A Novel Algorithm for Global Optimization: Rat Swarm Optimizer

Gaurav Dhiman[1,*], Meenakshi Garg[1], Atulya Nagar[2], Vijay Kumar[3], Mohammad Dehghani[4]
[1,*]Department of Computer Science, Government Bikram College of Commerce, Patiala-147001, Punjab, INDIA
Email: gdhiman0001@gmail.com
[1]Department of Computer Science, Government Bikram College of Commerce, Patiala-147001, Punjab, INDIA
Email: meenagarg82@gmail.com
[2]Pro-Vice-Chancellor for Research and Dean of the Faculty of Science, Liverpool Hope University, Hope Park, Liverpool L16 9JD, UNITED KINGDOM
Email: atulya.nagar@hope.ac.uk
[3]Department of Computer Science and Engineering, National Institute of Technology, Hamirpur-177001, Himachal Pradesh, INDIA
Email: vijaykumarchahar@gmail.com
[4]Department of Electrical and Electronics Engineering, Shiraz University of Technology, Shiraz, IRAN
Email: adanbax@gmail.com

Abstract
This paper presents a novel bio-inspired optimization algorithm called Rat Swarm Optimizer (RSO) for solving the challenging optimization problems. The main inspiration of this optimizer is the chasing and attacking behaviors of rats in nature. This paper mathematically models these behaviors and benchmarks on a set of 38 test problems to ensure its applicability on different regions of search space. The RSO algorithm is compared with eight well-known optimization algorithms to validate its performance. It is then employed on six real-life constrained engineering design problems. The convergence and computational analysis are also investigated to test exploration, exploitation, and local optima avoidance of proposed algorithm. The experimental results reveal that the proposed RSO algorithm is highly effective in solving real world optimization problems as compared to other well-known optimization algorithms.
Note that the source codes of the proposed technique are available at: `http://www.dhimangaurav.com`
Keywords: Optimization; Metaheuristics; Swarm-intelligence; Benchmark test functions; Engineering design problems.

1. Introduction

For real world problems, stochastic optimization methods have been employed for solving various combinatorial problems. These optimization problems are non-linear, multimodal, computationally expensive, and possess large solution spaces to solve traditional methods (A. Kaur, Jain, & Goel, 2017, 2019, n.d.; A. Kaur, 2019; H. Kaur et al., 2019; Dhiman & Kumar, 2017a; Singh & Dhiman, 2018a; Dhiman & Kumar, 2018d; Singh & Dhiman, 2018b). Metaheuristic algorithms are able to solve such complex problems (Che, Liu, & Yu, 2019; Dhiman & Kumar, 2018b; Dhiman & Kaur, 2018; Singh, Rabadiya, & Dhiman, 2018; Dhiman & Kumar, 2018a; A. Kaur, Kaur, & Dhiman, 2018; Singh, Dhiman, & Kaur, 2018; Li, He, & Li, 2019; Asghari, Rahmani, & Javadi, 2020; Ramirez-Atencia & Camacho, 2019) in a reasonable amount of time. Nowadays, there has been a lot of interest to develop metaheuristic optimization algorithms (Dhiman, Guo, & Kaur, 2018; Dhiman & Kumar, 2019b; Dhiman & Kaur, 2019b; Dhiman & Kumar, 2019a; Dhiman, Singh, Kaur, & Maini, 2019; Dhiman, 2019b; Singh et al., 2019; Dhiman, 2019a, 2019c; Dehghani, Montazeri, Malik, Dhiman, & Kumar, 2019; Maini & Dhiman, 2018; Pallavi & Dhiman, 2018; Garg & Dhiman, 2020; S. Kaur, Awasthi, Sangal, & Dhiman, 2020) which are computationally inexpensive, flexible, and gradient free (Ragmani, Elomri, Abghour, Moussaid, & Rida, 2019; D. Yang, Wang, Tian, & Zhang, 2020; Balasubramanian & Marichamy, 2020). These techniques have been classified into three categories (Dhiman, Soni, Pandey, Slowik, & Kaur, 2020; Dehghani et al., 2020; Chandrawat, Kumar, Garg, Dhiman, & Kumar, 2017; Singh & Dhiman, 2017; Dhiman & Kaur, 2017; Verma, Kaur, Dhiman, & Kaur, 2018; A. Kaur & Dhiman, 2019; Dhiman & Kaur, 2019a; Dhiman & Kumar, 2019c): Evolutionary based, Physical based, and Swarm-intelligence based algorithms.

Evolutionary based algorithms mimic the evolutionary

processes in nature such as reproduction, mutation, recombination, and selection. These algorithms are based on the survival of fittest candidate in a population for a given environment. This process continues over a number of generations until the satisfactory solution is not found. Some of the most popular evolutionary algorithms are Genetic Programming (GP) (Koza, 1992), Genetic Algorithms (GA) (Bonabeau, Dorigo, & Theraulaz, 1999), Differential Evolution (DE) (Storn & Price, 1997), Evolution Strategy (ES) (Beyer & Schwefel, 2002), and Biogeography-Based Optimizer (BBO) (Simon, 2008).

The physical based algorithms are inspired by physical processes. These processes are defined according to physics rules such as electromagnetic force, gravitational force, heating and cooling of materials, inertia force, and so on. The few of the popular physical based algorithms are Gravitational Search Algorithm (GSA) (Rashedi, Nezamabadi-pour, & Saryazdi, 2009), Simulated Annealing (SA) (Kirkpatrick, Gelatt, & Vecchi, 1983), Charged System Search (CSS) (Kaveh & Talatahari, 2010a), Big-Bang Big-Crunch (BBBC) (Erol & Eksin, 2006), Black Hole (BH) (Hatamlou, 2013) algorithm, Artificial Chemical Reaction Optimization Algorithm (ACROA) (Alatas, 2011), Ray Optimization (RO) algorithm (Kaveh & Khayatazad, 2012), Small-World Optimization Algorithm (SWOA) (Du, Wu, & Zhuang, 2006), Curved Space Optimization (CSO) (Moghaddam, Moghaddam, & Cheriet, 2012), Central Force Optimization (CFO) (Formato, 2009), and Galaxy-based Search Algorithm (GbSA) (Shah Hosseini, 2011).

The swarm-intelligence based algorithms are inspired by the collective intelligence of groups. This intelligence is present in flock of birds, school of fishes, ant colonies, and the like. The most popular algorithm of swarm-intelligence technique is Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995) which is inspired by the social behaviors of fish, birds, animals and so forth in nature. Each particle in this algorithm can move throughout the search space and update its current position with respect to global best solution (Anitha & Kaarthick, 2019). There are also other popular swarm-intelligence based techniques which are Ant Colony Optimization (Dorigo, Birattari, & Stutzle, 2006), Bee Collecting Pollen Algorithm (BCPA) (Lu & Zhou, 2008), Wolf pack search algorithm (C. Yang, Tu, & Chen, 2007), Monkey Search (Mucherino & Seref, 2007), Dolphin Partner Optimization (DPO) (Shiqin, Jianjun, & Guangxing, 2009), Cuckoo Search (CS) (X. S. Yang & Deb, 2009), Firefly Algorithm (FA) (X.-S. Yang, 2010a), Bat-inspired Algorithm (BA) (X.-S. Yang, 2010b), Spotted Hyena Optimizer (SHO) (Dhiman & Kumar, 2017b), and Hunting Search (HUS) (Oftadeh, Mahjoob, & Shariatpanahi, 2010).

In addition, there are also some other advantages of swarm-intelligence based algorithms such as: (1) These algorithms includes very few operators as compared to evolutionary based algorithms; (2) Swarm-intelligence based algorithms are able to maintain the information about the whole search space and very easy to implement; (3) These algorithms have less parameters that makes the algorithms utilize less memory space; (4) The computational efficiency of these algorithms is low as compared to other metaheuristics. There are other swarm-intelligence techniques which are listed in Table 1.

However, every optimization algorithm needs to address and maintains a good balance between the exploration and exploitation phases of a search space (Alba & Dorronsoro, 2005; Olorunda & Engelbrecht, 2008). The exploration phase investigates the different promising regions in a given search space whereas in exploitation phase the optimal solutions are searched around the promising regions (Lozano & Garcia-Martinez, 2010). Whereas, the performance of one optimization algorithm does not guarantee to to be equally good for other real-life problems (Wolpert & Macready, 1997). Therefore, proper balancing between the exploration/exploitation motivates us to develop a novel swarm-intelligence based optimization algorithm for solving real-life approaches. This paper presents a novel bio-inspired based metaheuristic algorithm named as Rat Swarm Optimizer (RSO) for global optimization problems. The Rat Swarm Optimizer (RSO) is inspired by the chasing and attacking behaviors of rats. The performance of RSO algorithm is tested on thirty-eight benchmark test functions and six real constrained optimization design problems. The results reveal that the performance of RSO is better than the other well-known optimization algorithms.

The rest of this paper is structured as follows: Section 2 presents the proposed RSO algorithm. Section 3 covers the results and discussion. In Section 4, the performance of RSO is tested on six constrained engineering design problems and compared it with other competitor algorithms. Finally, the conclusion and some future research directions are given in Section 5.

2. Rat Swarm Optimizer (RSO)

2.1. Inspiration

Rats are long tailed and medium sized rodents which are different in terms of size and weight. There are two main species of rat: Black rat and Brown rat. In rats family, the male rats are called bucks while female rats are called does. Rats are generally socially intelligent by nature. They groom each other and involve in various activities such as jumping, chasing, tumbling, and boxing. Rats are territorial animals which live in a group of both males and females. The behavior of rats is very aggressive in many cases which may result in the death of some animals. This aggressive behavior is the main motivation of this work while chasing

and fighting with prey. In this research, the chasing and fighting behaviors of rats are mathematically modeled to design RSO algorithm and perform optimization.

2.2. Mathematical model and optimization algorithm

This subsection describes the behavior of rat i.e., chasing and fighting. Then the proposed RSO algorithm is outlined.

2.2.1. Chasing the prey. Generally, rats are social animals who chase the prey in a group through their social agonistic behavior. To define this behavior mathematically, we assume that the best search agent has the knowledge of location of prey. The other search agents can update their positions with respect to best search agent obtained so far. The following equations are proposed in this mechanism:

$$\vec{P} = A \cdot \vec{P_i}(x) + C \cdot (\vec{P_r}(x) - \vec{P_i}(x)) \tag{1}$$

where $\vec{P_i}(x)$ defines the positions of rats and $\vec{P_r}(x)$ is the best optimal solution.

However, A and C parameters are calculated as follows:

$$A = R - x \times \Big(\frac{R}{Max_{Iteration}}\Big) \quad \text{where, } x = 0, 1, 2, \ldots, Max_{Iteration} \tag{2}$$

$$C = 2 \cdot rand() \tag{3}$$

Therefore, R and C are random numbers between $[1, 5]$ and $[0, 2]$, respectively. The parameters A and C are responsible for better exploration and exploitation over the course of iterations.

2.2.2. Fighting with prey. In order to mathematically define the fighting process of rats with prey, the following equation has been proposed:

$$\vec{P_i}(x + 1) = \mid \vec{P_r}(x) - \vec{P} \mid \tag{4}$$

where $\vec{P_i}(x + 1)$ defines the updated next position of rat. It saves the best solution and updates the positions of other search agents with respect to the best search agent. Fig. 1 shows the effect of Eqs. (1) and (4) in three dimensional environment. In this figure, the rat (A, B) can update its position towards the position of prey (A^*, B^*). By adjusting the parameters, as shown in Eqs. (2) and (3), the different number of positions can be reached about the current position. However, this concept can also be extended in n-dimensional environment.

Therefore, the exploration and exploitation are guaranteed by the adjusted value of parameters A and C. The proposed RSO algorithm saves the optimal solution with fewest operators. The pseudo code of the proposed RSO algorithm is presented in Algorithm.

Algorithm : Rat Swarm Optimizer

Input: the rats population P_i $(i = 1, 2, \ldots, n)$
Output: the optimal search agent
procedure RSO
Initialize the parameters A, C, and R
Calculate the fitness value of each search agent
$P_r \leftarrow$ the best search agent
  while $(x < Max_{Iteration})$ **do**
    for each search agent **do**
      Update the position of current search agent by Eq. (4)
    end for
    Update parameters A, C, and R
    Check if there is any search agent which goes beyond the given search space and then adjust it
    Calculate the fitness of each search agent
    Update P_r if there is a better solution than previous optimal solution
    $x \leftarrow x + 1$
  end while
return P_r
end procedure

2.3. Steps and flowchart of RSO

The steps and flowchart (see Fig. 2) of RSO are discussed below:

Step 1: Initialize the rats population P_i where $i = 1, 2, \ldots, n$.
Step 2: Choose the initial parameters of RSO: A, C, and R.
Step 3: Now, calculate the fitness value of each search agent.
Step 4: The best search agent is then explored in the given search space.
Step 5: Update the positions of search agents using Eq. (4).
Step 6: Check whether any search agent goes beyond the boundary limit of a search space and then amend it.
Step 7: Again, calculate the updated search agent fitness value and update the vector P_r if there is a better solution than previous optimal solution.
Step 8: Stop the algorithm if the stopping criteria is satisfied. Otherwise, return to Step 5.
Step 9: Return the best obtained optimal solution.

2.4. Computational complexity

In this subsection, the computational time and space complexity of proposed RSO algorithm are discussed.

2.4.1 Time complexity.

1. The initialization of RSO population needs $O(n \times d)$ time where n indicates the number of iterations and d

defines the dimension of a test function to adjust the solutions within the boundary.

2. In the next step, the fitness calculation of each search agent requires $O(Max_{Iteration} \times n \times d)$ time where $Max_{Iteration}$ is the maximum number of iterations to simulate the proposed RSO algorithm.

3. Repeat Steps 1 and 2 until the satisfactory results is found which needs $O(N)$ time.

Therefore, the overall time complexity of RSO algorithm is $O(Max_{Iteration} \times n \times d \times N)$.

2.4.2. Space complexity. The space complexity of RSO algorithm is the maximum amount of space to be utilized at any one time as considered during its initialization process. Hence, the total space complexity of RSO algorithm is $O(n \times d)$.

3. Experimental Results and Discussion

This section covers the experimentation on thirty eight benchmark test functions to demonstrate the performance of proposed RSO algorithm. The detailed description of these benchmarks are discussed below.

3.1. Benchmark test functions

The proposed algorithm is evaluated on thirty eight benchmark test functions which are divided into four main categories: Unimodal (Digalakis & Margaritis, 2001), Multimodal (X.-S. Yang, 2010a), Fixed-dimension Multimodal (Digalakis & Margaritis, 2001; X.-S. Yang, 2010a), and CEC-15 special session functions (Chen et al., 2014). These functions are described in Tables 3-6.

