Engineering with Computers
https://doi.org/10.1007/s00366-020-01067-y

ORIGINAL ARTICLE

Selection scheme sensitivity for a hybrid Salp Swarm Algorithm: analysis and applications

Laith Abualigah[1] · Mohammad Shehab[2] · Ali Diabat[3,4] · Ajith Abraham[5,6]

Received: 4 November 2019 / Accepted: 19 May 2020

Abstract

This paper proposes a hybrid version of the Salp Swarm Algorithm (SSA) and the hill climbing (HC) technique using various selection schemes to solve engineering design problems. The proposed algorithm consists of two stages. In the first stage, the basic SSA is hybridized with HC local search to improve its exploitation capabilities; we refer to the hybridized algorithm as HSSA. In the second stage, a selection scheme is applied to enhance the exploration capabilities of the hybrid SSA. Six popular selection schemes were investigated, and the proportional selection scheme was selected as it yielded the best performance. We refer to the hybridized SSA along with the proportional selection scheme as PHSSA. To validate the performance of the proposed algorithms, a series of experiments were conducted using thirty benchmark functions and four engineering design problems. The investigations using benchmark functions revealed that HSSA overcame the weaknesses of the local search in the basic SSA. Moreover, PHSSA enhanced performance by providing an appropriate balance between exploration and exploitation as well as maintaining the diversity of the solutions and avoiding premature convergence. Finally, PHSSA produced results on engineering design problems that were at least comparable and in many cases superior to SSA and similar algorithms in the literature.

Keywords Salp Swarm Algorithm · Hill climbing · Selection schemes · Hybridization · Meta-heuristic algorithms · Optimization problems

1 Introduction

Optimization problems have garnered increased attention in the last years because of their different and complex nature; these problems have been observed in several areas such as computer science and engineering applications and even in finance applications and decision-making [1, 2]. Numerous various meta-heuristic optimization algorithms (MOAs) have been introduced in the literature and employed to solve different kinds of problems (i.e., optimization and real-world problems). MOAs are designated by their nature and characterized by their capability in solving optimization problems at low computational costs [3]. In the real-world applications, decision-makers seek for intelligent methods that can support their necessary works by providing optimal decisions. Usually, any optimization problem requires optimal decision-making to be done efficiently by either determining the minimum cost value or even determining the maximum profit ratio [4].

MOAs are common techniques that tackle various kinds of optimization problems. They can transact with any optimization problems using several learning parameters based on the natural selection methods controlled by special coefficients to explore a wide-search space (global search) and to exploit an intensive-search space (local search) [5]. Based on these facts, many stochastic search methods such as Markov chains, swap mutation, differential evolution, Gaussian random number, levy flights, hill-climbing, and other different

✉ Laith Abualigah
laythdyabat@aau.edu.jo

1 Faculty of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan

2 Computer Science Department, Aqaba University of Technology, Aqaba 77110, Jordan

3 Division of Engineering, New York University Abu Dhabi, Saadiyat Island, Abu Dhabi 129188, United Arab Emirates

4 Department of Civil and Urban Engineering, Tandon School of Engineering, New York University, Brooklyn, NY 11201, USA

5 Department of Computer Science, University of Pretoria, Pretoria 0002, South Africa

6 Machine Intelligence Research Labs (MIR Labs), 3rd Street NW, P.O. Box 2259, Auburn, WA 98071, USA

Published online: 09 July 2020

algorithms components have been engaged in a new version of the MOA to enhance its performance. MOAs are grouped into Evolutionary-Based Optimization Algorithms (EAs), Swarm-Based Optimization Algorithms (SAs), and Trajectory-Based Optimization Algorithms (TAs) [6]. There are several new swarm-intelligence-based MOAs in the domain such as, but not limited to, Salp Swarm Algorithm (SSA) [7], Ant Lion Optimizer [8], Moth-Flame Optimization Algorithm [9], and Sine Cosine Algorithm (SCA) [10].

SSA is a novel swarm optimization algorithm introduced in 2017 for solving several optimization problems with a single objective and multiple objectives [7]. The main motivation of SSA is the swarming habits of salps while traveling and looking for supplies in oceans. This algorithm is experimented on various mathematical optimization functions to prove their effective behaviors in determining the optimal solution for optimization problems. Yıldız and Yıldız [11] introduced several MOAs including SSA to address shape problems in the automotive industry. Abbassi et al. [12] introduced an efficient method based on SSA for extracting the parameters of the electrical comparable line of PV cell-based double-diode design. Qais et al. [13] introduced a novel modification of the SSA tested by several common benchmark functions to validate its performance.

Singh et al. [14] proposed a new hybrid method based on using the Salp Swarm Algorithm with Sine Cosine Algorithm, called HSSASCA. The proposed method is found to improve the convergence execution with the exploration and exploitation of search strategies. In the proposed method (HSSASCA), the position of SSA in the available search space is renewed utilizing the position locations of SCA; therefore, the optimal solutions are taken based on the sine or cosine use. Rizk-Allah et al. [15] introduced a new version of the SSA based on a modified Arctan transformation. This version has two characteristics regarding the transfer function, namely multiplicity and mobility. By this version, the exploration and exploitation abilities are improved. The proposed algorithm acquired better performance compared with variants of transfer functions for determining global optimization problems. Kaur et al. [16] introduced a new multi-objective optimization method based on using Spotted Hyena Optimizer (MOSHO) and Salp Swarm Algorithm (SSA), called HMOSHSSA. The proposed algorithm used the exploration ability of the basic MOSHO to examine the available search space thoroughly, and the follower selection approach of the basic SSA is used to obtain the global best solution, which can make the convergence faster. Twenty-four benchmark test functions are utilized to evaluate the performance of the proposed HMOSHSSA. The obtained results are compared to the other seven multi-objective algorithms. The results proved that the proposed HMOSHSSA acquires promising results and beats other methods. Moreover, the results showed that the performance of proposed HMOSHSSA toward addressing real-life multi-objective optimization problems is better than similar competitor methods.

