Engineering with Computers (2021) 37:323–353
https://doi.org/10.1007/s00366-019-00826-w

ORIGINAL ARTICLE

ESA: a hybrid bio-inspired metaheuristic optimization approach for engineering problems

Gaurav Dhiman[1,2]

Received: 27 March 2019 / Accepted: 10 July 2019 / Published online: 19 July 2019

Abstract
In this paper, a hybrid bio-inspired metaheuristic optimization approach namely emperor penguin and salp swarm algorithm (ESA) is proposed. This algorithm imitates the huddling and swarm behaviors of emperor penguin optimizer and salp swarm algorithm, respectively. The efficiency of the proposed ESA is evaluated using scalability analysis, convergence analysis, sensitivity analysis, and ANOVA test analysis on 53 benchmark test functions including classical and IEEE CEC-2017. The effectiveness of ESA is compared with well-known metaheuristics in terms of the optimal solution. The proposed ESA is also applied on six constrained and one unconstrained engineering problems to evaluate its robustness. The results reveal that ESA offers optimal solutions as compared to the other competitor algorithms.

Keywords Metaheuristics · Optimization · Emperor penguin optimizer · Salp swarm algorithm · Engineering problems

1 Introduction

During the last few decades, various algorithms have been proposed to solve a variety of engineering optimization problems [1–21]. These optimization problems are very complex in nature because they have more than one local optimum solution. These problems are categorized into various categories whether they are constrained or unconstrained, discrete or continuous, static or dynamic, single or multi-objective.

In order to increase the efficiency and accuracy of these problems [22–26], researchers have encouraged to rely on metaheuristic algorithms [27–29]. Metaheuristics become more popular in various field because they do not require gradient information and bypass the local optima problem.

Metaheuristics are classified into two main categories: single-solution and multiple-solution. In single-solution-based algorithms, the searching process starts with one candidate solution, whereas in multiple-solution-based algorithm, the optimization performs using a set of solutions (i.e., population). Multiple-solution or population-based metaheuristics have advantages over single-solution-based metaheuristics. These are as follows:

- The searching process starts with random generated population, i.e, a set of multiple solutions.
- The multiple solutions can share the information between each other around the search space and avoid local optimal solutions.
- The exploration capability of multiple-solution or population-based metaheuristics is better than the single-solution-based metaheuristics.

The key phases of metaheuristic algorithms are exploration and exploitation. The exploration phase ensures that algorithm investigates the different promising regions in a given search space, whereas exploitation ensures the searching of optimal solutions around the promising regions. However, it is difficult to balance between these phases due to its stochastic nature. Therefore, the fine-tuning of these two phases is required to achieve the near-optimal solutions.

In recent years, a large number of metaheuristic algorithms have been developed. However, there is no single algorithm present which can solve all types of optimization problems. Some algorithms provide better optimal results as compared to the others. Therefore, developing a new metaheuristic algorithm is an open problem. This is the one

✉ Gaurav Dhiman
gaurav.dhiman@thapar.edu

[1] Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala, Punjab 147004, India

[2] Department of Computer Science, Government Bikram College of Commerce, Patiala, Punjab 147004, India

fact which can motivates us to develop a novel metaheuristic algorithm for solving optimization problems.

This paper presents a hybrid bio-inspired metaheuristic algorithm named as emperor penguin and salp swarm algorithm (ESA). It is inspired by the huddling and swarm behavior of emperor penguin optimizer (EPO) [30] and salp swarm algorithm (SSA) [31], respectively. The main contributions of this work are as follows:

- A hybrid bio-inspired swarm algorithm (ESA) is proposed.
- The proposed ESA is implemented and tested on 53 benchmark test functions (i.e., classical and CEC-2017).
- The performance of ESA is compared with well-known metaheuristics using sensitivity analysis, convergence analysis, and ANOVA test analysis.
- The robustness of proposed ESA and other metaheuristics are examined for solving engineering problems.

The rest of this paper is structured as follows: Sect. 2 presents the background and related works of optimization problems. The proposed ESA algorithm is discussed in Sect. 3. The experimental results and discussion is presented in Sect. 4. Section 5 focuses on the applications of ESA in engineering problems. Finally, the conclusion and some future research directions are given in Sect. 6.

2 Background and related works

This section firstly describes the recently developed EPO and SSA algorithms followed by related works in the field of optimization.

2.1 Emperor penguin optimizer (EPO)

Emperor penguins are social animals that perform various activities for living like hunting, foraging in groups. Emperor penguins perform huddling during extreme winters in the Antarctic to survive. Each penguin contributes equally while huddling depicting the sense of collectiveness and unity in their social behavior [32]. The huddling behavior can be summarized as below [30]:

- Create and discover huddling boundary.
- Compute the temperature around the huddle.
- Calculate the distance between each penguin.
- Effective mover is relocated.

2.1.1 Mathematical modeling

The main objective of modeling is to identify effective mover. L-shape polygon plane is considered as the shape of the huddle. After the effective mover is identified, the boundary of the huddle is again computed.

2.1.1.1 Generate and determine the huddle boundary To map the huddling behavior of emperor penguins, the first thing we need to consider is their polygon-shaped grid boundary. Every penguin is surrounded by at least two penguins while huddling. The huddling boundary is decided by the direction and speed of wind flow. Wind flow is generally faster as compared to penguins movement. Mathematically huddling boundary can be formulated as: let η represents the velocity of wind and χ represents the gradient of η:

$$\chi = \nabla\eta. \tag{1}$$

Vector α is integrated with η to obtain complex potential:

$$G = \eta + i\alpha, \tag{2}$$

where i represents the imaginary constant and G defines the polygon plane function.

2.1.1.2 Temperature profile around the huddle Emperor penguins perform huddling to conserve their energy and maximize huddle temperature $T = 0$ if $X > 0.5$ and $T = 1$ if $X < 0.5$, where X is the polygon radius. This temperature measure helps to perform exploration and exploitation task among emperor penguins. The temperature is computed as:

$$T' = \left(T - \frac{\text{Max}_{\text{itr}}}{y - \text{Max}_{\text{itr}}}\right)$$
$$T = \begin{cases} 0, & \text{if } X > 0.5 \\ 1, & \text{if } X < 0.5, \end{cases} \tag{3}$$

where y represents the current iteration, defines the current iteration, Max_{itr} represents the maximum count of iterations, X is the radius, and T is the time require to identify best optimal solution.

2.1.1.3 Distance between emperor penguins After the huddling boundary is computed, distance between the emperor penguin is calculated. The current optimal solution is the solution with higher fitness value than previous optimum solution. The search agents update their positions corresponding to current optimal solution. The position updation can be mathematically represented as:

$$\vec{M}_{\text{ep}} = \text{Abs}\left(N(\vec{A}) \cdot \vec{Q}(x) - \vec{A} \cdot \vec{Q_{\text{ep}}}(x)\right), \tag{4}$$

where $\vec{M}_{\text{ep}}$ denotes the distance between the emperor penguin and best fittest search agent (i.e., with less fitness value), x represents the ongoing iteration. $\vec{X}$ and $\vec{A}$ help to avoid collision among penguin. $\vec{Q}$ represents the best optimal solution (i.e., fittest emperor penguin), $\vec{Q_{\text{ep}}}$ represents the position vector of emperor penguin. $N()$ denotes the social

forces that helps to identify best optimal solution. The vectors $\vec{X}$ and $\vec{A}$ are calculated as follows:

$$\vec{X} = (M \times (T' + R_{\text{grid}}(\text{Accuracy})) \times \text{Rand}()) - T' \tag{5}$$

$$R_{\text{grid}}(\text{Accuracy}) = \text{Abs}(\vec{Q} - \vec{Q}_{\text{ep}}) \tag{6}$$

$$\vec{C} = \text{Rand}(), \tag{7}$$

where M is the movement parameter that maintains a gap between search agents for collision avoidance. The value of parameter M is set to 2. T' is the temperature profile around the huddle, $P_{\text{grid}}(\text{Accuracy})$ defines the polygon grid accuracy by comparing the difference between emperor penguins, and Rand() is a random function lies in the range of [0, 1].

The function $S()$ is calculated as follows:

$$N(\vec{A}) = \left(\sqrt{f \cdot \mathrm{e}^{-x/l} - \mathrm{e}^{-x}}\right)^2, \tag{8}$$

where e defines the expression function. f and l are control parameters for better exploration and exploitation. The values of f and l lie in the range of [2, 3] and [1.5, 2], respectively. Note that it has been observed that EPO algorithm provides better results between these ranges.

2.1.1.4 Relocate the mover The best obtained optimal solution (mover) is used to update the position of emperor penguins. The selected moves lead to the movement of other search agents in a search space. To find next position of a emperor penguin, following equations are used:

$$\vec{Q}_{\text{ep}}(x+1) = \vec{Q}(x) - \vec{X} \cdot \vec{M}_{\text{ep}}, \tag{9}$$

where $\vec{Q}_{\text{ep}}(x+1)$ denotes the updated position of emperor penguin.

2.2 Salp swarm algorithm (SSA)

Salp swarm algorithm is a metaheuristic bio-inspired optimization algorithm developed by Mirjalili et al. [31]. This algorithm is based on the swarming behavior of salps when navigating and foraging in the deep sea. This swarming behavior is mathematically modeled named as salp chain. This chain is divided into two groups: leader and followers. The leader leads the whole chain from the front while the followers follow each other. The updated position of the leader in a n-dimensional search environment is described as follows:

$$x_i^1 = \begin{cases} F_i + c_1((ub_i - lb_i)c_2 + lb_i), & c_3 \geq 1 \\ F_i - c_1((ub_i - lb_i)c_2 + lb_i), & c_3 < 1, \end{cases} \tag{10}$$

where x_i^1 represents the first position of salp, i.e., leader in the ith dimension, F_i is the position of food source, ub_i and lb_i are the lower bound and upper bound of ith dimension, respectively. However, c_1, c_2, and c_3 are random numbers.

The coefficient c_1 is responsible for better exploration and exploitation which is defined as follows:

$$c_1 = 2\mathrm{e}^{-\left(\frac{4l}{L}\right)^2}, \tag{11}$$

where l represents the current iteration and L is the maximum number of iterations; whereas, the parameters c_2 and c_3 are random numbers in range [0, 1].

To update the position of followers, the following equations are defined as follows:

$$x_i^j = \frac{1}{2}AT^2 + V_0T, \quad j \geq 2, \tag{12}$$

where x_i^j shows the position of follower, T represents the time and V_0 represents the initial speed. The parameter A is calculated as follows:

$$\begin{aligned} A &= \frac{V_{\text{final}}}{V_0} \\ V &= \frac{x - x_0}{T}. \end{aligned} \tag{13}$$

Considering $V_0 = 0$, the following equation can be expressed as:

$$x_i^j = \frac{1}{2}(x_i^j + x_i^{j-1}). \tag{14}$$

The SSA algorithm is able to solve high-dimensional problems using low computational efforts.

2.3 Related works

Multiple-solution-based metaheuristic algorithms are further classified into three categories such as evolutionary-based, physics-based, and swarm-based algorithms (see Fig. 1). The former one is generic population-based metaheuristic which is inspired from biological evolution, i.e., mutation, recombination, and selection. These do not make any assumptions about fitness landscape. The most popular evolutionary algorithm is genetic algorithm (GA) [33]. The evolution starts with randomly generated individuals from the given population. The fitness of each individual is computed in

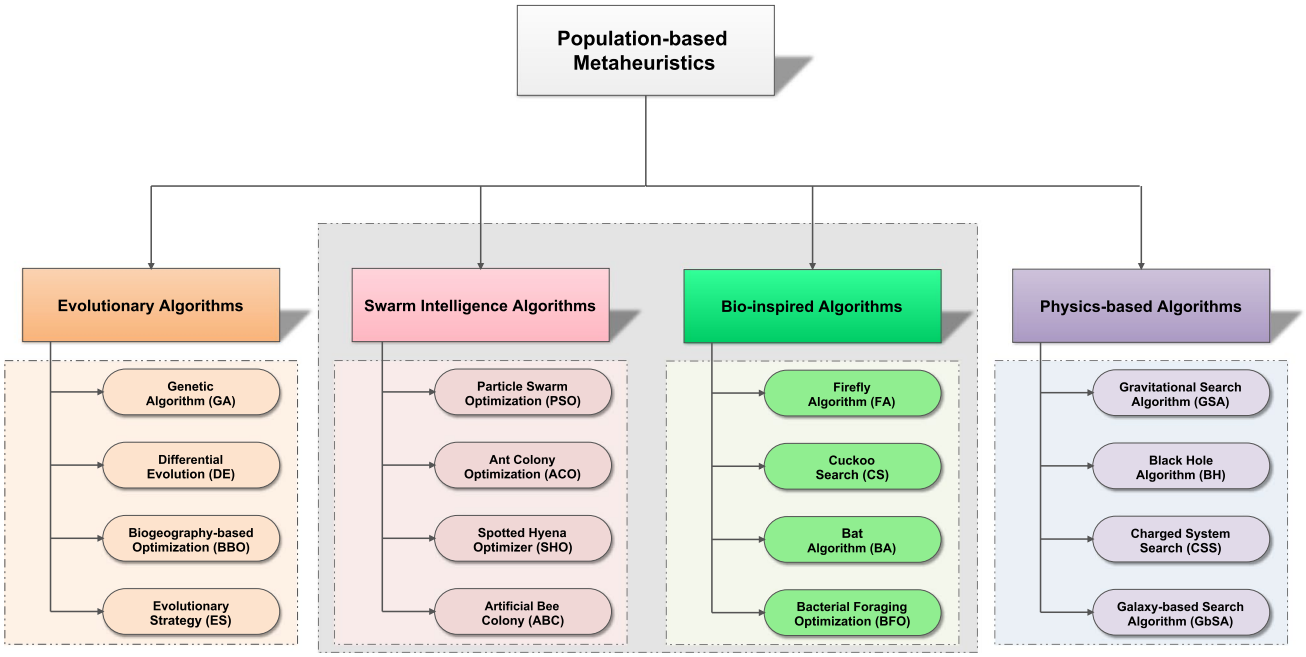

Fig. 1 Classification of population-based metaheuristic algorithms

each generation. The crossover and mutation operators are applied on individual to create a new population. The best individuals can generate a new population during the course of iterations. However, compared to other stochastic methods, genetic algorithm has advantage that it can be parallelized with little effort and not necessarily remain trapped in a sub-optimal local maximum or minimum of the target function. GA may provides local minima of a function that can steer the search in the wrong direction for some of the optimization problems. differential evolution (DE) [34] is another evolutionary-based metaheuristic algorithm that optimizes a problem by maintaining a candidate solutions and creates new candidate solutions by combining the existing ones. It can keep the candidate solution which has best fitness value for optimization problem. It has an ability to handle non-differentiable and non-linear cost functions. There are only few parameters to steer the minimization problem. The parameter tuning is a main challenge in DE because same parameters may not guarantee the global optimum solution. Apart from these, some of the other popular evolutionary-based algorithms are genetic programming (GP) [35], evolution strategy (ES) [36], and biogeography-based optimizer (BBO) [37].

The second category is physics-based algorithms in which each search agent can move throughout the search space according to physics rules such as gravitational force, electromagnetic force, inertia force, and many more. The well-known physics-based metaheuristic algorithms are simulated annealing (SA) [38] and gravitational search algorithm (GSA) [39]. Simulated annealing is inspired from annealing in metallurgy that involves heating and controlled cooling attributes of a material. These attributes depend on its thermodynamic free energy. SA is advantageous in terms to deal with non-linear models and noisy data. The main advantage of SA over other search methods is its ability to search the global optimal solution. However, it suffers from high computational time especially if the fitness function is very complex and non-linear in nature. Gravitational search algorithm is based on the law of gravity and mass interactions. The population solutions are interact with each other through the gravity force and their performance is measured by its mass. GSA requires only two input parameters to adjust, i.e., mass and velocity. It is easy to implement. The ability to find near the global optimum solution makes GSA differ from the other optimization algorithms. However, it suffers from computational time and convergence problem if the initial population is not generated well. Some of the other popular algorithms are: big-bang big-crunch (BBBC) [40], charged system search (CSS) [41], black hole (BH) [42] algorithm, central force optimization (CFO) [43], small-world optimization algorithm (SWOA) [44], artificial chemical reaction optimization algorithm (ACROA) [45], ray optimization (RO) algorithm [46], galaxy-based search algorithm (GbSA) [47], and curved space optimization (CSO) [48].