3.2. State-of-the-art algorithms for comparison

To validate the performance of the proposed RSO algorithm, the eight well-known optimization algorithms are used for comparison.

- **Spotted Hyena Optimizer (SHO) (Dhiman & Kumar, 2017b):** Spotted Hyena Optimizer (SHO) is a bio-inspired based optimization algorithm proposed by (Dhiman & Kumar, 2017b). It shows the searching, encircling, and hunting behaviors of spotted hyena in nature. The search agents can update their positions with a group of optimal solutions rather than one optimal solution. The algorithm was applied on constrained and unconstrained real-life engineering problems and benchmark test functions.

- **Grey Wolf Optimizer (GWO) (Mirjalili, Mirjalili, & Lewis, 2014):** GWO is a another bio-inspired based algorithm inspired by the behaviors of grey wolves. GWO employed four types of grey wolves: alpha, beta, delta, and omega which shows the hunting, searching, encircling, and attacking behaviors for optimization problems. Further, the performance of GWO was tested on well-known test functions and classical engineering design problems.

- **Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995):** Particle Swarm Optimization is a popular population based optimization algorithm which is inspired by the social behavior of birds and animals. In PSO, each particle can move around the search space with respect to global optimal solution and updates its current position. There are only a few parameters in this algorithm to adjust for better exploration and exploitation.

- **Moth-flame Optimization (MFO) (Mirjalili, 2015):** Moth-flame Optimization (MFO) is a bio-inspired optimization algorithm motivated by the navigation method of moths in nature. It maintains a fixed angle with respect to moon for travelling long distances. The performance of MFO was tested on constrained test problems to find the global optimum.

- **Multi-Verse Optimizer (MVO) (Mirjalili, Mirjalili, & Hatamlou, 2016):** Multi-verse Optimizer (MVO) is a physics based optimization algorithm proposed by (Mirjalili et al., 2016) which is inspired by the theory of multi-verse in physics and consists three main concepts i.e., white hole, black hole, and wormhole. The concepts of white hole and black hole are responsible for exploration and wormholes appropriate for exploitation in the search spaces.

- **Sine Cosine Algorithm (SCA) (Mirjalili, 2016):** Sine Cosine Algorithm (SCA) is proposed by Mirjalili that generates multiple solutions using mathematical model such as sine and cosine functions for solving optimization problems. The convergence behavior of SCA is very high and computational complexity is low which is helpful for local optima avoidance.

- **Gravitational Search Algorithm (GSA) (Rashedi et al., 2009):** Gravitational Search Algorithm (GSA) is proposed by (Rashedi et al., 2009) which is based on the Newton's law of gravitation and law of motion. This algorithm has an ability to find global optimum because it requires only few parameters such as position, inertial mass, active gravitational mass, and passive gravitational mass.

- **Genetic Algorithm (GA) (Bonabeau et al., 1999):** Genetic Algorithm (GA) is an evolutionary algorithm inspired by the theory of natural selection. It consists of three operators such as selection, crossover, and mutation to find the near optimal solutions.

3.3. Experimental setup

The parameter setting of proposed RSO algorithm and other optimization algorithms (i.e., SHO, GWO, PSO, MFO, MVO, SCA, GSA, and GA) is shown in the Table 7. The whole experimentation process and reported algorithms are implemented in Matlab R2018b version in the environment of Microsoft Windows 10 on Core i7 processor with 3.20 GHz and 16 GB memory.

3.4. Performance comparison

In order to demonstrate the performance of proposed RSO algorithm, its results are tested on unimodal, multimodal, fixed-dimension multimodal, and CEC-15 special session benchmark test functions.

3.4.1. Evaluation of functions $F_1 - F_7$ (*Exploitation*). The functions $F_1 - F_7$ are unimodal test problems which have the capability for better exploitation and find the best optimal solution. Table 8 reveals that RSO is very competitive as compared to other competitor algorithms. In particular, it maintains better results for functions F_2, F_5, and F_7.

3.4.2. Evaluation of functions $F_8 - F_{23}$ (*Exploration*). Multimodal test functions have the ability to determine the exploration of an optimization algorithm. Tables 9 and 10 show the results for functions multimodal and fixed-dimension multimodal functions which demonstrate the exploration capability of RSO algorithm. RSO is most efficient in nine test functions such as F_8, F_{10}, F_{11}, F_{14}, F_{15}, F_{16}, F_{17}, F_{18}, F_{19}, F_{20}, F_{22}, and F_{23} as well as very competitive results in rest of test problems.

3.4.3. Evaluation of CEC-15 functions ($CEC1 - CEC15$). This special session test suite is devoted to the real approaches for solving single objective optimization problems. These test functions are considered as black-box problems with bound constraints. Table 11 reveals that RSO algorithm is efficient for functions $CEC-1$, $CEC-3$, $CEC-7$, $CEC-8$, $CEC-9$, $CEC-10$, $CEC-11$, $CEC-12$, $CEC-13$, $CEC-14$, and $CEC-15$. The boxplot comparison and results on CEC benchmark test functions are shown in Fig. 3.

The results for functions $F_1 - F_{23}$ and $CEC-15$ show that RSO is the best optimizer for most of the cases as compared with other competitor algorithms.

3.5. Convergence analysis

The convergence curve analysis is investigated for better understanding the behaviors of RSO algorithm. These behaviors have been analysed into three stages.

In the initial stage, RSO converges very quickly through out the search space as shown in F_1, F_5, F_7, F_{11}, and F_{13} test functions.

In second stage, RSO converges towards the optimum during final iterations which is shown in F_{21} and F_{23} test functions.

In last step, RSO convergence very expressively from the initial steps of iterations as shown in functions F_3, F_9, F_{15}, F_{17}, and F_{19}.

These results reveal that RSO algorithm maintains a proper balance between exploration and exploitation to find the optimal results.

The convergence curves of RSO, SHO, GWO, PSO, MFO, MVO, SCA, GSA, and GA are compared and presented in Fig. 4 which shows that RSO is very competitive and high success rate as compared with other metaheuristic techniques for solving optimization problems.

3.6. Scalability study

This subsection presents the scalability analysis on various benchmark test functions. The dimensionality of these test functions varies from 30-50, 50-80, and 80-100. Fig. 5 shows the performance of RSO algorithm with different behaviors on different dimensionality. It has been observed that the proposed algorithm is applicable on high dimensional environment.

3.7. Statistical testing

Apart from the basic statistical analysis, the Wilcoxon ranksum test statistical test is performed at 5% level of significance. The $p-$values, which are less than 0.05, demonstrate the superiority of RSO algorithm. The results of the Wilcoxon ranksum test test are tabulated in Table 2. Overall, the results reveal that RSO performs better than other optimization algorithms in the literature.

4. RSO for Engineering Design Problems

In this section, six real-life constrained engineering design problems have been discussed. These problems are pressure vessel, speed reducer, welded beam, tension/compression spring, 25-bar truss, and rolling element bearing design problems. These optimization problems have different constraints and handles infeasible solutions with low computational efforts (Coello, 2002). These problems are compared with other reported algorithms in the literature to validate the performance of proposed algorithm.

4.1. Pressure vessel design

This problem was proposed by Kannan and Kramer (Kannan & Kramer, 1994) to minimize the total cost of material. The schematic view of pressure vessel is shown in Fig. 6 which are capped at both the ends by hemispherical heads. There are four design variables in this problem ($y_1 - y_4$):

- (y_1, thickness of the shell) T_s.

- (y_2, thickness of the head) T_h.
- (y_3, inner radius) R.
- (y_4, length of the cylindrical section) L.

where R and L are continuous design variables and T_s and T_h are integer numeric values which are multiples of 0.0625 *in*. The mathematical formulation of this problem is defined as follows:

$$\begin{aligned}
&\text{Consider } \vec{y} = [y_1\ y_2\ y_3\ y_4] = [T_s\ T_h\ R\ L],\\
&\text{Minimize } f(\vec{y}) = 0.6224y_1y_3y_4 + 1.7781y_2y_3^2\\
&+ 3.1661y_1^2y_4 + 19.84y_1^2y_3,\\
&\text{Subject to}\\
&g_1(\vec{y}) = -y_1 + 0.0193y_3 \le 0,\\
&g_2(\vec{y}) = -y_3 + 0.00954y_3 \le 0,\\
&g_3(\vec{y}) = -\pi y_3^2y_4 - \frac{4}{3}\pi y_3^3 + 1,296,000 \le 0,\\
&g_4(\vec{y}) = y_4 - 240 \le 0,\\
&\text{Variable range}\\
&0 \le y_1 \le 99,\\
&0 \le y_2 \le 99,\\
&0 \le y_3 \le 200,\\
&0 \le y_4 \le 200,
\end{aligned} \tag{5}$$

From Table 12, RSO obtains the best optimal solution among other reported algorithms such as SHO, GWO, PSO, MFO, MVO, SCA, GSA, and GA. According to the results, RSO achieves near optimal design with minimum cost.

Whereas, Table 13 represents the statistical results for pressure vessel design problem. The results show that RSO outperforms all other competitor algorithms. The convergence behavior of this design problem is shown in Fig. 7 which reveals that proposed algorithm is able to converge very efficiently in the initial steps of iterations.

4.2. Speed reducer design problem

The speed reducer design problem is a more challenging problem because it has seven design variables (Gandomi & Yang, 2011). This optimization problem is a minimization problem which can minimize the weight of speed reducer as shown in Fig. 8. The constraints of this design problem are (Mezura-Montes & Coello, 2005):

- Bending stress of the gear teeth.
- Surface stress.
- Transverse deflections of the shafts.
- Stresses in the shafts.

There are seven design variables ($y_1 - y_7$) which are face width (b), module of teeth (m), number of teeth in the pinion (z), length of the first shaft between bearings (l_1), length of the second shaft between bearings (l_2), the diameter of first (d_1) shafts, and the diameter of second shafts (d_2).

The mathematical formulation of this problem is formulated as follows:

$$\begin{aligned}
&\text{Minimize } f(\vec{y}) = 0.7854y_1y_2^2(3.3333y_3^2 + 14.9334y_3 - 43.0934)\\
&- 1.508y_1(y_6^2 + y_7^2) + 7.4777(y_6^3 + y_7^3) + 0.7854(y_4y_6^2 + y_5y_7^2),\\
&\text{Subject to}\\
&g_1(\vec{y}) = \frac{27}{y_1y_2^2y_3} - 1 \le 0,\\
&g_2(\vec{y}) = \frac{397.5}{y_1y_2^2y_3^2} - 1 \le 0,\\
&g_3(\vec{y}) = \frac{1.93y_4^3}{y_2y_6^4y_3} - 1 \le 0,\\
&g_4(\vec{y}) = \frac{1.93y_5^3}{y_2y_7^4y_3} - 1 \le 0,\\
&g_5(\vec{y}) = \frac{[(745(y_4/y_2y_3))^2 + 16.9\times10^6]^{1/2}}{110y_6^3} - 1 \le 0,\\
&g_6(\vec{y}) = \frac{[(745(y_5/y_2y_3))^2 + 157.5\times10^6]^{1/2}}{85y_7^3} - 1 \le 0,\\
&g_7(\vec{y}) = \frac{y_2y_3}{40} - 1 \le 0,\\
&g_8(\vec{y}) = \frac{5y_2}{y_1} - 1 \le 0,\\
&g_9(\vec{y}) = \frac{y_1}{12y_2} - 1 \le 0,\\
&g_{10}(\vec{y}) = \frac{1.5y_6 + 1.9}{y_4} - 1 \le 0,\\
&g_{11}(\vec{y}) = \frac{1.1y_7 + 1.9}{y_5} - 1 \le 0,\\
&\textit{where},\\
&2.6 \le y_1 \le 3.6,\ 0.7 \le y_2 \le 0.8,\ 17 \le y_3 \le 28,\ 7.3 \le y_4 \le 8.3,\\
&7.3 \le y_5 \le 8.3,\ 2.9 \le y_6 \le 3.9,\ 5.0 \le y_7 \le 5.5
\end{aligned} \tag{6}$$

The comparison results with various optimization algorithms for the best obtained optimal solution are tabulated in Table 14 and the statistical results are given in Table 15. To analyze these results, it concludes that RSO algorithm is best optimizer for speed reducer design problem. While, RSO algorithm obtains best convergence behavior during number of generations and achieves better results than other competitor methods as shown in Fig. 9.

4.3. Welded beam design

The objective of this design problem is to minimize the fabrication cost of the welded beam (see Fig. 10). The optimization constraints of welded beam are shear stress (τ) and

bending stress (θ) in the beam, buckling load (P_c) on the bar, end deflection (δ) of the beam. There are four optimization design variables of this problem which are as follows:

- Thickness of weld (h)
- Length of the clamped bar (l)
- Height of the bar (t)
- Thickness of the bar (b)

The mathematical formulation is described as follows:

$$\begin{aligned}
&\text{Consider } \vec{y} = [y_1\ y_2\ y_3\ y_4] = [h\ l\ t\ b],\\
&\text{Minimize } f(\vec{y}) = 1.10471 y_1^2 y_2 + 0.04811 y_3 y_4 (14.0 + y_2),\\
&\text{Subject to}\\
&g_1(\vec{y}) = \tau(\vec{y}) - \tau_{max} \le 0,\\
&g_2(\vec{y}) = \sigma(\vec{y}) - \sigma_{max} \le 0,\\
&g_3(\vec{y}) = \delta(\vec{y}) - \delta_{max} \le 0,\\
&g_4(\vec{y}) = y_1 - y_4 \le 0,\\
&g_5(\vec{y}) = P - P_c(\vec{y}) \le 0,\\
&g_6(\vec{y}) = 0.125 - y_1 \le 0,\\
&g_7(\vec{y}) = 1.10471 y_1^2 + 0.04811 y_3 y_4 (14.0 + y_2) - 5.0 \le 0,
\end{aligned}$$

Variable range

$$\begin{aligned}
&0.05 \le y_1 \le 2.00,\\
&0.25 \le y_2 \le 1.30,\\
&2.00 \le y_3 \le 15.0,
\end{aligned} \tag{7}$$

The comparison results for the best obtained solution by proposed and reported algorithms (i.e., SHO, GWO, PSO, MVO, SCA, GSA, GA, and HS) are presented in Table 16. The statistical results of the proposed and competitor algorithms is given in Table 17 which reveals the better performance of RSO and requires low computational cost to find the best optimal design.

