Biological survival optimizer: a new nature-inspired computation technique for engineering optimization

qingyang zhang (hgd_qy@126.com)
Jiangsu Normal University
Xiangyu He
Jiangsu Normal University
Shengxiang Yang
De Montfort University
Shouyong Jiang
University of Aberdeen
yongquan Dong
Jiangsu Normal University
Hui Song
Jiangsu Normal University

Research Article

Keywords: biological survival optimizer, engineering structural problem, heuristic, escape behavior

Posted Date: December 12th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1620703/v2

Additional Declarations:
Appendix 1 is not available with this version.

Version of Record: A version of this preprint was published at Soft Computing on February 13th, 2023. See the published version at https://doi.org/10.1007/s00500-023-07851-4.

Biological survival optimizer: a new nature-inspired computation technique for engineering optimization

Qingyang Zhang[a], Xiangyu He[a], Shengxiang Yang[b], Shouyong Jiang[c], Yongquan Dong[a], Hui Song[a]

[a] *School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, CO 221116 China*
[b] *School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, United Kingdom*
[c] *School of Computer Science, University of Aberdeen, Aberdeen, CO AB24 3FX United Kingdom*

Abstract

This study proposes a novel and lightweight bio-inspired computation technique named biological survival optimizer (BSO), which simulates the escape behavior of prey in the natural environment. This algorithm consists of two important courses, escape phase and adjustment phase. Specifically, in the escape phase, each search agent is required to updates its location using the best, the worst and a neighboring individual of the population. The adjustment phase is implemented using the simplex algorithm for search better location of the worst agent within a small region. The effectiveness of the BSO is validated on the CEC2017 benchmark problems and three classical engineering structural problems. Simulation comparison results considering both convergence and accuracy simultaneously show that BSO can present competitive performance compared with other state-of-art optimization techniques.

Keywords: biological survival optimizer, engineering structural problem, heuristic, escape behavior

1. Introduction

Global optimization is a process of searching the best solutions of a defined problem with maximum or minimum objection functions involved in different areas. With the rapid development of science and knowledge, optimization problems have been endowed with diverse characteristics, such as non-convex, discontinue, high-dimension and so on, therefore, the demand of high quality optimization algorithms becomes more urgent than before. Inspired by different biology mechanism or nature phenomena, scholars have designed diverse optimization algorithms. In general, these optimization algorithms can be grouped into derivation-based and stochastic algorithms. The former utilizes substantial gradient information of objective function to built the search direction for obtaining better solutions. For some simple or ideal models, derivation-based algorithms can obtain competitive results using less computation cost. However, they also show several disadvantages, premature convergence, gradient dependence, instability for complicated or difficult optimization problem. The latter can deal with the drawbacks mentioned above based on individual co-operation mechanism, since stochastic algorithms use a search population of random solutions sampled in feasible region to approximate better candidate solution.

The main feature of stochastic algorithm includes exploration and exploitation during the whole optimization course [1]. Exploration ability aims to help the search agents for exploring promising areas in feasible space broadly. On the contrary, exploitation ability refers to guide the population agents for converging toward the best member in the promising areas obtained in exploration phase. It is worth mentioning that a good tradeoff between exploration and exploitation should be maintained during the search process, since favoring exploration benefits to improve diversity for local optima avoidance, and emphasizing exploitation benefits to a faster convergence.

Recently, a growing number of new optimization algorithms or modifying the existing ones are proposed in this field. The reason of this phenomenon may derived from The No Free Lunch (NFL)[2], which indicates that designing a general-purpose optimization algorithm is impossible. It is obvious that the NFL theorem makes this field of research open, scholars are encouraged to modify the current methodologies for optimizing various problems or design novel algorithms for generating promising results with respect to the current algorithms.

The main work of this paper includes as follows, on the one hand, two different behaviors were extracted by simulating the escape behavior of Oryx named escape phase and adjustment phase, which are modeled mathematically. Then, biological survival optimizer (BSO) is proposed. On the other hand, unlike to other stochastic search algorithms, BSO is a parameterless method and utilizes the simple algorithm as local generation strategy for searching promising individuals. The advantage of this operator is that it not only performs effective local search but also generates agents that are better than the previous ones, which are widely used in various algorithms.

The following provides the basic organizational structure of this paper: Section 2 provides the literature review of the existing optimization algorithms. The background and the principles of BSO are summarized in Section 3. In Section 4, CEC2017 test problems are employed to evaluate the effectiveness of BSO. In Section 5, the proposed algorithm is utilized to solve

*Corresponding author: Qingyang Zhang
Email address: hgd_qy@126.com (Qingyang Zhang)

three constrained engineering design issues. Section 6 concludes the main work of this paper.

2. Literature review

Over the past two decades, inspired by various laws or phenomena of nature, Many heuristic optimization methods have been designed and applied to solve different practical problems successfully. According to the inspiration, the existing optimization methods can be classified into three different categories: evolution-based, physics-based and swarm-based algorithms.

Evolution-based algorithm is a kind of methods inspired by the evolution mechanism of nature. Genetic algorithm (GA), a representative method in this branch, mimics the Darwinian evolution theory [3-4]. Crossover and mutation operators are the main mechanisms for generating new offspring individuals and improving the quality of search population over the course of iterations in GA. Biogeography-Based Optimizer (BBO) imitates natural biogeography phenomenon that biological species tend to migrate to better habitats for living [5]. Some other algorithms are evolution strategy (ES)[6] and Differential Evolution (DE)[7].

Physics-based algorithms, the second category, simulate physical phenomena or rules, which is different from the Evolution-based algorithms. Gravitational Search Algorithm (GSA) is designed by imitating the Newtonian gravity and the laws of motion. All agents update their positions by the gravitational attraction force between them during iteration [8]. The heavier the mass, the bigger the attractive force. Big-Bang Big-Crunch (BBBC) is designed by imitating the big bang and big crunch principles, which is utilized to generate random directions and gather the search agents for moving toward the best point obtained so far [9]. Some other algorithms are Multi-Verse Optimizer (MVO) simulates the theory of multi-verse [10], Artificial raindrop algorithm (ARA) simulates the phenomenon of natural rainfall [11], Mine Blast algorithm (MBA) simulates the concept of mine bomb explosion [12]. States of Matter Search (SMS) devises from the simulation of the states of matter concept [13], Ray optimization (RO) simulates the Snell's light refraction law [14].

Swarm-based algorithms, the third category, mimic the swarm behaviors of animals in nature. Particle swarm optimization (PSO), one of the most population algorithms, imitates the foraging actions of bird in nature [15-16]. PSO utilizes multiple agents for generating new search individuals, which means that each search agent updates position considering its own best position obtained so far as well as the best position of the search swarm found so far. Bat algorithm (BA) is designed by imitating ultrasonic echolocation foraging behavior, which is used for sensing distance and to differentiate between prey and obstacles. Two update formula and random walking strategy are the core parts of the algorithm [17]. Grey wolf optimizer (G-WO) imitates the leadership hierarchy and hunting mechanism of grey wolves. The group of wolves is classified into four hierarchies, alpha, beta, delta, and omega. Adaptive control parameters and different computation operators are implemented to perform optimization [18]. Some other algorithms are Moth-flame optimization (MFO) simulates the flight mechanism of Moths [19], Cuckoo Search (CS) simulates breeding parasitism behavior [20], Sine Cosine Algorithm (SCA) simulates mathematical model based on sine and cosine functions [21], and whale optimization algorithm (WOA) simulate the social habits of humpback whales [22].

This section list shows that there are many swarm-based algorithms proposed so far, and most of them are inspired by hunting and search phenomena. To the best of our knowledge, there is no swarm-based method imitating the escaping behavior of Oryx, which motivates us to design a new optimization technique by modelling the escaping behavior mathematically for solving benchmark functions and engineering design problems.

3. Biological Survival Optimizer (BSO)

This section mainly includes two segments, on the one hand, the background of BSO is described broadly. On the other hand, the proposed calculation model of BSO is presented in detail.

3.1. Background

The Oryx is one of the most intelligent animals in nature. In the real world, escaping behavior is an common phenomenon, where Oryx needs to cooperate and exchange information together for survival. On the one hand, with the purpose of improving the successful probability of prey, predator usually tends to attack the weak side of Oryx group. On the other hand, the leaders of Oryx group guide other group individuals away from the hunting. Besides that, the individuals should keep close to their neighbors and should avoid collisions with their neighbors. We use a topological structure for simulating the interactive relationships between individuals, no matter how close or how far away those individuals are. However, some Oryx individuals may not escape from the hunting successfully. In other words, each individual has a certain escape probability. The stronger the individual is, the higher successful probability is. After getting away from the threat, the group arrives a new place temporarily. To avoid being attacked easily, the leaders must guide the individual in weaker side. toward a better and suitable location [23-30]. Fig.1 presents four inspiration curves of the whole process. Specifically, (a) a image of Oryx group; (b) the Oryx group is getting away from chase by lion; (c) the leaders help to adjust the position of the worst individual; (d) Arrive a safe place.

We have to admit that there are many behaviors or rules in real world. However, for simplicity, as mentioned before, the proposed algorithm only considers the following rules in this paper.