The third category is swarm-based algorithms which are inspired by the collective behavior of social creatures. This collective intelligence is based on the interaction of swarm with each other. These are easier to implement than the evolutionary-based algorithms due to number of operators (i.e., selection, crossover, and mutation).

The most popular algorithm is particle swarm optimization (PSO) which was proposed by Kennedy and Eberhart [49]. In PSO, particles move around the search space using the combination of best solutions [50]. The whole process is repeated until the termination criterion is satisfied. The main advantage of PSO is that it has no overlapping and mutation computation. During simulation, the most optimist particle can transmit information among the other particles. However, it suffers from the stagnation problem.

Ant colony optimization (ACO) is another popular swarm intelligence algorithm which was proposed by Dorigo [51]. The main inspiration behind this algorithm is the social behavior of ants in ant colony. The social intelligence of ants is to find the shortest path between the source food and nest. ACO is able to solve the travelling salesman and similar problems in an efficient way that can be advantageous of ACO over the other approaches. The theoretical analysis of a problem is very difficult using ACO because the computational cost is high during convergence.

Bat-inspired algorithm (BA) [52] is inspired by the echolocation behavior of bats. Another well-known swarm-based metaheuristic is artificial bee colony (ABC) algorithm [53] which is inspired by the collective behavior of bees to find the food sources. Spotted hyena optimizer (SHO) [16] is a bio-inspired metaheuristic algorithm that mimics the searching, hunting, and attacking behaviors of spotted hyenas in nature. The main concept behind this technique is the social relationship and collective behavior of spotted hyenas for hunting strategy. Cuckoo search (CS) [54] is inspired by the obligate brood parasitism of cuckoo species. These species lay their eggs in the nest of other species. Each egg and a cuckoo egg represent a solution and a new solution, respectively.

Emperor penguin optimizer (EPO) [30] is a recently developed bio-inspired metaheuristic algorithm that mimics the huddling behaviors of emperor penguins. The main steps of EPO are to generate huddle boundary, compute temperature around the huddle, calculate the distance, and find the effective mover.

Grey wolf optimizer (GWO) [55] is a very popular bio-inspired based algorithm for solving real-life constrained problems. Grey wolf optimizer (GWO) is inspired by the behaviors of grey wolves. It mimics the leadership, hierarchy, and hunting mechanisms of grey wolves. GWO employs four types of grey wolves namely, alpha, beta, delta, and omega for optimization problems. The hunting, searching, encircling, and attacking mechanisms are also implemented. Further, to investigate the performance of GWO algorithm, it was tested on well-known test functions and classical engineering design problems.

Multi-verse optimizer (MVO) is a promising optimization algorithm proposed by Mirjalili et al. [56]. It is inspired by the theory of multi-verse in physics which consists of three main concepts, i.e., white hole, black hole, and worm hole. The concepts of white hole and black hole are appropriate for exploration and worm hole helps in the exploitation of the given search spaces.

Sine cosine algorithm (SCA) is proposed by Mirjalili [57] for solving numerical optimization problems. SCA generates multiple random solutions and fluctuate them towards the best optimal solution using mathematical models such as sine and cosine functions. The convergence speed of SCA is very high which is helpful for local optima avoidance.

The other well-known metaheuristic algorithms are fireworks algorithm (FWA) [58–61], monkey search [62], Bacterial foraging optimization algorithm [63], firefly algorithm (FA) [64], fruit fly optimization algorithm (FOA) [65], golden section line search algorithm [66], Fibonacci search method [67], bird mating optimizer (BMO) [68], Krill Herd (KH) [69], artificial fish-swarm algorithm (AFSA) [70], Dolphin partner optimization (DPO) [71], bee collecting pollen algorithm (BCPA) [72], and hunting search (HS) [73].

3 Proposed algorithm

In this section, the motivation and brief justification of the proposed algorithm are described in detail.

3.1 Motivation

Nature has inspired many researchers in many ways and thus is a rich source of inspiration. Nowadays, most new algorithms are nature-inspired because they have been developed by drawing inspiration from nature. The main source of inspiration for developing new algorithms is nature. Almost all new algorithms can be referred as nature-inspired algorithms. The majority of nature-inspired algorithms are based on some characteristics of biological system. Therefore, the largest fraction of nature-inspired algorithms are biology or bio-inspired. Among bio-inspired algorithms, a special class of algorithms have been developed by drawing inspiration from swarm intelligence. It has been observed from the literature that multiple-solution or population-based swarm

intelligence algorithms are able to solve real-life optimization problems. They are able to explore throughout the search space, and exploit the global optimum. However, population-based techniques are more reliable than single-solution-based techniques because of more function evaluations.

According to no free lunch theorem [74], there is no optimization algorithm which is able to solve all optimization problems. This fact will attract the researchers of different fields to propose a new optimization algorithm. These motivate us to propose a new population-based metaheuristic algorithm.

The researchers have pointed out convergence and diversity difficulties for real-life problems. Hence, there is a need to develop an algorithm that maintains the convergence and diversity. In this paper, the navigation and foraging behaviors of SSA algorithm is used to maintain the diversity. The reasons to select these behaviors over others are:

1. SSA algorithm eliminates the problem of missing selection individuals.
2. The values of these behaviors are directly optimized, without any need for niching, that helps to maintain the diversity.
3. SSA ensures that any approximation set that has high-quality value for a particular problem contains all optimal solutions.

However, the calculation of SSA parameters requires high computational effort. To resolve this problem, EPO algorithm is employed. SSA suffers from overhead of maintaining the necessary information. For this, huddling behavior of EPO algorithm is used for maintaining the information. Therefore, a novel hybrid algorithm is proposed that utilizes the features of both EPO and SSA.

3.2 Hybrid emperor penguin and salp swarm algorithm (ESA)

The first step is to initialize the population and initial parameters of ESA algorithm as explained in Table 1. After the initialization, objective value of each search agent is calculated using *FITNESS* function as defined in line 4 of Algorithm 1. The best search agent is explored from the given search space. Further, the huddling behavior is defined using Eq. (9) until the suitable result is found for each search agent. In line 6 of Algorithm 1, position of each search agent is updated. Now, the leader and follower selection approaches

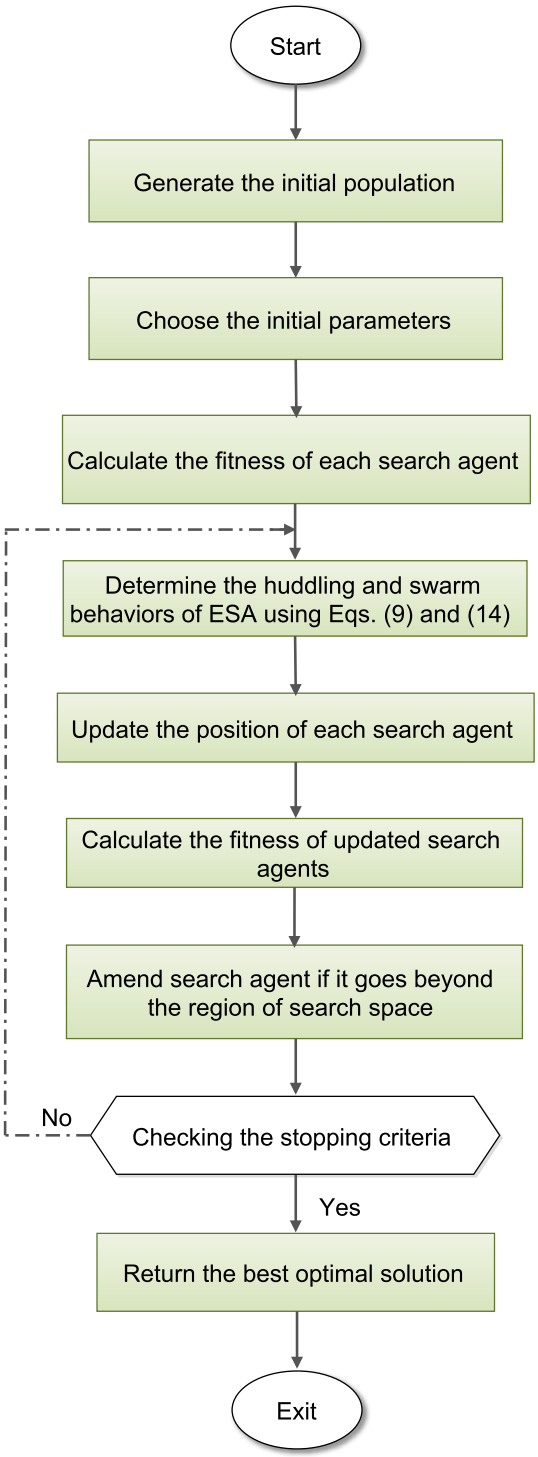

Fig. 2 Flowchart of the proposed ESA algorithm

Table 1 Parameter settings for algorithms

#	Algorithms	Parameters	Values
1.	Spotted hyena optimizer (SHO)	Search agents	80
		Control parameter ($\vec{h}$)	[5, 0]
		$\vec{M}$ constant	[0.5, 1]
		Number of generations	1000
2.	Grey wolf optimizer (GWO)	Search agents	80
		Control parameter ($\vec{a}$)	[2, 0]
		Number of generations	1000
3.	Particle swarm optimization (PSO)	Number of particles	50
		Inertia coefficient	0.75
		Cognitive and social coeff	1.8, 2
		Number of generations	1000
4.	Multi-verse optimizer (MVO)	Search agents	50
		Wormhole existence prob.	[0.2, 1]
		Travelling distance rate	[0.6, 1]
		Number of generations	1000
5.	Sine cosine algorithm (SCA)	Search agents	50
		Number of elites	2
		Number of generations	1000
6.	Gravitational search algorithm (GSA)	Search agents	30
		Gravitational constant	100
		Alpha coefficient	20
		Number of generations	1000
7.	Salp swarm algorithm (SSA)	Population size	80
		c_1, c_2, c_3	[0, 1]
		Number of generations	1000
8.	Emperor penguin optimizer (EPO)	Search agents	80
		Temperature profile (T')	[1, 1000]
		$\vec{A}$ constant	[-1.5, 1.5]
		Function $S()$	[0, 1.5]
		Parameter M	2
		Parameter f	[2, 3]
		Parameter l	[1.5, 2]
		Number of generations	1000

are applied to update the positions of search agents using Eq. (14).

Again, the objective value of each search agent is calculated to find the optimal solutions. The condition is checked whether any search agent goes beyond the boundary in a given search space and if it happens then adjust it. Calculate the updated search agent objective value and update the parameters if there is a better solution from the previous one. The algorithm will be stopped when the stopping criterion is satisfied. This criterion is defined by user for how long the algorithm will be run, i.e., maximum number of iterations. Finally, the optimal solution is returned, after the stopping criterion is satisfied (see Fig. 2).

The pseudo-code of ESA algorithm is shown in Algorithm 1. There are some interesting points about the proposed ESA algorithm which are given below:

- $N()$, A, and V assist the candidate solutions to behave more randomly in a search space and are responsible in avoiding conflicts between search agents.
- The convergence behaviors of common optimization algorithms suggest that the exploitation tends to increase the speed of convergence, while exploration tends to decrease the convergence rate of the algorithm. Therefore, the possibility of better exploration and exploitation is done by the adjusted values of $N()$, A, and V.
- The huddling and swarm behaviors of ESA in a search region defines the effectively collective behavior.

Algorithm 1 Hybrid emperor penguin and salp swarm algorithm

Input: Population $\vec{P_p}$
Output: Optimal fitness value
procedure ESA
Initialize the parameters $N(), c_1, c_2, c_3$, and Max_{itr}
  while $(x < Max_{itr})$ **do**
    FITNESS$(\vec{P_p})$ /* Compute the fitness of each search agent using **FITNESS** function*/
    for j ← 1 to n **do**
      Update the positions of search agents using Eq. (9)
    end for
    Apply the leader and follower selection approach to the updated positions of search agents using Eq. (14)
    FITNESS$(\vec{P_p})$ /* Again compute the fitness value of updated search agents using **FITNESS** function*/
    Amend search agent which goes beyond the region of search space
    $x \leftarrow x + 1$
  end while
return $\vec{P}$
end procedure

procedure FITNESS$(\vec{P_p})$
  for i ← 1 to n **do**
    $FIT[i] \leftarrow FITNESS\_FUNCTION(\vec{P_p})$
  end for
  $FIT_{best} \leftarrow$ **BEST**$(FIT[])$ /* Compute the best fitness value using **BEST** function */
return FIT_{best}
end procedure

procedure BEST$(FIT[])$
  $best \leftarrow FIT[0]$
  for i ← 1 to n **do**
    if$(FIT[i] < best)$ **then**
      $best \leftarrow FIT[i]$
    end if
  end for
return $best$
end procedure

3.3 Computational complexity

In this subsection, the computational complexity of proposed ESA algorithm is discussed. Both the time and space complexities of the proposed algorithm are given below.

3.3.1 Time complexity

1. Population initialization process requires $\mathcal{O}(n \times d)$ time, where n indicates the population size and d indicates the dimension of a given problem.
2. The fitness of each agent requires $\mathcal{O}(\text{Max}_{\text{itr}} \times n \times d)$ time, where Max_{itr} is the maximum number of iterations to simulate the proposed algorithm.
3. It requires $\mathcal{O}(N)$ time, where N defines the huddling and swarm behaviors of EPO and SSA for better exploration and exploitation.

Hence, the total time complexity of ESA algorithm is $\mathcal{O}(\text{Max}_{\text{itr}} \times n \times d \times N)$.

3.3.2 Space complexity

The space complexity of ESA algorithm is the maximum amount of space used at any one time which is considered during its initialization process. Thus, the total space complexity of ESA algorithm is $\mathcal{O}(n \times d)$.

4 Experimental results and discussion

This section describes the experimentation on 53 standard benchmark test functions to evaluate the performance of proposed algorithm. The detailed description of these benchmarks are presented below. Further, the results are compared with well-known metaheuristic algorithms.

4.1 Benchmark test functions

The 53 benchmark test functions are applied on the proposed algorithm to demonstrate its applicability and efficiency.

Table 2 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, and CEC-2017 benchmark test functions using different simulation runs (i.e., 100, 500, 800, and 1000)

Iterations	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	$C-1$	$C-15$
100	2.11E−19	7.03E+01	3.21E−04	8.07E−02	− 2.11E+01	3.40E+03	2.60E+07
500	5.44E−21	6.01E+01	1.20E−04	7.97E−02	− 2.97E+01	3.15E+03	2.56E+06
800	4.00E−26	5.52E+01	7.21E−05	6.00E−02	− 3.11E+01	2.94E+03	2.00E+06
1000	**1.40E−28**	**4.01E+00**	**3.10E−06**	**2.87E−02**	**− 3.47E+00**	**2.15E+02**	**1.43E+05**

The best-obtained results are in bold

Table 3 The obtained optimal values on unimodal, multimodal, fixed-dimension multimodal, and CEC-2017 benchmark test functions where the number of iterations is fixed as 1000

Search agents	Functions						
	F_1	F_5	F_{11}	F_{17}	F_{23}	$C-1$	$C-15$
30	1.51E−16	7.01E+00	3.11E−03	5.27E−01	− 2.53E+00	1.33E+02	2.86E+06
50	5.43E−18	5.31E+01	3.31E−03	8.10E−01	− 2.80E+00	4.21E+02	1.21E+05
80	**2.20E**−29	**5.00E+00**	**3.21E**−07	**4.12E**−03	**− 3.49E+00**	**1.23E+02**	**1.42E+04**
100	5.97E−23	5.42E+00	8.32E−04	6.18E−01	− 3.13E+00	2.96E+02	2.18E+05

The best-obtained results are in bold

The number of search agents is varied from 30 to 100

These functions are divided into six main categories: unimodal [75], multimodal [64], fixed-dimension multimodal [64, 75], and IEEE CEC-2017 [76] test functions. The descriptions of these test functions are given in "Appendix". In "Appendix", Dim and Range indicate the dimension of the function and boundary of the search space, respectively. $f_{\min}$ denotes the minimization function.

"Appendix" shows the characteristics of unimodal, multimodal, fixed-dimension multimodal, and CEC-2017 benchmark test functions. The seven test functions (F_1–F_7) are included in the first category of unimodal test functions. These functions have only one global optimum. The second category consists of six test functions (F_8–F_{13}) and third category includes ten test functions (F_{14}–F_{23}). There are multiple local solutions in these categories which are useful for examining the local optima problem. The fourth category consists of 30 CEC-2017 benchmark test functions ($C1$–$C30$).