By observing Fig. 11, RSO achieves the far optimal solution and high success rate for welded beam design problem.

4.4. Tension/compression spring design problem

The objective of this problem is to minimize the tension/compression spring weight as shown in Fig. 12. The optimization constraints of this problem are:

- Shear stress.
- Surge frequency.
- Minimum deflection.

There are three design variables of this problem: wire diameter (d), mean coil diameter (D), and the number of active coils (N). The mathematical formulation of this problem is described as follows:

$$\begin{aligned}
&\text{Consider } \vec{y} = [y_1\ y_2\ y_3] = [d\ D\ N],\\
&\text{Minimize } f(\vec{y}) = (y_3 + 2) y_2 y_1^2,\\
&\text{Subject to}\\
&g_1(\vec{y}) = 1 - \frac{y_2^3 y_3}{71785 y_1^4} \le 0,\\
&g_2(\vec{y}) = \frac{4y_2^2 - y_1 y_2}{12566(y_2 y_1^3 - y_1^4)} + \frac{1}{5108 y_1^2} \le 0,\\
&g_3(\vec{y}) = 1 - \frac{140.45 y_1}{y_2^2 y_3} \le 0,\\
&g_4(\vec{y}) = \frac{y_1 + y_2}{1.5} - 1 \le 0,
\end{aligned} \tag{8}$$

The best obtained solution by above mentioned competitor and proposed algorithm is given in Table 18. The results show that RSO performs better than other optimization algorithms. The statistical analysis of tension/compression spring design is presented in Table 19 which shows the efficiency of RSO to find the best optimal design.

In Fig. 13, RSO algorithm achieves the near optimal solution during the initial stage of iterations and yields better results than other optimization algorithms.

4.5. 25-bar truss design

The truss design problem is very popular optimization problem in the literature. As shown in Fig. 16, there are 10 nodes which are fixed and 25 bars cross-sectional members which are grouped into eight categories:

- Group 1: A_1
- Group 2: A_2, A_3, A_4, A_5
- Group 3: A_6, A_7, A_8, A_9
- Group 4: A_{10}, A_{11}
- Group 5: A_{12}, A_{13}
- Group 6: A_{14}, A_{15}, A_{17}
- Group 7: $A_{18}, A_{19}, A_{20}, A_{21}$
- Group 8: $A_{22}, A_{23}, A_{24}, A_{25}$

The other variables which effects on this problem are as follows:

- $p = 0.0272$ N/cm^3 (0.1 lb/in.^{3})
- $E = 68947$ MPa (10000 Ksi)
- Displacement limitation = 0.35 in.
- Maximum displacement = 0.3504 in.

- Design variable set = $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.6, 2.8, 3.0, 3.2, 3.4\}$

The member stress limitations for this truss is shown in Table 20.

The leading conditions of 25-bar truss is listed in Table 21. However, Table 22 reveals that RSO obtains best optimal weight which is better than other competitor algorithms.

The statistical results also show that RSO outperforms than other optimization algorithms. The effective convergence behavior of this problem is shown in Fig. 14.

4.6. Rolling element bearing design problem

The main objective of this problem is to maximize the dynamic load carrying capacity of a rolling element bearing as shown in Fig. 15. There are 10 decision variables of this design problem which are pitch diameter (D_m), ball diameter (D_b), number of balls (Z), inner (f_i) and outer (f_o) raceway curvature coefficients, K_{Dmin}, K_{Dmax}, ε, e, and ζ.

The mathematical formulation is described as follows:

$$
\begin{aligned}
&\text{Maximize } C_d = f_c Z^{2/3} D_b^{1.8} \qquad if\ D \le 25.4mm\\
&C_d = 3.647 f_c Z^{2/3} D_b^{1.4} \qquad if\ D > 25.4mm\\
&\text{Subject to}\\
&g_1(\vec{y}) = \frac{\phi_0}{2sin^{-1}(D_b/D_m)} - Z + 1 \le 0,\\
&g_2(\vec{y}) = 2D_b - K_{Dmin}(D-d) \ge 0,\\
&g_3(\vec{y}) = K_{Dmax}(D-d) - 2D_b \ge 0,\\
&g_4(\vec{y}) = \zeta B_w - D_b \le 0,\\
&g_5(\vec{y}) = D_m - 0.5(D+d) \ge 0,\\
&g_6(\vec{y}) = (0.5+e)(D+d) - D_m \ge 0,\\
&g_7(\vec{y}) = 0.5(D - D_m - D_b) - \varepsilon D_b \ge 0,\\
&g_8(\vec{y}) = f_i \ge 0.515,\\
&g_9(\vec{y}) = f_o \ge 0.515,\\
&where,\\
&f_c = 37.91\left[1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\right.\right.\\
&\left.\left.\left(\frac{f_i(2f_o-1)}{f_o(2f_i-1)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3}\\
&\times\left[\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{1/3}}\right]\left[\frac{2f_i}{2f_i-1}\right]^{0.41}\\
&x = [\{(D-d)/2 - 3(T/4)\}^2 + \{D/2 - T/4 - D_b\}^2 - \{d/2 + T/4\}^2]\\
&z = 2\{(D-d)/2 - 3(T/4)\}\{D/2 - T/4 - D_b\}\\
&\phi_o = 2\pi - 2cos^{-1}\left(\frac{x}{z}\right)\\
&\gamma = \frac{D_b}{D_m}, \quad f_i = \frac{r_i}{D_b}, \quad f_o = \frac{r_o}{D_b}, \quad T = D - d - 2D_b\\
&D = 160, \quad d = 90, \quad B_w = 30, \quad r_i = r_o = 11.033\\
&0.5(D+d) \le D_m \le 0.6(D+d), \quad 0.15(D-d) \le D_b\\
&\le 0.45(D-d), \quad 4 \le Z \le 50, \quad 0.515 \le f_i\ and\ f_o \le 0.6,\\
&0.4 \le K_{Dmin} \le 0.5, \quad 0.6 \le K_{Dmax} \le 0.7, \quad 0.3 \le e \le 0.4,\\
&0.02 \le e \le 0.1, \quad 0.6 \le \zeta \le 0.85
\end{aligned}
\tag{9}
$$

The comparison results of best optimal solution with different optimization algorithm is tabulated in Table 23. The statistical results for reported algorithms with proposed RSO is compared in Table 24.

In particular, Fig. 17 reveals that RSO algorithm is capable to achieve the near optimal solution. While analysing these results, the proposed RSO algorithm is detected as the best optimizer over other optimizers.

In summary, the results on the six real-life engineering design problems shows that RSO is able to solve various high-dimensional challenging problems and has the capability to handle various combinatorial optimization problems (COPs). Therefore, RSO is the best optimization algorithm under low computational costs and fast convergence speed towards the optimum.

5. Conclusion

This paper presents a novel swarm-intelligence based optimization algorithm called Rat Swarm Optimizer (RSO). The proposed RSO algorithm is tested on thirty eight benchmark test functions to evaluate the exploration and exploitation phases for avoiding local optimum.

The results on the unimodal and multimodal test functions reveal the superior exploitation and exploration capability of the RSO algorithm, respectively. Finally, the algorithm is benchmarked on very challenging CEC-15 special session with bound constraints benchmark test functions. The results show that the RSO is the best optimizer which provides very competitive results as compared with other well-known metaheuristics such as SHO, GWO, PSO, MFO, MVO, SCA, GSA, and GA. In particular, the computational complexity in terms of time and space complexity and convergence behavior have also been analyzed. The statistical measurements have been discussed to demonstrate the superiority of proposed algorithm as compared with other metaheuristics.

Moreover, the proposed algorithm has been applied on six real-life constrained engineering design problems (i.e., pressure vessel, speed reducer, welded beam, tension/compression spring, 25-bar truss, and rolling element bearing design) which shows that the RSO algorithm has high performance capability in unknown search spaces.

There are several research directions which can be recommended for future works. The binary version of the RSO algorithm is the one motivation for future work. Also, extension this algorithm to solve multi-objective as well as many-objective optimization problems can also be seen as a future contribution.

Conflict of interest

The authors declare that they have no conflict of interest.

Acknowledgement

The first and corresponding author **Dr. Gaurav Dhiman** would like to thanks to his father **Mr. Rajinder Dhiman** and mother **Mrs. Kamlesh Dhiman** for their support and divine blessings on him.

References

Alatas, B. (2011). Acroa: Artificial chemical reaction optimization algorithm for global optimization. *Expert Systems with Applications*, *38*(10), 13170 - 13180.

Alba, E., & Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *9*(2), 126-142.

Anitha, P., & Kaarthick, B. (2019). Oppositional based laplacian grey wolf optimization algorithm with svm for data mining in intrusion detection system. *Journal of Ambient Intelligence and Humanized Computing*, 1–12.

Asghari, P., Rahmani, A. M., & Javadi, H. H. S. (2020). Privacy-aware cloud service composition based on qos optimization in internet of things. *Journal of Ambient Intelligence and Humanized Computing*, 1–26.

Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. *Computers & Structures*, *169*, 1–12.

Askarzadeh, A., & Rezazadeh, A. (2013). A new heuristic optimization algorithm for modeling of proton exchange membrane fuel cell: bird mating optimizer. *International Journal of Energy Research*, *37*(10), 1196–1204.

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In *Ieee congress on evolutionary computation* (p. 4661-4667).

Balasubramanian, S., & Marichamy, P. (2020). An efficient medical data classification using oppositional fruit fly optimization and modified kernel ridge regression algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 1–11.

Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies – a comprehensive introduction. *Natural Computing*, *1*(1), 3–52.

Bichon, C. V. C. B. J. (2004). Design of space trusses using ant colony optimization. *Journal of Structural Engineering*, *130*(5), 741-751.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). Swarm intelligence: From natural to artificial systems. *Oxford University Press, Inc.*.

Chandrawat, R. K., Kumar, R., Garg, B., Dhiman, G., & Kumar, S. (2017). An analysis of modeling and optimization production cost through fuzzy linear programming problem with symmetric and right angle triangular fuzzy number. In *Proceedings of sixth international conference on soft computing for problem solving* (pp. 197–211).

Che, G., Liu, L., & Yu, Z. (2019). An improved ant colony optimization algorithm based on particle swarm optimization algorithm for path planning of autonomous underwater vehicle. *Journal of Ambient Intelligence and Humanized Computing*, 1–6.

Chen, Q., Liu, B., Zhang, Q., Liang, J., Suganthan, P., & Qu, B. (2014). Problem definitions and evaluation criteria for cec 2015 special session on bound constrained single-objective computationally expensive numerical optimization. *Technical Report, Nanyang Technological University*.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, *191*(11-12), 1245 - 1287.

Dai, C., Zhu, Y., & Chen, W. (2007). Seeker optimization algorithm. *International Conference on Computational Intelligence and Security*, 167–176.

Dehghani, M., Montazeri, Z., Malik, O., Al-Haddad, K., Guerrero, J. M., & Dhiman, G. (2020). A new methodology called dice game optimizer for capacitor placement in distribution systems. *Electrical Engineering and Electromechanics*(1), 61–64.

Dehghani, M., Montazeri, Z., Malik, O. P., Dhiman, G., & Kumar, V. (2019). Bosa: Binary orientation search algorithm. *International Journal of Innovative Technology and Exploring Engineering*, *9*, 5306–5310.

Dhiman, G. (2019a). Esa: a hybrid bio-inspired metaheuristic optimization approach for engineering problems. *Engineering with Computers*, 1–31.

Dhiman, G. (2019b). Moshepo: a hybrid multi-objective approach to solve economic load dispatch and micro grid problems. *Applied Intelligence*, 1–19.

Dhiman, G. (2019c). *Multi-objective metaheuristic approaches for data clustering in engineering application (s)* (Unpublished doctoral dissertation).

Dhiman, G., Guo, S., & Kaur, S. (2018). Ed-sho: A framework for solving nonlinear economic load power dispatch problem using spotted hyena optimizer. *Modern Physics Letters A*, *33*(40).

Dhiman, G., & Kaur, A. (2017). Spotted hyena optimizer for solving engineering design problems. In *2017 international conference on machine learning and data science (mlds)* (pp. 114–119).

Dhiman, G., & Kaur, A. (2018). Optimizing the design of airfoil and optical buffer problems using spotted hyena optimizer. *Designs*, *2*(3), 28.

Dhiman, G., & Kaur, A. (2019a). A hybrid algorithm based on particle swarm and spotted hyena optimizer for global optimization.

In *Soft computing for problem solving* (pp. 599–615). Springer.

Dhiman, G., & Kaur, A. (2019b). Stoa: A bio-inspired based optimization algorithm for industrial engineering problems. *Engineering Applications of Artificial Intelligence*, *82*, 148–174.

Dhiman, G., & Kumar, V. (n.d.). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, *In Press*.

Dhiman, G., & Kumar, V. (2017a). Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, *114*, 48–70.

Dhiman, G., & Kumar, V. (2017b). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, *114*(Supplement C), 48 - 70.

Dhiman, G., & Kumar, V. (2018a). Astrophysics inspired multi-objective approach for automatic clustering and feature selection in real-life environment. *Modern Physics Letters B*, *32*(31).

Dhiman, G., & Kumar, V. (2018b). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowledge-Based Systems*, *159*, 20–50.

Dhiman, G., & Kumar, V. (2018c). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowledge-Based Systems*, *159*, 20–50.

Dhiman, G., & Kumar, V. (2018d). Multi-objective spotted hyena optimizer: A multi-objective optimization algorithm for engineering problems. *Knowledge-Based Systems*, *150*, 175–197.

Dhiman, G., & Kumar, V. (2019a). Knrvea: A hybrid evolutionary algorithm based on knee points and reference vector adaptation strategies for many-objective optimization. *Applied Intelligence*, *49*(7), 2434–2460.

Dhiman, G., & Kumar, V. (2019b). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, *165*, 169–196.

Dhiman, G., & Kumar, V. (2019c). Spotted hyena optimizer for solving complex and non-linear constrained engineering problems. In *Harmony search and nature inspired optimization algorithms* (pp. 857–867). Springer.

Dhiman, G., Singh, P., Kaur, H., & Maini, R. (2019). Dhiman: A novel algorithm for economic d ispatch problem based on optimization met h od us i ng m onte carlo simulation and a strophysics co n cepts. *Modern Physics Letters A*, *34*(04).

Dhiman, G., Soni, M., Pandey, H. M., Slowik, A., & Kaur, H. (2020). A novel hybrid hypervolume indicator and reference vector adaptation strategies based evolutionary algorithm for many-objective optimization. *Engineering with Computers*, 1–19.

