007/s00500-017-2695-3

60. Oliva, D., Hinojosa, S., Elaziz, M. A., Ortega-Sánchez, N. (2018). Context based image segmentation using antlion optimization and sine cosine algorithm. *Multimedia Tools and Applications, 77(19),* 25761–25797. https://doi.org/10.1007/s11042-018-5815-x

61. Tawhid, M. A., Savsani, V. (2019). Multi-objective sine-cosine algorithm (MO-SCA) for multi-objective engineering design problems. *Neural Computing and Applications, 31(2),* 915–929. https://doi.org/10.1007/s00521-017-3049-x

62. Issa, M., Hassanien, A. E., Oliva, D., Helmi, A., Ziedan, I. et al. (2018). ASCA-PSO: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment. *Expert Systems with Applications, 99,* 56–70. https://doi.org/10.1016/j.eswa.2018.01.019

63. Khalilpourazari, S., Pasandideh, S. H. R. (2020). Sine–cosine crow search algorithm: Theory and applications. *Neural Computing and Applications, 32(12),* 1–18.

64. Chen, K., Zhou, F., Yin, L., Wang, S., Wang, Y. et al. (2018). A hybrid particle swarm optimizer with sine cosine acceleration coefficients. *Information Sciences, 422,* 218–241. https://doi.org/10.1016/j.ins.2017.09.015

65. Nenavath, H., Jatoth, R. K. (2018). Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking. *Applied Soft Computing, 62,* 1019–1043. https://doi.org/10.1016/j.asoc.2017.09.039

66. Elaziz, M. A., Oliva, D., Xiong, S. (2017). An improved opposition-based sine cosine algorithm for global optimization. *Expert Systems with Applications, 90,* 484–500. https://doi.org/10.1016/j.eswa.2017.07.043

67. Chen, H., Wang, M., Zhao, X. (2020). A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems. *Applied Mathematics and Computation, 369,* 124872. https://doi.org/10.1016/j.amc.2019.124872

68. Chen, H., Heidari, A. A., Zhao, X., Zhang, L., Chen, H. (2020). Advanced orthogonal learning-driven multi-swarm sine cosine optimization: Framework and case studies. *Expert Systems with Applications, 144,* 113113. https://doi.org/10.1016/j.eswa.2019.113113