4.2 Experimental setup

The proposed ESA is compared with well-known algorithms namely spotted hyena optimizer (SHO) [16], grey wolf optimizer (GWO) [55], particle swarm optimization (PSO) [49], multi-verse optimizer (MVO) [56], sine cosine algorithm (SCA) [57], gravitational search algorithm (GSA) [39], salp swarm algorithm (SSA) [31], emperor penguin optimizer (EPO) [30], and jSO [77]. The parameter values of these algorithms are set as they are recommended in their original papers. Table 1 shows the parameter settings of competitor algorithms. The experimentation has been done on Matlab R2014a (8.3.0.532) version using 64-bit Core i7 processor with 3.20 GHz and 8 GB main memory (Tables 2, 3).

4.3 Performance comparison

In order to demonstrate the effectiveness of the proposed algorithm, it is compared with well-known optimization algorithms on unimodal, multimodal, fixed-dimension multimodal, and CEC-2017 benchmark test functions. The average and standard deviation of the best optimal solution are mentioned in tables. For each benchmark test function, ESA algorithm utilizes 30 independent runs in which each run employs 1000 iterations.

4.3.1 Evaluation of test functions F_1–F_7

The unimodal test functions (F_1–F_7) are used to assess the exploitation capability of metaheuristic algorithm. Table 4 shows the mean and standard deviation of best optimal solution obtained from the above-mentioned algorithms on unimodal test functions. For F_1, F_2, and F_3 test functions, SHO is the best optimizer whereas ESA is the second best optimizer in terms of mean and standard deviation. ESA provides better results for F_4, F_5, F_6, and F_7 benchmark test functions. It is observed from results that ESA is very competitive as compared with other competitor algorithms and has better exploitation capability to find the best optimal solution very efficiently.

4.3.2 Evaluation of test functions F_8–F_{23}

Multimodal test functions have an ability to evaluate the exploration of an optimization algorithm. Tables 5 and 6 depict the performance of above-mentioned algorithms on multimodal test functions (F_8–F_{13}) and

Table 4 Mean and standard deviation of best optimal solution for 30 independent runs on unimodal benchmark test functions

F	ESA		SHO		GWO		PSO		MVO		SCA		GSA		SSA		EPO	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std
F_1	4.21E−29	6.30E−30	**0.00E+00**	**0.00E+00**	4.61E−23	7.37E−23	4.98E−09	1.40E−08	2.81E−01	1.11E−01	3.55E−02	1.06E−01	1.16E−16	6.10E−17	1.95E−12	2.01E−11	5.71E−28	8.31E−29
F_2	7.21E−40	1.02E−40	**0.00E+00**	**0.00E+00**	1.20E−34	1.30E−34	7.29E−04	1.84E−03	3.96E−01	1.41E−01	3.23E−05	8.57E−05	1.70E−01	9.29E−01	6.53E−18	5.10E−17	6.20E−40	3.32E−40
F_3	3.15E−20	4.10E−20	**0.00E+00**	**0.00E+00**	1.00E−14	4.10E−14	1.40E+01	7.13E+00	4.31E+01	8.97E+00	4.91E+03	3.89E+03	4.16E+02	1.56E+02	7.70E−10	7.36E−09	2.05E−19	9.17E−20
F_4	**1.31E**−20	**4.95E**−21	7.78E−12	8.96E−12	2.02E−14	2.43E−14	6.00E−01	1.72E−01	8.80E−01	2.50E−01	1.87E+01	8.21E+00	1.12E+00	9.89E−01	9.17E+01	5.67E+01	4.32E−18	3.98E−19
F_5	**5.03E+00**	**5.51E−02**	8.59E+00	5.53E−01	2.79E+01	1.84E+00	4.93E+01	3.89E+01	1.18E+02	1.43E+02	7.37E+02	1.98E+03	3.85E+01	3.47E+01	5.57E+02	4.16E+01	5.07E+00	4.90E−01
F_6	**5.11E**−20	**2.32E**−21	2.46E−01	1.78E−01	6.58E−01	3.38E−01	9.23E−09	1.78E−08	3.15E−01	9.98E−02	4.88E+00	9.75E−01	1.08E−16	4.00E−17	3.15E−01	9.98E−02	7.01E−19	4.39E−20
F_7	**3.70E**−06	**6.11E**−07	3.29E−05	2.43E−05	7.80E−04	3.85E−04	6.92E−02	2.87E−02	2.02E−02	7.43E−03	3.88E−02	5.79E−02	7.68E−01	2.77E+00	6.79E−04	3.29E−03	2.71E−05	9.26E−06

The best-obtained results are in bold

Table 5 Mean and standard deviation of best optimal solution for 30 independent runs on multimodal benchmark test functions

F	ESA		SHO		GWO		PSO		MVO		SCA		GSA		GA		EPO	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std
F_8	**− 8.92E+02**	5.85E+01	− 1.16E+02	**2.72E+01**	− 6.14E+02	9.32E+01	− 6.01E+02	1.30E+02	− 6.92E+02	9.19E+01	− 3.81E+02	2.83E+01	− 2.75E+02	5.72E+01	− 5.11E+02	4.37E+01	− 8.76E+02	5.92E+01
F_9	5.71E−02	5.37E−02	**0.00E+00**	**0.00E+00**	4.34E−01	1.66E+00	4.72E+01	1.03E+01	1.01E+02	1.89E+01	2.23E+01	3.25E+01	3.35E+01	1.19E+01	1.23E−01	4.11E+01	6.90E−01	4.81E−01
F_{10}	**6.01E−18**	1.50E−15	2.48E+00	1.41E+00	1.63E−14	**3.14E−15**	3.86E−02	2.11E−01	1.15E+00	7.87E−01	1.55E+01	8.11E+00	8.25E−09	1.90E−09	5.31E−11	1.11E−10	8.03E−16	2.74E−14
F_{11}	5.01E−06	6.32E−06	**0.00E+00**	**0.00E+00**	2.29E−03	5.24E−03	5.50E−03	7.39E−03	5.74E−01	1.12E−01	3.01E−01	2.89E−01	8.19E+00	3.70E+00	3.31E−06	4.23E−05	4.20E−05	4.73E−04
F_{12}	3.01E−05	3.70E−04	3.68E−02	1.15E−02	3.93E−02	2.42E−02	**1.05E−10**	**2.06E−10**	1.27E+00	1.02E+00	5.21E+01	2.47E+02	2.65E−01	3.14E−01	9.16E−08	4.88E−07	5.09E−03	3.75E−03
F_{13}	**0.00E+00**	**0.00E+00**	9.29E−01	9.52E−02	4.75E−01	2.38E−01	4.03E−03	5.39E−03	6.60E−02	4.33E−02	2.81E+02	8.63E+02	5.73E−32	8.95E−32	6.39E−02	4.49E−02	0.00E+00	0.00E+00

The best-obtained results are in bold

Table 6 Mean and standard deviation of best optimal solution for 30 independent runs on fixed-dimension multimodal benchmark test functions

F	ESA		SHO		GWO		PSO		MVO		SCA		GSA		SSA		EPO	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std
F_{14}	**1.04E+00**	3.10E−03	9.68E+00	3.29E+00	3.71E+00	3.86E+00	2.77E+00	2.32E+00	9.98E+01	**9.14E−12**	1.26E+00	6.86E−01	3.61E+00	2.96E+00	4.39E+00	4.41E−02	1.08E+00	4.11E−02
F_{15}	**7.11E**−04	**5.07E**−04	9.01E−03	1.06E−03	3.66E−02	7.60E−02	9.09E−03	2.38E−03	7.15E−02	1.26E−01	1.01E−02	3.75E−03	6.84E−02	7.37E−02	7.36E−02	2.39E−03	8.21E−03	4.09E−03
F_{16}	− 1.02E+00	9.80E−07	**− 1.03E+00**	**2.86E**−11	− 1.02E+00	7.02E−09	− 1.02E+00	0.00E+00	− 1.02E+00	4.74E−08	− 1.02E+00	3.23E−05	− 1.02E+00	0.00E+00	− 1.02E+00	4.19E−07	− 1.02E+00	9.80E−07
F_{17}	**3.97E**−01	4.30E−05	3.97E−01	2.46E−01	3.98E−01	7.00E−07	3.97E−01	9.03E−16	3.98E−01	1.15E−07	3.98E−01	7.61E−04	3.98E−01	1.13E−16	3.98E−01	**3.71E**−17	3.98E−01	5.39E−05
F_{18}	**3.00E+00**	**1.10E**−08	3.00E+00	9.05E+00	3.00E+00	7.16E−06	3.00E+00	6.59E−05	3.00E+00	1.48E+01	3.00E+00	2.25E−05	3.00E+00	3.24E−02	3.00E+00	6.33E−07	3.00E+00	1.15E−08
F_{19}	− **3.88E+00**	5.11E−08	− 3.71E+00	4.39E−01	− 3.84E+00	1.57E−03	− 3.80E+00	**3.37E**−15	− 3.77E+00	3.53E−07	− 3.75E+00	2.55E−03	− 3.86E+00	4.15E−01	− 3.81E+00	4.37E−10	− 3.86E+00	6.50E−07
F_{20}	−2.86E+00	5.17E−01	− 1.44E+00	5.47E−01	− 3.27E+00	7.27E−02	− **3.32E+00**	2.66E−01	− 3.23E+00	**5.37E**−02	− 2.84E+00	3.71E−01	− 1.47E+00	5.32E−01	− 2.39E+00	4.37E−01	− 2.81E+00	7.11E−01
F_{21}	− 7.05E+00	1.25E+00	− 2.08E+00	**3.80E**−01	− **9.65E+00**	1.54E+00	− 7.54E+00	2.77E+00	− 7.38E+00	2.91E+00	− 2.28E+00	1.80E+00	− 4.57E+00	1.30E+00	− 5.19E+00	2.34E+00	−8.07E+00	2.29E+00
F_{22}	− **12.71E+00**	4.16E−02	− 1.61E+00	**2.04E**−04	− 1.04E+00	2.73E−04	− 8.55E+00	3.08E+00	− 8.50E+00	3.02E+00	− 3.99E+00	1.99E+00	− 6.58E+00	2.64E+00	− 2.97E+00	1.37E−02	− 10.01E+00	3.97E−02
F_{23}	−3.52E+00	2.12E−03	−1.68E+00	2.64E−01	− **1.05E+01**	**1.81E**−04	− 9.19E+00	2.52E+00	− 8.41E+00	3.13E+00	− 4.49E+00	1.96E+00	− 9.37E+00	2.75E+00	− 3.10E+00	2.37E+00	− 3.41E+00	1.11E−02

The best-obtained results are in bold

fixed-dimension multimodal test functions (F_{14}–F_{23}), respectively. From these tables, it can be seen that ESA is able to find optimal solution for nine test problems (i.e., $F_8, F_{10}, F_{13}, F_{14}, F_{15}, F_{17}, F_{18}, F_{19}$, and F_{22}). For F_9, F_{11}, and F_{16} test functions, SHO provides better optimal results than ESA. ESA is the second best algorithm on these test functions. PSO attains best optimal solution for F_{12} and F_{20} test functions. For F_{21} test function, GWO and ESA are the first and second best optimization algorithms, respectively. The results reveal that ESA obtains competitive results in majority of the test problems and has better exploration capability.

4.3.3 Evaluation of IEEE CEC-2017 test functions (C_1–C_30)

This special session is devoted to the algorithms and techniques for solving real parameter single objective optimization without making use of the exact equations of the test functions. Table 7 shows the performance of proposed ESA algorithm and other competitive approaches on IEEE CEC-2017 test functions. The results reveal that ESA achieves best optimal solution for most of the CEC-2017 benchmark test functions (i.e., *C*-4, *C*-5, *C*-8, *C*-10, *C*-11, *C*-12, *C*-13, *C*-20, *C*-22, *C*-25, *C*-26, *C*-29, *C*-30).

4.4 Convergence analysis

Figure 3 shows the convergence curves of proposed ESA and other competitor algorithm. It is shown that ESA is very competitive over benchmark test functions. ESA has three different convergence behaviors. In the initial stage of iterations, ESA converges more quickly in the search space due to its adaptive mechanism. In the second step, ESA converges towards the optimum when final iteration reaches. The last step shows the express convergence from the initial step of iterations. The results reveal that ESA algorithm maintains a proper balance between exploration and exploitation to find the global optimum.

4.5 Sensitivity analysis

The proposed ESA algorithm uses four parameters namely, maximum number of iterations and number of search agents. The sensitivity investigation of these parameters has been discussed by varying their values and keeping other parameters fixed.

1. Maximum number of iterations: ESA algorithm was run for different number of iterations. The values of $\text{Max}_{\text{iteration}}$ used in experimentation are 100, 500, 800, and 1000. Table 2 shows the effect of iterations over benchmark test functions. The results reveal that ESA converges towards the optimum when the number of iterations is increased.

Table 7 Mean and standard deviation of best optimal solution for 30 independent runs on CEC-2017 benchmark test functions