Digalakis, J., & Margaritis, K. (2001). On benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics*, *77*(4), 481-506.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization - artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine*, *1*, 28–39.

Du, H., Wu, X., & Zhuang, J. (2006). Small-world optimization algorithm for function optimization. *Springer Berlin Heidelberg*, 264–273.

Erol, O. K., & Eksin, I. (2006). A new optimization method: Big bang-big crunch. *Advances in Engineering Software*, *37*(2), 106 - 111.

Formato, R. A. (2009). Central force optimization: A new deterministic gradient-like optimization metaheuristic. *Opsearch*, *46*(1), 25–51.

Gandomi, A. H. (2014). Interior search algorithm (isa): A novel approach for global optimization. *ISA Transactions*, *53*(4), 1168 - 1183.

Gandomi, A. H., & Yang, X.-S. (2011). Benchmark problems in structural optimization. *Springer Berlin Heidelberg*, 259–281.

Garg, M., & Dhiman, G. (2020). Deep convolution neural network approach for defect inspection of textured surfaces. *Journal of the Institute of Electronics and Computer*, *2*, 28–38.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001, February). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68.

Ghorbani, N., & Babaei, E. (2014). Exchange market algorithm. *Applied Soft Computing*, *19*, 177 - 187.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, *1*(3), 190–206.

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences*, *222*, 175 - 184.

He, S., Wu, Q. H., & Saunders, J. R. (2006). A novel group search optimizer inspired by animal behavioural ecology. In *Ieee international conference on evolutionary computation* (p. 1272-1278).

Kannan, B., & Kramer, S. N. (1994). An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of mechanical design*, *116*(2), 405–411.

Kashan, A. H. (2009, Dec). League championship algorithm: A new algorithm for numerical function optimization. In *International conference of soft computing and pattern recognition* (p. 43-48).

Kaur, A. (2019). A systematic literature review on empirical analysis of the relationship between code smells and software quality attributes. *Archives of Computational Methods in Engineering*, 1–30.

Kaur, A., & Dhiman, G. (2019). A review on search-based tools and techniques to identify bad code smells in object-oriented systems. In *Harmony search and nature inspired optimization algorithms* (pp. 909–921). Springer.

Kaur, A., Jain, S., & Goel, S. (n.d.). Sp-j48: a novel optimization and machine-learning-based approach for solving complex problems: special application in software engineering for detecting code smells. *Neural Computing and Applications*, 1–19.

Kaur, A., Jain, S., & Goel, S. (2017). A support vector machine based approach for code smell detection. In *2017 international conference on machine learning and data science (mlds)* (pp. 9–14).

Kaur, A., Jain, S., & Goel, S. (2019). Sandpiper optimization algorithm: a novel approach for solving real-life engineering problems. *Applied Intelligence*, 1–38.

Kaur, A., Kaur, S., & Dhiman, G. (2018). A quantum method for dynamic nonlinear programming technique using schrödinger equation and monte carlo approach. *Modern Physics Letters B*, *32*(30).

Kaur, H., Peel, A., Acosta, K., Gebril, S., Ortega, J. L., & Sengupta-Gopalan, C. (2019). Comparison of alfalfa plants overexpressing glutamine synthetase with those overexpressing sucrose phosphate synthase demonstrates a signaling mechanism integrating carbon and nitrogen metabolism between the leaves and nodules. *Plant direct*, *3*(1), e00115.

Kaur, S., Awasthi, L. K., Sangal, A., & Dhiman, G. (2020). Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Engineering Applications of Artificial Intelligence*, *90*, 103541.

Kaveh, A., & Farhoudi, N. (2013, May). A new optimization method: Dolphin echolocation. *Adv. Eng. Softw.*, *59*, 53–70.

Kaveh, A., & Khayatazad, M. (2012). A new meta-heuristic method: Ray optimization. *Computers and Structures*, *112-113*, 283 - 294.

Kaveh, A., & Mahdavi, V. (2014). Colliding bodies optimization: A novel meta-heuristic method. *Computers and Structures*, *139*, 18 - 27.

Kaveh, A., & Talatahari, S. (2009). Size optimization of space trusses using big bang-big crunch algorithm. *Computers and Structures*, *87*(17-18), 1129 - 1140.

Kaveh, A., & Talatahari, S. (2010a). A novel heuristic optimization method: charged system search. *Acta Mechanica*, *213*(3), 267–289.

Kaveh, A., & Talatahari, S. (2010b). Optimal design of skeletal structures via the charged system search algorithm. *Structural and Multidisciplinary Optimization*, *41*(6), 893–911.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of ieee international conference on neural networks* (pp. 1942–1948).

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Koza, J. R. (1992). Genetic programming: On the programming of computers by means of natural selection. *MIT Press*.

Kumar, V., Chhabra, J. K., & Kumar, D. (2014a). Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. *Journal of Computational Science*, *5*(2), 144 - 155.

Kumar, V., Chhabra, J. K., & Kumar, D. (2014b). Variance-based harmony search algorithm for unimodal and multimodal optimization problems with application to clustering. *Cybernetics and Systems*, *45*(6), 486-511.

Li, X., He, F., & Li, W. (2019). A cloud-terminal-based cyber-physical system architecture for energy efficient machining process optimization. *Journal of Ambient Intelligence and Humanized Computing*, *10*(3), 1049–1064.

Lozano, M., & Garcia-Martinez, C. (2010). Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. *Computers and Operations Research*, *37(3)*, 481–497.

Lu, X., & Zhou, Y. (2008). A novel global convergence algorithm: Bee collecting pollen algorithm. *4th International Conference on Intelligent Computing, Springer*, 518–525.

Maini, R., & Dhiman, G. (2018). Impacts of artificial intelligence on real-life problems. *International Journal of Advance Research and Innovative Ideas in Education*, *4*, 291–295.

Martin, R., & Stephen, W. (2006). Termite: A swarm intelligent routing algorithm for mobilewireless ad-hoc networks. In *Stigmergic optimization* (pp. 155–184). Berlin, Heidelberg: Springer Berlin Heidelberg.

Mezura-Montes, E., & Coello, C. A. C. (2005). Useful infeasible solutions in engineering optimization with evolutionary algorithms. *Springer Berlin Heidelberg*, 652–662.

Mirjalili, S. (2015, November). Moth-flame optimization algorithm. *Know.-Based Syst.*, *89*(C), 228–249.

Mirjalili, S. (2016). Sca: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120 - 133.

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016, February). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.*, *27*(2), 495–513.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46 - 61.

Moghaddam, F. F., Moghaddam, R. F., & Cheriet, M. (2012). Curved space optimization: A random search based on general relativity theory. *Neural and Evolutionary Computing*.

Moosavian, N., & Roodsari, B. K. (2014). Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks. *Swarm and Evolutionary Computation*, *17*, 14 - 24.

Mucherino, A., & Seref, O. (2007). Monkey search: a novel metaheuristic search for global optimization. *AIP Conference Proceedings*, *953*(1).

Oftadeh, R., Mahjoob, M., & Shariatpanahi, M. (2010). A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search. *Computers and Mathematics with Applications*, *60*(7), 2087 - 2098.

Olorunda, O., & Engelbrecht, A. P. (2008). Measuring exploration/exploitation in particle swarms using swarm diversity. *IEEE Congress on Evolutionary Computation*, 1128-1134.

Pallavi, & Dhiman, G. (2018). Impact of foreign direct investment on the profitability: A study of scheduled commercial banks in india. *Computational and Applied Mathematics Journal*, *4*, 27–30.

Pan, W.-T. (2012, February). A new fruit fly optimization algorithm: Taking the financial distress model as an example. *Know.-Based Syst.*, *26*, 69–74.

Ragmani, A., Elomri, A., Abghour, N., Moussaid, K., & Rida, M. (2019). Faco: a hybrid fuzzy ant colony optimization algorithm for virtual machine scheduling in high-performance cloud computing. *Journal of Ambient Intelligence and Humanized Computing*, 1–13.

Ramezani, F., & Lotfi, S. (2013). Social-based algorithm. *Applied Soft Computing*, *13*(5), 2837 - 2856.

Ramirez-Atencia, C., & Camacho, D. (2019). Constrained multi-objective optimization for multi-uav planning. *Journal of Ambient Intelligence and Humanized Computing*, *10*(6), 2467–2484.

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*(13), 2232 - 2248.

Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2013). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing*, *13*(5), 2592 - 2612.

Schutte, J., & Groenwold, A. (2003). Sizing design of truss structures using particle swarms. *Structural and Multidisciplinary Optimization*, *25*(4), 261–269.

Shah Hosseini, H. (2011). Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation. *International Journal of Computational Science and Engineering*, *6*, 132–140.

Shiqin, Y., Jianjun, J., & Guangxing, Y. (2009). A dolphin partner optimization. *Proceedings of the WRI Global Congress on Intelligent Systems*, 124–128.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, *12*(6), 702-713.

Singh, P., & Dhiman, G. (2017). A fuzzy-lp approach in time series forecasting. In *International conference on pattern recognition and machine intelligence* (pp. 243–253).

Singh, P., & Dhiman, G. (2018a). A hybrid fuzzy time series forecasting model based on granular computing and bio-inspired optimization approaches. *Journal of computational science*, *27*, 370–385.

Singh, P., & Dhiman, G. (2018b). Uncertainty representation using fuzzy-entropy approach: Special application in remotely sensed high-resolution satellite images (rshrsis). *Appl. Soft Comput.*, *72*, 121–139.

Singh, P., Dhiman, G., Guo, S., Maini, R., Kaur, H., Kaur, A., ... Singh, N. (2019). A hybrid fuzzy quantum time series and linear programming model: Special application on taiex index dataset. *Modern Physics Letters A*, *34*(25).

Singh, P., Dhiman, G., & Kaur, A. (2018). A quantum approach for time series data based on graph and schrödinger equations methods. *Modern Physics Letters A*, *33*(35).

Singh, P., Rabadiya, K., & Dhiman, G. (2018). A four-way decision-making system for the indian summer monsoon rainfall. *Modern Physics Letters B*, *32*(25).

Storn, R., & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359.

Tan, Y., & Zhu, Y. (2010). Fireworks algorithm for optimization. *Springer Berlin Heidelberg*, 355–364.

Verma, S., Kaur, S., Dhiman, G., & Kaur, A. (2018). Design of a novel energy efficient routing framework for wireless nanosensor networks. In *2018 first international conference on secure cyber computing and communication (icsccc)* (pp. 532–536).

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82.

Yang, C., Tu, X., & Chen, J. (2007). Algorithm of marriage in honey bees optimization based on the wolf pack search. *International Conference on Intelligent Pervasive Computing*, 462-467.

Yang, D., Wang, X., Tian, X., & Zhang, Y. (2020). Improving monarch butterfly optimization through simulated annealing strategy. *Journal of Ambient Intelligence and Humanized Computing*, 1–12.

Yang, X.-S. (2010a). Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation*, *2*(2), 78–84.

Yang, X.-S. (2010b). A new metaheuristic bat-inspired algorithm. *Springer Berlin Heidelberg*, 65–74.

Yang, X. S., & Deb, S. (2009). Cuckoo search via levy flights. In *World congress on nature biologically inspired computing* (p. 210-214).

Table 1
Swarm-intelligence based optimization algorithms.

Algorithms	Years
Tabu (Taboo) Search (TS) (Glover, 1989)	1989
Harmony Search (HS) (Geem, Kim, & Loganathan, 2001)	2001
Termite Algorithm (TA) (Martin & Stephen, 2006)	2005
Group Search Optimizer (GSO) (He, Wu, & Saunders, 2006)	2006
Seeker Optimization Algorithm (RSO) (Dai, Zhu, & Chen, 2007)	2007
Imperialist Competitive Algorithm (ICA) (Atashpaz-Gargari & Lucas, 2007)	2007
League Championship Algorithm (LCA) (Kashan, 2009)	2009
Firework Algorithm (Tan & Zhu, 2010)	2010
Fruit fly Optimization Algorithm (FOA) (Pan, 2012)	2012
Bird Mating Optimizer (BMO) (Askarzadeh & Rezazadeh, 2013)	2012
Dolphin Echolocation (DE) algorithm (Kaveh & Farhoudi, 2013)	2013
Social-Based Algorithm (SBA) (Ramezani & Lotfi, 2013)	2013
Mine Blast Algorithm (MBA) (Sadollah, Bahreininejad, Eskandar, & Hamdi, 2013)	2013
Variance-Based Harmony Search (Kumar, Chhabra, & Kumar, 2014b)	2014
Parameter Adaptive Harmony Search (PAHS) (Kumar, Chhabra, & Kumar, 2014a)	2014
Exchange Market Algorithm (EMA) (Ghorbani & Babaei, 2014)	2014
Colliding Bodies Optimization (CBO) (Kaveh & Mahdavi, 2014)	2014
Interior Search Algorithm (ISA)(Gandomi, 2014)	2014
Soccer League Competition (SLC) algorithm (Moosavian & Roodsari, 2014)	2014
Crow Search Algorithm (CSA) (Askarzadeh, 2016)	2016
Emperor Penguin Optimizer (EPO) (Dhiman & Kumar, 2018c)	2018
Seagull Optimization Algorithm (SOA) (Dhiman & Kumar, n.d.)	2018

Table 2
p−values obtained from the Wilcoxon ranksum test for CEC-15 benchmark test functions.

F	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA
CEC-1	0.0009	0.0004	0.0004	0.0003	0.0067	0.0001	0.0004	0.0060
CEC-2	0.0012	0.0003	0.0003	0.0050	0.0002	0.0085	0.0035	0.0050
CEC-3	0.0002	0.0250	0.0102	0.0053	0.0006	0.0006	0.0007	0.0004
CEC-4	0.0042	0.0226	0.0029	0.0076	0.0023	0.0040	0.0002	0.0001
CEC-5	0.0006	0.0369	0.0005	0.0004	0.0009	0.2267	0.0093	0.0552
CEC-6	0.0076	0.0007	0.0038	0.0097	0.0269	0.4118	0.0072	0.0423
CEC-7	0.0087	0.0092	0.0005	0.0043	0.0022	0.0010	0.0004	0.0003
CEC-8	0.0032	0.5671	0.0009	0.0842	0.0080	0.0001	0.0028	0.0077
CEC-9	0.0001	0.0001	0.0077	0.0095	0.0001	0.0001	0.0003	0.0001
CEC-10	0.0001	0.0001	0.0001	0.0523	0.0440	0.0083	0.0004	0.0009
CEC-11	0.0002	0.0086	0.0009	0.0004	0.0007	0.0023	0.0055	0.0095
CEC-12	0.0001	0.0005	0.0001	0.0001	0.0637	0.0001	0.0042	0.0277
CEC-13	0.0001	0.0251	0.0001	0.0046	0.0048	0.0001	0.0004	0.0006
CEC-14	0.0001	0.0080	0.0036	0.0033	0.0796	0.0001	0.0055	0.0006
CEC-15	0.0009	0.0063	0.0076	0.0198	0.0119	0.0001	0.0076	0.0004

Table 3
Unimodal benchmark test functions.