1. The weaker side of Oryx group is attacked by predator easily. (Rule 1)
2. Each member has a escape probability value. (Rule 2)
3. Each search individual updates its position according to the best individual and its neighbors.(Rule 3)
4. Two best members guide the individual in weaker side to modify its current position. (Rule 4)

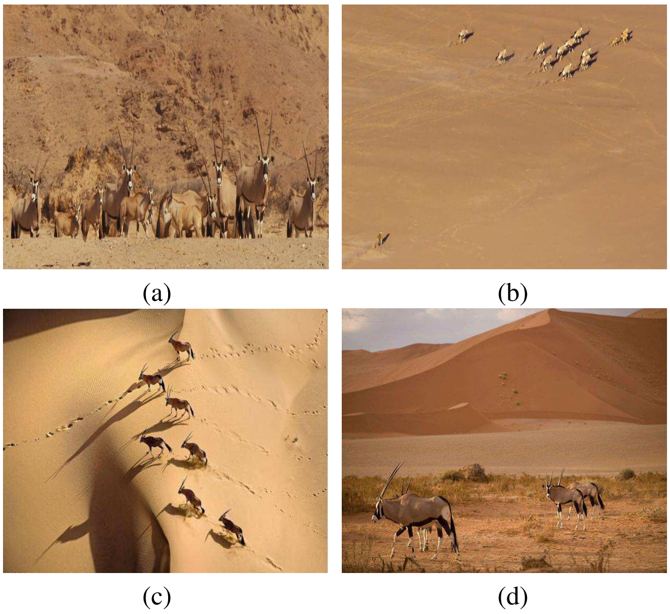

Figure 1: The inspiration curves of BSO (a)Oryx group (b)Getting away from chase by lion (c)Adjust the position of member (d)Arrive a safe place

3.2. Calculation model

In this section, the calculation models including escape phase and adjustment phase are first presented through converting the above rules properly. Then, we outline the pseudo codes and discuss the basic principle of BSO.

3.2.1. Group generation

It is assumed that the search population (Pop) is comprised of N feasible solutions ($X_i, i = 1, 2, 3..., N$), which are generated randomly using the following equations.

$$X_i(t) = (x_i^1(t), x_i^2(t), ..., x_i^D(t)), i = 1, 2, ..., N \tag{1}$$

$$x_i^k(t) = LB^k + r \times (UB^k - LB^k), k = 1, 2, ..., D \tag{2}$$

$$Pop(t) = (X_1(t), X_2(t), ..., X_N(t))$$

where N and D refers to the member of search members and dimension of issue, respectively. t denotes the current number of generation, r denotes a random number generated from (0, 1). LB and UB are the lower and upper boundaries of variables, respectively.

3.2.2. Escape phase

This section mainly simulates the behavior that the leaders help other species members away from the hunting. Due to predator usually attacks the weak side of population (Rule 1). The individual in the weak side is caught by the hunter easily. In BSO, we use the roulette strategy to set probability for determining whether the current agent will be caught (Rule 2). Obviously, in terms of fitness, the better the individual is, the lower probability it has. If the current individual is caught, it will be replaced by a new one. Besides that, Rule 3 shows that each agent modifies its position according to the best individual and its neighbors. To define the neighborhood structure of an individual, a ring topology structure (see Fig.2) is utilized in this paper [31]. If the index of individual is i, the index of the selected neighbor is $i + 1$. If the index of agent is N, the index of the selected neighbor is 1. Here, we assume that the best individual has the best fitness, the worst member is opposite. The following provides the calculation equations.

$$newX_i = X_i + \delta_i(pop^{best} - pop^{worst}) + \varphi_i(X_{neigh,i} - X_i) \tag{3}$$

where each number of δ_i and φ_i are generated from (0,1), pop^{best} and pop^{worst} are the best and worst individual of population, respectively, $X_{neigh,i}$ is the neighbor of X_i.

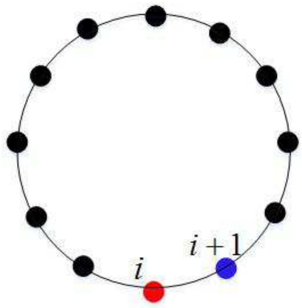

Figure 2: The ring topology structure

Fig.4 (a) depicts that the pursuer attacks the animal population labeled by black dot. In Fig.4 (b), it is assumed that the best individual (pop^{best}) is described by blue, the worst (pop^{worst}) is red, the current individual (X_i) is purple, and its neighbor ($X_{neigh,i}$) is green. They move in the opposite direction of attack and modify their positions according to Eq. (3).

3.2.3. Adjustment phase

The phase main simulates the Rule 4, the leaders guide the other members in the weak side of the population toward safer place. This action is commanded by the two best individuals of the population. From the view of optimization, this phase aims to make the worst individual better with the assistance of two best solutions of the population. Therefore, we define that this course is implemented using the simplex algorithm (Fig.3)[32], which is a popular technique launched by the previous solutions.

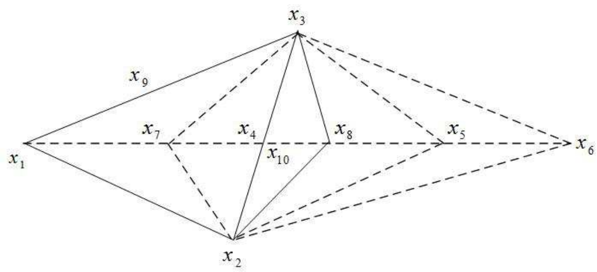

Figure 3: Schematic view of the simplex method

As shown in Fig.4 (c), the best member and the second best agent are labeled by blue and pink, respectively. The worst

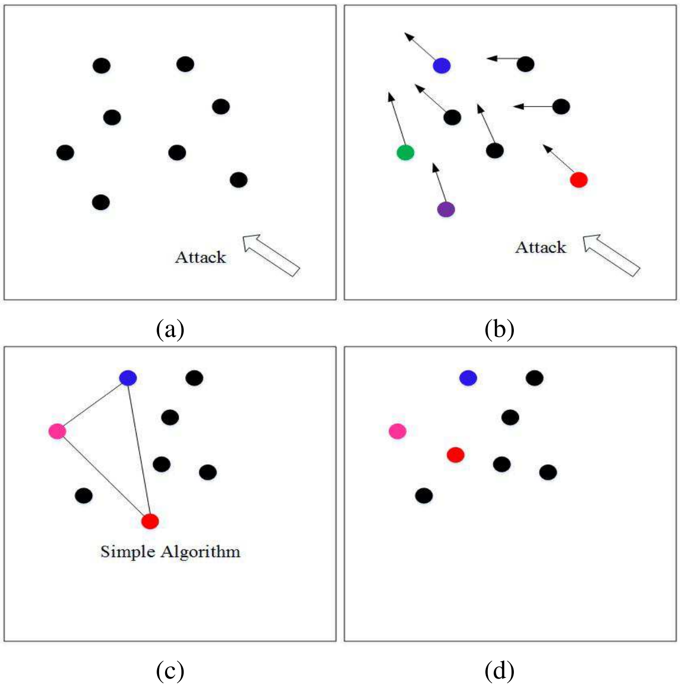

Figure 4: The whole search course curves of BSO

Algorithm 1 Biological Survival Optimizer

Input: N: the population size;
D: the dimension of optimization problem;
UB, LB: the lower and upper bounds of variables, respectively;
Control factors: Expansion factor (α), Contraction factor(β);
$MaxIter$: the termination criterion ;
Generate an initial population($Pop = X_1, X_2, ..., X_N$) using Eq.(1) and (2);
Evaluate the objective function values: $f(X_1), f(X_2), ..., f(X_N)$;
while the termination criterion is not satisfied **do**
  sort the population based on fitness;
  set the probability (Pr) of each agent using roulette;
  (Escape Phase)
  Find the best solution (pop^{best}) and the worst solution pop^{worst} of population;
  Find the neighbor of each agent
  for $i = 1 : N$ (each solution (X_i) in population) **do**
    if $rand < Pr(i)$ **then**
      Generate new solution($newX_i$)using Eq.(2);
    else
      Generate new solution($newX_i$)using Eq.(3);
    end if
  end for
  Evaluate $f(newX)$
  if $f(newX) < f(X_i)$ **then**
    $X_i = newX$;
    $f(X_i) = f(newX)$;
  end if
  (Adjustment Phase)
  Find the two best members and the worst individual of population
  Execute the simplex algorithm (**Appendix 1**) based on the three solutions ;
end while
  Output the best candidate solution;

individual (red dot) moves toward new place with the help of the other best members using the simple algorithm. The red dot will arrive a new position as shown in Fig.4 (d).

In other words, the movement within the region may ensure that the search individual will find better position. Finally, after the above mentioned phases, the initialization population can move toward a new domain. The quality of population will be improved over the course of generation. The main pseudocode of BSO is summarized in Algorithm 1.

3.3. Discussion

Similar to other heuristic algorithms, BSO also employs a population of candidate solutions initialized randomly in the search space to proceed to the global optimum. The difference is that the simple algorithm, a determination operator, is integrated into BSO. Totally, BSO has two main courses, escape phase and adjustment phase.