69. Guo, W., Wang, Y., Zhao, F., Dai, F. (2019). Riesz fractional derivative elite-guided sine cosine algorithm. *Applied Soft Computing, 81,* 105481. https://doi.org/10.1016/j.asoc.2019.04.044
70. Gupta, S., Deep, K. (2019). Improved sine cosine algorithm with crossover scheme for global optimization. *Knowledge-Based Systems, 165,* 374–406. https://doi.org/10.1016/j.knosys.2018.12.008
71. Gupta, S., Deep, K. (2019). A hybrid self-adaptive sine cosine algorithm with opposition based learning. *Expert Systems with Applications, 119,* 210–230. https://doi.org/10.1016/j.eswa.2018.10.050
72. Long, W., Wu, T., Liang, X., Xu, S. (2019). Solving high-dimensional global optimization problems using an improved sine cosine algorithm. *Expert Systems with Applications, 123,* 108–126. https://doi.org/10.1016/j.eswa.2018.11.032
73. Chen, H., Yang, C., Heidari, A. A., Zhao, X. (2020). An efficient double adaptive random spare reinforced whale optimization algorithm. *Expert Systems with Applications, 154,* 113018. https://doi.org/10.1016/j.eswa.2019.113018
74. Abd Elaziz, M. E., Ewees, A. A., Oliva, D., Duan, P., Xiong, S. (2017). A hybrid method of sine cosine algorithm and differential evolution for feature selection. *Neural Information Processing*, Cham: Springer International Publishing.
75. Qu, C., Zeng, Z., Dai, J., Yi, Z., He, W. (2018). A modified sine-cosine algorithm based on neighborhood search and greedy levy mutation. *Computational Intelligence and Neuroscience, 2018*. https://doi.org/10.1155/2018/4231647
76. Hussien, A. G., Heidari, A. A., Ye, X., Liang, G., Chen, H. et al. (2022). Boosting whale optimization with evolution strategy and Gaussian random walks: An image segmentation method. *Engineering with Computers,* 1–45. https://doi.org/10.1007/s00366-021-01542-0
77. Su, H., Zhao, D., Yu, F., Heidari, A. A., Zhang, Y. et al. (2022). Horizontal and vertical search artificial bee colony for image segmentation of COVID-19 X-ray images. *Computers in Biology and Medicine, 142,* 105181. https://doi.org/10.1016/j.compbiomed.2021.105181
78. Zeng, G. Q., Lu, K. D., Dai, Y. X. Zhang, Z. J., Chen, M. R. et al. (2014). Binary-coded extremal optimization for the design of PID controllers. *Neurocomputing, 138,* 180–188. https://doi.org/10.1016/j.neucom.2014.01.046
79. Dong, R., Chen, H., Heidari, A. A., Turabieh, H., Mafarja, M. et al. (2021). Boosted kernel search: Framework, analysis and case studies on the economic emission dispatch problem. *Knowledge-Based Systems, 233,* 107529. https://doi.org/10.1016/j.knosys.2021.107529
80. Wu, S. H., Zhan, Z. H., Zhang, J. (2021). SAFE: Scale-adaptive fitness evaluation method for expensive optimization problems. *IEEE Transactions on Evolutionary Computation, 25(3),* 478–491. https://doi.org/10.1109/TEVC.2021.3051608
81. Zhao, F., Di, S., Cao, J., Tang, J., Jonrinaldi. (2021). A novel cooperative multi-stage hyper-heuristic for combination optimization problems. *Complex System Modeling and Simulation, 1(2),* 91–108. https://doi.org/10.23919/CSMS.2021.0010
82. Cai, Z., Gu, J., Luo, J., Zhang, Q., Chen, H. et al. (2019). Evolving an optimal kernel extreme learning machine by using an enhanced grey wolf optimization strategy. *Expert Systems with Applications, 138,* 112814. https://doi.org/10.1016/j.eswa.2019.07.031
83. Yu, H., Cheng, X., Chen, C., Heidari, A. A., Liu, J. et al. (2022). Apple leaf disease recognition method with improved residual network. *Multimedia Tools and Applications, 81(6),* 7759–7782. https://doi.org/10.1007/s11042-022-11915-2
84. Lai, X., Zhou, Y. (2020). Analysis of multiobjective evolutionary algorithms on the biobjective traveling salesman problem (1,2). *Multimedia Tools and Applications, 79(1–3),* 1–22. https://doi.org/10.1007/s11042-020-09399-z
85. Hu, J., Chen, H., Heidari, A. A., Wang, M., Zhang, X. et al. (2021). Orthogonal learning covariance matrix for defects of grey wolf optimizer: Insights, balance, diversity, and feature selection. *Knowledge-Based Systems, 213,* 106684. https://doi.org/10.1016/j.knosys.2020.106684

86. Hu, J., Gui, W., Heidari, A. A., Cai, Z., Liang, G. et al. (2022). Dispersed foraging slime mould algorithm: Continuous and binary variants for global optimization and wrapper-based feature selection. *Knowledge-Based Systems, 237,* 107761. https://doi.org/10.1016/j.knosys.2021.107761

87. Deng, W., Zhang, X., Zhou, Y., Liu, Y., Zhou, X. et al. (2022). An enhanced fast non-dominated solution sorting genetic algorithm for multi-objective problems. *Information Sciences, 585,* 441–453. https://doi.org/10.1016/j.ins.2021.11.052

88. Tu, L., Yang, L. Z. (2013). Feature selection based on ant colony optimization for image classification. *Applied Mechanics and Materials, 319*, 337–342.