Function	Dim	Range	f_{min}
$F_1(y) = \sum_{i=1}^{N} y_i^2$	30	[-100, 100]	0
$F_2(y) = \sum_{i=1}^{N} \|y_i\| + \prod_{i=1}^{N} \|y_i\|$	30	[-10, 10]	0
$F_3(y) = \sum_{i=1}^{N} (\sum_{j-1}^{i} y_j)^2$	30	[-100, 100]	0
$F_4(y) = max_i\{\|y_i\|, 1 \leq i \leq N\}$	30	[-100, 100]	0
$F_5(y) = \sum_{i=1}^{N-1} [100(y_{i+1} - y_i^2)^2 + (y_i - 1)^2]$	30	[-30, 30]	0
$F_6(y) = \sum_{i=1}^{N} (\|y_i + 0.5\|)^2$	30	[-100, 100]	0
$F_7(y) = \sum_{i=1}^{N} iy_i^4 + random[0, 1]$	30	[-1.28, 1.28]	0

Table 4
Multimodal benchmark test functions.

Function	Dim	Range	f_{min}
$F_8(y) = \sum_{i=1}^{N} -y_i sin(\sqrt{\|y_i\|})$	30	[-500, 500]	-418.982×5
$F_9(y) = \sum_{i=1}^{N} [y_i^2 - 10cos(2\pi y_i) + 10]$	30	[-5.12, 5.12]	0
$F_{10}(y) = -20exp\Big(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} y_i^2}\Big) - exp\Big(\frac{1}{N}\sum_{i=1}^{N} cos(2\pi y_i)\Big) + 20 + e$	30	[-32, 32]	0
$F_{11}(y) = \frac{1}{4000}\sum_{i=1}^{N} y_i^2 - \prod_{i=1}^{N} cos\Big(\frac{y_i}{\sqrt{i}}\Big) + 1$	30	[-600, 600]	0
$F_{12}(y) = \frac{\pi}{N}\{10sin(\pi x_1) + \sum_{i=1}^{N-1}(x_i - 1)^2[1 + 10sin^2(\pi x_{i+1})] + (x_n - 1)^2\} + \sum_{i=1}^{N} u(y_i, 10, 100, 4)$ $x_i = 1 + \frac{y_{i+1}}{4}$ $u(y_i, a, k, m) = \begin{cases} k(y_i - a)^m & y_i > a \\ 0 & -a < y_i < a \\ k(-y_i - a)^m & y_i < -a \end{cases}$	30	[-50, 50]	0
$F_{13}(y) = 0.1\{sin^2(3\pi y_1) + \sum_{i=1}^{N}(y_i - 1)^2[1 + sin^2(3\pi y_i + 1)] + (y_n - 1)^2[1 + sin^2(2\pi y_n)]\} + \sum_{i=1}^{N} u(y_i, 5, 100, 4)$	30	[-50, 50]	0
$F_{14}(y) = -\sum_{i=1}^{N} sin(y_i) \cdot \Big(sin\Big(\frac{i.y_i^2}{\pi}\Big)\Big)^{2m}, m = 10$	30	$[0, \pi]$	-4.687
$F_{15}(y) = \Big[e^{-\sum_{i=1}^{N}(y_i/\beta)^{2m}} - 2e^{-\sum_{i=1}^{N} y_i^2}\Big] \cdot \prod_{i=1}^{N} cos^2 y_i, m = 5$	30	[-20, 20]	-1
$F_{16}(y) = \{[\sum_{i=1}^{N} sin^2(y_i)] - exp(-\sum_{i=1}^{N} y_i^2)\} \cdot exp[-\sum_{i=1}^{N} sin^2\sqrt{\|y_i\|}]$	30	[-10, 10]	-1

Table 5
Fixed-dimension multimodal benchmark test functions.

Function	Dim	Range	f_{min}
$F_{14}(y) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(y_i - a_{ij})^6}\right)^{-1}$	2	[-65, 65]	1
$F_{15}(y) = \sum_{i=1}^{11} \left[a_i - \frac{y_1(b_i^2 + b_i y_2)}{b_i^2 + b_i y_3 + y_4}\right]^2$	4	[-5, 5]	0.00030
$F_{16}(y) = 4y_1^2 - 2.1y_1^4 + \frac{1}{3}y_1^6 + y_1y_2 - 4y_2^2 + 4y_2^4$	2	[-5, 5]	-1.0316
$F_{17}(y) = \left(y_2 - \frac{5.1}{4\pi^2}y_1^2 + \frac{5}{\pi}y_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)cosy_1 + 10$	2	[-5, 5]	0.398
$F_{18}(y) = [1 + (y_1 + y_2 + 1)^2(19 - 14y_1 + 3y_1^2 - 14y_2 + 6y_1y_2 + 3y_2^2)]\times$ $[30 + (2y_1 - 3y_2)^2 \times (18 - 32y_1 + 12y_1^2 + 48y_2 - 36y_1y_2 + 27y_2^2)]$	2	[-2, 2]	3
$F_{19}(y) = -\sum_{i=1}^{4} c_i exp(-\sum_{j=1}^{3} a_{ij}(y_j - p_{ij})^2)$	3	[1, 3]	-3.86
$F_{20}(y) = -\sum_{i=1}^{4} c_i exp(-\sum_{j=1}^{6} a_{ij}(y_j - p_{ij})^2)$	6	[0, 1]	-3.32
$F_{21}(y) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.1532
$F_{22}(y) = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.4028
$F_{23}(y) = -\sum_{i=1}^{1} 0[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.536

Table 6
CEC-15 benchmark test functions.

Function	Dim	Range	f_{min}
$CEC-1$ = Rotated Bent Cigar Function	30	[-100, 100]	100
$CEC-2$ = Rotated Discus Function	30	[-10, 10]	200
$CEC-3$ = Shifted and Rotated Weierstrass Function	30	[-100, 100]	300
$CEC-4$ = Shifted and Rotated Schwefel's Function	30	[-100, 100]	400
$CEC-5$ = Shifted and Rotated Katsuura Function	30	[-30, 30]	500
$CEC-6$ = Shifted and Rotated HappyCat Function	30	[-100, 100]	600
$CEC-7$ = Shifted and Rotated HGBat Function	30	[-1.28, 1.28]	700
$CEC-8$ = Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	30	[-500, 500]	800
$CEC-9$ = Shifted and Rotated Expanded Scaffer's F6 Function	30	[-5.12, 5.12]	900
$CEC-10$ = Hybrid Function 1 ($N = 3$)	30	[-32, 32]	1000
$CEC-11$ = Hybrid Function 2 ($N = 4$)	30	[-600, 600]	1100
$CEC-12$ = Hybrid Function 3 ($N = 5$)	30	[-50, 50]	1200
$CEC-13$ = Composition Function 1 ($N = 5$)	30	[-50, 50]	1300
$CEC-14$ = Composition Function 2 ($N = 3$)	2	[-65.536, 65.536]	1400
$CEC-15$ = Composition Function 3 ($N = 5$)	4	[-5, 5]	1500

Table 7
Parameters values of algorithms.

Algorithms	Parameters	Values
# (*For all algorithms*)	Search Agents	30
	Number of Generations	1000
Rat Swarm Optimizer (RSO)	Control Parameter (R)	[1, 5]
	Constant Parameter C	[0, 2]
Spotted Hyena Optimizer (SHO)	Control Parameter ($\vec{h}$)	[5, 0]
	$\vec{M}$ Constant	[0.5, 1]
Grey Wolf Optimizer (GWO)	Control Parameter ($\vec{a}$)	[2, 0]
Particle Swarm Optimization (PSO)	Inertia Coefficient	0.75
	Cognitive and Social Coeff	1.8, 2
Moth-Flame Optimization (MFO)	Convergence Constant	[-1, -2]
	Logarithmic Spiral	0.75
Multi-Verse Optimizer (MVO)	Wormhole Existence Prob.	[0.2, 1]
	Travelling Distance Rate	[0.6, 1]
Sine Cosine Algorithm (SCA)	Number of Elites	2
Gravitational Search Algorithm (GSA)	Gravitational Constant	100
	Alpha Coefficient	20
Genetic Algorithm (GA)	Crossover and Mutation	0.9, 0.05

Table 8
The obtained average and standard deviation results of RSO on unimodal benchmark test functions.

F	RSO	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA
F_1	6.09E-32	**0.00E+00**	4.69E-47	4.98E-09	3.15E-04	2.81E-01	3.55E-02	1.16E-16	1.95E-12
	(5.67E-35)	**(0.00E+00)**	(7.30E-45)	(1.40E-08)	(5.99E-04)	(1.11E-01)	(1.06E-01)	(6.10E-17)	(2.01E-11)
F_2	**0.00E+00**	0.00E+00	1.20E-24	7.29E-04	3.71E+01	3.96E-01	3.23E-05	1.70E-01	6.53E-18
	(0.00E+00)	(0.00E+00)	(1.30E-21)	(1.84E-03)	(2.16E+01)	(1.41E-01)	(8.57E-05)	(9.29E-01)	(5.10E-17)
F_3	1.10E-18	**0.00E+00**	1.00E-14	1.40E+01	4.42E+03	4.31E+01	4.91E+03	4.16E+02	7.70E-10
	(4.47E-19)	**(0.00E+00)**	(4.10E-14)	(7.13E+00)	(3.71E+03)	(8.97E+00)	(3.89E+03)	(1.56E+02)	(7.36E-09)
F_4	4.67E-07	7.78E-12	**2.02E-14**	6.00E-01	6.70E+01	8.80E-01	1.87E+01	1.12E+00	9.17E+01
	(1.96E-08)	(8.96E-12)	**(2.43E-14)**	(1.72E-01)	(1.06E+01)	(2.50E-01)	(8.21E+00)	(9.89E-01)	(5.67E+01)
F_5	**6.13E+00**	8.59E+00	2.79E+01	4.93E+01	3.50E+03	1.18E+02	7.37E+02	3.85E+01	5.57E+02
	(7.97E-01)	(5.53E-01)	(1.84E+00)	(3.89E+01)	(3.98E+03)	(1.43E+02)	(1.98E+03)	(3.47E+01)	(4.16E+01)
F_6	6.37E-07	2.46E-01	6.58E-01	9.23E-09	1.66E-04	3.15E-01	4.88E+00	**1.08E-16**	3.15E-01
	(7.30E-06)	(1.78E-01)	(3.38E-01)	(1.78E-08)	(2.01E-04)	(9.98E-02)	(9.75E-01)	**(4.00E-17)**	(9.98E-02)
F_7	**9.49E-06**	3.29E-05	7.80E-04	6.92E-02	3.22E-01	2.02E-02	3.88E-02	7.68E-01	6.79E-04
	(1.83E-05)	(2.43E-05)	(3.85E-04)	(2.87E-02)	(2.93E-01)	(7.43E-03)	(5.79E-02)	(2.77E+00)	(3.29E-03)

Table 9
The obtained average and standard deviation results of RSO on multimodal benchmark test functions.

F	RSO	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA
F_8	**-8.57E+03**	-1.16E+03	-6.14E+03	-6.01E+03	-8.04E+03	-6.92E+03	-3.81E+03	-2.75E+03	-5.11E+03
	(4.23E+02)	**(2.72E+02)**	(9.32E+02)	(1.30E+03)	(8.80E+02)	(9.19E+02)	(2.83E+02)	(5.72E+02)	(4.37E+02)
F_9	1.57E+02	**0.00E+00**	4.34E-01	4.72E+01	1.63E+02	1.01E+02	2.23E+01	3.35E+01	1.23E-01
	(7.39E+01)	**(0.00E+00)**	(1.66E+00)	(1.03E+01)	(3.74E+01)	(1.89E+01)	(3.25E+01)	(1.19E+01)	(4.11E+01)
F_{10}	**7.40E-17**	2.48E+00	1.63E-14	3.86E-02	1.60E+01	1.15E+00	1.55E+01	8.25E-09	5.31E-11
	(6.42E+00)	(1.41E+00)	**(3.14E-15)**	(2.11E-01)	(6.18E+00)	(7.87E-01)	(8.11E+00)	(1.90E-09)	(1.11E-10)
F_{11}	**0.00E+00**	0.00E+00	2.29E-03	5.50E-03	5.03E-02	5.74E-01	3.01E-01	8.19E+00	3.31E-06
	(0.00E+00)	(0.00E+00)	(5.24E-03)	(7.39E-03)	(1.74E-01)	(1.12E-01)	(2.89E-01)	(3.70E+00)	(4.23E-05)
F_{12}	5.52E-01	3.68E-02	3.93E-02	**1.05E-10**	1.26E+00	1.27E+00	5.21E+01	2.65E-01	9.16E-08
	(8.40E+00)	(1.15E-02)	(2.42E-02)	**(2.06E-10)**	(1.83E+00)	(1.02E+00)	(2.47E+02)	(3.14E-01)	(4.88E-07)
F_{13}	6.05E-02	9.29E-01	4.75E-01	4.03E-03	7.24E-01	6.60E-02	2.81E+02	**5.73E-32**	6.39E-02
	(7.43E-01)	(9.52E-02)	(2.38E-01)	(5.39E-03)	(1.48E+00)	(4.33E-02)	(8.63E+02)	**(8.95E-32)**	(4.49E-02)

Table 10
The obtained average and standard deviation results of RSO on fixed-dimension multimodal benchmark test functions.