F	ESA		SHO		GWO		PSO		MVO		SCA		GSA		SSA		jSO	
	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std	Ave	Std
C-1	1.60E+04	1.31E+07	2.38E+05	2.28E+07	2.12E+05	2.18E+07	4.47E+04	4.83E+06	1.57E+05	2.73E+07	6.16E+04	5.12E+06	7.75E+05	3.17E+07	3.30E+06	8.47E+07	**0.00E+00**	**0.00E+00**
C-2	6.80E+05	1.44E+09	3.23E+04	4.29E+06	5.75E+05	6.13E+07	9.51E+03	1.18E+05	1.07E+03	1.56E+05	1.53E+04	1.13E+05	7.43E+07	2.43E+09	4.68E+03	1.19E+04	**0.00E+00**	**0.00E+00**
C-3	3.30E+02	1.26E−04	3.30E+02	3.86E−03	3.30E+02	7.18E−03	3.30E+02	8.71E−03	3.30E+02	9.24E−03	3.30E+02	3.29E−03	3.30E+02	7.63E−03	3.30E+02	1.21E−06	**0.00E+00**	**0.00E+00**
C-4	**3.45E+01**	5.71E+02	4.21E+02	1.81E+02	4.26E+02	1.13E+02	4.19E+02	3.06E+01	4.36E+02	1.27E+02	4.28E+02	1.13E+02	4.52E+02	7.82E+02	4.49E+02	7.35E+02	5.62E+01	**4.88E+01**
C-5	**1.41E+01**	2.16E+03	9.23E+02	1.95E+03	9.30E+02	1.88E+03	8.75E+02	2.26E+03	1.43E+04	3.55E+04	1.19E+04	2.91E+03	1.86E+03	2.40E+03	1.85E+03	2.89E+03	1.64E+01	**3.46E+00**
C-6	2.15E+03	1.15E+05	1.39E+04	1.25E+05	2.36E+03	2.55E+05	1.96E+03	1.03E+04	7.45E+03	3.92E+04	3.92E+04	2.54E+04	2.40E+04	2.51E+05	3.01E+05	2.80E+07	**1.09E**−06	**2.62E**−06
C-7	7.12E+02	**5.60E**−02	7.12E+02	6.86E−02	7.12E+02	7.17E−02	7.12E+03	7.85E−02	7.12E+02	1.20E+01	7.12E+03	9.50E−02	7.16E+02	9.17E−02	7.18E+02	1.42E+01	**6.65E+01**	3.47E+00
C-8	**1.57E+01**	2.44E+04	1.96E+03	1.08E+04	3.59E+04	2.14E+04	3.53E+03	2.87E+04	9.03E+03	8.84E+05	2.68E+04	1.71E+04	6.83E+03	3.46E+04	6.17E+04	4.91E+08	1.70E+01	**3.14E+00**
C-9	1.10E+03	1.61E+02	1.10E+04	1.53E−02	1.10E+04	1.38E−02	1.10E+03	7.33E−02	1.10E+04	2.30E−01	1.10E+03	5.39E−03	1.10E+03	9.89E−02	1.10E+04	5.43E+01	**0.00E+00**	**0.00E+00**
C-10	**1.33E+03**	2.61E+05	2.10E+03	2.83E+04	4.10E+04	2.92E+04	3.37E+03	1.94E+04	8.49E+04	1.22E+05	2.72E+03	1.88E+04	9.01E+04	8.93E+04	3.52E+04	1.84E+06	3.14E+03	**3.67E+02**
C-11	**1.45E+01**	1.51E+02	1.48E+03	2.52E+02	1.50E+04	5.91E+02	1.45E+03	1.22E+03	1.47E+04	8.07E+02	1.49E+04	5.52E+02	1.45E+04	1.21E+03	1.51E+03	7.83E+02	2.79E+01	**3.33E+00**
C-12	**1.40E+03**	7.60E+01	1.40E+04	7.99E−02	1.40E+03	**6.79E−02**	1.40E+04	6.04E−02	1.40E+03	9.24E−02	1.40E+05	8.17E−02	1.41E+03	1.64E+01	1.41E+06	2.15E+01	1.68E+03	5.23E+02
C-13	**1.40E+01**	**6.53E−06**	1.40E+02	2.86E−05	1.40E+06	1.02E−05	1.40E+03	5.54E−04	1.40E+04	1.14E−04	1.40E+03	2.53E−05	1.40E+04	3.88E−04	1.45E+02	4.80E+02	3.06E+01	2.12E+01
C-14	3.32E+03	2.22E+04	4.35E+04	1.83E+04	7.39E+03	2.55E+04	7.20E+03	3.22E+04	7.70E+04	1.39E+04	7.44E+04	2.57E+04	7.61E+03	1.62E+04	9.40E+03	4.14E+03	**2.50E+01**	**1.87E+00**
C-15	1.70E+03	5.79E+02	1.70E+03	3.86E+01	1.71E+04	4.04E+01	1.70E+03	**2.76E**−05	1.71E+06	1.23E+02	1.70E+03	1.90E−03	1.72E+06	3.74E+01	1.74E+06	1.22E+01	**2.39E+01**	2.49E+00
C-16	2.50E+05	2.21E+09	3.28E+05	3.18E+09	3.02E+06	3.08E+09	5.37E+05	5.73E+08	2.47E+05	3.63E+09	7.06E+05	6.02E+09	8.65E+05	4.07E+09	4.20E+06	9.37E+09	**4.51E+02**	**1.38E+02**
C-17	7.70E+06	2.34E+07	4.13E+04	5.19E+04	6.65E+06	7.03E+05	8.41E+04	2.08E+04	2.97E+04	2.46E+04	2.43E+05	2.03E+03	8.33E+06	3.33E+07	5.58E+03	2.09E+03	**2.83E+02**	**8.61E+01**
C-18	4.20E+02	2.16E−04	4.20E+02	4.76E−04	4.20E+02	8.08E−04	4.20E+02	9.61E−04	4.20E+02	8.14E−04	4.20E+03	4.19E−04	4.20E+02	8.53E−04	4.20E+03	**2.11E**−07	**2.43E+01**	2.02E+00
C-19	5.10E+02	6.61E+02	5.11E+03	2.71E+02	5.16E+02	2.03E+02	**1.09E+01**	**4.90E**−01	5.26E+02	2.17E+02	5.18E+03	2.03E+02	5.42E+02	8.72E+02	5.39E+03	8.25E+02	1.41E+01	2.26E+00
C-20	**1.17E+02**	3.06E+01	8.13E+03	2.85E+01	8.20E+02	**2.78E+00**	7.65E+02	3.16E+01	2.33E+03	4.45E+02	2.09E+03	3.81E+01	2.76E+04	3.30E+02	2.75E+03	3.79E+01	1.40E+02	7.74E+01
C-21	3.05E+03	2.05E+06	2.29E+03	2.15E+04	3.26E+04	3.45E+05	2.86E+03	2.93E+03	8.35E+04	4.82E+03	4.82E+04	3.44E+04	3.30E+04	3.41E+04	4.91E+06	3.70E+07	**2.19E+02**	**3.77E+00**
C-22	**8.02E+02**	**6.50E**−02	8.02E+02	7.76E−01	8.02E+03	8.07E−01	8.02E+02	8.75E−01	8.02E+02	2.10E+01	8.02E+03	8.40E−01	8.06E+03	8.07E−02	8.08E+02	2.32E+00	1.49E+03	1.75E+03
C-23	2.47E+03	3.34E+04	2.86E+03	2.98E+04	4.49E+04	3.04E+04	4.43E+04	3.77E+04	8.93E+04	9.74E+04	3.58E+03	2.61E+04	7.73E+03	4.36E+04	7.07E+04	5.81E+06	**4.30E+02**	**6.24E+00**
C-24	2.00E+04	2.51E+02	2.00E+04	2.43E−02	2.00E+04	2.28E−02	2.00E+04	7.23E−03	2.00E+04	3.20E−02	2.00E+04	**6.29E**−03	2.00E+04	8.79E−02	2.00E+04	6.33E+01	**5.07E+02**	4.13E+00
C-25	**2.23E+02**	3.51E+05	3.00E+04	3.73E+04	5.00E+04	3.82E+04	4.27E+04	2.84E+04	9.39E+04	2.12E+05	3.62E+04	2.78E+04	8.91E+04	6.83E+04	4.42E+06	2.74E+06	4.81E+02	**2.80E+00**
C-26	**1.05E+03**	**2.41E+01**	2.38E+04	3.42E+02	2.40E+04	6.81E+02	2.35E+05	2.12E+03	2.37E+04	9.97E+02	2.39E+04	6.42E+02	2.35E+04	2.11E+03	2.41E+04	8.73E+02	1.13E+03	5.62E+01
C-27	2.30E+04	8.50E+01	2.30E+04	8.89E−02	2.30E+04	**7.69E**−02	2.30E+04	7.94E−02	2.30E+04	8.14E−02	2.30E+04	9.07E−02	2.31E+04	3.54E+01	2.31E+04	3.05E+00	**5.11E+02**	1.11E+01
C-28	5.30E+04	**7.43E**−06	5.30E+04	3.76E−05	5.30E+04	3.92E−05	5.30E+04	6.44E−04	5.30E+04	2.04E−04	5.30E+04	3.43E−05	5.30E+04	4.78E−04	5.35E+04	4.70E+02	**4.60E+02**	6.84E+00
C-29	**3.22E+02**	3.12E+04	5.25E+04	2.73E+04	8.29E+03	3.45E+04	8.10E+04	4.12E+04	8.60E+03	2.29E+05	8.34E+04	3.47E+05	8.51E+03	2.52E+04	8.30E+04	5.04E+03	3.63E+02	**1.32E+01**
C-30	**2.60E+04**	6.69E+02	2.60E+04	4.76E+01	2.61E+04	5.94E+01	2.60E+04	**3.66E**−04	2.61E+04	2.13E+02	2.60E+04	2.80E−03	2.62E+04	4.64E+01	2.64E+04	2.12E+02	6.01E+05	2.99E+04

The best-obtained results are in bold

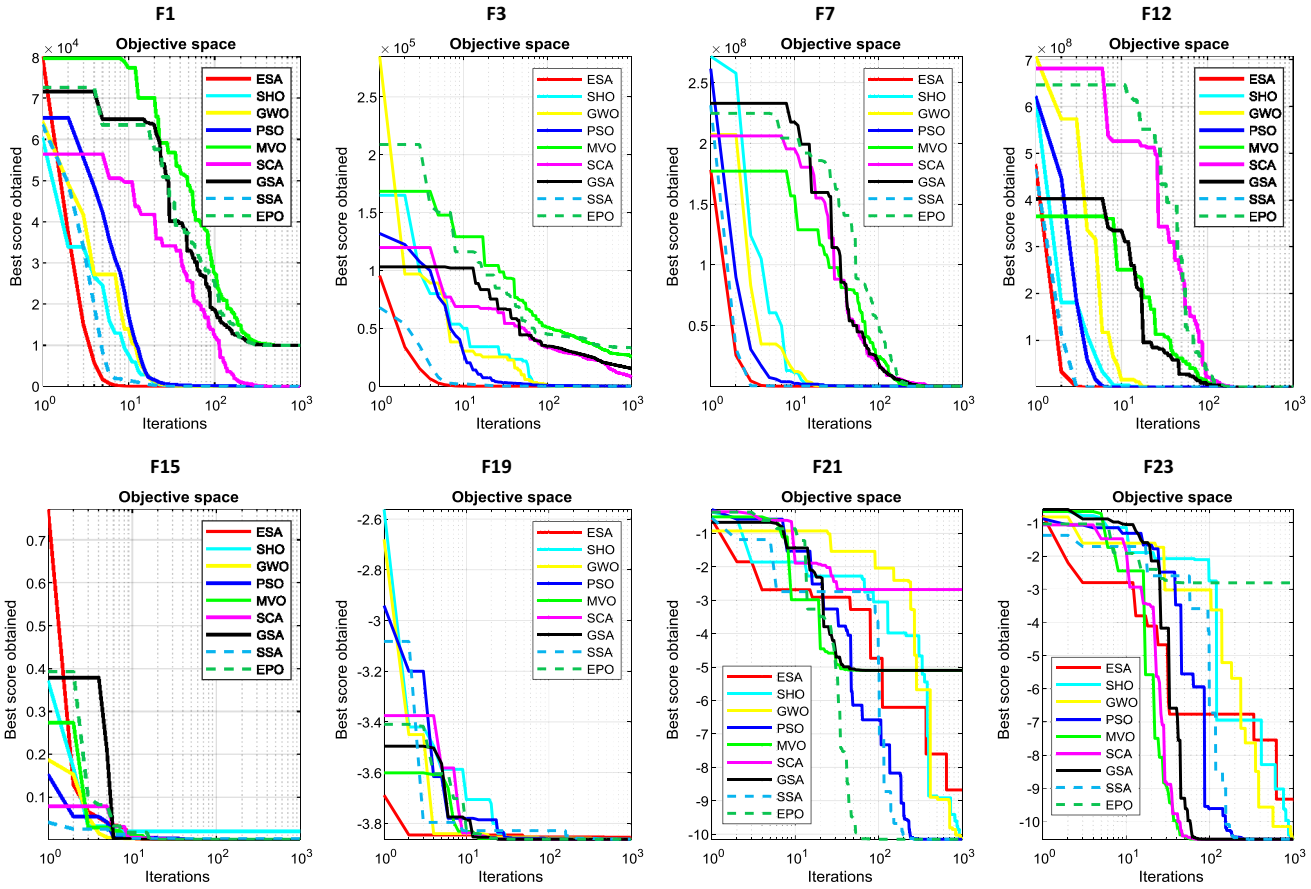

Fig. 3 Convergence analysis of the proposed ESA and other competitor algorithms on benchmark test problems

2. Number of search agents: ESA algorithm was run for different values of search agent (i.e., 30, 50, 80, 100). Table 3 shows the effect of number of search agents on benchmark test functions. It is observed from table that the value of objective function decreases with the increase in number of search agents.

4.6 Scalability study

This subsection describes the effect of scalability on various test functions by using proposed ESA and other competitive algorithms. The dimensionality of the test functions is made to vary between the different ranges. Figure 4 shows the performance of ESA and other algorithms on scalable benchmark test functions. It is observed that the performance of ESA is not too much degraded when the dimensionality of search space is increased. The results reveal that the performance of ESA is least affected with the increase in dimensionality of search space. This is due to better capability of the proposed ESA for balancing between exploration and exploitation.

4.7 Statistical testing

Apart from standard statistical analysis such as mean and standard deviation, ANOVA test has been conducted. ANOVA test is used to determine whether the results obtained from proposed algorithm are different from other competitor algorithms in a statistically significant way. The sample size for ANOVA test is 30 with 95% confidence of interval. A p value determines whether the given algorithm is statistically significant or not. If the p value of the given algorithm is less than 0.05, then the corresponding algorithm is statistically significant. Table 8 shows the analysis of ANOVA test on the benchmark test functions. It is observed from Table 8 that the p value obtained from ESA is much smaller than 0.05 for all the benchmark test functions. Therefore, the proposed ESA is statistically different from the other competitor algorithms.

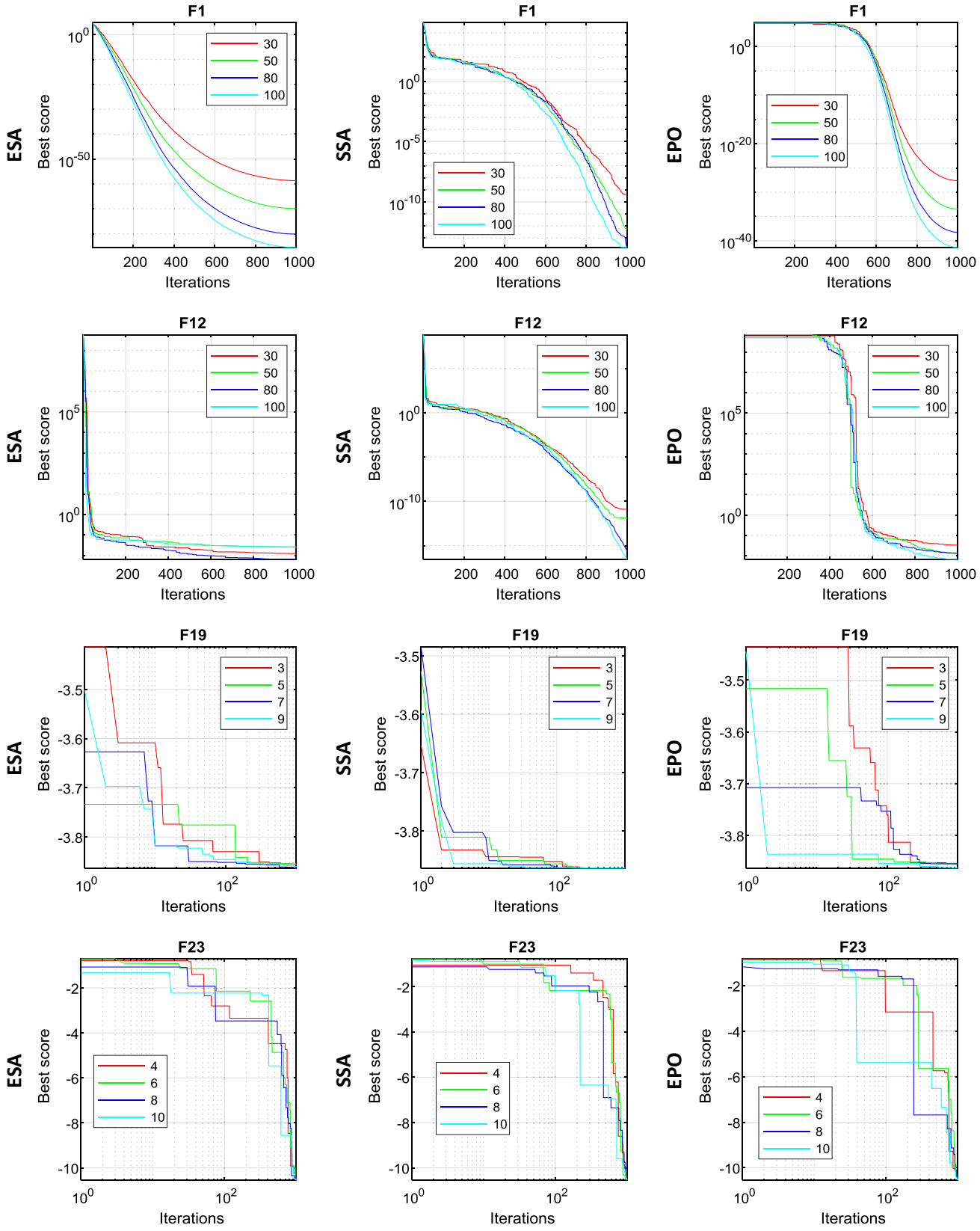

Fig. 4 Effect of scalability on the performance of ESA, SSA, and EPO algorithms

Table 8 ANOVA test results

F	p value	ESA	SHO	GWO	PSO	MVO	SCA	GSA	SSA	EPO
F_1	1.71E−21	SHO, GWO, PSO, MVO, SCA, GSA,GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_2	2.61E−23	SHO, GWO, PSO, MVO, SCA,GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA
F_3	1.51E−35	SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, GWO, PSO, SCA, GSA, GA	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA
F_4	4.82E−70	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_5	2.18E−12	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, PSO, MVO, SCA, GSA, GA,EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_6	8.90E−14	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA, GA, EPO	ESA, SHO, PSO, SCA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, GA
F_7	4.12E−75	SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA	ESA, SHO, GWO, PSO, SCA, GSA, GA
F_8	5.52E−19	SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, GWO, MVO, SCA, GSA, EPO	ESA, SHO, PSO, MVO, SCA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, EPO	ESA, SHO, PSO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, PSO, MVO, SCA
F_9	2.41E−65	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA,GSA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA
F_{10}	1.11E−80	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA	ESA, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GA, EPO	ESA, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, SCA	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA

Table 8 (continued)