In the literature, most MOAs stuck in local exploration and suffer from slow convergence speed [17, 18]. Abualigah et al. [19, 20] introduced a novel hybrid clustering method based on multi-objective Krill Herd Algorithm with a local search to address the clustering problem. As well, in [21] introduced hybrid Krill Herd Algorithm with the k-means clustering. Several different datasets taken from the Laboratory of Computational Intelligence are used to evaluate the production of the proposed algorithms. The obtained results revealed that the proposed methods in both studies acquired better performance compared with other optimization methods. Bairathi and Gopalani [22] introduced a new enhanced method using opposition-based SSA. To enhance the achievement of SSA, opposition-based learning is proposed in SSA. The introduced algorithm is assessed on various numerical functions and is compared with similar optimization algorithms. Moghdani et al. [23] proposed a modified algorithm based on using volleyball premier league (VPL) and Sine Cosine Algorithm (SCA), called VPLSCA. The SCA operators are incorporated into the general procedure of the VPL to take the advantages of both search paradigms and tackle the limitations of the conventional VPL algorithm. The twenty-five benchmark functions and three optimization problems are used to validate the performance of the proposed algorithm. The obtained results demonstrated that the proposed VPLSCA achieved very reasonable and promising outcomes compared to other similar methods published in the literature.

Several aspects make global optimization problems hard to solve: the area of the search regions increases exponentially as dimensionality increases, which is the main problem that encountered the optimization methods [24]. However, there is no theory to determine which algorithms can gain the best performance; indeed, combined algorithms (i.e., hybridized, modified, etc.) enhance the performance of the basic algorithms [25]. SSA has shown superior performance for solving several problems with small or even medium dimensionality. However, two main weaknesses are recognized in the performance trajectory of the basic version of the SSA: loss of the solutions' diversity, which produces tenacious premature convergence and slow convergence manner. Because of these weaknesses, SSA requires further refinements, to be modified or hybridized with other algorithms components or local search techniques, to avert the early convergence (premature) for enhancing its performance. As a result, an improved method, using several promising selection schemes for the hybridization of SSA and hill-climbing (HC) search strategy called HSSA, is proposed and investigated using global optimization problems.

In this paper, the main contributions are summarized as follows:

- A new hybridization method using SSA and HC (HSSA) is developed to improve the exploitation search.
- Alternative selection methods in the hybrid SSA for global optimization problems are investigated to maintain the diversity of the solutions.
- The performance of the proposed algorithms is tested on several global optimization problems.
- Experimental results proved that the proposed HSSA with the proportional selection scheme robustly solved the problems and produced superior performance.

The rest of this paper is organized as follows: Sect. 2 explains the optimization problem, SSA, HC, and selection schemes. Section 3 shows the general structure of the proposed algorithm. Section 4 shows experimental results and discussions. Finally, conclusions and future directions are presented in Sect. 5.

2 Materials and methods

In this section, the optimization problem, SSA, HC, and selection schemes are explained in the following subsections.

2.1 Problem definition

Generally, there is a mathematical representation for any optimization problem, which is presented in the following equations [26, 27]. This presentation can be utilized to form the most optimization problems in a general design as follows:

$$\min f(x), \text{where} \quad x = \{1, 2, \ldots, k\} \in S \tag{1}$$

$$\text{subject to}: g_i(x) > 0, j = (1, 2, \ldots, J) \tag{2}$$

$$h_k(x) = 0, k = (1, 2, \ldots, K) \tag{3}$$

$$L_i \leq x_i \leq U_i, i = (1, 2, \ldots, n) \tag{4}$$

where S denotes the solution space; $g_i(x)$ and $h_k(x)$ denote the difference constraints and balance constraints. J denotes the number of difference constraints, K denotes the number of balance constraints, and $[L_i, U_i]$ denotes the lower (L_i) and upper (U_i) boundaries of the ith variable.

If the problem presentation was without constraints (i.e., $J = 0$ and $K = 0$), then the optimization is supposed to be an unconstrained optimization problem, when $J > 0$ or $K > 0$, the problem is supposed to be a constrained optimization problem.

2.2 The conventional salp swam algorithm

This section presents the SSA and describes its main elements. It also includes explanations on the convergence, exploitation, and exploration of this algorithm.

2.2.1 Inspiration of SSA

Over 1.2 million species of marine organisms are already cataloged in a central database [28]. Most of these species have the same behaviors and features, such as communicating methods, locomotor performance, and looking for food. Salp is a kind of marine organism which belongs to the family of Salpidae. Its shape is highly similar to jellyfishes, cylindrical shape with openings at the end which pump water through their gelatinous bodies to move and feed by internal feeding filters. Figure 1a shows the shape of a salp.

As mentioned above, the marine organisms share some behavior such as swarming behavior, for example, the school of fish [29], while for salps it is called a salp chain (see Fig. 1b). Although their living environments are extremely difficult to access, the biological researchers believe that the reasons for their behavior is to help the salps achieve better locomotion and foraging.

2.2.2 The procedure of basic Salp Swarm Algorithm

Salp Swarm Algorithm (SSA) is a population-based optimization method proposed by Mirjalili et al. [7]. The behavior of the SSA can be explored by computing it to the salp chain searching for optimal food sources (i.e., the target of this swarm is a food source in the search space called **F**). In SSA, according to the individuals' (i.e., salps) positions in the chain, they are divided into either leaders or followers.