F	RSO	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA
F_{14}	**6.51E-01**	9.68E+00	3.71E+00	2.77E+00	2.21E+00	9.98E-01	1.26E+00	3.61E+00	4.39E+00
	(6.17E-05)	(3.29E+00)	(3.86E+00)	(2.32E+00)	(1.80E+00)	**(9.14E-12)**	(6.86E-01)	(2.96E+00)	(4.41E-02)
F_{15}	**2.28E-04**	9.01E-04	3.66E-03	9.09E-04	1.58E-03	7.15E-03	1.01E-03	6.84E-03	7.36E-03
	(4.61E-04)	(1.06E-04)	(7.60E-03)	(2.38E-04)	(3.50E-03)	(1.26E-02)	(3.75E-04)	(7.37E-03)	(2.39E-04)
F_{16}	**-1.08E+01**	-1.06E+01	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.04E+00
	(3.34E-13)	(2.86E-11)	(7.02E-09)	(0.00E+00)	(0.00E+00)	(4.74E-08)	(3.23E-05)	(0.00E+00)	(4.19E-07)
F_{17}	**4.99E-02**	3.97E-01	3.98E-01	3.97E-01	3.98E-01	3.98E-01	3.99E-01	3.98E-01	3.98E-01
	(6.17E-07)	(2.46E-01)	(7.00E-07)	(9.03E-16)	(1.13E-16)	(1.15E-07)	(7.61E-04)	(1.13E-16)	**(3.71E-17)**
F_{18}	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	3.00E+00	5.70E+00	3.00E+00	3.01E+00	3.01E+00
	(8.20E-08)	(9.05E+00)	(7.16E-06)	(6.59E-05)	**(4.25E-15)**	(1.48E+01)	(2.25E-05)	(3.24E-02)	(6.33E-07)
F_{19}	**-3.90E+00**	-3.75E+00	-3.86E+00	-3.90E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.22E+00	-3.30E+00
	(3.87E-10)	(4.39E-01)	(1.57E-03)	(3.37E-15)	**(3.16E-15)**	(3.53E-070	(2.55E-03)	(4.15E-01)	(4.37E-10)
F_{20}	**-3.32E+00**	-1.44E+00	-3.27E+00	-3.32E+00	-3.23E+00	-3.23E+00	-2.84E+00	-1.47E+00	-2.39E+00
	(3.31E-02)	(5.47E-01)	(7.27E-02)	(2.66E-01)	(6.65E-02)	(5.37E-02)	(3.71E-01)	(5.32E-01)	(4.37E-01)
F_{21}	-2.05E+01	-2.08E+00	**-9.65E+00**	-7.54E+00	-6.20E+00	-7.38E+00	-2.28E+00	-4.57E+00	-5.19E+00
	(3.66E+00)	**(3.80E-01)**	(1.54E+00)	(2.77E+00)	(3.52E+00)	(2.91E+00)	(1.80E+00)	(1.30E+00)	(2.34E+00)
F_{22}	**-1.86E+01**	-1.61E+01	-1.04E+01	-8.55E+00	-7.95E+00	-8.50E+00	-3.99E+00	-6.58E+00	-2.97E+00
	(2.00E-04)	(2.04E-04)	(2.73E-04)	(3.08E+00)	(3.20E+00)	(3.02E+00)	(1.99E+00)	(2.64E+00)	(1.37E-02)
F_{23}	**-1.06E+01**	-1.68E+00	-1.05E+01	-9.19E+00	-7.50E+00	-8.41E+00	-4.49E+00	-9.37E+00	-3.10E+00
	(5.30E+00)	(2.64E-01)	**(1.81E-04)**	(2.52E+00)	(3.68E+00)	(3.13E+00)	(1.96E+00)	(2.75E+00)	(2.37E+00)

Table 11
The obtained average and standard deviation results of RSO on CEC-15 benchmark test functions.

F	RSO	SHO	GWO	PSO	MFO	MVO	SCA	GSA	GA
CEC-1	**4.19E+05**	2.28E+06	2.02E+06	4.37E+05	1.47E+06	6.06E+05	7.65E+06	3.20E+07	8.89E+06
	(4.19E+06)	(2.18E+06)	(2.08E+06)	**(4.73E+05)**	(2.63E+06)	(5.02E+05)	(3.07E+06)	(8.37E+06)	(6.95E+06)
CEC-2	6.97E+05	3.13E+05	5.65E+06	9.41E+03	1.97E+04	1.43E+04	7.33E+08	**4.58E+03**	2.97E+05
	(2.29E+06)	(4.19E+05)	(6.03E+06)	(1.08E+04)	(1.46E+04)	(1.03E+04)	(2.33E+08)	**(1.09E+03)**	(2.85E+03)
CEC-3	**3.20E+02**	3.20E+02	3.20E+02	3.20E+02	3.20E+02	3.20E+02	3.20E+02	3.20E+02	3.20E+02
	(5.96E-03)	(3.76E-02)	(7.08E-02)	(8.61E-02)	(9.14E-02)	(3.19E-02)	(7.53E-02)	**(1.11E-05)**	(2.78E-02)
CEC-4	4.10E+02	4.11E+02	4.16E+02	**4.09E+02**	4.26E+02	4.18E+02	4.42E+02	4.39E+02	6.99E+02
	(8.41E+01)	(1.71E+01)	(1.03E+01)	**(3.96E+00)**	(1.17E+01)	(1.03E+01)	(7.72E+00)	(7.25E+00)	(6.43E+00)
CEC-5	9.14E+02	9.13E+02	9.20E+02	**8.65E+02**	1.33E+03	1.09E+03	1.76E+03	1.75E+03	1.26E+03
	(6.60E+02)	(1.85E+02)	**(1.78E+02)**	(2.16E+02)	(3.45E+02)	(2.81E+02)	(2.30E+02)	(2.79E+02)	(1.86E+02)
CEC-6	3.50E+03	1.29E+04	2.26E+04	**1.86E+03**	7.35E+03	3.82E+03	2.30E+04	3.91E+06	2.91E+05
	(4.00E+04)	(1.15E+04)	(2.45E+04)	**(1.93E+03)**	(3.82E+03)	(2.44E+03)	(2.41E+04)	(2.70E+06)	(1.67E+05)
CEC-7	**7.02E+02**	7.02E+02	7.02E+02	7.02E+02	7.02E+02	7.02E+02	7.06E+02	7.08E+02	7.08E+02
	(4.81E-01)	(6.76E-01)	(7.07E-01)	(7.75E-01)	(1.10E+00)	(9.40E-01)	(9.07E-01)	(1.32E+00)	(2.97E+00)
CEC-8	**1.47E+03**	1.86E+03	3.49E+03	3.43E+03	9.93E+03	2.58E+03	6.73E+03	6.07E+05	5.79E+04
	(2.07E+03)	(1.98E+03)	(2.04E+03)	(2.77E+03)	(8.74E+03)	**(1.61E+03)**	(3.36E+03)	(4.81E+05)	(2.76E+04)
CEC-9	**1.00E+03**	1.00E+03	1.00E+03	1.00E+03	1.00E+03	1.00E+03	1.00E+03	1.00E+03	1.00E+03
	(6.97E-01)	(1.43E-01)	(1.28E-01)	(7.23E-02)	(2.20E-01)	**(5.29E-02)**	(9.79E-01)	(5.33E+00)	(3.97E+00)
CEC-10	**1.59E+03**	2.00E+03	4.00E+03	3.27E+03	8.39E+03	2.62E+03	9.91E+03	3.42E+05	4.13E+04
	(2.30E+04)	(2.73E+03)	(2.82E+03)	(1.84E+03)	(1.12E+04)	**(1.78E+03)**	(8.83E+03)	(1.74E+05)	(2.39E+04)
CEC-11	**1.33E+03**	1.38E+03	1.40E+03	1.35E+03	1.37E+03	1.39E+03	1.35E+03	1.41E+03	1.36E+03
	(1.44E+01)	(2.42E+01)	(5.81E+01)	(1.12E+02)	(8.97E+01)	(5.42E+01)	(1.11E+02)	(7.73E+01)	(5.39E+01)
CEC-12	**1.30E+03**	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.31E+03	1.31E+03	1.31E+03
	(4.91E+00)	(7.89E-01)	**(6.69E-01)**	(6.94E-01)	(9.14E-01)	(8.07E-01)	(1.54E+00)	(2.05E+00)	(1.65E+00)
CEC-13	**1.30E+03**	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.30E+03	1.35E+03	1.35E+03
	(5.09E-05)	(2.76E-04)	(1.92E-04)	(5.44E-03)	(1.04E-03)	(2.43E-04)	(3.78E-03)	(4.70E+01)	(3.97E+01)
CEC-14	**3.56E+03**	4.25E+03	7.29E+03	7.10E+03	7.60E+03	7.34E+03	7.51E+03	9.30E+03	8.96E+03
	(6.12E+04)	(1.73E+03)	(2.45E+03)	(3.12E+03)	(1.29E+03)	(2.47E+03)	(1.52E+03)	**(4.04E+02)**	(6.32E+03)
CEC-15	**1.60E+03**	1.60E+03	1.61E+03	1.60E+03	1.61E+03	1.60E+03	1.62E+03	1.64E+03	1.63E+03
	(6.00E-03)	(3.76E+00)	(4.94E+00)	**(2.66E-07)**	(1.13E+01)	(1.80E-02)	(3.64E+00)	(1.12E+01)	(3.67E+01)

Table 12
Comparison results for pressure vessel design problem.

Algorithms	Optimum variables				Optimum cost
	T_s	T_h	R	L	
RSO	0.775967	0.383127	40.313297	200.00000	**5878.5395**
SHO	0.778210	0.384889	40.315040	200.00000	5885.5773
GWO	0.779035	0.384660	40.327793	199.65029	5889.3689
PSO	0.778961	0.384683	40.320913	200.00000	5891.3879
MVO	0.845719	0.418564	43.816270	156.38164	6011.5148
SCA	0.817577	0.417932	41.74939	183.57270	6137.3724
GSA	1.085800	0.949614	49.345231	169.48741	11550.2976
GA	0.752362	0.399540	40.452514	198.00268	5890.3279
HS	1.099523	0.906579	44.456397	179.65887	6550.0230

Table 13
Statistical results of proposed RSO and competitor algorithms for pressure vessel design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
RSO	**5878.5395**	**5881.5355**	**5887.3933**	167.041	**5880.0051**
SHO	5885.5773	5887.4441	5892.3207	**002.893**	5886.2282
GWO	5889.3689	5891.5247	5894.6238	013.910	5890.6497
PSO	5891.3879	6531.5032	7394.5879	534.119	6416.1138
MVO	6011.5148	6477.3050	7250.9170	327.007	6397.4805
SCA	6137.3724	6326.7606	6512.3541	126.609	6318.3179
GSA	11550.2976	23342.2909	**33226.2526**	5790.625	24010.0415
GA	5890.3279	6264.0053	7005.7500	496.128	6112.6899
HS	6550.0230	6643.9870	8005.4397	657.523	7586.0085

Table 14
Comparison results for speed reducer design problem.

Algorithms	Optimum variables							Optimum cost
	b	m	z	l_1	l_2	d_1	d_2	
RSO	3.50112	0.7	17	7.3	7.8	3.32323	5.24567	**2993.0027**
SHO	3.50159	0.7	17	7.3	7.8	3.35127	5.28874	2998.5507
GWO	3.506690	0.7	17	7.380933	7.815726	3.357847	5.286768	3001.288
PSO	3.500019	0.7	17	8.3	7.8	3.352412	5.286715	3005.763
MVO	3.508502	0.7	17	7.392843	7.816034	3.358073	5.286777	3002.928
SCA	3.508755	0.7	17	7.3	7.8	3.461020	5.289213	3030.563
GSA	3.600000	0.7	17	8.3	7.8	3.369658	5.289224	3051.120
GA	3.510253	0.7	17	8.35	7.8	3.362201	5.287723	3067.561
HS	3.520124	0.7	17	8.37	7.8	3.366970	5.288719	3029.002

Table 15
Statistical results of proposed RSO and competitor algorithms for speed reducer design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
RSO	**2993.0027**	**2998.597**	3000.678	**1.75697**	**2995.357**
SHO	2998.5507	2999.640	3003.889	1.93193	2999.187
GWO	3001.288	3005.845	3008.752	5.83794	3004.519
PSO	3005.763	3105.252	3211.174	79.6381	3105.252
MVO	3002.928	3028.841	3060.958	13.0186	3027.031
SCA	3030.563	3065.917	3104.779	18.0742	3065.609
GSA	3051.120	3170.334	**3363.873**	92.5726	3156.752
GA	3067.561	3186.523	3313.199	17.1186	3198.187
HS	3029.002	3295.329	3619.465	57.0235	3288.657

Table 16
Comparison results for welded beam design.

Algorithms	Optimum variables				Optimum cost
	h	*t*	*l*	*b*	
RSO	0.205397	3.465789	9.034571	0.201097	**1.722789**
SHO	0.205563	3.474846	9.035799	0.205811	1.725661
GWO	0.205678	3.475403	9.036964	0.206229	1.726995
PSO	0.197411	3.315061	10.00000	0.201395	1.820395
MVO	0.205611	3.472103	9.040931	0.205709	1.725472
SCA	0.204695	3.536291	9.004290	0.210025	1.759173
GSA	0.147098	5.490744	10.00000	0.217725	2.172858
GA	0.164171	4.032541	10.00000	0.223647	1.873971
HS	0.206487	3.635872	10.00000	0.203249	1.836250

Table 17
Statistical results of proposed RSO and competitor algorithms for welded beam design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
RSO	**1.722789**	**1.725097**	1.726987	0.015748	**1.723697**
SHO	1.725661	1.725828	1.726064	**0.000287**	1.725787
GWO	1.726995	1.727128	1.727564	0.001157	1.727087
PSO	1.820395	2.230310	**3.048231**	0.324525	2.244663
MVO	1.725472	1.729680	1.741651	0.004866	1.727420
SCA	1.759173	1.817657	1.873408	0.027543	1.820128
GSA	2.172858	2.544239	3.003657	0.255859	2.495114
GA	1.873971	2.119240	2.320125	0.034820	2.097048
HS	1.836250	1.363527	2.035247	0.139485	1.9357485

Table 18
Comparison results for tension/compression spring design.

Algorithms	Optimum variables			Optimum cost
	d	*D*	*N*	
RSO	0.051075	0.341987	12.0667	**0.012655697**
SHO	0.051144	0.343751	12.0955	0.012674000
GWO	0.050178	0.341541	12.07349	0.012678321
PSO	0.05000	0.310414	15.0000	0.013192580
MVO	0.05000	0.315956	14.22623	0.012816930
SCA	0.050780	0.334779	12.72269	0.012709667
GSA	0.05000	0.317312	14.22867	0.012873881
GA	0.05010	0.310111	14.0000	0.013036251
HS	0.05025	0.316351	15.23960	0.012776352

Table 19
Statistical results of proposed RSO and competitor algorithms for tension/compression spring design.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
RSO	**0.012655697**	**0.012665789**	0.012667980	0.057894	**0.012666369**
SHO	0.012674000	0.012684106	0.012715185	**0.000027**	0.012687293
GWO	0.012678321	0.012697116	0.012720757	0.000041	0.012699686
PSO	0.013192580	0.014817181	**0.017862507**	0.002272	0.013192580
MVO	0.012816930	0.014464372	0.017839737	0.001622	0.014021237
SCA	0.012709667	0.012839637	0.012998448	0.000078	0.012844664
GSA	0.012873881	0.013438871	0.014211731	0.000287	0.013367888
GA	0.013036251	0.014036254	0.016251423	0.002073	0.013002365
HS	0.012776352	0.013069872	0.015214230	0.000375	0.012952142

Table 20
Member stress limitations for 25-bar truss design problem.