The first phase consists of a ring topology structure, the roulette strategy and a stochastic operator. The ring topology, inspired from the notion of neighborhood, connects the index of individual, where the first individual is the neighbor of the last individual and vice versa. It also can be regarded as a neighborhood technique, which aims to maintain diversity of the solutions and improve the exploration capability of algorithm. The roulette strategy is very close to Darwinian evolution theory, which means that the worst individual is eliminated easily, while the better member can survive. We use this strategy to keep the elitism solution and determine whether the worse individual will be replaced by a new one with a probability. It is generally known that to prevent premature convergence during the process of evolution, it is necessary to sample the whole search space systematically and generate new solutions diverse enough. Therefore, the stochastic search operator is used to improve diversity of the population and guarantee that each agent can move toward other places randomly. This not only generates some potential solutions distributed in the search space, but also improves the probability of finding the optimal solution. The second phase is implemented by the simplex algorithm, which simulates the Rule 4. Another important reason is that the simplex algorithm is a good algorithm, which can generate better solutions in each generation. According to the characteristics, the simplex algorithm is able to guide the search agents to move toward better place from generation to generation. It not only benefits to strengthen the local search, but also provides accurate search direction during the optimization process. Besides that, BSO is also a parameter-less method except two basic parameters used in the simple algorithm, which makes the algorithm simple and convenient.

4. Numerical experiments

This section aims to evaluate the effectiveness of BSO by solving a set of CEC2017 test problems with various characteristics.

4.1. Benchmark Problems

The CEC2017 benchmark problems is very suitable to evaluate the performance of an optimization algorithm comprehensively, since it has diverse features including multimodal, non-separable, asymmetrical and different variables subcomponents. According to the literature [33], these functions can be classified into four categories, unimodal ($f_1 - f_2$), multimodal($f_3 - f_9$), hybrid ($f_{10} - f_{19}$) and composition ($f_{20} - f_{29}$) parts. More details about the basic characteristics of these problems can be found in corresponding literature.

4.2. Evaluation indicator

This study adopts the following indicators for evaluating the effectiveness of algorithm.

- f_{mean} and *std*: refer to the average value and standard deviation of the function error, respectively;

- Wilcoxon's rank sum tests [34]: determines whether there exist statistically different between two algorithms for each problem at a 0.05 significance level, p-value less than 0.05 means that the effectiveness of two competitive methods is statistically different with 95% ($h = 1$), otherwise, there is no significant difference ($h = 0$). Besides that, the relevant comparison results are recorded at the bottom of each Table, and ‡, † and ≀ used in this manuscript indicate that the performance of BSO is better than, worse than and similar to that of the corresponding algorithm, respectively.

4.3. Parameter settings

By reference [33], the stopping condition is set to the maximum number of iterations ($MaxIter$), which is defined as $10,000$ for all algorithms. 30 runs are independently carried out to reduce the computation error. In addition, eight existing optimization algorithms are utilized to compare the performance of BSO, the following provides the relevant descriptions and involved parameters according to corresponding references, DEDVR[35], MGSCA[36], MGWO[37], WSSA[38], GLFGWO[39], QGDA[40], SGLSCA[41], CMAES[42], Two important parameters used in the simple algorithm are set as $\alpha = 2$, $\beta = 0.5$ in BSO, respectively.

4.4. Experiment results

The experimental results calculated by all algorithms are recorded in Table 1-4. In addition, with the purpose of comparing the convergence rate of BSO with the other compared algorithms, several evolution curves of the benchmark problems are shown in Fig.5-6.

For $f_1 - f_2$ unimodal problems, it is can be observed from Table 1 that although the proposed algorithm has the same f_{mean} and *std* values as CMAES on f_2, CMAES has the best f_{mean} value and standard deviation with respective to its peers. According to the obtained p-value and h-value results, BSO performs significantly better than the other seven compared algorithms, performs similar to CMAES on f_2. Fig.5 plots the convergence graphs of f_1, as may be observed, the potential convergence performance of BSO is slightly slower than QGDA and GLFGWO at the beginning of generations, but it is second and outperforms other competitors finally.

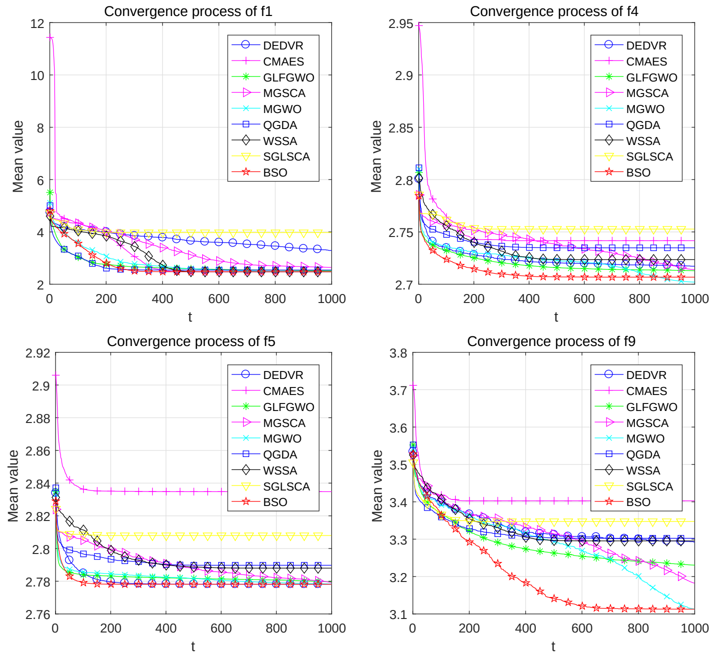

Figure 5: Convergence curves of all computation algorithms on unimodal and multimodal functions

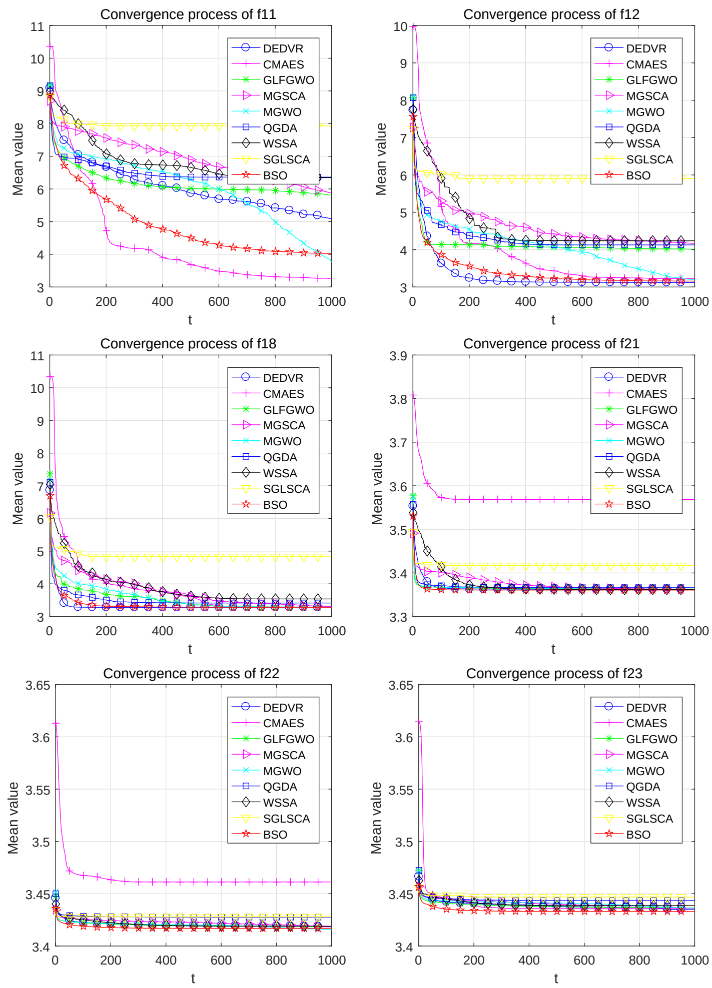

Figure 6: Convergence curves of all computation algorithms on hybrid and composition functions

Table 1: Experiment results obtained by all methods on CEC2017 unimodal functions

Function	Index	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
f_1	f_{mean}	1.0319e+03‡	4.2261e+06‡	1.9800e+04‡	3.6504e+03‡	4.8786e+05‡	1.8372e+08‡	3.5722e+09‡	**1.0000e+02**†	4.6116e+02
	std	3.8638e+02	1.0583e+07	1.0076e+04	3.9511e+03	1.9762e+06	5.0147e+08	1.0923e+09	**0**	8.7671e+02
	p	5.5999e-07	3.0199e-11	3.0199e-11	7.0430e-07	3.6897e-11	3.0199e-11	3.0199e-11	1.2118e-12	-
	h	1	1	1	1	1	1	1	1	-
f_2	f_{mean}	1.9440e+03‡	4.4142e+02‡	3.0043e+02‡	3.0000e+02‡	3.2334e+02‡	3.4845e+02‡	9.4498e+03‡	**3.0000e+02**$^{\wr}$	**3.0000e+02**
	std	4.7565e+02	9.7015e+01	1.0002e+01	2.3479e-12	2.1443e+01	6.7504e+01	2.8366e+03	**0**	**0**
	p	1.2118e-12	1.2118e-12	1.2118e-12	1.1941e-12	1.2118e-12	1.2118e-12	1.2118e-12	9.7285e-01	-
	h	1	1	1	1	1	1	1	0	-
	$\ddagger$	-	2	2	2	2	2	2	0	-
	$\dagger$	-	0	0	0	0	0	0	1	-
	$\wr$	-	0	0	0	0	0	0	1	-

Table 2: Experiment results obtained by all methods on CEC2017 simple multimodal functions