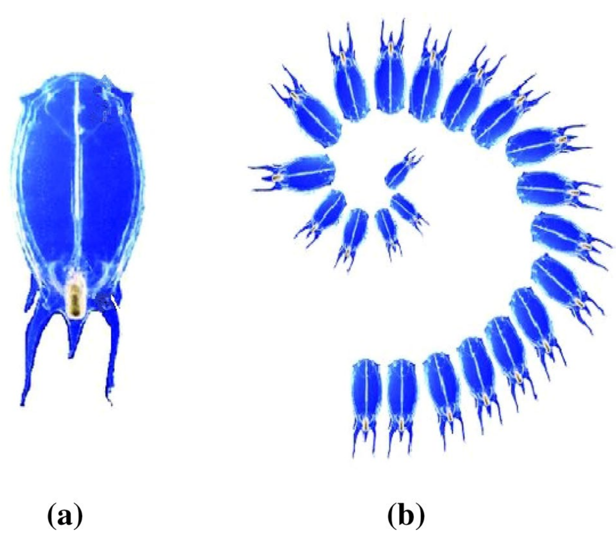


Fig. 1 **a** Individual salp, **b** swarm of salps (salps chain)

The chain is started with a leader and the followers follow it to guide them in their movements [30].

Algorithm 1 shows the pseudocode of SSA, where it can be noted the simplicity of SSA and its similarity to other swarm intelligent algorithms. Where it starts by initializing the salp population, the swarm X of n salps is represented in Eq. (5) as two-dimensional matrix. Then, it calculates the fitness of each salp to determine the salp with the best fitness (i.e., leader). The leader position is updated using Eq. (6).

$$X_i = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_d^1 \\ x_1^2 & x_2^2 & \dots & x_d^2 \\ \vdots & \vdots & \dots & \\ x_1^n & x_2^n & \dots & x_d^n \end{bmatrix} \tag{5}$$

$$x_i^1 = \begin{cases} y_i + r_1((ub_i - lb_i)r_2 + lb_i) & r_3 \geq 0 \\ y_i + r_1((ub_i - lb_i)r_2 + lb_i) & r_3 < 0 \end{cases} \tag{6}$$

where the x_i^1 is the position of the first salp in the ith dimension, and y_i is the food position in the ith dimension. lb_i and ub_i represent the lower bound and the upper bound of the ith dimension, and the coefficient r_1 is calculated by Eq. (7), r_2, r_3 are random numbers between [0,1].

$$r_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \tag{7}$$

where L is the maximum iterations, l is the current iteration. It is worth to mention that the coefficient r_1 is very important in SSA because it balances exploration and exploitation during the entire search process. Regarding the followers, Eq. (8) shows the update of their positions:

$$x_i^j = \frac{1}{2}\lambda t^2 + \delta_0 t \tag{8}$$

where $j \geq 2$, x_i^j refers to the position of the jth salp in the ith dimension, δ_0 is an initial speed t is the time, $\lambda = \frac{\delta_{\text{final}}}{\delta_0}$, where $\delta = \frac{x - x_0}{t}$. In Optimization, the time indicates the iteration. So, the discrepancy between iterations is equal to 1. Considering the assumption that $\delta_0 = 0$, the following equation is employed for this issue.

$$x_i^j = \frac{1}{2}(x_i^j + x_i^{j-1}) \tag{9}$$

where $j \geq 2$. In case some salps move outside of the search space, Eq. (10) illustrates how to bring them back to the search space.

$$x_i^j = \begin{cases} l^j & \text{if } x_i^j \leq l^j \\ u^j & \text{if } x_i^j \geq u^j \\ x_i^j & \text{otherwise} \end{cases}. \tag{10}$$

Algorithm 1 Salp Swarm Algorithm

```
Require: Initialize the salp population x_i(i = 1, 2, ..., n) consider ub and lb.
while (End condition is not satisfied) do
   Calculate the fitness of each search salp
   F=the best search solution
   Update r_1 by Equation (7)
   for (each salp (x_i)) do
      if (i==1) then
         Update the position of the leading salp by Equation (6)
      else
         Update the position of the followers salp by Equation (9)
      end if
   end for
   Verify the position of salps based on the upper and lower bounds
end while
return F
```

2.2.3 Exploration and exploitation in SSA

Exploration and exploitation, global search and local search, and diversification and intensification [31]. These word pairings are very common in the optimization algorithms, and there is no algorithm that doesn't contain at least one of these pairs [32]. In general, the exploration aims to discover the promising areas of the search landscape, prevent solutions from stagnating in a local optimum, and maximize the probability of detecting the global optimum, while the exploitation aims to obtain even better solutions from good ones through discovering the neighborhood of each solution [32].

In SSA, the responsible parameter of balance between exploration and exploitation is called coefficient r_1 which has been calculated in Eq. (7). However, SSA suffers from a problem in exploitation which leads to the slow convergence rate [33, 34].

2.3 Hill climbing: local search technique

The hill climbing (HC) technique, called local search, is the most simplistic form of local development methods. It begins with one random initial solution (x), iteratively proceeds by moving from the current solution to a better neighboring solution till it reaches a local optimum (i.e., the local optimal solution does not have a better neighboring solution, no improvement in fitness function). It only takes downhill progress where the fitness function of a neighboring solution should be better than the current solution. Consequently, it can converge to the local optima fast and suddenly. However, it can quickly get stuck in local optima, which in most situations is not satisfactory. Algorithm 2 presents the pseudocode of the HC technique. After creating the first solution x and through the iterative improvement process, a group of neighboring solutions is created utilizing the procedure $Improve(N(x))$. This procedure seeks to discover the enhanced neighboring solution from the group of the neighbors utilizing any used acceptance rule such as first improvement, best improvement, sidewalk, and random walk. But, all of these rules are stopped in local optima.

Algorithm 2 Hill climbing technique

```
1: The initial solution x
2: x_i=LB_i+ (UB_i-LB_i) * U(0,1), ∀_i=(1, 2, ...., N)
3: Calculate fitness function F(x)
4: while (End condition is not satisfied) do
5:     x'=Improve((N(x))
6:     if F(x') ≤ F(x) then
7:         x=x'
8:     end if
9: end while
10: return x
```

2.4 Selection schemes

In this section, the selection schemes that are described that used in this paper.