F	p value	ESA	SHO	GWO	PSO	MVO	SCA	GSA	SSA	EPO
F_{11}	4.26E−47	SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, EPO	ESA, SHO, GWO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, GA
F_{12}	2.90E−51	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA
F_{13}	2.20E−18	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, EPO	ESA, SHO, PSO, MVO, SCA, GA
F_{14}	3.24E−41	SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA
F_{15}	7.15E−13	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, GSA, EPO	ESA, SHO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA	ESA, SHO, GWO, PSO, MVO, GA, EPO	ESA, SHO, PSO, SCA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA	ESA, SHO, GWO, PSO, MVO, SCA
F_{16}	4.27E−38	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, GSA	ESA, SHO, GWO, PSO, MVO, SCA	ESA, SHO, GWO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA
F_{17}	5.50E−48	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GA, EPO	ESA, SHO, PSO, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA
F_{18}	9.60E−42	SHO, GWO, PSO, MVO, GSA, GA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GSA, GA, EPO	ESA, SHO, PSO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, SHO, PSO, MVO, SCA,EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{19}	4.41E−36	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GA, EPO	ESA, SHO, GWO, SCA, GSA, GA, EPO	ESA, SHO, PSO, SCA, GA, EPO	ESA, SHO, PSO, MVO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, EPO	ESA, SHO, PSO, SCA, GSA, EPO	ESA, GWO, PSO, MVO, SCA, GSA, GA
F_{20}	7.70E−80	SHO, GWO, PSO, MVO, SCA, GSA	ESA, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, PSO, MVO, SCA, GSA,	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, PSO, GSA, GA	ESA, SHO, GWO, MVO, SCA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA

Table 8 (continued)

F	p value	ESA	SHO	GWO	PSO	MVO	SCA	GSA	SSA	EPO
F_{21}	6.16E−68	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, GSA, GA, EPO	ESA, SHO, PSO, MVO, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, GA, EPO	ESA, SHO, PSO, SCA, GA	ESA, SHO, GWO, PSO, MVO, SCA, GSA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA
F_{22}	2.10E−73	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, PSO, MVO, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GA, EPO	ESA, GWO, PSO, SCA, GSA, GA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA
F_{23}	3.15E−30	SHO, GWO, PSO, MVO, SCA, GSA, GA, EPO	ESA, GWO, MVO, SCA, GSA, EPO	ESA, SHO, PSO, MVO, SCA, GSA, GA, EPO	ESA, SHO, GWO, MVO, SCA, GSA, GA, EPO	ESA, SHO, PSO, GSA, GA, EPO	ESA, SHO, GWO, PSO, MVO, GSA, GA	ESA, SHO, MVO, SCA, GA, EPO	ESA, SHO, GWO, PSO, MVO, SCA, GSA	ESA, SHO, GWO, PSO, MVO, SCA, GSA, GA

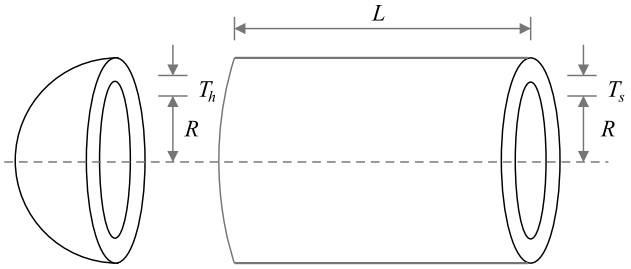

Fig. 5 Schematic view of pressure vessel problem

5 ESA for engineering problems

The proposed ESA algorithm has been tested and validated on six constrained and one unconstrained engineering problems. These are pressure vessel, speed reducer, welded beam, tension/compression spring, 25-bar truss, rolling element bearing, and displacement of loaded structure design problems [78, 79]. These optimization design problems have different constraints with different nature. Different types of penalty functions are used to handle these problems such as static penalty, dynamic penalty, annealing penalty, adaptive penalty, co-evolutionary penalty, and death penalty [80].

However, death penalty function handles the solution which can violate the constraints. This function assigns the fitness value as zero to discard the infeasible solutions during optimization, i.e., it does not employ any information about infeasible solutions. Due to its low computational complexity and simplicity, ESA algorithm is equipped with death penalty function to handle both constrained and unconstrained engineering design problems.

5.1 Constrained engineering problems

This subsection describes six constrained engineering problems and compared it with other competitor approaches. The statistical analysis of these problems is also done to validate the efficiency and effectiveness of proposed algorithm.

5.1.1 Pressure vessel design problem

This problem was first proposed by Kannan and Kramer [81] to minimize the total cost consisting of material, forming, and welding of a cylindrical vessel. The schematic view of pressure vessel problem is shown in Fig. 5 which is capped at both ends by hemispherical heads. There are four design variables of this problem:

- T_s (z_1, thickness of the shell).
- T_h (z_2, thickness of the head).
- R (z_3, inner radius).
- L (z_4, length of the cylindrical section without considering the head).

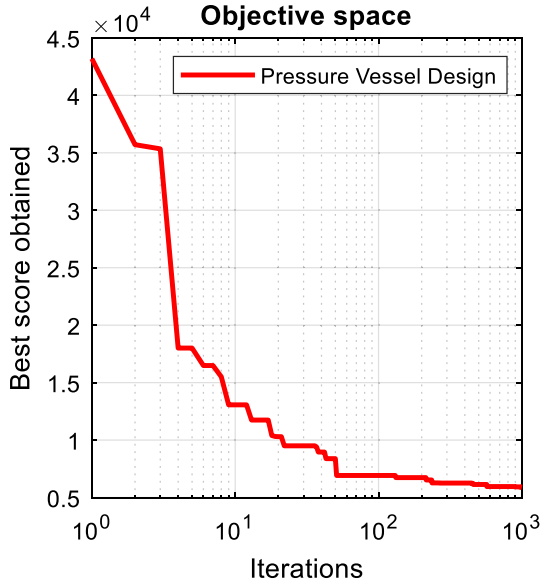

Fig. 6 Convergence analysis of ESA for pressure vessel design problem

Among these four design variables, R and L are continuous variables. T_s and T_h are integer values which are multiples of 0.0625 in. The mathematical formulation of this problem is given below:

$$\begin{aligned}
&\text{Consider } \vec{z} = [z_1\ z_2\ z_3\ z_4] = [T_s\ T_h\ R\ L],\\
&\text{Minimize } f(\vec{z}) = 0.6224 z_1 z_3 z_4 + 1.7781 z_2 z_3^2\\
&\qquad\qquad + 3.1661 z_1^2 z_4 + 19.84 z_1^2 z_3,\\
&\text{Subject to:}\\
&g_1(\vec{z}) = -z_1 + 0.0193 z_3 \le 0,\\
&g_2(\vec{z}) = -z_3 + 0.00954 z_3 \le 0,\\
&g_3(\vec{z}) = -\pi z_3^2 z_4 - \frac{4}{3}\pi z_3^3 + 1{,}296{,}000 \le 0,\\
&g_4(\vec{z}) = z_4 - 240 \le 0,\\
&\text{where}\\
&1 \times 0.0625 \le z_1,\ z_2 \le 99 \times 0.0625,\ 10.0 \le z_3,\ z_4 \le 200.0.
\end{aligned} \tag{15}$$

Table 10 Statistical results obtained from different algorithms for pressure vessel design problem

Algorithms	Best	Mean	Worst
ESA	**5879.9558**	**5882.5248**	5885.6581
EPO	5880.0700	5884.1401	5891.3099
SHO	5885.5773	5887.4441	5892.3207
GWO	5889.3689	5891.5247	5894.6238
PSO	5891.3879	6531.5032	7394.5879
MVO	6011.5148	6477.3050	7250.9170
SCA	6137.3724	6326.7606	6512.3541
GSA	11550.2976	23342.2909	**33226.2526**
SSA	5890.3279	6264.0053	7005.7500

The best-obtained results are in bold

Table 9 reveals the obtained best comparison between ESA and other competitor algorithms such as EPO, SHO, GWO, PSO, MVO, SCA, GSA, and SSA. The proposed ESA provides optimal solution at $z_{1-4} = (0.778092, 0.383236, 40.315052, 200.00000)$ with corresponding fitness value as $f(z_{1-4}) = 5879.9558$. From this table, it can be seen that, ESA algorithm is able to find best optimal design with minimum cost.

The statistical results of pressure vessel design problem are tabulated in Table 10. It can be seen from Table 10 that ESA surpassed other algorithms for providing the best solution in terms of best, mean, and median. The convergence behavior obtained by proposed ESA for best optimal design is shown in Fig. 6.

5.1.2 Speed reducer design problem

The speed reducer design problem is a challenging benchmark problem due to its seven design variables [82] as shown in Fig. 7. The objective of this problem is to minimize the weight of speed reducer subject to constraints [83]:

Table 9 Comparison of best solution obtained from different algorithms for pressure vessel design problem

Algorithms	Optimum variables				Optimum cost
	T_s	T_h	R	L	
ESA	0.778092	0.383236	40.315052	200.00000	**5879.9558**
EPO	0.778099	0.383241	40.315121	200.00000	5880.0700
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
SSA	0.752362	0.399540	40.452514	198.00268	5890.3279

The best-obtained result is in bold

Table 11 Comparison of best solution obtained from different algorithms for speed reducer design problem

Algorithms	Optimum variables							Optimum cost
	b	m	p	l_1	l_2	d_1	d_2	
ESA	3.50120	0.7	17	7.3	7.8	3.33415	5.26531	**2993.9584**
EPO	3.50123	0.7	17	7.3	7.8	3.33421	5.26536	2994.2472
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
SSA	3.510253	0.7	17	8.35	7.8	3.362201	5.287723	3067.561

The best-obtained result is in bold

Table 12 Statistical results obtained from different algorithms for speed reducer design problem

Algorithms	Best	Mean	Worst
ESA	**2993.9584**	**2996.002**	2999.569
EPO	2994.2472	2997.482	2999.092
SHO	2998.5507	2999.640	3003.889
GWO	3001.288	3005.845	3008.752
PSO	3005.763	3105.252	3211.174
MVO	3002.928	3028.841	3060.958
SCA	3030.563	3065.917	3104.779
GSA	3051.120	3170.334	**3363.873**
SSA	3067.561	3186.523	3313.199

The best-obtained results are in bold

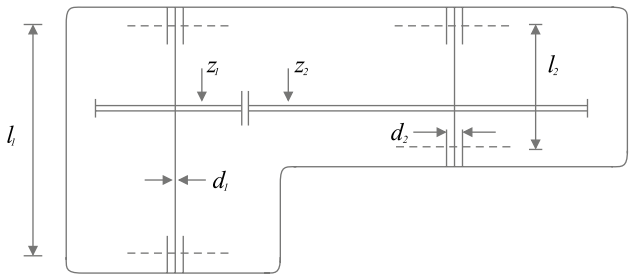

Fig. 7 Schematic view of speed reducer problem

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

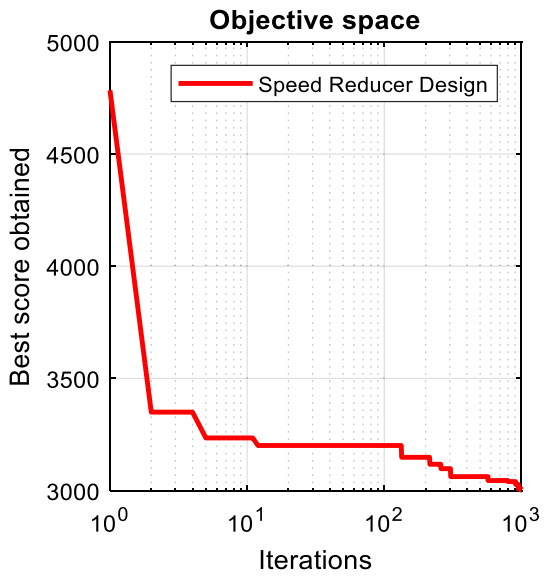

Fig. 8 Convergence analysis of ESA for speed reducer design problem

There are seven design variables (z_1–z_7) such as face width (b), module of teeth (m), number of teeth in the pinion (p), length of the first shaft between bearings (l_1), length of the second shaft between bearings (l_2), diameter of first (d_1) shafts, and diameter of second shafts (d_2). The mathematical formulation of this problem is formulated as follows:

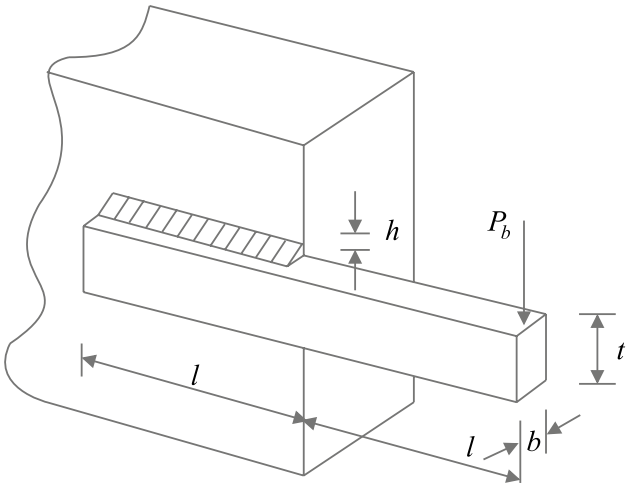

Fig. 9 Schematic view of welded beam problem

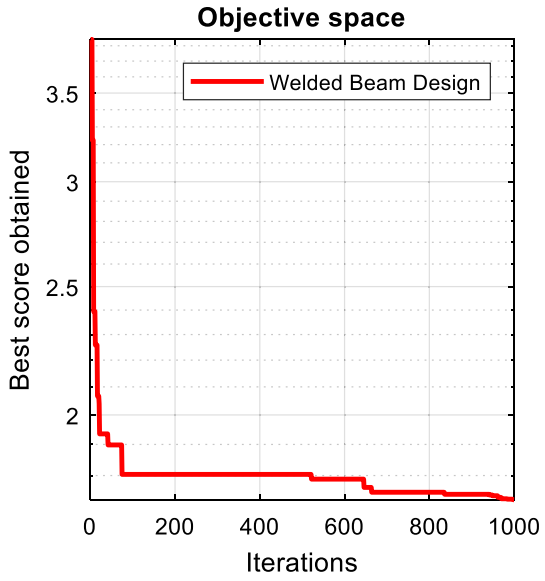

Fig. 10 Convergence analysis of ESA for welded beam design problem

Consider $\vec{z} = [z_1\ z_2\ z_3\ z_4\ z_5\ z_6\ z_7] = [b\ m\ p\ l_1\ l_2\ d_1\ d_2]$,

Minimize $f(\vec{z}) = 0.7854 z_1 z_2^2 (3.3333 z_3^2 + 14.9334 z_3 - 43.0934) - 1.508 z_1 (z_6^2 + z_7^2) + 7.4777 (z_6^3 + z_7^3) + 0.7854 (z_4 z_6^2 + z_5 z_7^2)$,

Subject to:

$$g_1(\vec{z}) = \frac{27}{z_1 z_2^2 z_3} - 1 \le 0,$$

$$g_2(\vec{z}) = \frac{397.5}{z_1 z_2^2 z_3^2} - 1 \le 0,$$

$$g_3(\vec{z}) = \frac{1.93 z_4^3}{z_2 z_6^4 z_3} - 1 \le 0,$$

$$g_4(\vec{z}) = \frac{1.93 z_5^3}{z_2 z_7^4 z_3} - 1 \le 0,$$

$$g_5(\vec{z}) = \frac{[(745(z_4/z_2 z_3))^2 + 16.9 \times 10^6]^{1/2}}{110 z_6^3} - 1 \le 0,$$

$$g_6(\vec{z}) = \frac{[(745(z_5/z_2 z_3))^2 + 157.5 \times 10^6]^{1/2}}{85 z_7^3} - 1 \le 0,$$

$$g_7(\vec{z}) = \frac{z_2 z_3}{40} - 1 \le 0,$$

$$g_8(\vec{z}) = \frac{5 z_2}{z_1} - 1 \le 0,$$

$$g_9(\vec{z}) = \frac{z_1}{12 z_2} - 1 \le 0,$$

$$g_{10}(\vec{z}) = \frac{1.5 z_6 + 1.9}{z_4} - 1 \le 0,$$

$$g_{11}(\vec{z}) = \frac{1.1 z_7 + 1.9}{z_5} - 1 \le 0,$$

where

$$2.6 \le z_1 \le 3.6,\ 0.7 \le z_2 \le 0.8,\ 17 \le z_3 \le 28,\ 7.3 \le z_4 \le 8.3,\ 7.3 \le z_5 \le 8.3,\ 2.9 \le z_6 \le 3.9,\ 5.0 \le z_7 \le 5.5. \tag{16}$$