Function	Index	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
f_3	f_{mean}	4.0301e+02‡	4.0732e+02‡	4.0387e+02‡	4.0434e+02‡	4.0783e+02‡	4.2439e+02‡	5.6436e+02‡	**4.0000e+02**†	4.0004e+02
	std	1.3954e-01	2.7442e+01	4.2161e-01	1.4809e+00	3.0968e+00	5.0127e+01	6.4458e+01	**0**	1.5096e-02
	p	3.0199e-11	3.0199e-11	3.0199e-11	5.5707e-10	3.0199e-11	3.0199e-11	3.0199e-11	1.2118e-12	-
	h	1	1	1	1	1	1	1	1	-
f_4	f_{mean}	5.2146e+02‡	5.1762e+02‡	**5.0355e+02**†	5.2908e+02‡	5.1631e+02‡	5.4301e+02‡	5.6585e+02‡	5.5157e+02‡	5.0904e+02
	std	2.1212e+00	6.2897e+00	**1.5048e+00**	1.3297e+01	8.2291e+00	1.9088e+01	9.0383e+00	6.7282e+01	3.8641e+00
	p	4.9024e-11	2.9831e-07	1.4137e-08	6.5869e-11	5.0544e-06	5.9778e-11	2.9747e-11	5.6126e-05	-
	h	1	1	1	1	1	1	1	1	-
f_5	f_{mean}	**6.0000e+02**‡	6.0143e+02‡	6.0015e+02‡	6.1376e+02‡	6.0193e+02‡	6.1626e+02‡	6.4262e+02‡	6.8356e+02‡	**6.0000e+02**
	std	6.1728e-06	1.2792e+00	9.3692e-02	1.0078e+01	1.7962e+00	1.1214e+01	6.4100e+00	2.9818e+01	**2.0868e-06**
	p	6.6716e-11	1.9879e-11	1.9879e-11	1.9879e-11	1.9779e-11	1.9879e-11	1.9879e-11	1.9879e-11	-
	h	1	1	1	1	1	1	1	1	-
f_6	f_{mean}	7.3365e+02‡	7.3485e+02‡	**7.1752e+02**†	7.4821e+02‡	7.3101e+02‡	7.4236e+02‡	8.1405e+02‡	1.1649e+03‡	7.1771e+02
	std	5.0879e+00	1.0534e+01	**2.4691e+00**	1.5366e+01	8.6605e+00	1.4267e+01	1.3692e+01	5.2062e+02	4.3839e+00
	p	1.4643e-10	1.5581e-08	1.0035e-03	6.0658e-11	1.5581e-08	3.0199e-11	3.0199e-11	1.5581e-08	-
	h	1	1	1	1	1	1	1	1	-
f_7	f_{mean}	8.2285e+02‡	8.1681e+02‡	**8.0422e+02**†	8.2862e+02‡	8.1712e+02‡	8.2623e+02‡	8.5274e+02‡	8.6766e+02‡	8.0779e+02
	std	3.6567e+01	5.1419e+00	**2.0365e+00**	6.9803e+00	7.2748e+00	1.0669e+01	9.8752e+00	5.9622e+01	3.1801e+00
	p	2.9747e-11	2.8865e-09	4.9140e-05	4.4354e-11	2.5739e-08	1.3103e-10	2.9747e-11	9.9816e-10	-
	h	1	1	1	1	1	1	1	1	-
f_8	f_{mean}	9.0000e+02‡	9.0820e+02‡	9.0001e+02‡	9.5237e+02‡	9.0866e+02‡	9.3377e+02‡	1.4395e+03‡	4.2185e+03‡	**9.0000e+02**
	std	3.6348e-09	1.4796e+01	1.9278e-02	9.7694e+01	1.2484e+01	5.8966e+01	1.8562e+02	1.3868e+03	**0**
	p	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	-
	h	1	1	1	1	1	1	1	1	-
f_9	f_{mean}	1.9604e+03‡	1.5229e+03‡	1.2977e+03$^{\wr}$	1.9744e+03‡	1.7004e+03‡	2.0066e+03‡	2.2256e+03‡	2.5278e+03‡	**1.2964e+03**
	std	1.4274e+02	2.3080e+02	1.3024e+02	2.5098e+02	3.1052e+02	3.2275e+02	1.9424e+02	4.7510e+02	**2.1766e+02**
	p	6.6955e-11	2.3885e-04	6.5204e-01	2.6099e-10	1.6062e-06	2.6099e-10	3.6897e-11	4.5043e-11	-
	h	1	1	0	1	1	1	1	1	-
	$\ddagger$	7	7	3	7	7	7	7	6	-
	$\dagger$	0	0	3	0	0	0	0	1	-
	$\wr$	0	0	1	0	0	0	0	0	-

Table 2 summarizes the calculation results on $f_3 - f_9$, it is obvious that CMAES obtains the best results on f_3, although BSO performs slightly weaker than MGWO on f_4, f_6 and f_7, BSO performs significantly better than other compared algorithms on most simple multimodal problems according to the Wilcoxon's test values. This situation is ascribed to the powerful exploration capability. BSO has the simplex algorithm and two main stochastic search operators, which are designed to guarantee that each search individual is able to move toward potential regions randomly, and generate promising offspring agent. Besides that, it can be observed from Fig.5 that BSO has better convergence performance with respect to its competitors.

Different from the above two types of functions, hybrid problems ($f_{10} - f_{19}$) usually require various techniques to optimize different subcomponents partitioned on variables. The statistical and comparison results recorded in Table 3 demonstrate that BSO has better performance than its competitors on f_{10}, f_{13} and f_{14}, performs slightly worse than DEDVR on five test problems and performs similar to MGWO on f_{11}, f_{15} and f_{16}. Besides that, Fig.6 provides three evolution graphes, it is appear that although BSO is slight worse than CMAES and MGWO, it is significant better than other compared methods over the course of iterations.

The last category is composition functions ($f_{20} - f_{29}$) with the properties of non-separable, asymmetrical and different properties around different local optima, they are much difficult to be optimized effectively. It is can be summarized from the experimental data provided in Table 4 that the proposed algorithm obtain the best results on f_{21}, f_{23} and f_{26} problems. Although BSO is slightly weaker than DEDVR and MGWO considering f_{24} and f_{29}, respectively, there are no great differences between them according to the Wilcoxon's test results. From the given convergence graphs in Fig.6, BSO has the best convergence rate compared with its peers.

4.5. Sensitivity analysis

As mentioned before, the proposed algorithm consists of two different key components. This subsection aims to discuss the role that each component plays in dealing with CEC2017 benchmark functions. Specifically, To demonstrate that the stochastic search operator has important effect on the proposed

Table 3: Experiment results obtained by all methods on CEC2017 simple hybrid functions