2.4.1 Tournament selection scheme

Tournament selection is among the most popular selection methods in Genetic Algorithms. It was initially proposed by Grefenstette and Baker [35]. Algorithm 3 shows the principle of tournament selection work, which starts from the random selection of t individuals from $P(t)$ population and then proceeds to the selection of the best individual from tournament t. This procedure is repeated N times. The best choice is frequently between two individuals, and this scheme is called binary tournament, where the choice is between t individuals called tournament size [36]. In other words, the efficiency of tournament selection scheme based on the value of t. For instance, increasing the value of t will increase the diversity which leads to an increase in the quality of the selected solution, and vice versa [37].

Algorithm 3 Tournament Selection Scheme

Input: The population $P(_T)$ the tournament size $t \in i1, 2, ..., N$
Output: The population after selection $P(_T)'$

Description:
tournament $(J_1, ..., J_N)$:
for $i \leftarrow 1$ *to* N *do*
$J'_i \leftarrow$ best fit individual out of t randomly piked individuals from $\{J_1, ..., J_N\}$;
endfor
return $\{J'_1, ..., J'_N\}$

There are several merits of the tournament selection scheme. For instance, low susceptibility to a takeover by dominant individuals [38], it has efficient time complexity [39], and no requirement for fitness scaling or sorting [37].

2.4.2 Proportional selection scheme

The proportional selection scheme or so-called Roulette wheel has been proposed in [40]. In other words, each element reserves a section in the roulette wheel, where the section's size is proportional with the element's fitness. The mechanism of this method is choosing the probability based on the comparison between the fitness values of any solution and the fitness value of the stored solution in SSA. As shown in Algorithm 4, r has been selected from $U(0,1)$. Then, s_i has accumulative determined the probabilities, and the following equation shows the probability of solution x.

$$P_i = \frac{F(x_i)}{\sum_{j=1}^{\text{swarm size}} f(x_j)} \tag{11}$$

Algorithm 4 Proportional Selection Scheme

Input: The population $P(_T)$, $r \in U(0,1)$
Output: The population after selection $P(_T)'$

Description:

proportional $(J_1, ..., J_N)$:
$s_0 \leftarrow 0$
for $i \leftarrow 1$ *to* N *do*
  $s_i \leftarrow s_{i-1} + P_i$
endfor
for $i \leftarrow 1$ *to* N *do*
  $r \leftarrow random\,[0, s_N]$
  $J_i' \leftarrow J_i such that s_{i-1} \leqslant r < s_i$
endfor
return $\{J_1', ..., J_N'\}$

The advantage of proportional selection is that it offers a chance for each element to be chosen. In contrast, in population converges, it suffers from selection pressure [39].

2.4.3 Linear ranking selection scheme

To overcome the limitation of the proportional selection scheme, Grefenstette and Baker proposed linear ranking selection scheme [35]. It arranges the solutions based on their fitness ranks. Equation (12) shows the mechanism of calculation and the selection probability by linear mapping of the solution ranks.

$$P_i = \frac{1}{N} \times \left(\eta^+ - \left(\eta^+ - \eta^-\right) \times \frac{i-1}{N-1}\right), \quad i \in 1, \ldots, N \tag{12}$$

where i is the rank of solution location x^j, η^- is the expected value of the worst location, η^+ is the expected value of the best location. Both of η^- and η^+ set the slope of the linear function. More details are shown in Algorithm 5.

Algorithm 5 Linear Ranking Selection Scheme

Input: The population $P(_T)$ and the reproduction rate of the worst individual $\eta- \in [1, 0]$
Output: The population after selection $P(_T)'$

Description:

$linear\_ranking(J_1, ..., J_N)$:
$\bar{J} \leftarrow sorted\ population\ \ J\ \ according\ fitness\ with\ worst\ individual\ at\ the\ first\ position$
$s_0 \leftarrow 0$
for $i \leftarrow 1$ *to* N *do*
  $s_i \leftarrow s_{i-1} + p_i$ $(Equation\ 12)$
endfor
for $i \leftarrow 1$ *to* N *do*
  $r \leftarrow random\,[0, s_N]$
  $J_i' \leftarrow \bar{J}_i such that s_{i-1} \leqslant r < s_i$
endfor
return $\{J_1', ..., J_N'\}$

The expected results of the linear ranking selection scheme with small η^+ are close to the binary tournament selection. However, the linear ranking selection scheme with big η^+ suffers from a stronger selection pressure (i.e., more efficient time complexity) [41].

2.4.4 Exponential ranking selection scheme

Unlike the linear ranking scheme, the exponential ranking selection arranges the probabilities of the ranked elements by exponentially weighted. The major of the exponent c which is situated between (0, 1), where it based on parameter s. For instance, the best solution has a value of $c_1 = 1$, followed by the second solution with $c_2 = s$ ($s = 0.99$), the third solution has $c_3 = s^2$, and so on until the worst solution has $c_{\text{swarm size}} = s^{\text{swarm size}-1}$ [42]. Probabilities of the individuals are calculated by

$$p_i = \frac{c^{N-i}}{\sum_{j=1}^{N} c^{N-j}} \quad i \in \{1, 2, \dots, N\} \tag{13}$$

The $\sum_{j-1}^{N} c^{N-j}$ normalizes the probabilities to ensure that $\sum_{i=1}^{N} c^{N-j} p_i = 1$. As $\sum_{j-1}^{N} c^{N-j} = \frac{c^N-1}{C-1}$ it will be as the following equation:

$$p_i = \frac{c-1}{c^N - 1} C^{N-i} \quad i \in \{1, 2, \dots, N\} \tag{14}$$

Algorithm 6 illustrates the exponential ranking selection algorithm; the similarity of structure between linear ranking selection and exponential ranking selection can be noticed, while the difference lies in the calculation of the selection probabilities.