Table 13 Comparison of best solution obtained from different algorithms for welded beam design problem

Algorithms	Optimum variables				Optimum cost
	h	*l*	*t*	*b*	
ESA	0.203296	3.471148	9.035107	0.201150	**1.721026**
EPO	0.205411	3.472341	9.035215	0.201153	1.723589
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
SSA	0.164171	4.032541	10.00000	0.223647	1.873971

The best-obtained result is in bold

Table 14 Statistical results obtained from different algorithms for welded beam design problem

Algorithms	Best	Mean	Worst
ESA	**1.721026**	**1.725023**	1.727208
EPO	1.723589	1.725124	1.727211
SHO	1.725661	1.725828	1.726064
GWO	1.726995	1.727128	1.727564
PSO	1.820395	2.230310	**3.048231**
MVO	1.725472	1.729680	1.741651
SCA	1.759173	1.817657	1.873408
GSA	2.172858	2.544239	3.003657
SSA	1.873971	2.119240	2.320125

The best-obtained results are in bold

Table 15 Comparison of best solution obtained from different algorithms for tension/compression spring design problem

Algorithms	Optimum variables			Optimum cost
	d	*D*	*P*	
ESA	0.051080	0.342895	12.0895	**0.012655526**
EPO	0.051087	0.342908	12.0898	0.012656987
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
SSA	0.05010	0.310111	14.0000	0.013036251

The best-obtained result is in bold

Table 16 Statistical results obtained from different algorithms for tension/compression spring design problem

Algorithms	Best	Mean	Worst
ESA	**0.012655526**	**0.012677562**	0.012667896
EPO	0.012656987	0.012678903	0.012667902
SHO	0.012674000	0.012684106	0.012715185
GWO	0.012678321	0.012697116	0.012720757
PSO	0.013192580	0.014817181	**0.017862507**
MVO	0.012816930	0.014464372	0.017839737
SCA	0.012709667	0.012839637	0.012998448
GSA	0.012873881	0.013438871	0.014211731
SSA	0.013036251	0.014036254	0.016251423

The best-obtained results are in bold

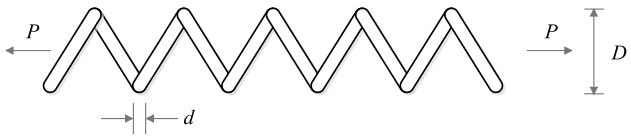

Fig. 11 Schematic view of tension/compression spring problem

Table 11 shows the comparison of the best obtained optimal solution with various optimization algorithms. The proposed ESA algorithm provides optimal solution at $z_{1-7} = (3.50120, 0.7, 17, 7.3, 7.8, 3.33415, 5.26531)$ with corresponding fitness value as $f(z_{1-7}) = 2993.9584$. The statistical results of ESA and competitor optimization algorithms are given in Table 12.

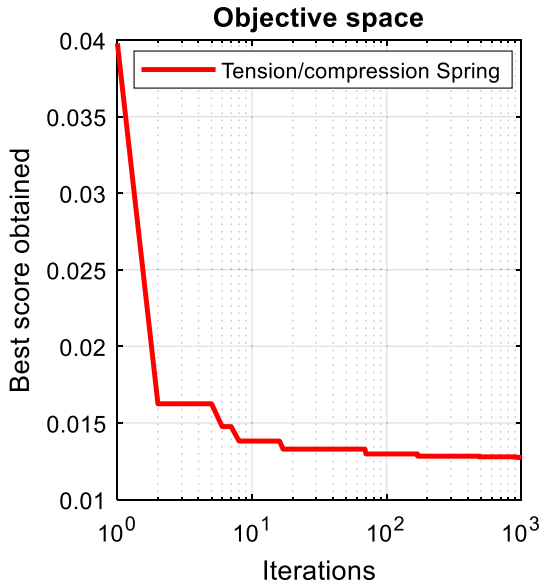

Fig. 12 Convergence analysis of ESA for tension/compression spring design problem

The results show that ESA outperforms than other metaheuristic optimization algorithms. Figure 8 shows the convergence behavior of ESA on speed reducer design problem.

5.1.3 Welded beam design problem

The main objective of this design problem is to minimize the fabrication cost of welded beam as shown in Fig. 9. The optimization constraints of welded beam are shear stress (τ), bending stress (θ) in the beam, buckling load (P_c) on the bar, and end deflection (δ) of the beam. There are four design variables (z_1–z_4) of this problem.

- h (z_1, thickness of weld)
- l (z_2, length of the clamped bar)
- t (z_3, height of the bar)
- b (z_4, thickness of the bar)

The mathematical formulation is described as follows:

Table 17 Member stress limitations for 25-bar truss design problem

Element group	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)

Consider $\vec{z} = [z_1\ z_2\ z_3\ z_4] = [h\ l\ t\ b]$,

Minimize $f(\vec{z}) = 1.10471 z_1^2 z_2 + 0.04811 z_3 z_4 (14.0 + z_2)$,

Subject to:

$g_1(\vec{z}) = \tau(\vec{z}) - 13{,}600 \le 0,$

$g_2(\vec{z}) = \sigma(\vec{z}) - 30{,}000 \le 0,$

$g_3(\vec{z}) = \delta(\vec{z}) - 0.25 \le 0,$

$g_4(\vec{z}) = z_1 - z_4 \le 0,$

$g_5(\vec{z}) = 6000 - P_c(\vec{z}) \le 0,$

$g_6(\vec{z}) = 0.125 - z_1 \le 0,$

$g_7(\vec{z}) = 1.10471 z_1^2 + 0.04811 z_3 z_4 (14.0 + z_2) - 5.0 \le 0,$

where

$0.1 \le z_1,\ 0.1 \le z_2,\ z_3 \le 10.0,\ z_4 \le 2.0,$

$$\tau(\vec{z}) = \sqrt{(\tau')^2 + (\tau'')^2 + (l\tau'\tau'')/\sqrt{0.25(l^2 + (h+t)^2)}},$$

$$\tau' = \frac{6000}{\sqrt{2}hl}, \qquad \sigma(\vec{z}) = \frac{504{,}000}{t^2 b}, \qquad \delta(\vec{z}) = \frac{65{,}856{,}000}{(30 \times 10^6) b t^3},$$

$$\tau'' = \frac{6000(14 + 0.5l)\sqrt{0.25(l^2 + (h+t)^2)}}{2[0.707hl(l^2/12 + 0.25(h+t)^2)]},$$

$$P_c(\vec{z}) = 64{,}746.022(1 - 0.0282346t) t b^3. \tag{17}$$

The obtained best comparison between proposed ESA and other metaheuristics is presented in Table 13. Among other algorithms, the proposed ESA provides optimal solution at $z_{1-4} = (0.203296, 3.471148, 9.035107, 0.201150)$ with corresponding fitness value equal to $f(z_{1-4}) = 1.721026$. Table 14 shows the statistical comparison of the proposed algorithm and other competitor algorithms. ESA shows superiority to other algorithms in terms of best, mean, and median.

Table 19 Statistical results obtained from different algorithms for 25-bar truss design problem

Groups	ESA	ACO [86]	PSO [87]	CSS [88]	BB-BC [89]
*A*1	0.01	0.01	0.01	0.01	0.01
*A*2–*A*5	2.007	2.042	2.052	2.003	1.993
*A*6–*A*9	3.001	3.001	3.001	3.007	3.056
*A*10–*A*11	0.01	0.01	0.01	0.01	0.01
*A*12–*A*13	0.01	0.01	0.01	0.01	0.01
*A*14–*A*17	0.661	0.684	0.684	0.687	0.665
*A*18–*A*21	1.620	1.625	1.616	1.655	1.642
*A*22–*A*25	2.668	2.672	2.673	2.66	2.679
Best weight	**544.92**	545.03	545.21	545.10	545.16
Average weight	**545.13**	545.74	546.84	545.58	545.66
Std. dev.	**0.401**	0.94	1.478	0.412	0.491

The best-obtained results are in bold

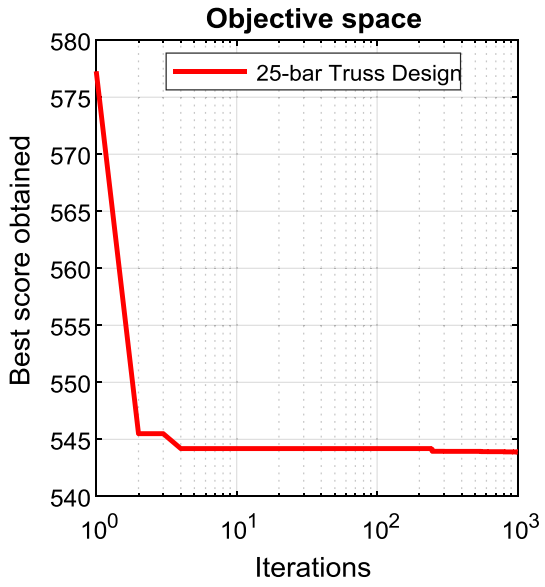

Fig. 13 Convergence analysis of ESA for 25-bar truss design problem

Table 18 Two loading conditions for the 25-bar truss design problem

Node	Case 1			Case 2		
	P_xKips (kN)	P_yKips (kN)	P_zKips (kN)	P_xKips (kN)	P_yKips (kN)	P_zKips (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

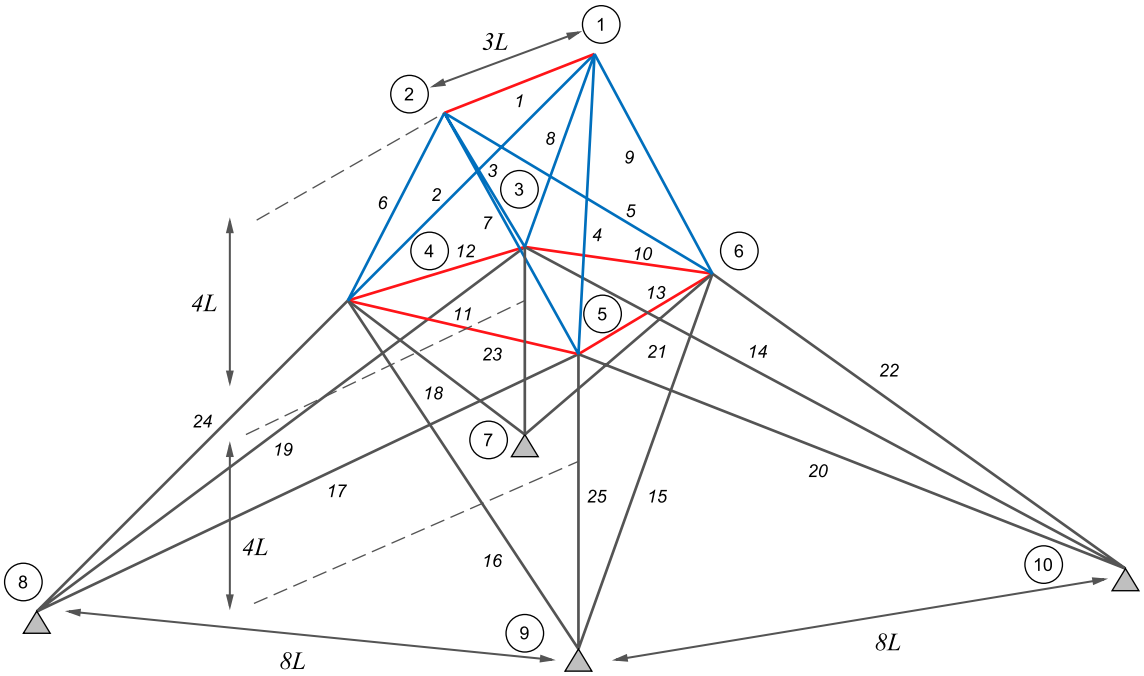

Fig. 14 Schematic view of 25-bar truss problem

Table 20 Comparison of best solution obtained from different algorithms 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	
ESA	125	21.41750	10.94109	0.510	0.515	0.4	0.7	0.3	0.02	0.6	**85070.085**
EPO	125	21.41890	10.94113	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85,067.983
SHO	125	21.40732	10.93268	0.515	0.515	0.4	0.7	0.3	0.02	0.6	85,054.532
GWO	125.6199	21.35129	10.98781	0.515	0.515	0.5	0.68807	0.300151	0.03254	0.62701	84,807.111
PSO	125	20.75388	11.17342	0.515	0.515000	0.5	0.61503	0.300000	0.05161	0.60000	81,691.202
MVO	125.6002	21.32250	10.97338	0.515	0.515000	0.5	0.68782	0.301348	0.03617	0.61061	84,491.266
SCA	125	21.14834	10.96928	0.515	0.515	0.5	0.7	0.3	0.02778	0.62912	83,431.117
GSA	125	20.85417	11.14989	0.515	0.517746	0.5	0.61827	0.304068	0.02000	0.624638	82,276.941
SSA	125	20.77562	11.01247	0.515	0.515000	0.5	0.61397	0.300000	0.05004	0.610001	82,773.982

The best-obtained result is in bold

Figure 10 shows the convergence analysis of best optimal solution obtained from ESA for welded beam design problem.

5.1.4 Tension/compression spring design problem

The objective of this design problem is to minimize the tension/ compression spring weight (see Fig. 11). The optimization constraints of this problem are described as follows:

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

Table 21 Statistical results obtained from different algorithms for rolling element bearing design problem

Algorithms	Best	Mean	Worst
ESA	**85,070.085**	**85,045.953**	86,553.485
EPO	85,067.983	85,042.352	86,551.599
SHO	85,054.532	85,024.858	85,853.876
GWO	84,807.111	84,791.613	84,517.923
PSO	81,691.202	50,435.017	**32,761.546**
MVO	84,491.266	84,353.685	84,100.834
SCA	83,431.117	81,005.232	77,992.482
GSA	82,276.941	78,002.107	71,043.110
SSA	82,773.982	81,198.753	80,687.239

The best-obtained results are in bold

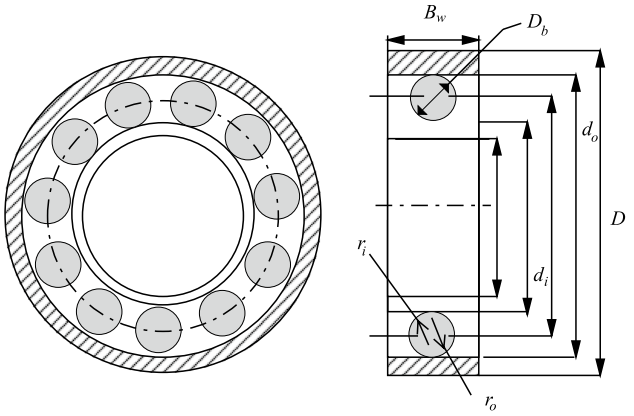

Fig. 15 Schematic view of rolling element bearing problem

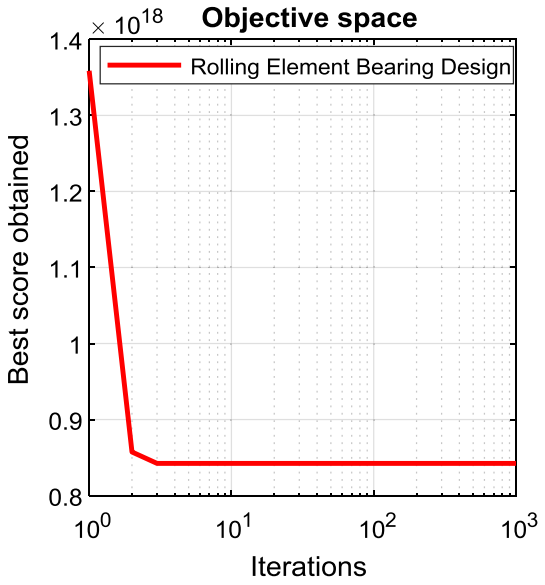

Fig. 16 Convergence analysis of ESA for rolling element bearing design problem

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

Table 22 Comparison of best solution obtained from different algorithms for displacement of loaded structure problem

Algorithms	Optimum cost (π)
ESA	**167.2635**
EPO	168.8231
SHO	168.8889
GWO	170.3645
PSO	170.5960
MVO	169.3023
SCA	169.0032
GSA	176.3697
SSA	171.3674

The best-obtained result is in bold

Consider $\vec{z} = [z_1\ z_2\ z_3] = [d\ D\ P]$,

Minimize $f(\vec{z}) = (z_3 + 2)z_2 z_1^2$,

Subject to:

$$
\begin{aligned}
g_1(\vec{z}) &= 1 - \frac{z_2^3 z_3}{71785 z_1^4} \le 0,\\
g_2(\vec{z}) &= \frac{4z_2^2 - z_1 z_2}{12566(z_2 z_1^3 - z_1^4)} + \frac{1}{5108 z_1^2} \le 0,\\
g_3(\vec{z}) &= 1 - \frac{140.45 z_1}{z_2^2 z_3} \le 0,\\
g_4(\vec{z}) &= \frac{z_1 + z_2}{1.5} - 1 \le 0,
\end{aligned}
\tag{18}
$$

where

$0.05 \le z_1 \le 2.0,\ 0.25 \le z_2 \le 1.3,\ 2.0 \le z_3 \le 15.0.$

Table 15 shows the comparison for the best solution obtained from the proposed ESA and other competitor algorithms in terms of design variables and objective values. ESA obtained best solution at design variables $z_{1-3} = (0.051080, 0.342895, 12.0895)$ with an objective function value of $f(z_{1-3}) = 0.012655526$. The results reveal that ESA performs better than the other competitor algorithms. The statistical results of tension/compression spring design problem for the reported algorithms are compared and tabulated in Table 16. It can be seen from Table 16 that ESA provides better statistical results than the other optimization algorithms in terms of best, mean, and median.