Function	Index	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
f_{10}	f_{mean}	1.1057e+03‡	1.1151e+03‡	1.1033e+03‡	1.1889e+03‡	1.1189e+03‡	1.1670e+03‡	1.8949e+03‡	1.1747e+03‡	**1.1013e+03**
	std	1.0402e+00	7.2890e+00	1.4765e+00	8.5037e+01	1.1331e+01	5.9502e+01	3.0846e+02	4.7645e+01	**1.3658e+00**
	p	2.3715e-10	3.6897e-11	4.8011e-07	3.0199e-11	3.6897e-11	3.0199e-11	3.0199e-11	3.0180e-11	-
	h	1	1	1	1	1	1	1	1	-
f_{11}	f_{mean}	1.2200e+05‡	7.5625e+05‡	5.7710e+03≀	2.2254e+06‡	6.4264e+05‡	2.3054e+06‡	8.5132e+07‡	**1.8196e+03**†	1.0425e+04
	std	5.4748e+04	1.2051e+06	3.1402e+03	2.4257e+06	7.9201e+05	5.2739e+06	5.9002e+07	**3.0055e+02**	1.5451e+04
	p	4.9752e-11	8.1014e-10	5.7459e-02	9.9186e-11	1.0702e-09	7.3803e-10	3.0199e-11	1.0907e-05	-
	h	1	1	0	1	1	1	1	1	-
f_{12}	f_{mean}	**1.3194e+03**†	1.4637e+04‡	1.5714e+03‡	1.7273e+04‡	1.0310e+04‡	1.3416e+04‡	8.0589e+05‡	1.6669e+03‡	1.4407e+03
	std	**3.9944e+00**	1.1205e+04	1.5995e+02	1.1622e+04	6.4914e+03	9.2361e+03	7.7186e+05	1.7989e+02	1.7997e+02
	p	2.5974e-05	3.0199e-11	1.8916e-04	3.0199e-11	3.0199e-11	3.3384e-11	3.0199e-11	8.2919e-06	-
	h	1	1	1	1	1	1	1	1	-
f_{13}	f_{mean}	1.4523e+03‡	1.4879e+03‡	1.4470e+03‡	1.5706e+03‡	1.4769e+03‡	1.4953e+03‡	3.8262e+03‡	1.5025e+03‡	**1.4289e+03**
	std	2.7682e+01	3.2775e+01	1.2736e+01	1.5509e+02	2.6420e+01	4.2734e+01	1.2448e+03	1.5219e+02	**1.2017e+01**
	p	3.4971e-09	1.7769e-10	1.2860e-06	3.0199e-11	3.8202e-10	5.4941e-11	3.0199e-11	2.0283e-07	-
	h	1	1	1	1	1	1	1	1	-
f_{14}	f_{mean}	1.5733e+03‡	1.6541e+03‡	1.5410e+03‡	2.3745e+03‡	1.6108e+03‡	2.1643e+03‡	6.5653e+03‡	1.6281e+03‡	**1.5246e+03**
	std	6.3252e+01	1.7567e+02	1.9387e+01	7.7494e+02	1.1700e+02	1.0846e+03	2.8249e+03	9.1079e+01	**2.9586e+01**
	p	3.1573e-05	4.8011e-07	1.0035e-03	3.3384e-11	7.0430e-07	3.3384e-11	3.0199e-11	3.0811e-08	-
	h	1	1	1	1	1	1	1	1	-
f_{15}	f_{mean}	**1.6013e+03**†	1.6525e+03‡	1.6099e+03‡	1.6980e+03‡	1.6772e+03‡	1.8447e+03‡	2.0133e+03‡	1.8617e+03‡	1.6193e+03
	std	**5.0948e-01**	4.9493e+01	1.2461e+01	9.1508e+01	6.7634e+01	1.4750e+02	7.3641e+01	1.7255e+01	3.2407e+01
	p	1.1228e-02	2.1327e-05	4.6427e-01	1.8608e-06	3.0939e-06	1.8567e-09	3.0199e-11	1.5581e-08	-
	h	1	1	0	1	1	1	1	1	-
f_{16}	f_{mean}	1.7791e+03‡	1.7395e+03‡	1.7253e+03‡	1.7629e+03‡	1.7603e+03‡	1.7722e+03‡	1.8308e+03‡	1.8610e+03‡	**1.7298e+03**
	std	5.5023e+01	1.2164e+01	9.2811e+00	1.5649e+01	3.0490e+01	3.9448e+01	2.7842e+01	1.7017e+02	**1.9892e+01**
	p	4.1825e-09	1.4932e-04	7.9584e-01	2.9215e-09	2.5721e-07	1.1023e-08	8.9934e-11	1.8567e-09	-
	h	1	1	0	1	1	1	1	1	-
f_{17}	f_{mean}	**1.8115e+03**†	3.1469e+04‡	3.9474e+03‡	2.7241e+04‡	1.7953e+04‡	2.3253e+04‡	2.7890e+06‡	1.9000e+03≀	1.9467e+03
	std	**2.6486e+00**	1.4379e+04	3.7592e+03	1.4925e+04	1.2230e+04	1.5074e+04	3.0927e+06	4.9960e+01	1.6381e+02
	p	3.0199e-11	3.0199e-11	1.2023e-08	3.0199e-11	3.3384e-11	3.6897e-11	3.0199e-11	5.0114e-01	-
	h	1	1	1	1	1	1	1	0	-
f_{18}	f_{mean}	**1.9007e+03**†	2.0424e+03‡	1.9238e+03‡	3.4512e+03‡	1.9496e+03‡	2.5892e+03‡	6.7892e+04‡	1.9432e+03‡	1.9144e+03
	std	**1.6273e-01**	4.6810e+02	1.5957e+01	2.5705e+03	3.3129e+01	1.1593e+03	4.3396e+04	2.5426e+01	2.1034e+01
	p	3.0199e-11	2.4386e-09	3.0059e-04	7.3891e-11	9.2603e-09	1.6132e-10	3.0199e-11	5.1857e-07	-
	h	1	1	1	1	1	1	1	1	-
f_{19}	f_{mean}	**2.0000e+03**†	2.0368e+03‡	2.0174e+03‡	2.1145e+03‡	2.0596e+03‡	2.1613e+03‡	2.2601e+03‡	2.4745e+03‡	2.0136e+03
	std	**3.5883e-10**	1.0302e+01	8.4467e+00	5.9893e+01	3.4872e+01	6.8849e+01	3.7570e+01	2.5073e+02	1.0446e+01
	p	5.2847e-10	1.9512e-10	1.5007e-02	3.0104e-11	6.0473e-11	3.0104e-11	3.0104e-11	3.0104e-11	-
	h	1	1	1	1	1	1	1	1	-
	‡	5	10	7	10	10	10	10	8	-
	†	5	0	0	0	0	0	0	1	-
	≀	0	0	3	0	0	0	0	1	-

Table 5: Results obtained by different versions of BSO on unimodal functions

Function	Index	BSOV	BSOVV	BSO
f_1	f_{mean}	2.6190e+04‡	2.6691e+04‡	**4.6116e+02**
	std	7.4271e+03‡	8.9279e+03‡	**8.7671e+02**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_2	f_{mean}	5.4363e+03‡	5.4697e+03‡	**3.0000e+02**
	std	5.1681e+03‡	4.8619e+03‡	**0**
	p	1.2118e-12	1.2118e-12	-
	h	1	1	-
	‡	2	2	-
	†	0	0	-
	≀	0	0	-

strategy, BSOV is designed by removing the stochastic operator, in the other words, BSOV just has the second search mechanism. Similarly, to study the role of the simple algorithm, BSO is also modified by excluding the second search strategy, called BSOVV. These two variants are compared with the original BSO, and Tables 5-8 report the corresponding computing and comparison results.

It is not difficult to observe from the results that BSO outperforms two modified variants (BSOV and BSOVV) for almost all test problems. This means that each search strategy indeed helps improve the quality of population in varying environments. The reason may originate from the fact that the stochastic operator is designed by using different individuals, which helps to generate promising solutions to some extent. The comparisons between the two different variants and the original BSO illustrate that each part has an significant effect on the performance of BSO, and removing any of them reduces performance. Therefore, it is necessary to combine them together and format the BSO algorithm.

Apart from the component analysis, there are two important parameters in BSO named the expansion factor (α) and contraction factor (β). The discussion on the influence of two parameters will not appear here. The reason is that much effort has been devoted to designing effective parameter combinations, and different parameters may suitable for different problems. This paper only defines a base case of $\alpha = 2$ and $\beta = 0.5$ for computing the optimization problems. In our future work, scholars who are interested in this can also investigate it and further improve the search performance of the algorithm.

5. Engineering optimization problems

To further explore the effectiveness of the proposed algorithm in practice, three widely used practical engineering problems

Table 4: Experiment results obtained by all methods on CEC2017 Composition functions