Algorithm 6 Exponential Ranking Selection Scheme

Input: The population $P(_T)$ and the ranking base $c \in [1, 0]$
Output: The population after selection $P(_T)'$

Description:

$exponential\_\ ranking(c, J_1, ..., J_N)$:
$\bar{J} \leftarrow sorted\ population\ \ J\ \ according\ fitness\ with\ worst\ individual\ at\ the\ first\ position$
$s_0 \leftarrow 0$
for $i \leftarrow 1\ to\ N\ do$
  $s_i \leftarrow s_{i-1} + p_i$ $(Equation\ 3.6)$
endfor
for $i \leftarrow 1\ to\ N\ do$
  $r \leftarrow random\,[0, s_N]$
  $J_i' \leftarrow \bar{J}_i such that s_{i-1} \leqslant r < s_i$
endfor
return $\{J_1', ..., J_N'\}$

2.4.5 Greedy-based selection scheme

The greedy selection scheme is called global best which was initially applied by Kennedy in PSO [43]. The technicality of greedy selection focuses to choose the three best solutions: x_α, x_β, and x_γ to avoid the local optima. Algorithm 7 shows the pseudocode of the greedy selection scheme.

Algorithm 7 Greedy-based Selection Scheme

Input: The population $P(_T)$
Output: The population after selection $P(_T)'$

Description:
for $j \leftarrow 1$ *to* $|J|$ *do*
for $t \leftarrow 1$ *to* $|T|$ *do*
$w_{jt} \leftarrow 0$
endfor
endfor
for $i \leftarrow 1$ *to* $|I|$ *do*
Obtain a new patrol using DP and let $a^* = a^*_{jt}$ be the obtained optimal patrol
Assign patrol a^* to team i
foreach (j, t) *with* $a^* = a^*_{jt} = 1$ *do*
$w_{jt} \leftarrow 1$
endfor
endfor
return $\{J'_1, ..., J'_N\}$

As mentioned above, the greedy scheme chooses the best three solutions and ignores the other solutions. Therefore, the diversity of the search space might be lost which leads to prematurely converge and quickly stagnate without efficient results.

2.4.6 Truncation selection scheme

Truncation selection is considered the simplest selection scheme compared to other selection schemes. The truncation chooses elements and saves a certain percentage until reaches the population size [44]. This selection is equal to (μ, λ)-selection utilized in development strategies with $T = \frac{\mu}{\lambda}$ [45].

Table 1 Description of unimodal benchmark functions

No.	Function	Equation	Range	$f_{\min}$
f1	Beale	$f_1(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	$[-4.5, 4.5]$	0
f2	Watson	$f_2(x) = \sum_{i=0}^{29} \left\{ \sum_{j=0}^{4}((j-1)\alpha_i^j x_{j+1}) - \left[\sum_{j=0}^{5} \alpha_i^j x_{j+1}\right]^2 - 1 \right\}^2 + x_1^2$	$[-5, 5]$	0.002288
f3	Dixon and Price	$f_3(x) = (x_1 - 1)^2 + \sum_{i=2}^{d} i(2x_i^2 - x_{i-1})^2$	$[-10, 10]$	0
f4	Quartic with noise	$f_4(x) = \sum_{i=1}^{30} ix^4 + random[0, 1)$	$[-1.28, 1.28]$	0
f5	Schwefel 1.2	$f_5(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]$	0
f6	Schwefel 2.22	$f_6(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-100, 100]$	0
f7	Schwefel 2.21	$f_7(x) = \sum_{i=1}^{n} \lvert x_i \rvert$	$[-100, 100]$	0
f8	Sphere	$f_8(x) = \sum_{i=1}^{d} x_i^2$	$[-5.12, 5.12]$	0
f9	Step	$f_9(x) = \sum_{i=1}^{n} \lfloor x_i^2 \rfloor$	$[-100, 100]$	0
f10	Zakharov	$f_{10}(x) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^4$	$[-5, 10]$	0

Table 2 Description of multimodal benchmark functions

No.	Function	Equation	Range	$f_{\min}$
f11	Easom	$f_{11}(x) = -\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2-(x_{2-\pi})^2)$	$[-100, 100]$	0
f12	Shubert	$f_{12}(x) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2+i)\right)$	$[-10, 10]$	− 186.7309
f13	Wolfe	$f_{13}(x) = \frac{3}{4}(x_1^2+x_2^2-x_1\cdot x_2)^{0.75}+x_3$	$[0, 2]$	0
f14	Colville	$f_{14}(x) = 100(x_1^2-x_2)^2+(x_1-1)^2+(x_3-1)^2+90(x_3^2-x_4)+10.1((x_2-1)^2)+19.8(x_2-1)(x_4-1)$	$[-10, 10]$	0
f15	Ackley	$f_{15}(x) = -\alpha\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right)+\alpha+\exp(1))^{**}$	$[-32.768, 32.768]$	0
f16	Griewank	$f_{16}(x) = \sum_{i=1}^{d}\frac{x_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$[-600, 600]$	0
f17	Levy	$f_{17}(x) = sin^2(\pi w_1)+\sum_{i=1}^{d-1}(w_i-1)^2[1+10\sin^2(\pi w_i+1)]+(w_d-1)^2[1+sin^2(2\pi w_d)]^{**}$	$[-10, 10]$	0
f18	Perm	$f_{18}(x) = \sum_{i=1}^{d}\left(\sum_{j=1}^{d}(j+\beta)\left(x_j^i-\frac{1}{j^i}\right)\right)^2$	$[-d, d]$	0
f19	Rastrigin	$f_{19}(x) = 10d+\sum_{i=1}^{d}\left[x_i^2-10\cos(2\pi x_i)\right]$	$[-5.12, 5.12]$	0
f20	Rosenbrock	$f_{20}(x) = \sum_{i=1}^{d-1}\left[100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right]$	$[-5, 10]$	0
f21	Egg Holder	$f_{21}(x) = -(x_2+47)sin\left(\sqrt{\left\|x_2+\frac{x_1}{2}+47\right\|}\right)-x_1 sin\left(\sqrt{\left\|x_1-(x_2+47)\right\|}\right)$	$[-5.12, 5.12]$	− 959.6407
f22	Michalewicz	$f_{22}(x) = -\sum_{i=1}^{d} sin(x_i)sin^{2m}\left(\frac{ix_i^2}{\pi}\right), m = 10$	$[0,\pi]$	− 1.8013