Groups	Compressive stress limitations Ksi (MPa)	Tensile stress limitations Ksi (MPa)
Group 1	35.092 (241.96)	40.0 (275.80)
Group 2	11.590 (79.913)	40.0 (275.80)
Group 3	17.305 (119.31)	40.0 (275.80)
Group 4	35.092 (241.96)	40.0 (275.80)
Group 5	35.092 (241.96)	40.0 (275.80)
Group 6	6.759 (46.603)	40.0 (275.80)
Group 7	6.959 (47.982)	40.0 (275.80)
Group 8	11.082 (76.410)	40.0 (275.80)

Table 21
Two loading conditions for 25-bar truss design problem.

Node	Case 1			Case 2		
	$P_x Kips(kN)$	$P_y Kips(kN)$	$P_z Kips(kN)$	$P_x Kips(kN)$	$P_y Kips(kN)$	$P_z Kips(kN)$
1	0.0	20.0 (89)	-5.0 (22.25)	1.0 (4.45)	10.0 (44.5)	-5.0 (22.25)
2	0.0	-20.0 (89)	-5.0 (22.25)	0.0	10.0 (44.5)	-5.0 (22.25)
3	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0
6	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0

Table 22
Statistical results of proposed RSO with literature for 25-bar truss design problem.

Group	RSO	ACO (Bichon, 2004)	PSO (Schutte & Groenwold, 2003)	CSS (Kaveh & Talatahari, 2010b)	BB-BC (Kaveh & Talatahari, 2009)
A1	**0.01**	0.01	0.01	0.01	0.01
A2-A5	**1.848**	2.042	2.052	2.003	1.993
A6-A9	**3.000**	3.001	3.001	3.007	3.056
A10-A11	**0.01**	0.01	0.01	0.01	0.01
A12-A13	**0.01**	0.01	0.01	0.01	0.01
A14-A17	**0.657**	0.684	0.684	0.687	0.665
A18-A21	1.627	1.625	**1.616**	1.655	1.642
A22-A25	2.661	2.672	2.673	**2.66**	2.679
Best weight	**543.57**	545.03	545.21	545.10	545.16
Average weight	**546.20**	545.74	546.84	545.58	545.66
Std. dev.	**0.388**	0.94	1.478	0.412	0.491

Table 23
Comparison results for rolling element bearing design problem.

Algorithms	Optimum variables										Opt. cost
	D_m	D_b	Z	f_i	f_o	K_{Dmin}	K_{Dmax}	ε	e	ζ	
RSO	125	21.41769	10.94027	0.515	0.515	0.4	0.7	0.3	0.02	0.6	**85069.021**
SHO	125	21.40732	10.93268	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85054.532
GWO	125.6199	21.35129	10.98781	0.515	0.515	0.5	0.68807	0.300151	0.03254	0.62701	84807.111
PSO	125	20.75388	11.17342	0.515	0.515000	0.5	0.61503	0.300000	0.05161	0.60000	81691.202
MVO	125.6002	21.32250	10.97338	0.515	0.515000	0.5	0.68782	0.301348	0.03617	0.61061	84491.266
SCA	125	21.14834	10.96928	0.515	0.515	0.5	0.7	0.3	0.02778	0.62912	83431.117
GSA	125	20.85417	11.14989	0.515	0.517746	0.5	0.61827	0.304068	0.02000	0.624638	82276.941
GA	125	20.77562	11.01247	0.515	0.515000	0.5	0.61397	0.300000	0.05004	0.610001	82773.982
HS	125	20.87123	11.16697	0.515	0.516000	0.5	0.61951	0.301128	0.05024	0.614531	81569.527

Table 24
Statistical results of proposed RSO and competitor algorithms for rolling element bearing design problem.

Algorithms	Best	Mean	Worst	Std. Dev.	Median
RSO	**85069.021**	**85041.998**	86548.613	1789.14	**85054.497**
SHO	85054.532	85024.858	85853.876	0186.68	85040.241
GWO	84807.111	84791.613	84517.923	0137.186	84960.147
PSO	81691.202	50435.017	**32761.546**	**13962.150**	42287.581
MVO	84491.266	84353.685	84100.834	0392.431	84398.601
SCA	83431.117	81005.232	77992.482	1710.777	81035.109
GSA	82276.941	78002.107	71043.110	3119.904	78398.853
GA	82773.982	81198.753	80687.239	1679.367	8439.728
HS	81569.527	80397.998	79412.779	1756.902	8347.009

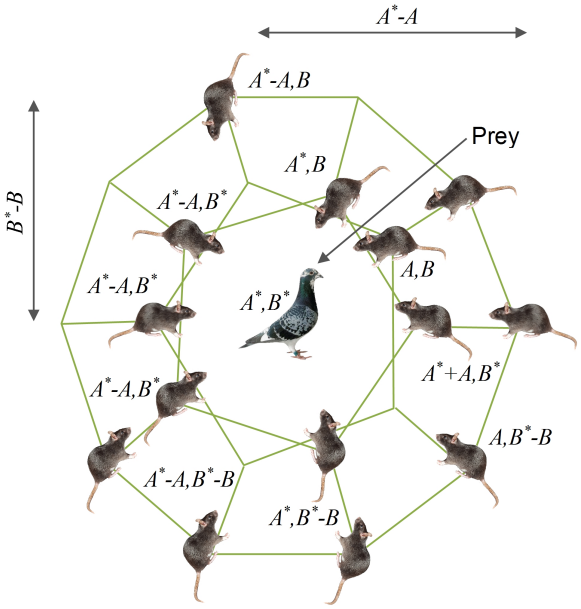

Figure 1. 3D position vectors of rats.

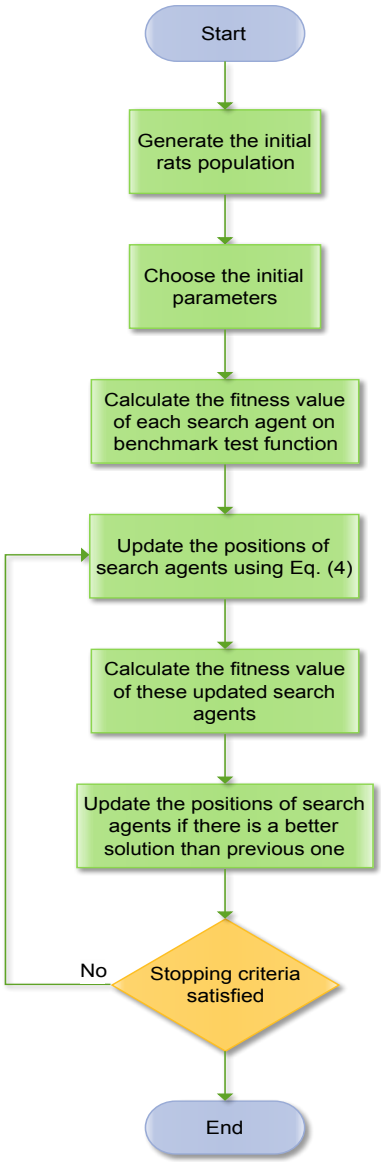

Figure 2. Flowchart of the proposed RSO algorithm.

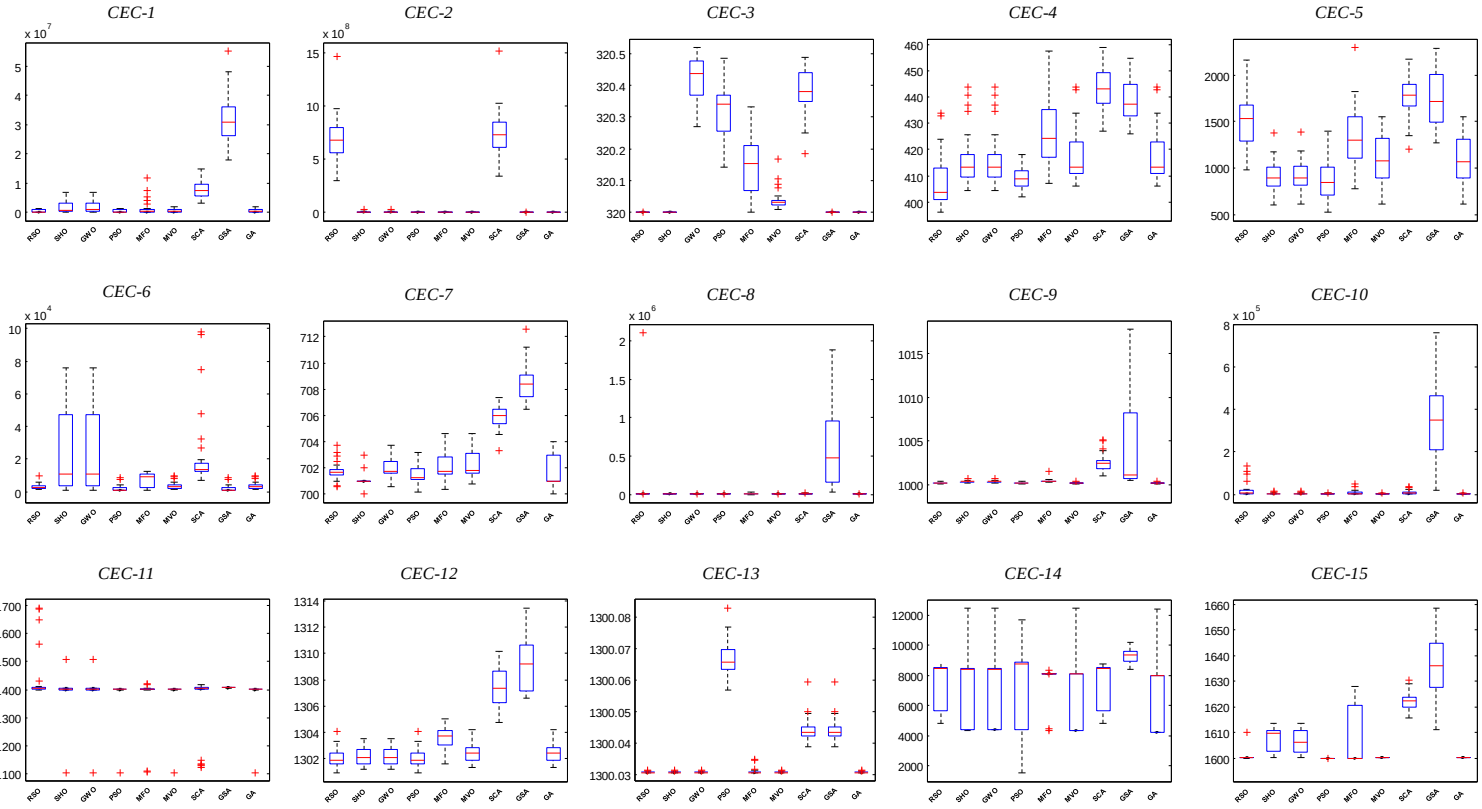

Figure 3. Boxplot analysis of proposed RSO algorithm on CEC benchmark test functions.

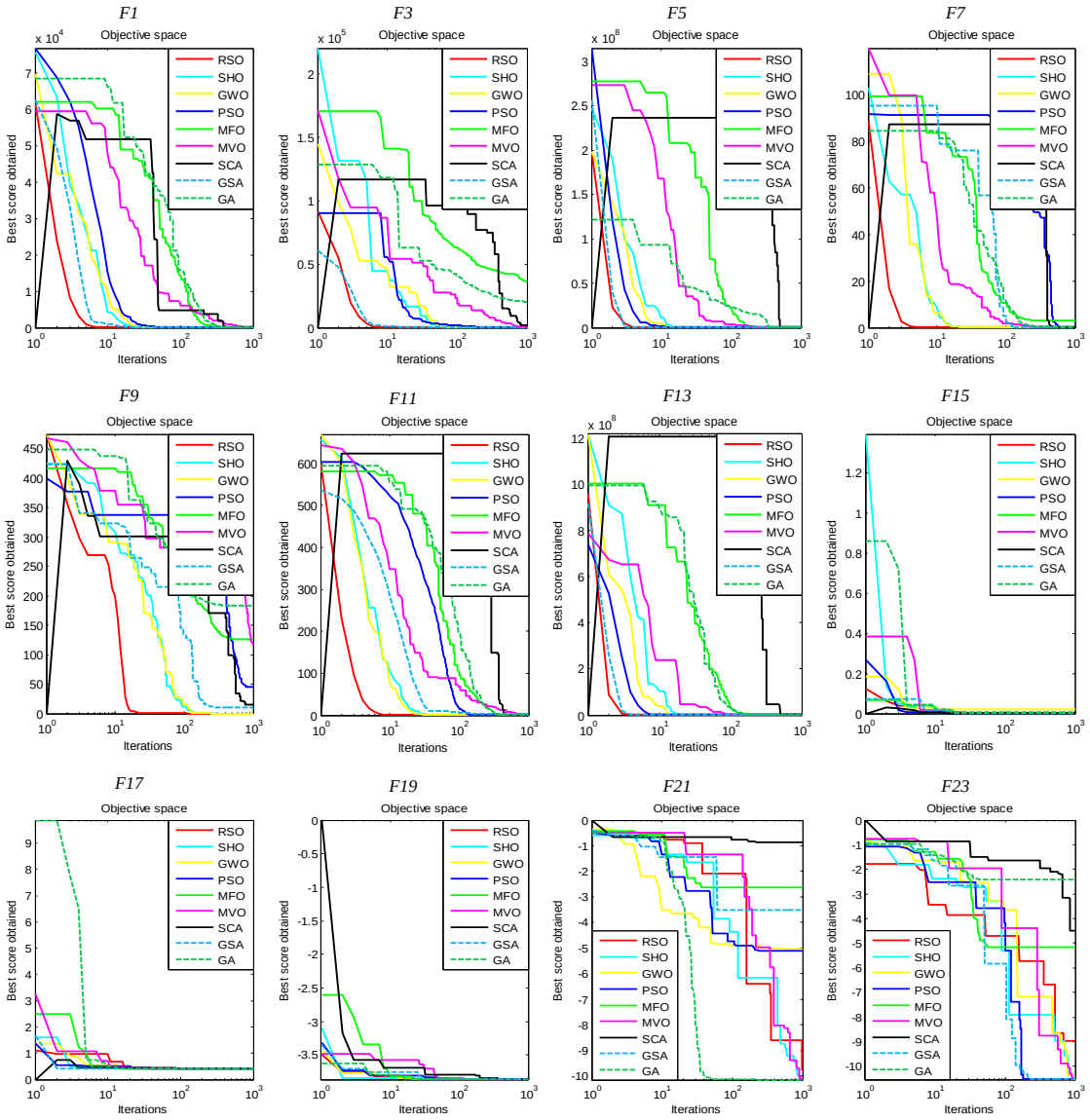

Figure 4. Convergence analysis of proposed RSO algorithm and competitor algorithms obtained on some of the benchmark test problems.