Function	Index	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
f_{20}	f_{mean}	2.2631e+03†	2.2571e+03$^{\wr}$	2.2596e+03†	**2.2108e+03**†	2.2948e+03$^{\wr}$	2.2706e+03$^{\wr}$	2.2475e+03†	2.3577e+03‡	2.3042e+03
	std	4.2121e+01	5.9784e+01	5.2951e+01	**3.1756e+01**	4.7238e+01	7.5002e+01	2.1824e+01	7.7927e+01	2.8500e+01
	p	1.3017e-03	5.7929e-01	1.2023e-08	9.0632e-08	8.2357e-02	9.1171e-01	8.4687e-09	5.9659e-03	-
	h	1	0	1	1	0	0	1	1	-
f_{21}	f_{mean}	2.3042e+03‡	2.2984e+03‡	2.2981e+03‡	2.3011e+03‡	2.3077e+03‡	2.3244e+03‡	2.6108e+03‡	3.7032e+03‡	**2.2943e+03**
	std	5.8261e-01	2.0934e+01	1.8483e+01	1.7564e+01	3.2662e+00	5.4777e+01	9.8475e+01	1.0902e+03	**2.5632e+01**
	p	4.0721e-11	2.8290e-08	1.4419e-03	1.5951e-07	3.3342e-11	3.0161e-11	3.0161e-11	9.7383e-04	-
	h	1	1	1	1	1	1	1	1	-
f_{22}	f_{mean}	2.6227e+03‡	2.6195e+03‡	**2.6056e+03**†	2.6239e+03‡	2.6210e+03‡	2.6767e+03‡	2.6788e+03‡	2.8921e+03‡	2.6112e+03
	std	3.2453e+00	8.0289e+00	**2.0200e+00**	7.7408e+00	6.2892e+00	3.8734e+01	7.8490e+00	5.0972e+02	4.2478e+00
	p	2.6099e-10	1.8682e-05	7.0881e-08	2.9215e-09	1.0666e-07	4.0772e-11	3.0199e-11	9.2113e-05	-
	h	1	1	1	1	1	1	1	1	-
f_{23}	f_{mean}	2.7210e+03‡	2.7283e+03‡	2.7308e+03‡	2.7457e+03‡	2.7316e+03‡	2.7762e+03‡	2.7978e+03‡	2.7292e+03‡	**2.7107e+03**
	std	6.7883e+01	7.4916e+01	2.7531e+00	4.7357e+01	6.3029e+01	1.0244e+02	3.2613e+01	4.3556e+01	**8.4185e+01**
	p	4.7120e-04	2.6795e-04	1.3843e-06	1.4908e-06	1.4419e-03	1.5951e-07	3.9613e-08	2.7530e-03	-
	h	1	1	1	1	1	1	1	1	-
f_{24}	f_{mean}	**2.9002e+03**$^{\wr}$	2.9267e+03‡	2.9233e+03$^{\wr}$	2.9296e+03‡	2.9264e+03‡	2.9228e+03$^{\wr}$	3.0949e+03‡	2.9284e+03‡	2.9207e+03
	std	**3.1173e+00**	2.1598e+01	2.3952e+01	2.4702e+01	2.3342e+01	2.3860e+01	6.4755e+01	2.3328e+01	2.4978e+01
	p	8.0725e-01	5.0775e-03	7.2402e-02	4.0564e-02	6.3692e-03	1.3726e-01	2.9991e-11	3.2003e-02	-
	h	0	1	0	1	1	0	1	1	-
f_{25}	f_{mean}	2.9000e+03†	2.9380e+03†	2.8969e+03$^{\wr}$	**2.8922e+03**†	2.9190e+03†	3.0436e+03$^{\wr}$	3.2686e+03‡	3.0913e+03$^{\wr}$	3.0131e+03
	std	8.8982e-08	3.9913e+01	**1.7824e+01**	9.2643e+01	7.5609e+01	3.4903e+02	9.2795e+01	4.1270e+02	2.7170e+02
	p	7.3246e-02	7.4573e-03	1.3194e-01	1.4157e-09	4.0744e-02	5.7930e-02	7.0900e-08	4.6235e-01	-
	h	1	1	0	1	1	0	1	0	-
f_{26}	f_{mean}	3.0992e+03‡	3.0952e+03‡	3.0991e+03‡	3.0939e+03$^{\wr}$	3.0962e+03‡	3.1377e+03‡	3.1099e+03‡	3.1231e+03‡	**3.0902e+03**
	std	1.5293e+01	1.4304e+01	3.1735e+00	3.1169e+00	9.1386e+00	4.4610e+01	4.5527e+00	1.4034e+02	**1.4199e+00**
	p	5.5246e-04	8.9756e-04	1.1395e-07	3.7242e-01	4.1068e-09	2.9506e-11	2.9506e-11	3.5236e-11	-
	h	1	1	1	0	1	1	1	1	-
f_{27}	f_{mean}	**3.1002e+03**†	3.2117e+03†	3.2438e+03$^{\wr}$	3.2095e+03†	3.3463e+03$^{\wr}$	3.3540e+03$^{\wr}$	3.3918e+03$^{\wr}$	3.2376e+03†	3.3272e+03
	std	**3.3724e-01**	7.3251e+01	1.5621e+02	1.0239e+02	1.0340e+02	1.4536e+02	5.8015e+01	1.4392e+02	1.6981e+02
	p	6.0083e-05	4.0640e-02	8.8752e-01	3.1444e-02	1.3457e-01	4.9813e-01	6.1689e-02	3.1582e-02	-
	h	1	1	0	1	0	0	0	1	-
f_{28}	f_{mean}	3.1943e+03‡	3.1610e+03‡	**3.1409e+03**†	3.2100e+03‡	3.1993e+03‡	3.3218e+03‡	3.2927e+03‡	3.2625e+03‡	3.1539e+03
	std	8.6483e+00	3.1496e+01	**9.3159e+00**	5.5219e+01	4.9965e+01	1.1496e+02	6.0272e+01	9.3095e+01	2.7673e+01
	p	3.0811e-08	4.5146e-02	3.9167e-02	1.1937e-06	7.0881e-08	2.8716e-10	1.3289e-10	4.5726e-09	-
	h	1	1	1	1	1	1	1	1	-
f_{29}	f_{mean}	7.9258e+04†	1.3130e+05†	**3.2456e+04**$^{\wr}$	1.1131e+05†	7.4018e+05‡	3.2929e+06‡	1.2713e+06‡	2.1635e+05$^{\wr}$	3.8183e+05
	std	4.4614e+04	3.0366e+05	**1.4904e+05**	2.5013e+05	7.9459e+05	4.1827e+06	7.9934e+05	3.9900e+05	5.7220e+05
	p	2.7084e-02	4.2064e-02	9.1171e-01	2.4155e-02	5.5601e-04	9.5299e-07	3.1192e-06	6.5667e-01	-
	h	1	1	0	1	1	1	1	0	-
	$\ddagger$	5	6	3	5	7	6	8	8	-
	$\dagger$	4	3	3	4	1	0	1	0	-
	$\wr$	1	1	4	1	2	4	1	2	-

Table 9: Experimental results of all methods for Tension/compression spring design problem

Variables	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
x_1	0.06899395	0.05003490	0.06899846	0.05000000	0.06900761	0.05655288	0.05000000	0.05155974	0.05146060
x_2	0.93343179	0.31815726	0.93356942	0.31277075	0.93392044	0.48540956	0.31474035	0.35361405	0.35124589
x_3	2.00000000	13.9736461	2.00000000	14.6634210	2.00000000	6.42231835	14.3898574	11.4733015	11.6171624
f_{mean}	0.01777315	0.01271599	0.01777653	0.01327191	0.01778504	0.01278983	0.01320437	0.01308636	**0.01267585**
std	3.5696e-14	1.4994e-05	2.1943e-06	6.8107e-04	5.3640e-06	9.4641e-05	9.9265e-05	5.6142e-04	**1.3885e-05**

are adopted [43-45]. Note that the stopping condition, the number of runs and the other relevant settings keep the same as that in previous subsection 4.3. In addition, to deal with the constrains involved in problems, an effective constraint handling technique is utilized [1]. Tables 9-11 summarize the experiment results considering the mean error value (f_{mean}) and standard deviation (std) on the corresponding statistical results.

5.1. case 1

The main target of compression spring problem aims to optimize the weight considering four different constraint conditions, which involves three control variables (x_1,x_2 and x_3). Eq.(4) and Fig.7 provide the mathematical model and architecture graph, respectively.

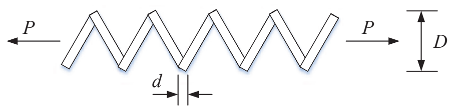

Figure 7: The schematic of the tension/compression spring design problem

$$
\begin{aligned}
&Min. f(x) = (x_3 + 2)x_2 x_1^2 \\
s.t.\quad &g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leqslant 0 \\
&g_2(x) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leqslant 0 \\
&g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leqslant 0 \\
&g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leqslant 0
\end{aligned}
\tag{4}
$$

Table 10: Experimental results of all methods for Pressure vessel design problem

Variables	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
x_1	0.96099558	0.78240189	0.96168907	1.32487442	0.96318757	1.08351033	1.23020054	0.78574452	0.77816864
x_2	0.47502063	0.38683703	0.47566403	0.65488612	0.47640431	0.53556283	0.61133921	0.38839392	0.38464916
x_3	49.7925162	40.5375510	49.8201449	68.6463432	49.9037386	56.1372281	63.5981424	40.7121513	40.3196187
x_4	99.9998919	197.002772	99.8162183	10.2633410	99.1322577	56.0581137	18.6301442	194.606888	200.000000
f_{mean}	6277.01738	6187.94631	6281.12298	7345.38581	6458.13436	6541.70494	21545.1717	6191.41452	**5885.33277**
std	4.1044e+04	5.1283e+02	1.4507e+00	1.1898e+03	3.5407e+02	5.5648e+02	7.9909e+03	2.5431e+02	**6.3882e-07**

Table 11: Experimental results of all methods for three-bar truss design problem

Variables	DEDVR	MGSCA	MGWO	WSSA	GLFGWO	QGDA	SGLSCA	CMAES	BSO
x_1	0.78867513	0.78729684	0.78872584	0.78850149	0.78916276	0.78826187	0.78872136	0.78867513	0.78869913
x_2	0.40824828	0.41216083	0.40810531	0.40873964	0.40687971	0.40941930	0.40811755	0.40824828	0.40818042
f_{mean}	**263.895843**	263.898089	263.896030	263.897180	263.897041	263.897738	265.705696	**263.895843**	**263.895843**
std	**1.7345e-13**	1.7897e-03	2.1630e-04	2.6230e-03	9.5081e-04	1.5539e-03	1.1997e+00	6.4193e-07	6.0960e-07

Table 6: Results obtained by different versions of BSO on multimodal functions

Function	Index	BSOV	BSOVV	BSO
f_3	f_{mean}	4.1485e+02‡	4.0968e+02‡	**4.0004e+02**
	std	2.6003e+01‡	1.1468e+01‡	**1.5096e-02**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_4	f_{mean}	5.2176e+02‡	5.1984e+02‡	**5.0904e+02**
	std	9.0752e+00‡	8.9914e+00‡	**3.8641e+00**
	p	1.0901e-08	3.7715e-07	-
	h	1	1	-
f_5	f_{mean}	6.0784e+02‡	6.0969e+02‡	**6.0000e+02**
	std	4.7525e+00‡	9.1876e+00‡	**2.0868e-06**
	p	1.9879e-11	1.9879e-11	-
	h	1	1	-
f_6	f_{mean}	7.2487e+02‡	7.2479e+02‡	**7.1771e+02**
	std	6.3683e+00‡	7.2606e+00‡	**4.3839e+00**
	p	5.4620e-06	4.9426e-05	-
	h	1	1	-
f_7	f_{mean}	8.1754e+02‡	8.1859e+02‡	**8.0779e+02**
	std	8.0160e+00‡	7.2340e+00‡	**3.1801e+00**
	p	1.1365e-07	4.1333e-09	-
	h	1	1	-
f_8	f_{mean}	9.3477e+02‡	9.5782e+02‡	**9.0000e+02**
	std	7.3327e+01‡	1.1255e+02‡	**0**
	p	1.2118e-12	1.2118e-12	-
	h	1	1	-
f_9	f_{mean}	1.9584e+03‡	2.0749e+03‡	**1.2964e+03**
	std	4.1013e+02‡	3.7811e+02‡	**2.1766e+02**
	p	2.3897e-08	9.9186e-11	-
	h	1	1	-
	‡	7	7	-
	†	0	0	-
	≀	0	0	-

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

In this problem, the results listed in Table 9 clearly show that the performance of BSO is remarkable with respect to other optimizers on different evaluate indicators. Although the standard deviation values of BSO is a little weaker than that of DEDVR, MGWO and GLFGWO, it generates the best 'f_{mean}' values, which means that the proposed algorithm is able to obtain a set of optimal design with minimum weight compared to other competitors.