**In f_{15}, $\alpha = 20$, $b = 0.2$, and $c = 2\pi$

In f_{17}, $w_i = 1+\frac{x_i-1}{4}$

Table 3 Description of fixed-dimension multimodal benchmark functions

No.	Function	Equation	Range	$f_{\min}$
f23	Branin	$f_{23}(x) = \left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	$[x_1 \in [-5, 10], x_2 \in [0, 15]]$	0.397887
f24	Goldstein Price	$f(x) = [1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]$ $\times[30+(2x_1-3x_2)\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	$[-2, 2]$	3
f25	Hartman 1	$f_{25}(x) = -\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{4}\alpha_{ij}(x_j-p_{ij})^2\right]$	$[-1, 3]$	−3.86
f26	Hartman 2	$f_{26}(x) = -\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{6}\alpha_{ij}(x_j-p_{ij})^2\right]$	$[0, 1]$	−3.32
f27	Kowalik	$f_{27}(x) = \sum_{i=0}^{10}\left[\alpha_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2 {}^{**}$	$[-5, 5]$	0.0003074861
f28	Shekel 1	$f_{28}(x) = -\sum_{i=1}^{5}\left[(x-\alpha_i)(x-\alpha_i)^T+c_i\right]^{-1}$	$[0, 10]$	−10.1532
f29	Shekel 2	$f_{29}(x) = -\sum_{i=1}^{7}\left[(x-\alpha_i)(x-\alpha_i)^T+c_i\right]^{-1}$	$[0, 10]$	−10.4028
f30	Shekel 3	$f_{30}(x) = -\sum_{i=1}^{10}\left[(x-\alpha_i)(x-\alpha_i)^T+c_i\right]^{-1}$	$[0, 10]$	−10.5363

**$\alpha = [4, 2, 1, 1/2, 1/4, 1/8, 1/10, 1/12, 1/14, 1/16]$

$b = [0.1957, 0.1947, 0.1735, 0.1600, 0.0844, 0.0627, 0.0456, 0.0342, 0.0323, 0.0235, 0.0246]$

Algorithm 8 Truncation Selection Scheme

Input: The population $P(_T)$, the truncation threshold $T \in [0, 1]$
Output: The population after selection $P(_T)'$
Description:

```
Truncation (T, J_1, ..., J_N) :
j̄ ← sorted population J according fitness with worst individual at the first posit
for i ← 1 to N do
    r ← random {[(1 − T) N], ···, N}
    j'_i ← j̄_r
endfor
return {J'_1, ..., J'_N}
```

From Truncation's pseudocode, it can be noticed the ease of implementation of this selection. However, it neglects the solutions with a low fitness value which have an ability to improve into better solutions. Therefore, may lead to premature convergence.

3 The proposed method

In the literature, the satisfactory effectiveness of the SSA has been tested in addressing various practical problems. The SSA has a satisfactory exploration (global search) ability because of the performance of random operators. These random operators can heighten the diversification (exploration) of the solutions throughout the early stages of the search processes. However, basic SSA still needs further improvements in terms of either exploration directions or even exploitation direction. The main idea here is to maintain the unique simplicity of the SSA and also improve its searches (i.e., exploratory and exploitative) abilities and characteristics.

In this proposal as shown in Algorithm 9, two effective mechanisms have been studied to promote the effectiveness of SSA in solving the optimization problems. It is known as being a powerful optimization method acting in this domain. In the first part, a new hybridization method, based on using SSA with HC local search called HSSA, is developed to improve the exploitation search abilities of the SSA. In the second part, six selection schemes have been investigated and studied for the utilization purposes in SSA in order to improve the performance of the SSA and avoid the drawbacks identified earlier. Generally, these two mechanisms are considered to decrease the likelihood of premature convergence and trapped in local optima of the basic SSA.

Algorithm 9 The proposed HSSA

```
Require: Initialize the salp population x_i(i = 1, 2, ..., n) consider ub and lb.
while (End condition is not satisfied) do
    Calculate the fitness of each search salp.
    Update r_1
    for (each salp (x_i)) do
        if (i==1) then
            Update the position of the leading salp
        else
            Update the position of the followers salp
        end if
    end for
    Verify the position of salps based on the upper and lower bounds
    Choose the initial solution (x) of HC technique from the solutions of the SSA using one of
    the proposed six selection schemes
    Calculate fitness function f(x)
    while (End condition is not satisfied) do
        x'=Improve((N(x))
        if F(x') ≤ F(x) then
            x=x'
        end if
    end while
    if the generated solution of HC (x') is better than the chosen solution from SSA (x) then
        Replace the SSA by the HC solution
    end if
end while
return the best x
```