Figure 12 shows the convergence behavior of best optimal solution obtained from proposed ESA.

5.1.5 25-bar truss design problem

The truss design problem is a popular optimization problem [84, 85] (see Fig. 14). There are 10 nodes and 25 bars

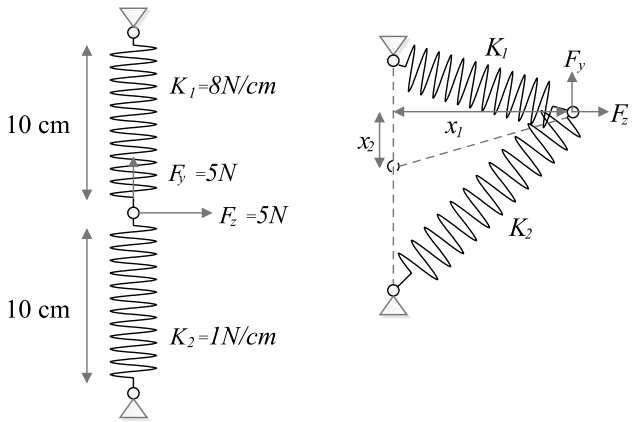

Fig. 17 Schematic view of displacement of loaded structure

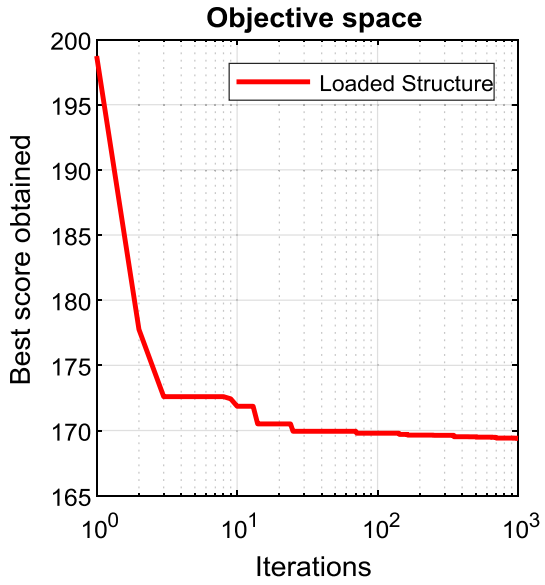

Fig. 18 Convergence analysis of ESA for displacement of loaded structure problem

Table 23 Statistical results obtained from different algorithms for displacement of loaded structure problem

Algorithms	Best	Mean	Worst
ESA	**167.2635**	**169.5362**	176.1128
EPO	168.8231	170.1309	**230.9721**
SHO	168.8889	170.3659	173.6357
GWO	170.3645	171.3694	174.3970
PSO	170.5960	174.6354	175.3602
MVO	169.3023	171.0034	174.3047
SCA	169.0032	171.7530	174.4527
GSA	176.3697	178.7521	179.5637
SSA	171.3674	172.0374	174.0098

The best-obtained results are in bold

cross-sectional members. These 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 affects on this problem are as follows:

- $p = 0.0272$ N/cm^3 (0.1 lb/in.^{3})
- $E = 68947$ MPa (10,000 Ksi)
- Displacement limitation = 0.35 in.
- Maximum displacement = 0.3504 in.
- Design variable set = $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.6, 2.8, 3.0, 3.2, 3.4\}$

Table 17 shows the member stress limitations for this problem. The loading conditions for 25-bar truss are presented in Table 18. The comparison of best obtained solutions among several algorithms is tabulated in Table 19. It can be seen that the proposed ESA is better than other algorithms in terms of best, average, and standard deviation. ESA converges very efficiently towards optimal solution as shown in Fig. 13.

5.1.6 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 depicted in Fig. 15. There are ten decision variables such as 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 (see Fig. 15). The mathematical representation of this problem is given below:

Table 24 Shekel's Foxholes function F_{14}

$(a_{ij}, i = 1, 2$ and $j = 1, 2, \ldots, 25)$								
$i\backslash j$	1	2	3	4	5	6	...	25
1	−32	−16	0	16	32	−32	...	32
2	−32	−32	−32	−32	−32	−16	...	32

Table 25 Hartman function F_{19}

i	$(a_{ij}, j = 1, 2, 3)$			c_i	$(p_{ij}, j = 1, 2, 3)$		
1	3	10	30	1	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.038150	0.5743	0.8828

Table 26 Shekel's Foxholes functions F_{21}, F_{22}, F_{23}

i	$(a_{ij}, j = 1,2,3,4)$				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

$$\text{Maximize } C_d = \begin{cases} f_c Z^{2/3} D_b^{1.8}, & \text{if } D \le 25.4\,\text{mm} \\ C_d = 3.647 f_c Z^{2/3} D_b^{1.4}, & \text{if } D > 25.4\,\text{mm} \end{cases}$$

Subject to:

$$g_1(\vec{z}) = \frac{\phi_0}{2sin^{-1}(D_b/D_m)} - Z + 1 \le 0,$$

$$g_2(\vec{z}) = 2D_b - K_{\text{Dmin}}(D-d) \ge 0,$$

$$g_3(\vec{z}) = K_{\text{Dmax}}(D-d) - 2D_b \ge 0,$$

$$g_4(\vec{z}) = \zeta B_w - D_b \le 0,$$

$$g_5(\vec{z}) = D_m - 0.5(D+d) \ge 0,$$

$$g_6(\vec{z}) = (0.5+e)(D+d) - D_m \ge 0,$$

$$g_7(\vec{z}) = 0.5(D - D_m - D_b) - \varepsilon D_b \ge 0,$$

$$g_8(\vec{z}) = f_i \ge 0.515,$$

$$g_9(\vec{z}) = f_o \ge 0.515,$$

where

$$f_c = 37.91\left[1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_o-1)}{f_o(2f_i-1)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3} \times \left[\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{1/3}}\right]\left[\frac{2f_i}{2f_i-1}\right]^{0.41} \quad (19)$$

$$x = [\{(D-d)/2 - 3(T/4)\}^2 + \{D/2 - T/4 - D_b\}^2 - \{d/2 + T/4\}^2]$$

$$y = 2\{(D-d)/2 - 3(T/4)\}\{D/2 - T/4 - D_b\}$$

$$\phi_o = 2\pi - 2\cos^{-1}\left(\frac{x}{y}\right)$$

$$\gamma = \frac{D_b}{D_m}, \quad f_i = \frac{r_i}{D_b}, \quad f_o = \frac{r_o}{D_b}, \quad T = D - d - 2D_b$$

$$D = 160, \quad d = 90, \quad B_w = 30, \quad r_i = r_o = 11.033$$

$$0.5(D+d) \le D_m \le 0.6(D+d), \quad 0.15(D-d) \le D_b \le 0.45(D-d), \quad 4 \le Z \le 50, \quad 0.515 \le f_i \text{ and } f_o \le 0.6,$$

$$0.4 \le K_{\text{Dmin}} \le 0.5, \quad 0.6 \le K_{\text{Dmax}} \le 0.7, \quad 0.3 \le e \le 0.4,$$

$$0.02 \le e \le 0.1, \quad 0.6 \le \zeta \le 0.85.$$

Table 20 shows the performance comparison of best obtained optimal solution. The proposed ESA provides optimal solution at $z_{1-10} = (125, 21.41750, 10.94109, 0.510, 0.515, 0.4, 0.7, 0.3, 0.02, 0.6)$ with corresponding fitness value equal to $f(z_{1-10}) = 85070.085$. The statistical results obtained for rolling element bearing design problem are compared and tabulated in Table 21. The results reveal that the proposed ESA gives the best solution with considerable improvement.

Figure 16 shows the convergence analysis of ESA algorithm and reveals that ESA is able to achieve best optimal solution.

5.2 Unconstrained engineering problem

This subsection describes the displacement of loaded structure design problem to minimize the potential energy.

5.2.1 Displacement of loaded structure design problem

A displacement is a vector which defines the shortest distance between initial and final position of a given point.

The objective of this problem is to minimize the potential energy for reducing the excess load of structure. The loaded structure that should have minimum potential energy ($f(\vec{z})$) is shown in Fig. 17. The problem can be stated as follows:

$$f(\vec{z}) = \text{Minimize}_{z_1, z_2}\ \pi$$

where

$$\pi = \frac{1}{2}K_1 u_1^2 + \frac{1}{2}K_2 u_2^2 - F_z z_1 - F_y z_2$$

$$K_1 = 8\,\text{N/cm}, K_2 = 1\,\text{N/cm}, F_y = 5\,\text{N}, F_z = 5\,\text{N}$$

$$u_1 = \sqrt{z_1^2 + (10 - z_2^2) - 10}, \quad u_2 = \sqrt{z_1^2 + (10 + z_2^2) - 10}. \quad (20)$$

Table 22 reveals the comparison of best optimal solution obtained from ESA and other metaheuristics including EPO, SHO, GWO, PSO, MVO, SCA, GSA, and SSA. The proposed ESA generates best optimum cost at $\pi = 167.2635$. It can be seen that ESA is able to minimize the potential energy for loaded structure problem.

The statistical results for the reported algorithms are tabulated in Table 23. From Table 23, it is noticed that the results obtained from ESA are far better than the other competitor

Table 27 Hartman function F_{20}

i	$(a_{ij}, j = 1,2,\ldots,6)$						c_i	$(p_{ij}, j = 1,2,\ldots,6)$					
1	10	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10	17	0.1	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	10	17	8	3	0.2348	0.1415	0.3522	0.2883	0.3047	0.6650
4	17	8	0.05	10	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

algorithms in terms of best, mean, and median. Figure 18 shows the convergence analysis of best solution obtained from proposed ESA algorithm (Tables 24, 25, 26, 27).

In summary, ESA is an effective optimizer for solving both constrained and unconstrained engineering design problems with low computational cost and fast convergence speed.

6 Conclusion and future works

This paper presents a hybrid swarm-based bio-inspired metaheuristic algorithm called emperor penguin and salp swarm algorithm (ESA). The fundamental concepts behind this algorithm are the huddling and swarm behaviors of EPO and SSA algorithms, respectively. The proposed ESA algorithm has been tested on fifty-three benchmark test functions. It is observed from statistical analysis that ESA attains global optimal solution with better convergence as compared to other competitive algorithms.

For CEC-2017 benchmark test functions, the performance of ESA is found accurate and consistent. The effect of scalability has also been investigated on the performance of ESA. The results reveal that the performance of ESA is less susceptible to scalability as compared to other algorithms. The sensitivity analysis has also been investigated on ESA.

Moreover, ESA is applied on six constrained and one unconstrained engineering design problems to show its effectiveness and efficacy. On the basis of results, it can be concluded that the proposed ESA is applicable to engineering design problems. In future, ESA may be extended for solving multi-objective optimization problems. The binary and many objective versions of ESA can be valuable contributions. ESA may also be extended for solving online large scale optimization and engineering applications.

Compliance with ethical standards

Conflict of interest The author declares that he has no conflict of interest.

Appendix: Unimodal, multimodal, and fixed-dimension multimodal benchmark test functions

Unimodal benchmark test functions

Sphere model

$$F_1(z) = \sum_{i=1}^{30} z_i^2$$
$$-100 \le z_i \le 100, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Schwefel's problem 2.22

$$F_2(z) = \sum_{i=1}^{30} |z_i| + \prod_{i=1}^{30} |z_i|$$
$$-10 \le z_i \le 10, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Schwefel's problem 1.2

$$F_3(z) = \sum_{i=1}^{30} \left(\sum_{j=1}^{i} z_j \right)^2$$
$$-100 \le z_i \le 100, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Schwefel's problem 2.21

$$F_4(z) = \max_i \{ |z_i|, 1 \le i \le 30 \}$$
$$-100 \le z_i \le 100, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Generalized Rosenbrock's function

$$F_5(z) = \sum_{i=1}^{29} [100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2]$$
$$-30 \le z_i \le 30, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Step function

$$F_6(z) = \sum_{i=1}^{30} (\lfloor z_i + 0.5 \rfloor)^2$$
$$-100 \le z_i \le 100, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Quartic function

$$F_7(z) = \sum_{i=1}^{30} i z_i^4 + \text{random}[0, 1]$$
$$-1.28 \le z_i \le 1.28, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Multimodal benchmark test functions

Generalized Schwefel's problem 2.26

$$F_8(z) = v_{i=1}^{30} - z_i \sin(\sqrt{|z_i|})$$
$$-500 \le z_i \le 500, \quad f_{\min} = -12569.5, \quad \text{Dim} = 30$$

Generalized Rastrigin's function

$$F_9(z) = \sum_{i=1}^{30} [z_i^2 - 10\cos(2\pi z_i) + 10]$$
$$-5.12 \le z_i \le 5.12, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Table 28 IEEE CEC-2017 benchmark test functions

No.	Functions	$f_{\min}$
C-1	Shifted and rotated bent cigar function	100
C-2	Shifted and rotated sum of different power function	200
C-3	Shifted and rotated Zakharov function	300
C-4	Shifted and rotated Rosenbrock's function	400
C-5	Shifted and rotated Rastrigin's function	500
C-6	Shifted and rotated expanded Scaffer's function	600
C-7	Shifted and rotated Lunacek Bi_Rastrigin function	700
C-8	Shifted and rotated non-continuous Rastrigin's function	800
C-9	Shifted and rotated Levy function	900
C-10	Shifted and rotated Schwefel's function	1000
C-11	Hybrid function1 ($N = 3$)	1100
C-12	Hybrid function2 ($N = 3$)	1200
C-13	Hybrid function3 ($N = 3$)	1300
C-14	Hybrid function4 ($N = 4$)	1400
C-15	Hybrid function5 ($N = 4$)	1500
C-16	Hybrid function6 ($N = 4$)	1600
C-17	Hybrid function6 ($N = 5$)	1700
C-18	Hybrid function6 ($N = 5$)	1800
C-19	Hybrid function6 ($N = 5$)	1900
C-20	Hybrid function6 ($N = 6$)	2000
C-21	Composition function1 ($N = 3$)	2100
C-22	Composition function2 ($N = 3$)	2200
C-23	Composition function3 ($N = 4$)	2300
C-24	Composition function4 ($N = 4$)	2400
C-25	Composition function5 ($N = 5$)	2500
C-26	Composition function6 ($N = 5$)	2600
C-27	Composition function7 ($N = 6$)	2700
C-28	Composition function8 ($N = 6$)	2800
C-29	Composition function9 ($N = 3$)	2900
C-30	Composition function10 ($N = 3$)	3000