5.2. Case 2

The main target of pressure vessel problem aims to optimize overall cost including material, forming and welding costs, which are expressed by four different constraint conditions, two discrete control variables (x_1 and x_2) and two continuous control variables (x_3 and x_4). Eq.(5) and Fig.8 provide the mathematical model and architecture graph, respectively.

Table 7: Results obtained by different versions of BSO on Hybrid functions

Function	Index	BSOV	BSOVV	BSO
f_{10}	f_{mean}	1.1759e+03‡	1.1793e+03‡	**1.1013e+03**
	std	3.8546e+01‡	3.6104e+01‡	**1.3658e+00**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_{11}	f_{mean}	2.0631e+05‡	1.7235e+06‡	**1.0425e+04**
	std	5.0873e+05‡	3.5606e+06‡	**1.5451e+04**
	p	1.1937e-06	1.6980e-08	-
	h	1	1	-
f_{12}	f_{mean}	2.0979e+04‡	9.7821e+03‡	**1.4407e+03**
	std	1.6778e+04‡	1.2767e+04‡	**1.7997e+02**
	p	3.0199e-11	3.3384e-11	-
	h	1	1	-
f_{13}	f_{mean}	4.0764e+04‡	5.6419e+03‡	**1.4289e+03**
	std	6.5606e+03‡	8.4343e+03‡	**1.2017e+01**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_{14}	f_{mean}	1.8689e+04‡	2.3807e+04‡	**1.5246e+03**
	std	2.2476e+04‡	3.3025e+04‡	**2.9586e+01**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_{15}	f_{mean}	1.9033e+03‡	1.8608e+03‡	**1.6193e+03**
	std	1.7681e+02‡	1.7894e+02‡	**3.2407e+01**
	p	9.7555e-10	2.4386e-09	-
	h	1	1	-
f_{16}	f_{mean}	1.8012e+03‡	1.8137e+03‡	**1.7298e+03**
	std	5.9060e+01‡	9.1546e+01‡	**1.9892e+01**
	p	1.0702e-09	7.3803e-10	-
	h	1	1	-
f_{17}	f_{mean}	2.5597e+04‡	2.5414e+04‡	**1.9467e+03**
	std	1.7230e+04‡	1.7241e+04‡	**1.6381e+02**
	p	3.0199e-11	3.3384e-11	-
	h	1	1	-
f_{18}	f_{mean}	1.3042e+04‡	1.7495e+04‡	**1.9144e+03**
	std	1.2275e+04‡	2.7275e+04‡	**2.1034e+01**
	p	3.0199e-11	3.0199e-11	-
	h	1	1	-
f_{19}	f_{mean}	2.1574e+03‡	2.1256e+03‡	**2.0136e+03**
	std	6.8306e+01‡	7.6621e+01‡	**1.0446e+01**
	p	3.0199e-11	3.0104-11	-
	h	1	1	-
	‡	10	10	-
	†	0	0	-
	≀	0	0	-

Table 8: Results obtained by different versions of BSO on Composition functions

Function	Index	BSOV	BSOVV	BSO
f_{20}	f_{mean}	2.2972e+03‡	**2.2863e+03**≀	2.3042e+03
	std	4.7518e+01‡	5.3210e+01≀	**2.8500e+01**
	p	4.2259e-03	3.7108e-01	-
	h	1	0	-
f_{21}	f_{mean}	2.3027e+03‡	2.3025e+03‡	**2.2943e+03**
	std	1.6113e+01‡	1.7252e+01‡	**2.5632e+01**
	p	1.0690e-09	2.0317e-09	-
	h	1	1	-
f_{22}	f_{mean}	2.6501e+03‡	2.6439e+03‡	**2.6112e+03**
	std	2.2711e+01‡	1.8447e+01‡	**4.2478e+00**
	p	1.4643e-10	1.0937e-10	-
	h	1	1	-
f_{23}	f_{mean}	2.7623e+03‡	2.7552e+03‡	**2.7107e+03**
	std	5.0628e+01‡	6.9058e+01‡	**8.4185e+01**
	p	2.9186e-09	2.0134e-08	-
	h	1	1	-
f_{24}	f_{mean}	2.9300e+03‡	2.9326e+03‡	**2.9207e+03**
	std	2.8083e+01‡	2.7825e+01‡	**2.4978e+01**
	p	2.7063e-02	3.9137e-02	-
	h	1	1	-
f_{25}	f_{mean}	**2.9960e+03**†	3.1126e+03‡	3.0131e+03
	std	**9.6000e+01**†	3.7122e+02‡	2.7170e+02
	p	4.3953e-04	4.3953e-04	-
	h	1	1	-
f_{26}	f_{mean}	3.1563e+03‡	3.1729e+03‡	**3.0902e+03**
	std	4.7452e+01‡	4.2231e+01‡	**1.4199e+00**
	p	2.9486e-11	2.9413e-11	-
	h	1	1	-
f_{27}	f_{mean}	3.5139e+03‡	3.5171e+03‡	**3.3272e+03**
	std	2.3893e+02‡	2.8614e+02‡	**1.6981e+02**
	p	4.3531e-03	7.1971e-03	-
	h	1	1	-
f_{28}	f_{mean}	3.2730e+03‡	3.2630e+03‡	**3.1539e+03**
	std	7.1359e+01‡	6.7949e+01‡	**2.7673e+01**
	p	1.2870e-09	1.2870e-09	-
	h	1	1	-
f_{29}	f_{mean}	2.5894e+06‡	2.5661e+06‡	**3.8183e+05**
	std	1.8642e+06‡	4.1742e+06‡	**5.7220e+05**
	p	9.0595e-08	5.5979e-07	-
	h	1	1	-
	‡	9	9	-
	†	1	0	-
	≀	0	1	-

$$
\begin{aligned}
& Min.f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 \\
& +19.84x_1^2x_3 \\
& s.t. \quad g_1(x) = 0.0193x_3 - x_1 \leqslant 0 \\
& \qquad g_2(x) = 0.00954x_3 - x_2 \leqslant 0 \\
& \qquad g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leqslant 0 \\
& \qquad g_4(x) = x_4 - 240 \leqslant 0 \\
& \qquad\qquad 0 \leqslant x_1, x_2 \leqslant 100,\ 10 \leqslant x_2, x_4 \leqslant 200
\end{aligned} \tag{5}
$$

Table 10 clearly testifies that the proposed algorithm delivers better results under the same run circumstance and termination criterion, and the superiority is statistically significant with respective to other competitors considering the obtained f_{mean} and standard deviation results. That means BSO is able to generate

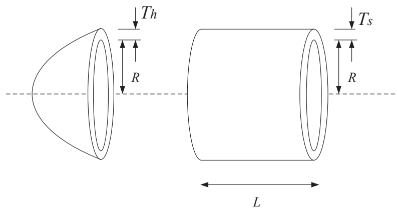

Figure 8: The schematic of the pressure vessel design problem

a set of parameter combinations such that the optimization objective is optimal.

5.3. case 3

The main target of three-bar truss design problem aims to optimize the weight considering stress, deflection, and buckling constraints, which are expressed by seven different constraint conditions, two decision variables(x_1 and x_2) and some control parameters. Eq.(6) and Fig.9 provide the mathematical model and architecture graph, respectively.

$$
\begin{aligned}
& Min.f(x) = (2\sqrt{2}x_1 + x_2) \times l \\
& s.t. \quad g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leqslant 0 \\
& \qquad g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leqslant 0 \\
& \qquad g_3(x) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \leqslant 0
\end{aligned} \tag{6}
$$

where, $l = 100cm$, $P = 2kN/cm^2$,$\sigma = 2kN/cm^2$, $0 \leqslant x_1 \leqslant 1$, $0 \leqslant x_2 \leqslant 1$.

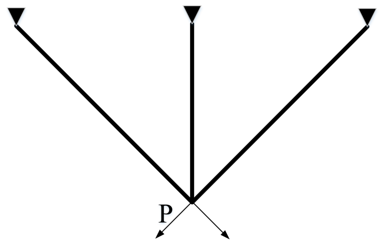

Figure 9: The schematic of the three-bar truss design problem

According to the experimental results listed in Table 11, although the standard deviation value of BSO is weaker than that of DEDVR, BSO, CMAES and DEDVR obtain the best results, and sufficiently outperform other optimization algorithms with the same termination criterion and run circumstance. Such evidence indicates that the proposed algorithm will be an attractive alternative optimizer for generating satisfactory results on challenging optimization problems in future.

6. Conclusion

In this paper, inspired by the Oryx escape and survival phenomenon in nature, a new heuristic optimization technique named biological survival optimizer is proposed. The simplex method and several stochastic operators are introduced for exploring and exploiting the feasible region effectively. The sensitivity analysis of the proposed algorithm is also discussed from the view of principle. To evaluate the performance of BSO, a recent test suite of CEC2017 benchmark functions with different characteristics and three different engineering design problems are utilized. Experimental results compared with eight existing optimization algorithms demonstrate that BSO has better performance than the other algorithms on most cases, showing the proposed algorithm has a good tracking ability. Besides that, the sensitivity analysis on the role of each component in the proposed algorithm is also analysed and discussed extensively. In future work, BSO will be further improved or modified as tool for addressing diverse practical applications in real world.