Table 4 The best normalized results for SSA with different dimensional spaces

Function	Dimensional spaces							
	5	10	15	20	25	30	35	40
F_1	**1.00**	1.73E+01	3.01E+01	1.88E+01	3.35E+00	5.18E+00	1.16E+01	2.82E+01
F_2	6.91E+00	**1.00**	8.58E+00	3.38E+01	1.42E+01	1.64E+01	8.59E+00	4.22E+01
F_3	**1.00**	1.15E+00	1.02E+00	1.11E+00	1.08E+00	1.17E+00	1.06E+00	1.23E+01
F_4	1.06E+00	4.23E+00	1.22E+01	3.31E+01	5.11E+00	**1.00**	7.01E+00	5.17E+00
F_5	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_6	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_7	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_8	6.99E+00	1.13E+01	2.55E+00	**1.00**	7.63E+00	5.63E+00	1.40E+00	1.94E+00
F_9	1.06E+01	1.06E+01	1.06E+01	1.06E+01	1.06E+01	**1.00**	1.06E+01	1.06E+01
F_{10}	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_{11}	**1.00**	3.46E+00	2.18E+00	2.18E+00	5.15E+00	1.10E+01	6.24E+00	3.67E+01
F_{12}	**1.00**	2.02E+00	6.82E+00	2.54E+00	1.13E+01	4.86E+00	7.52E+00	1.12E+00
F_{13}	3.97E+01	1.27+01	4.47E+00	**1.00**	1.46E+01	2.93E+01	1.08E+01	1.21E+01
F_{14}	1.05E+00	1.19E+00	1.21E+00	**1.00**	1.28E+00	1.29E+00	1.25E+00	1.40E+00
F_{15}	9.21E+00	**1.00**	1.32E+00	4.54E+00	5.25E+00	1.57E+00	8.78E+00	4.42E+00
F_{16}	1.48E+00	9.72E+00	2.40E+00	**1.00**	4.28E+00	3.77E+00	4.69E+00	4.93E+00
F_{17}	3.52E+03	3.01E+00	1.08E+00	**1.00**	1.41E+00	1.32E+00	1.09E+00	1.15E+00
F_{18}	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_{19}	2.74E+00	**1.00**	1.96E+00	2.49E+01	5.31E+01	1.01E+01	3.90E+01	1.97E+00
F_{20}	2.59E+00	4.71E+00	1.65E+00	**1.00**	6.83E+00	5.80E+01	3.08E+01	4.92E+01
F_{21}	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_{22}	4.00E+00	**1.00**	2.09E+00	4.91E+00	2.51E+00	1.74E+00	6.99E+00	9.67E+00
F_{23}	**1.00**	3.06E+00	4.97E+00	4.74E+00	1.37E+01	1.64E+01	4.92E+01	4.42E+01
F_{24}	**1.00**	9.09E+00	7.58E+00	1.31E+01	3.94E+01	4.22E+01	5.24E+01	8.63E+01
F_{25}	**1.00**	1.32E+00	1.32E+00	1.32E+00	1.32E+00	1.32E+00	1.32E+00	1.32E+00
F_{26}	1.04E+00	**1.00**	1.74E+00	1.43E+00	1.34E+00	1.23E+00	1.12E+00	1.17E+00
F_{27}	**1.00**	1.24E+00	1.34E+00	1.51E+00	1.73E+00	1.70E+00	1.74E+00	1.55E+00
F_{28}	**1.00**	1.19E+00	1.24E+00	1.16E+00	1.13E+00	1.16E+00	1.08E+00	1.11E+00
F_{29}	**1.00**	1.30E+00	1.23E+00	1.10E+00	1.24E+00	1.10E+00	1.02E+00	1.33E+00
F_{30}	**1.00**	1.02E+00	1.01E+00	1.02E+00	1.03E+00	1.01E+00	1.01E+00	1.01E+00
Total best	**17**	**11**	**6**	**12**	**6**	**8**	**6**	**6**

3.1 Computational complexity

Note that the computational complexity for running the proposed HSSA algorithm depends on the number of salp solutions (X), the dimensions (d), and the maximum number of repetitions (t). Hence, the computational complexity is $O(\text{HSSA}) = O(\text{Initialization}) + t \times O((\text{Salp Fitness})$ $+O(\text{Updating process of all salps positions with the used mechanisms}))$. The total computational complexity of computing the fitness value for all solutions is $O(X \times t)$. Updating the values of the positions of all solutions with new mechanisms is $O((\frac{1}{6}) \times X \times d^2)$. Consequently, the overall computational complexity of the proposed HSSA algorithm is $O(X \times d + (\frac{1}{6}) \times X \times d^2 + F \times X)$, where $\text{SSA} = O(X \times d + F \times X)$ and $HC = O(t(d \times 1 + F \times 1))$.

4 Experimental results and discussions

4.1 Experiments using benchmark functions

In this section, the performance of the proposed HSSA method is tested on one side and from another side is compared with various other well-known and new methods using two common optimization problems: various benchmark test problems and various engineering design problems. All implemented codes in this study have been conducted in the same custom using MATLAB 8.5.0.197613 (R2015a) and run on a computer machine with the Windows 10 64-bit professional and 8 GB RAM. For fair comparisons, 50 search agents and 1000 iterations are employed. The reported results over 30 independent runs are recorded. Note, the SSA parameters have been taken from the first paper [7].