89. Zhao, X., Li, D., Yang, B., Chen, H., Yang, X. et al. (2015). A two-stage feature selection method with its application. *Computers & Electrical Engineering, 47,* 114–125. https://doi.org/10.1016/j.compeleceng.2015.08.011

90. Deng, W., Liu, H., Xu, J., Zhao, H., Song, Y. (2020). An improved quantum-inspired differential evolution algorithm for deep belief network. *IEEE Transactions on Instrumentation and Measurement, 69(10),* 7319–7327.

91. Zhao, H., Liu, H., Xu, J., Deng, W. (2019). Performance prediction using high-order differential mathematical morphology gradient spectrum entropy and extreme learning machine. *IEEE Transactions on Instrumentation and Measurement, 69(7),* 4165–4172.

92. Shi, B., Ye, H., Zheng, L., Lyu, J., Chen, C. et al. (2021). Evolutionary warning system for COVID-19 severity: Colony predation algorithm enhanced extreme learning machine. *Computers in Biology and Medicine, 136,* 104698. https://doi.org/10.1016/j.compbiomed.2021.104698

93. Sun, Y., Xue, B., Zhang, M., Yen, G. G. (2020). Evolving deep convolutional neural networks for image classification. *IEEE Transactions on Evolutionary Computation, 24(2),* 394–407. https://doi.org/10.1109/TEVC.4235

94. Qiao, K., Liang, J., Yu, K., Yuan, M., Qu, B. et al. (2022). Self-adaptive resources allocation-based differential evolution for constrained evolutionary optimization. *Knowledge-Based Systems, 235,* 107653. https://doi.org/10.1016/j.knosys.2021.107653

95. Liang, J., Ban, X., Yu, K., Qu, B., Qiao, K. (2021). Differential evolution with rankings-based fitness function for constrained optimization problems. *Applied Soft Computing, 113,* 108016. https://doi.org/10.1016/j.asoc.2021.108016

96. Gao, D., Wang, G. G., Pedrycz, W. (2020). Solving fuzzy job-shop scheduling problem using de algorithm improved by a selection mechanism. *IEEE Transactions on Fuzzy Systems, 28(12),* 3265–3275. https://doi.org/10.1109/TFUZZ.91

97. Wang, G. G., Gao, D., Pedrycz, W. (2022). Solving multiobjective fuzzy job-shop scheduling problem by a hybrid adaptive differential evolution algorithm. *IEEE Transactions on Industrial Informatics, 18(12),* 8519–8528. https://doi.org/10.1109/TII.2022.3165636

98. Hu, K., Ye, J., Fan, E., Shen, S., Huang, L. et al. (2017). A novel object tracking algorithm by fusing color and depth information based on single valued neutrosophic cross-entropy. *Journal of Intelligent & Fuzzy Systems, 32(3),* 1775–1786. https://doi.org/10.3233/JIFS-152381

99. Hu, K., He, W., Ye, J., Zhao, L., Peng, H. et al. (2019). Online visual tracking of weighted multiple instance learning via neutrosophic similarity-based objectness estimation. *Symmetry, 11(6),* 832. https://doi.org/10.3390/sym11060832

100. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-based Systems, 96,* 120–133. https://doi.org/10.1016/j.knosys.2015.12.022

101. Vaishnav, P. K., Sharma, S., Sharma, P. (2021). Analytical review analysis for screening COVID-19 disease. *International Journal of Modern Research, 1(1),* 22–29.