Ackley's function

$$F_{10}(z) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} z_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{30} cos(2\pi z_i)\right) + 20 + e$$

$$-32 \le z_i \le 32, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Generalized Griewank function

$$F_{11}(z) = \frac{1}{4000}\sum_{i=1}^{30} z_i^2 - \prod_{i=1}^{30}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1$$

$$-600 \le z_i \le 600, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

Generalized penalized functions

- $$F_{12}(z) = \frac{\pi}{30}\{10\sin(\pi x_1) + \sum_{i=1}^{29}(x_i - 1)^2 \times [1 + 10\sin^2(\pi x_{i+1})] + (x_n - 1)^2\} + \sum_{i=1}^{30} u(z_i, 10, 100, 4)$$

$$-50 \le z_i \le 50, \quad f_{\min} = 0, \quad \text{Dim} = 30$$

- $$F_{13}(z) = 0.1\{\sin^2(3\pi z_1) + \sum_{i=1}^{29}(z_i - 1)^2 \times [1 + sin^2(3\pi z_i + 1)] + (z_n - 1)^2[1 + \sin^2(2\pi z_{30})]\} + \sum_{i=1}^{N} u(z_i, 5, 100, 4)$$

$$-50 \le z_i \le 50, \quad f_{\min} = 0, \quad \text{Dim} = 30,$$

where $x_i = 1 + \frac{z_i + 1}{4}$

$$u(z_i, a, k, m) = \begin{cases} k(z_i - a)^m & z_i > a \\ 0 & -a < z_i < a \\ k(-z_i - a)^m & z_i < -a \end{cases}$$

Fixed-dimension multimodal benchmark test functions

Shekel's Foxholes function

$$F_{14}(z) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(z_i - a_{ij})^6}\right)^{-1}$$

$$-65.536 \le z_i \le 65.536, \quad f_{\min} \approx 1, \quad \text{Dim} = 2$$

Kowalik's function

$$F_{15}(z) = \sum_{i=1}^{11}\left[a_i - \frac{z_1(b_i^2 + b_i z_2)}{b_i^2 + b_i z_3 + z_4}\right]^2$$

$$-5 \le z_i \le 5, \quad f_{\min} \approx 0.0003075, \quad \text{Dim} = 4$$

Six-hump camel-back function

$$F_{16}(z) = 4z_1^2 - 2.1z_1^4 + \frac{1}{3}z_1^6 + z_1z_2 - 4z_2^2 + 4z_2^4$$

$$-5 \le z_i \le 5, \quad f_{\min} = -1.0316285, \quad \text{Dim} = 2$$

Branin function

$$F_{17}(z) = \left(z_2 - \frac{5.1}{4\pi^2}z_1^2 + \frac{5}{\pi}z_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos z_1 + 10$$

$$-5 \le z_1 \le 10, \quad 0 \le z_2 \le 15, \quad f_{\min} = 0.398, \quad \text{Dim} = 2$$

Goldstein–Price function

$$F_{18}(z) = [1 + (z_1 + z_2 + 1)^2(19 - 14z_1 + 3z_1^2 - 14z_2 + 6z_1z_2 + 3z_2^2)] \times [30 + (2z_1 - 3z_2)^2 \times (18 - 32z_1 + 12z_1^2 + 48z_2 - 36z_1z_2 + 27z_2^2)]$$

$$-2 \le z_i \le 2, \quad f_{\min} = 3, \quad \text{Dim} = 2$$

Hartman's family

- $$F_{19}(z) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(z_j - p_{ij})^2\right)$$

  $$0 \le z_j \le 1, \quad f_{\min} = -3.86, \quad \text{Dim} = 3$$

- $$F_{20}(z) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(z_j - p_{ij})^2\right)$$

  $$0 \le z_j \le 1, \quad f_{\min} = -3.32, \quad \text{Dim} = 6$$

Shekel's Foxholes function

- $$F_{21}(z) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$$

  $$0 \le z_i \le 10, \quad f_{\min} = -10.1532, \quad \text{Dim} = 4$$

- $$F_{22}(z) = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$$

  $$0 \le z_i \le 10, \quad f_{\min} = -10.4028, \quad \text{Dim} = 4$$

- $$F_{23}(z) = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$$

  $$0 \le z_i \le 10, \quad f_{min} = -10.536, \quad Dim = 4$$

CEC-2017 benchmark test functions

The detailed descriptions of 15 well-known CEC-2017 benchmark test functions ($C1$–$C30$) are mentioned in Table 28.

References

1. Kaveh A, Shahrouzi M (2007) A hybrid ant strategy and genetic algorithm to tune the population size for efficient structural optimization. Eng Comput 24(3):237–254
2. Kaveh A, Shahrouzi M (2008) Dynamic selective pressure using hybrid evolutionary and ant system strategies for structural optimization. Int J Numer Methods Eng 73(4):544–563
3. Singh P, Rabadiya K, Dhiman G (2018) A four-way decision-making system for the indian summer monsoon rainfall. Mod Phys Lett B 32(25):1850304
4. Singh P, Dhiman G (2018) A hybrid fuzzy time series forecasting model based on granular computing and bio-inspired optimization approaches. J Comput Sci 27:370–385 [Online]. http://www.sciencedirect.com/science/article/pii/S1877750317300923
5. Singh P, Dhiman G, Kaur A (2018) A quantum approach for time series data based on graph and Schrödinger equations methods. Mod Phys Lett A 33(35):1850208
6. Kaur A, Kaur S, Dhiman G (2018) A quantum method for dynamic nonlinear programming technique using Schrödinger equation and Monte Carlo approach. Mod Phys Lett B 1850374
7. Dhiman G, Kaur A (2019) A hybrid algorithm based on particle swarm and spotted hyena optimizer for global optimization. Soft computing for problem solving. Springer, Berlin, pp 599–615
8. Dhiman G, Kumar V (2019) Spotted hyena optimizer for solving complex and non-linear constrained engineering problems. Harmony search and nature inspired optimization algorithms. Springer, Berlin, pp 857–867
9. Kaur A, Dhiman G (2019) A review on search-based tools and techniques to identify bad code smells in object-oriented systems. Harmony search and nature inspired optimization algorithms. Springer, Berlin, pp 909–921
10. Dhiman G, Kaur A (2017) Spotted hyena optimizer for solving engineering design problems. In: Machine learning and data science (MLDS), 2017 international conference on IEEE, pp 114–119
11. Dhiman G, Kumar V (2018) Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. Knowl Based Syst 150:175–197 [Online]. http://www.sciencedirect.com/science/article/pii/S0950705118301357
12. Dhiman G, Kaur A (2018) Optimizing the design of airfoil and optical buffer problems using spotted hyena optimizer. Designs 2(3):28
13. Dhiman G, Kumar V (2018) Knrvea: a hybrid evolutionary algorithm based on knee points and reference vector adaptation strategies for many-objective optimization. Appl Intell 1–27

14. Dhiman G, Guo S, Kaur S (2018) Ed-sho: a framework for solving nonlinear economic load power dispatch problem using spotted hyena optimizer. Mod Phys Lett A 33(40):1850239
15. Dhiman G, Kumar V (2018) Emperor penguin optimizer: a bio-inspired algorithm for engineering problems. Knowl Based Syst 159:20–50
16. Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70
17. Verma S, Kaur S, Dhiman G, Kaur A (2019) Design of a novel energy efficient routing framework for wireless nanosensor networks. In: 2018 first international conference on secure cyber computing and communication (ICSCCC). IEEE, pp 532–536
18. Dhiman G, Singh P, Kaur H, Maini R (2019) DHIMAN: a novel algorithm for economic dispatch problem based on optimization method using Monte Carlo simulation and a strophysics concepts. Mod Phys Lett A 34(04):1950032
19. Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174
20. Singh P, Dhiman G, Guo S, Maini R, Kaur H, Kaur A, Kaur H, Singh J, Singh N (2019) A hybrid fuzzy quantum time series and linear programming model: special application on Taiex index dataset. Mode Phys Lett A 1950201
21. Dhiman G (2019) MOSHEPO: a hybrid multi-objective approach to solve economic load dispatch and micro grid problems. Appl Intell
22. Chandrawat RK, 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. Springer, pp 197–211
23. Singh P, Dhiman G (2017) A fuzzy-LP approach in time series forecasting. In: International conference on pattern recognition and machine intelligence, Springer, pp 243–253
24. Dhiman G, Kumar V (2018) Astrophysics inspired multi-objective approach for automatic clustering and feature selection in real-life environment. Mod Phys Lett B 32(31):1850385
25. Dhiman G, Kumar V (2019) Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. Knowl Based Syst 165:169–196
26. Singh P, Dhiman G (2018) Uncertainty representation using fuzzy-entropy approach: special application in remotely sensed high-resolution satellite images (RSHRSIs). Appl Soft Comput 72:121–139 [Online]. http://www.sciencedirect.com/science/article/pii/S1568494618304265
27. Kaveh A, Rad SM (2010) Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design. Iran J Sci Technol 34(B1):15
28. Kaveh A, Zolghadr A (2012) Truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability. Comput Struct 102:14–27
29. Kaveh A, Javadi SM (2014) An efficient hybrid particle swarm strategy, ray optimizer, and harmony search algorithm for optimal design of truss structures. Period Polytech Civ Eng 58(2):155–171
30. Dhiman G, Kumar V (2018) Emperor penguin optimizer: a bio-inspired algorithm for engineering problems. Knowl Based Syst 159:20–50 [Online]. http://www.sciencedirect.com/science/article/pii/S095070511830296X
31. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
32. Waters A, Blanchette F, Kim AD (2012) Modeling huddling penguins. PLoS One 7(11):e50277
33. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–72
34. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359. https://doi.org/10.1023/A:1008202821328 **[Online]**
35. Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, New York
36. Beyer H-G, Schwefel H-P (2002) Evolution strategies—a comprehensive introduction. Nat Comput 1(1):3–52. https://doi.org/10.1023/A:1015059928466 **[Online]**
37. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
38. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
39. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248. http://www.sciencedirect.com/science/article/pii/S0020025509001200 **[Online]**
40. Erol OK, Eksin I (2006) A new optimization method: Big bang-big crunch. Adv Eng Softw 37(2):106–111. http://www.sciencedirect.com/science/article/pii/S0965997805000827 **[Online]**
41. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3–4):267–289
42. Hatamlou A (2013) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175–184. http://www.sciencedirect.com/science/article/pii/S0020025512005762 **[Online]**
43. Formato RA (2009) Central force optimization: a new deterministic gradient-like optimization metaheuristic. Opsearch 46(1):25–51. https://doi.org/10.1007/s12597-009-0003-4 **[Online]**
44. Du H, Wu X, Zhuang J (2006) Small-world optimization algorithm for function optimization. Springer, Berlin, pp 264–273
45. Alatas B (2011) ACROA: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 38(10):13 170–13 180. http://www.sciencedirect.com/science/article/pii/S0957417411006531 **[Online]**
46. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294
47. Shah Hosseini H (2011) Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation. Int J Comput Sci Eng 6:132–140
48. Moghaddam FF, Moghaddam RF, Cheriet M (2012) Curved space optimization: a random search based on general relativity theory. Neural Evol Comput
49. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, pp 1942–1948
50. Slowik A, Kwasnicka H (2017) Nature inspired methods and their industry applications–swarm intelligence algorithms. IEEE Trans Ind Inf 99:1–1
51. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization—artificial ants as a computational intelligence technique. IEEE Comput Intell Mag 1:28–39
52. Yang X-S (2010) A new metaheuristic bat-inspired algorithm. Springer, Berlin, pp 65–74
53. Karaboga D, Basturk B (2007) Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems. Springer, Berlin, pp 789–798
54. Yang XS, Deb S (2009) Cuckoo search via levy flights. In: World congress on nature biologically inspired computing, pp 210–214
55. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61. http://www.sciencedirect.com/science/article/pii/S0965997813001853 **[Online]**
56. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513. https://doi.org/10.1007/s00521-015-1870-7 **[Online]**

57. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133. http://www.sciencedirect.com/science/article/pii/S0950705115005043 **[Online]**
58. Tan Y, Zhu Y (2010) Fireworks algorithm for optimization. In: International conference in swarm intelligence. Springer, pp 355–364
59. Zheng S, Janecek A, Tan Y (2013) Enhanced fireworks algorithm. In: Evolutionary computation (CEC), 2013 IEEE congress on IEEE, pp 2069–2077
60. Ding K, Zheng S, Tan Y (2013) A GPU-based parallel fireworks algorithm for optimization. In: Proceedings of the 15th annual conference on genetic and evolutionary computation. ACM, pp 9–16
61. Zheng S, Janecek A, Li J, Tan Y (2014) Dynamic search in fireworks algorithm. In: Evolutionary computation (CEC), 2014 IEEE congress on IEEE, pp 3222–3229
62. Mucherino A, Seref O (2007) Monkey search: a novel metaheuristic search for global optimization. AIP conference proceedings 953(1)
63. Das S, Biswas A, Dasgupta S, Abraham A (2009) Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. Springer, Berlin, pp 23–55
64. Yang X-S (2010) Firefly algorithm, stochastic test functions and design optimisation. Int J Bio-Inspired Comput 2(2):78–84. https://doi.org/10.1504/IJBIC.2010.032124
65. Pan W-T (2012) A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowl Based Syst 26:69–74
66. Wang Y, Wu S, Li D, Mehrabi S, Liu H (2016) A part-of-speech term weighting scheme for biomedical information retrieval. J Biomed Inf 63:379–389. http://www.sciencedirect.com/science/article/pii/S1532046416301125 **[Online]**
67. Orozco-Henao C, Bretas A, Chouhy-Leborgne R, Herrera-Orozco A, Marin-Quintero J (2017) Active distribution network fault location methodology: a minimum fault reactance and fibonacci search approach. Int J Electr Power Energy Syst 84:232–241. http://www.sciencedirect.com/science/article/pii/S0142061516302307 **[Online]**
68. Askarzadeh A (2014) Bird mating optimizer: an optimization algorithm inspired by bird mating strategies. Commun Nonlinear Sci Numer Simul 19(4):1213–1228
69. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845
70. Neshat M, Sepidnam G, Sargolzaei M, Toosi AN (2014) Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications. Artif Intell Rev 42(4):965–997
71. Shiqin Y, Jianjun J, Guangxing Y (2009) A dolphin partner optimization. In: Proceedings of the WRI global congress on intelligent systems, pp 124–128
72. Lu X, Zhou Y (2008) A novel global convergence algorithm: bee collecting pollen algorithm. In: 4th international conference on intelligent computing, Springer, pp 518–525
73. Oftadeh R, Mahjoob M, Shariatpanahi M (2010) A novel meta-heuristic optimization algorithm inspired by group hunting of animals: hunting search. Comput Math Appl 60(7):2087–2098. http://www.sciencedirect.com/science/article/pii/S0898122110005419 **[Online]**
74. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
75. Digalakis J, Margaritis K (2001) On benchmarking functions for genetic algorithms. Int J Comput Math 77(4):481–506
76. Awad N, Ali M, Liang J, Qu B, Suganthan P (2016) Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective bound constrained real-parameter numerical optimization. In: Technical report, Nanyang Technological University Singapore
77. Brest J, Maučec MS, Bošković B (2017) Single objective real-parameter optimization: algorithm JSO. In: Evolutionary computation (CEC), 2017 IEEE congress on IEEE, pp 1311–1318
78. Kaveh A (2014) Advances in metaheuristic algorithms for optimal design of structures. Springer, Berlin
79. Kaveh A, Ghazaan MI (2018) Meta-heuristic algorithms for optimal design of real-size structures. Springer, Berlin
80. Coello CAC (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11–12):1245–1287. http://www.sciencedirect.com/science/article/pii/S0045782501003231 **[Online]**
81. Kannan B, Kramer SN (1994) An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J Mech Des 116(2):405–411
82. Gandomi AH, Yang X-S (2011) Benchmark problems in structural optimization. Springer, Berlin, pp 259–281
83. Mezura-Montes E, Coello CAC (2005) Useful infeasible solutions in engineering optimization with evolutionary algorithms. Springer, Berlin, pp 652–662
84. Kaveh A, Talatahari S (2009) A particle swarm ant colony optimization for truss structures with discrete variables. J Construct Steel Res 65(8–9):1558–1568
85. Kaveh A, Talatahari S (2009) Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. Comput Struct 87(5–6):267–283
86. Bichon CVCBJ (2004) Design of space trusses using ant colony optimization. J Struct Eng 130(5):741–751
87. Schutte J, Groenwold A (2003) Sizing design of truss structures using particle swarms. Struct Multidiscip Optim 25(4):261–269. https://doi.org/10.1007/s00158-003-0316-5 **[Online]**
88. Kaveh A, Talatahari S (2010) Optimal design of skeletal structures via the charged system search algorithm. Struct Multidiscip Optim 41(6):893–911
89. Kaveh A, Talatahari S (2009) Size optimization of space trusses using big bang-big crunch algorithm. Comput Struct 87(17–18):1129–1140