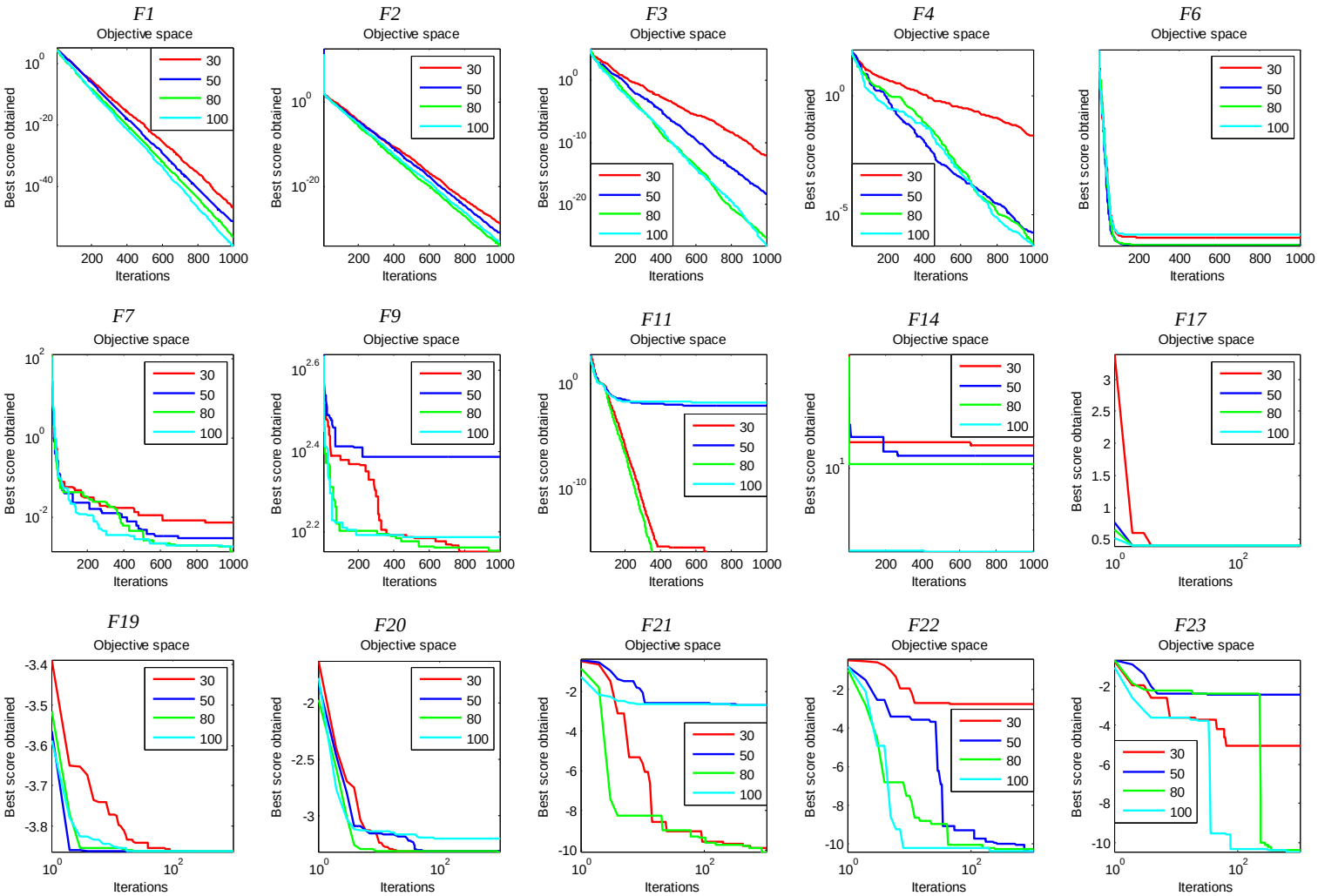

Figure 5. Scalability analysis of proposed RSO algorithm.

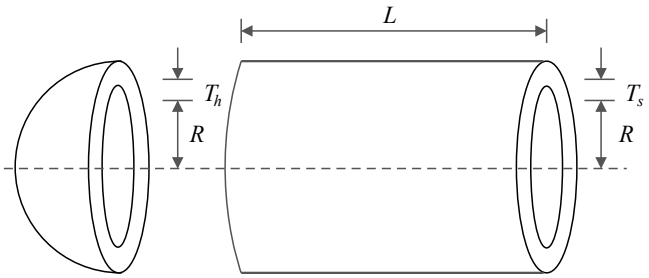

Figure 6. Schematic view of pressure vessel problem.

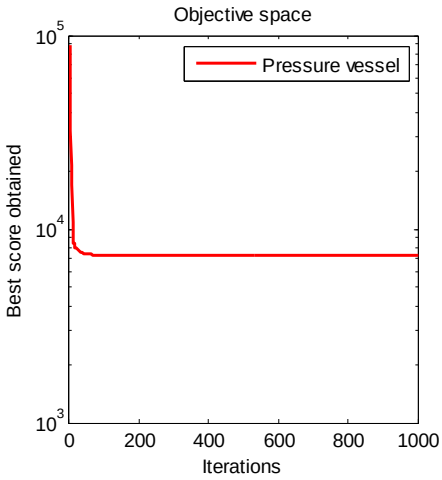

Figure 7. Analysis of proposed RSO for the pressure vessel problem.

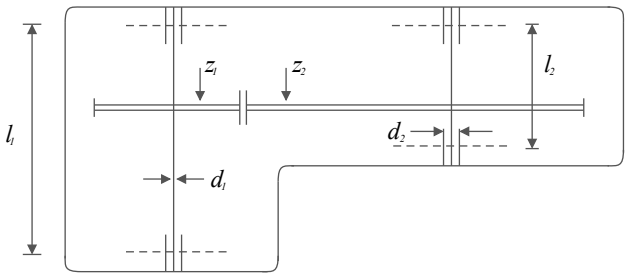

Figure 8. Schematic view of speed reducer problem.

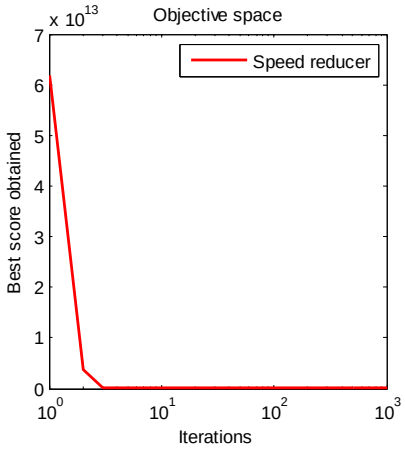

Figure 9. Analysis of proposed RSO for the speed reducer problem.

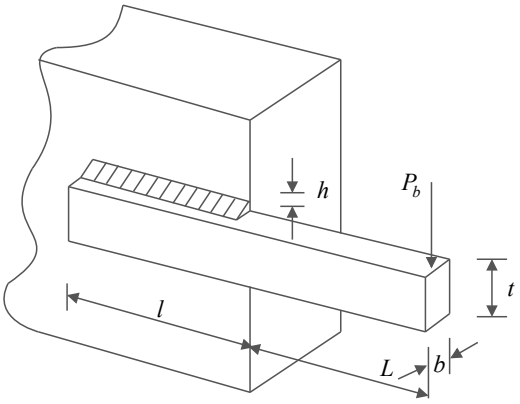

Figure 10. Schematic view of welded beam problem.

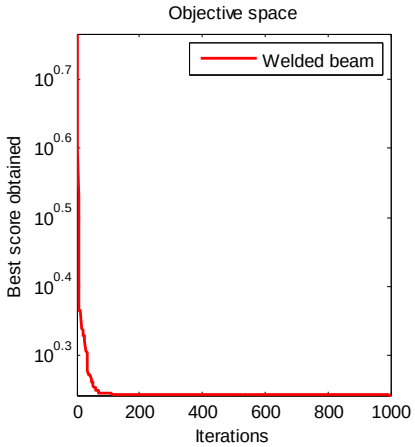

Figure 11. Analysis of proposed RSO for the welded beam problem.

Figure 12. Schematic view of tension/compression spring problem.

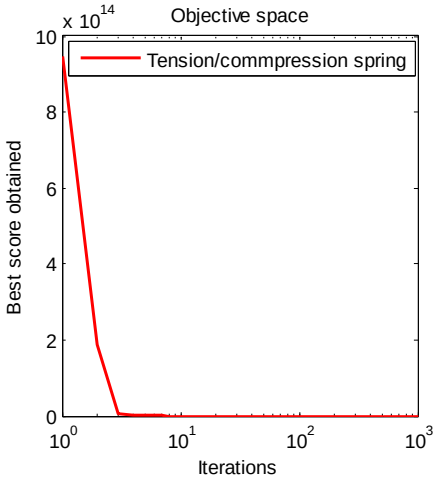

Figure 13. Analysis of proposed RSO for tension/compression spring problem.

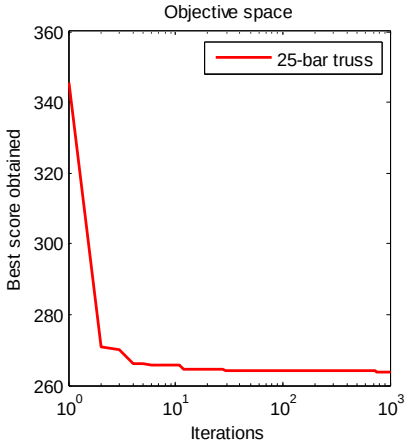

Figure 14. Analysis of proposed RSO for the 25-bar truss problem.

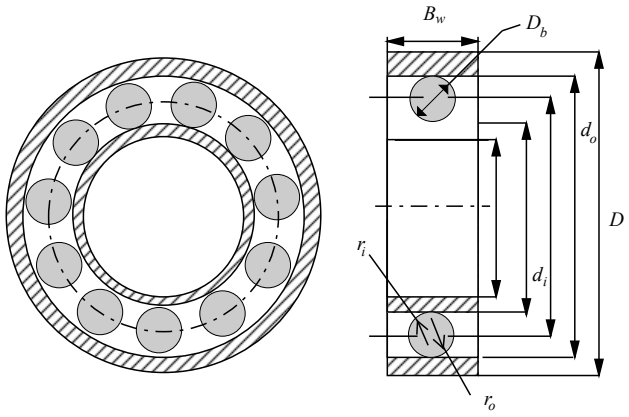

Figure 15. Schematic view of rolling element bearing problem.

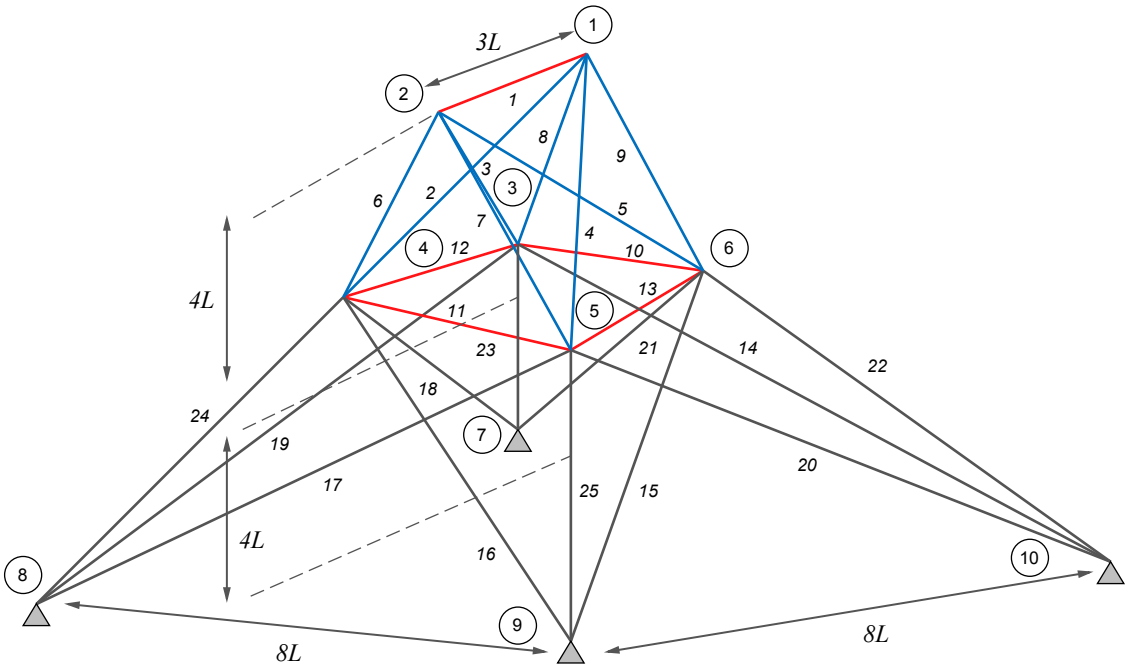

Figure 16. Schematic view of 25-bar truss design problem.

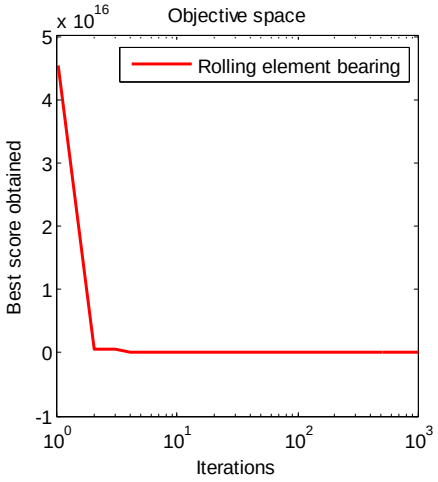

Figure 17. Analysis of proposed RSO for the rolling element bearing problem.