Acknowledgment

The authors express sincerely appreciation to the anonymous reviewers for their helpful opinions. This work is supported by the National Natural Science Foundation of China under Grants 62006103 and 61872168, in part by the Jiangsu national science research of high education under Grand 20KJB110021.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

All data/figure files are available after acceptance of the manuscript for publication.

Reference

[1] X. S. Yang, "Nature-inspired metaheuristic algorithms," in *Luniver press,* 2nd ed., 2010.

[2] D. H. Wolpert, and W. G. Macready, " No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[3] H. Akcan, "A genetic algorithm based solution to the Minimum-Cost Bounded-Error Calibration Tree problem," *Appl. Soft. Comput.*, vol. 73, pp. 83–95, 2018.

[4] M. Sami, N. Nejah, V. D. B. Adrien, "V. Thierry. Improved Many-objective optimization algorithms for the 3D indoor deployment problem," *Arab. J. Sci. Eng.*, vol. 44, pp. 3883–3904, 2019.

[5] A. H. Ali, W. Ejaz, and M. A. Taei, et al. "Solving MAX-SAT Problem by Binary Biogeograph-based Optimization Algorithm," *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2019.

[6] B. Rafal, " Handling bound constraints in CMA-ES: An experimental study" , *Swarm. Evol. Comput.*, vol. 52, 100627, 2020.

[7] C. Pinar, B. Erkan. "Bernstain-search differential evolution algorithm for numerical function optimization," *Expert. Syst. Appl.*, vol. 138, 112831, 2019.

[8] G. Ricardo, J. L. Luis, A. L. Julio. "A Memetic Chaotic Gravitational Search Algorithm for unconstrained global optimization problems," *Appl. Soft. Comput.*, vol. 79, pp. 14–29, 2019.

[9] P. Doddy, M. Y. Cheng, Y. W. Wu, et al. "Differential Big Bang - Big Crunch algorithm for construction-engineering design optimization," *Utomat. Constra.*, vol. 85, pp. 290–304, 2018.

[10] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimzer: a nature-inspired algorithm for global optimization," *Neural. comput. appl.*, vol. 27, no. 2, pp. 495–513, 2016.

[11] Q. Jiang, L. Wang, and Y. Lin, et al."An efficient multi-objective artificial raindrop algorithm and its application to dynamic optimization problems in chemical processes," *Appl. Soft. Comput.*, vol. 58, pp. 354–377, 2017.

[12] S. M. E. Abd, E. S. Ali. "Optimal locations and sizing of capacitors in radial distribution systems using mine blast algorithm," *Electr. Eng.*, vol. 100, pp. 1–9, 2016.

[13] J. K. Li, S. B. Song, Y. Tang, et al. "Approximate logic neuron model trained by states of matter search algorithm," *Knowl-based. syst.*, vol. 163, pp. 120–130, 2019.

[14] B. Amin, V. Vahid, S. Miadeza, P. S. Joao. "Multiobjective ray optimization algorithm as a solution strategy for solving non-convex problems: A power generation scheduling case study," *Elect. Power. Energy Syst.*, vol. 119, 105967, 2020.

[15] M. lsiet, G. Mohamed. "Self-adapting control parameters in particle swarm optimization," *Appl. Soft. Comput.*, vol. 83, 105653, 2019.

[16] M. Sami, N. Nejah, V. D. B. Adrien, V. Thierry. "A new multi-agent particle swarm algorithm based on birds accents for the 3D indoor deployment problem," *Isa. Trans.*, vol. 91, pp. 262–280, 2019.

[17] Q. Liu, L. Wu, et al, "A novel hybrid bat algorithm for solving continuous optimization problems," *Appl. Soft. Comput.*, vol. 73, pp. 67–82, 2018.

[18] C. Lu, L. Gao. J. Yi. "Grey wolf optimizer with cellular topological structure," in *Expert. Syst. Appl.*., vol. 107, pp. 89–114, 2018.

[19] W. Yamany, M. Fawzy, and A. Tharwat, et al, "Moth-flame optimization for training multi-layer perceptrons," in *Proc. ICENCO,* Cairo, Egypt, 2015, pp. 267–272.

[20] A. S. Joshi, K. Omkar, G. M. Kakandikar, V. M. Nandedkar. "Cuckoo Search Optimization- A Review," *Mater. today.*, vol. 4, pp. 7262–7269, Nov. 2017.

[21] A. L. Hafez, H. M. Zawbaa, E. Emary, and A. E. Hassanien, "Sine cosine optimization algorithm for feature selection," in *Proc. INISTA,* Sinaia, Romania, pp. 223–233, 2016.

[22] S. Mirjalili, and A. Lewis, "The Whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, 2016.

[23] IUCN SSC Antelope Specialist Group, " Oryx leucoryx," *IUCN Red List of Threatened Species. Version 2008.*

[24] O. Gilad, W. E. Grant, and S. David, " Simulated dynamics of Arabian Oryx (Oryx leucoryx) in the Israeli Negev: Effects of migration corridors and post-reintroduction changes in natality on population viability" , *Ecolo. Model.*, vol. 210, no. 1-2, pp. 169–178, Jan. 2008.

[25] M. Zafar-ul Islam, P.M. Basheer, W. Rahman,and A. Boug, " An attack by ratel mellivora capensis on pre-release asian Houbara bustards chlamydotis macqueenii in central saudi arabia" , *Small Carni. Conse.*, vol. 44, pp. 35–37, 2011.

[26] C. Stephen, T. Arshad, S. Ahmed, et al, "Incidental findings of Cysticercus tenuicollis metacestodes in five oryx species," in *Asian. Pac. J. Trop. Bio.*., vol. 6, pp. 90–92, 2016.

[27] F. Mads, M. Osma, W. Tobias, et al, "The hairy lizard: heterothermia affects anaesthetic requirements in the Arabian oryx ," *Vet. Anaesth. analg.*, vol. 44, pp. 899–904, 2017.

[28] D. Braha, " Global civil unrest: contagion, self-organization, and prediction" , *PLoSONE.*, vol. 7, no. 10, pp. 1–9, 2012.

[29] M. Ballerini, N. Cabibbo, R. M. Candelier, and V. Zdravkovic, "Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study.," in *Proc. Natl. Acad. Sci,* USA, 2008, pp. 1232–1237.

[30] E. Abdulhakeem, A. Mohammad, et al, "CIDR estrous synchronization in the Arabian Oryx" , *Theriogenology.*, vol. 132, pp. 113–117, 2019.

[31] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Cof. Evolu. Comput.,* Honolulu, USA, 2002, pp. 1671–1676.

[32] Y. Q. Zhou, Y. X. Zhou, Q. F. Luo, and M. Abdel-Basset, " A simplex method-based social spider optimization algorithm for clustering analysis," , *Eng. Appl. Artif. Intel.*, vol. 64,pp. 67–82, 2017.

[33] N. H. Awad, M. Z. Ali, P. N. Suganthan, J. J. Liang and B. Y. Qu, , "Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization," Nanyang Technological University, Jordan University of Science and Technology and Zhengzhou University, Tech. Rep., 2016.

[34] Y. Wang, Z.X. Cai, and Q.F Zhang, " Differential evolution with composite trial vector generation strategies and control parameters" , *IEEE Trans. Evol. Comput*., vol. 15, no. 1, pp. 55–66, 2011.

[35] A. Ghosh, S. Das, A. Kr. Das, and L. Gao, "Reusing the Past Difference Vectors in Differential EvolutionłA Simple But Significant Improvement" , *IEEE T. Cybernetics*, vol. 50, no. 11, pp. 4821–4834, 2020.

[36] S. Gupta and K Deep, and A. Engelbrecht, "A memory guided sine cosine algorithm for global optimization" , *ENG. Appl. Artif. Intel.*, vol. 93, 103718, 2020.

[37] S. Gupta and K Deep, "A memory-based Grey Wolf Optimizer for global optimization tasks" , *Appl. Soft. Comput.*, vol. 93, 106367, 2020.

[38] H. Ren, J. Li, H. l. Chen, C. Y. Li, "Adaptive levy-assisted salp swarm algorithm: Analysis and optimization case studies" , *Math. Comput. Simulat*., vol. 181, pp. 380–409, 2021.

[39] S. Gupta and K Deep, "Enhanced leadership-inspired grey wolf optimizer for global optimization problems" , *Eng. Comput.*, vol. 36, pp. 1777–1800, 2020.

[40] C. Yu, Z. Cai, X. Ye, M. Wang, X. Zhao, G. Liang, H. Chen, and C. Li, "Quantum-like mutation-induced dragonfly-inspired optimization approach" , *Math. Comput. Simulat*, vol. 178, pp. 259–289, 2020.

[41] W. Zhou, P. Wang, A. Heidari, M. Wang, X. Zhao and H. Chen, "Multi-core sine cosine optimization: Methods and inclusive analysis" , *Expert. Syst. Appl.*, vol. 164, 113974, 2021.

[42] N. Hansen, A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies" , *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, 2001.

[43] Q. Zhang, R. Wang, J. Yang, et al. "Biology migration algorithm: a new nature-inspired heuristic methodology for global optimization" , *Soft. Comput*., vol. 23, pp. 7333–7358, 2019.

[44] S. Mirjalili, "The Ant Lion Optimizer" , *Adv. Eng. Softw*., vol. 83, pp. 80–98, 2015.

[45] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm" , *Comput struct*., vol. 169, pp. 1–12, 2016.