102. Li, C., Li, J., Chen, H., Heidari, A. A. (2021). Memetic harris hawks optimization: Developments and perspectives on project scheduling and qos-aware web service composition. *Expert Systems with Applications, 171,* 114529. https://doi.org/10.1016/j.eswa.2020.114529

103. Ren, H., Li, J., Chen, H., Li, C. (2021). Stability of salp swarm algorithm with random replacement and double adaptive weighting. *Applied Mathematical Modelling, 95,* 503–523. https://doi.org/10.1016/j.apm.2021.02.002
104. Wilcoxon, F. (1992). Individual comparisons by ranking methods. In: Kotz, S., Johnson, N. L., *Breakthroughs in statistics: Methodology and distribution*, pp. 196–202. New York, NY: Springer.
105. Çelik, Y., Kutucu, H. (2018). Solving the tension/compression spring design problem by an improved firefly algorithm. *IDDM, 1(2255),* 1–7.
106. Deb, K. (1991). Optimal design of a welded beam via genetic algorithms. *AIAA Journal, 29(11),* 2013–2015. https://doi.org/10.1016/C2010-0-67103-3
107. Moss, D. R. (2004). *Pressure vessel design mannual*. Elsevier. https://doi.org/10.2514/3.10834
108. Morton, S., Webber, J. (1994). Optimal design of a composite i-beam. *Composite Structures, 28(2),* 149–168. https://doi.org/10.1016/0263-8223(94)90045-0
109. Gupta, V. K., Surendra Kumar, S., Anupriya, Ramesh Singh, R. (2022). Crime tracking system and people's safety in India using machine learning approaches. *International Journal of Modern Research, 2(1),* 1–7.
110. Shukla, S. K., Gupta, V. K., Joshi, K., Gupta, A., Singh, M. K. (2022). Self-aware execution environment model (SAE2) for the performance improvement of multicore systems. *International Journal of Modern Research, 2(1),* 17–27.

Appendix A. Tension/compression spring design problem

Minimize: $f(x) = (x_3 + 2)\, x_2 x_1^2$

Subject to: $g_1(x) = 1 - \left(x_2^3 x_3 / 71,785 x_1^4\right) \leq 0$

$g_2(x) = \left(4x_2^2 - x_1 x_2 / 12,566\left(x_2 x_1^3 - x_1^4\right) + \left(1/5108 x_1^2\right)\right) - 10 \leq 0$

$g_3(x) = 1 - \left(140.45 x_1 / x_2^2 x_3\right) \leq 0$

$g_4(x) = (x_2 + x_1)/1.5 - 1 \leq 0,$

Variable Range

$0.05 \leq x_1 \leq 2.00$

$0.25 \leq x_2 \leq 1.30$

$2.00 \leq x_3 \leq 15.00$

Appendix B. Welded beam design problem

Minimize: $f_1(x) = 1.10471 * x(1)^2 * x(2) + 0.04811 * x(3) * x(4) * (14.0 + *x(2))$

Subject to: $g_1(x) = \tau - 13600$

$g_2(x) = \boldsymbol{\sigma} - 30000$

$g_3(x) = x(1) - x(4)$

$g_4(x) = 6000 - p$

Variable Range

$0.125 \leq x_1 \leq 5$

$0.1 \leq x_2 \leq 10$

$0.1 \leq x_3 \leq 10$

$0.125 \leq x_4 \leq 5$

Appendix C. Pressure vessel design problem

Minimize: $f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

Subject to: $g_1(x) = -x_1 + 0.0193x$

$g_2(x) = -x_2 + 0/00954x_3 \leq 0$

$g_3(x) = -\pi x_3^2x_4 - (4/3)\pi x_3^3 + 1,296,000 \leq 0$

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

Variable Range

$0 \leq x_i \leq 100,\ i = 1, 2$

$0 \leq x_i \leq 200,\ i = 3, 4$

Appendix D. I-beam design problem

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

Subject to:

$g = 2bt_w + t_w(h - 2t_f) \leq 0$

Variable Range

$10 \leq x_1 \leq 50$

$10 \leq x_2 \leq 80$

$0.9 \leq x_3 \leq 5$

$0.9 \leq x_4 \leq 5$

Figure S1: Balance analysis of RDSCA and SCA
Table S1: CEC 2017 benchmark functions
Table S2: The calculated p-values from the Wilcoxon signed-rank test
Table S3: Results of TCSD problem
Table S4: Results of WBD problem
Table S5: Results of PVD problem
Table S6: Results of IBD problem