Table 5 The best normalized results for SSA with population sizes

Function	Population sizes							
	5	10	15	20	50	100	250	500
F_1	5.41E+01	**1.00**	1.17E+02	**1.00**	1.49E+01	9.91E+02	5.94E+02	1.93E+02
F_2	5.23E+01	7.13E+00	**1.00**	**1.00**	2.56E+00	**1.00**	2.60E+01	5.16E+01
F_3	1.04E+00	**1.00**	1.04E+00	**1.00**	1.17E+00	1.04E+00	1.04E+00	1.04E+00
F_4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_5	2.41E+01	8.37E+00	**1.00**	1.93E+00	2.19E+01	3.70E+00	1.12E+01	4.90E+01
F_6	1.74E+01	6.89E+00	2.39E+01	**1.00**	2.36E+01	6.65E+01	2.16E+01	3.22E+01
F_7	1.23E+00	1.04E+00	1.09E+00	1.90E+00	**1.00**	1.04E+00	1.24E+00	1.14E+00
F_8	1.01E+01	1.20E+01	1.09E+00	**1.00**	2.68E+01	3.80E+00	5.96E+00	1.62E+01
F_9	1.04E+00	2.57E+00	2.27E+00	**1.00**	5.68E+00	1.24E+00	1.10E+00	2.55E+00
F_{10}	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_{11}	1.33E+01	1.05E+01	1.86E+00	1.86E+00	**1.00**	2.87E+01	6.39E+00	2.58E+01
F_{12}	2.28E+00	7.58E+00	2.85E+00	**1.00**	2.53E+00	1.05E+00	5.00E+00	5.34E+00
F_{13}	**1.00**	1.32E+00	8.20E+00	**1.00**	**1.00**	**1.00**	**1.00**	1.29E+00
F_{14}	2.83E+01	2.49E+01	1.83E+00	**1.00**	1.66E+01	9.32E+01	1.08E+02	1.09E+02
F_{15}	2.23E+01	3.87E+01	1.02E+01	**1.00**	1.49E+00	1.28E+00	1.10E+01	1.12E+01
F_{16}	1.58E+00	1.53E+00	1.23E+00	**1.00**	1.15E+00	1.16E+00	1.20E+00	1.17E+00
F_{17}	2.23E+00	1.60E+00	1.42E+00	**1.00**	**1.00**	1.02E+00	1.11E+00	1.18E+00
F_{18}	2.69E+00	2.32E+00	2.54E+00	**1.00**	1.93E+00	1.81E+00	1.87E+00	1.66E+00
F_{19}	1.02E+02	7.62E+01	**1.00**	**1.00**	7.62E+00	1.52E+01	2.62E+01	2.26E+01
F_{20}	1.01E+00	1.04E+00	**1.00**	1.06E+00	1.02E+00	1.01E+00	1.02E+00	1.02E+00
F_{21}	1.14E+00	1.29E+00	1.28E+00	**1.00**	**1.00**	1.29E+00	1.24E+00	1.21E+00
F_{22}	1.42E+02	1.15E+01	1.16E+01	**1.00**	3.43E+00	2.79E+00	2.01E+00	1.24E+01
F_{23}	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F_{24}	1.32E+00	1.32E+00	**1.00**	1.27+00	1.35E+00	1.43E+00	1.22E+00	1.21E+00
F_{25}	1.11E+00	7.58E+00	1.25E+00	**1.00**	1.19E+00	**1.00**	1.10E+00	1.02E+00
F_{26}	**1.00**	1.02E+00	1.03E+00	**1.00**	**1.00**	1.02E+00	**1.00**	1.04E+00
F_{27}	1.21E+00	1.33E+00	1.01E+00	**1.00**	1.38E+00	2.71E+00	1.07E+00	1.55E+01
F_{28}	1.02E+00	1.16E+00	**1.00**	**1.00**	1.49E+00	1.60E+00	2.23E+00	1.44E+01
F_{29}	**1.00**	**1.00**	**1.00**	1.01E+00	1.01E+00	1.01E+00	1.29E+00	1.06E+00
F_{30}	1.60E+00	1.43E+00	1.01E+00	**1.00**	1.03E+00	1.03E+00	2.04E+00	6.81E+00
Total best	**6**	**6**	**10**	**24**	**9**	**6**	**5**	**3**

Table 6 The parameters values of the comparative algorithms

Algorithms	Parameter
ABC	*Colony size* = 50, *limit* = 1000
BA	$F_{\min} = 0$, $F_{\max} = 2$, $r = 0.5$, $A = 0.25$, $\alpha, \gamma = 0.9$
MFO	$P = 0.01$, $\beta = 1.5$
DE	$CR = 0.9$, $F = 0.6$
GA	*Crossover type is* 1; *crossover probability* = 1; *mutation probability* = 0.01
HS	HMCR = 0.9, PAR = 0.5, BW = 0.01
KH	$N^{\max} = 0.01$, $V_f = 0.02$, $D^{\max} = 0.005$, $C_t = 0.4$
GWO	$\alpha_0 = 2$

4.1.1 Experimental setting

The proposed HSSA method is verified based on using 30 classical benchmark test functions listed in Tables 1, 2, and 3. These well-known benchmarks include 30 test functions, which are classified into unimodal (optimization functions with only one local optimum) and multimodal (optimization functions that frequently contain multiple global and local optima) problems. Moreover, these functions are chosen with various dimensions and diverse difficulty levels including 10 scalable unimodal functions, 12 scalable multimodal functions, and 8 fixed-dimension multimodal functions. These features make the investigation process more fitting for testing the exploration and exploitation functions in the proposed method.

Table 7 Best, average (Avg), and standard deviation (Std) for comparing the proposed HSSA with basic SSA and other algorithms

Function	Metric	Comparative algorithms									
		ABC	BA	MFO	DE	GA	HS	KH	GWO	SSA	HSSA
F_1	Best	1.37E−05	3.00E−01	1.15E−01	7.77E−01	2.43E−05	4.27E+00	5.31E−02	3.13E−02	1.48E−04	3.28E−08
	Avg	2.99E−01	8.46E+00	2.82E+00	9.36E+00	1.21E−03	2.02E+01	5.29E+00	2.38E+00	2.39E−02	1.17E−06
	Std	1.07E−16	5.15E−01	5.68E+00	2.75E+02	1.08E−05	8.36E−01	2.37E+00	2.14E−02	1.87E+00	1.80E−06
F_2	Best	4.89E−06	2.56E−01	2.20E−02	7.17E−03	4.09E−06	2.15E−01	1.22E−01	3.53E−01	5.44E−03	2.08E−06
	Avg	2.85E−01	7.31E+00	1.49E+00	4.94E−01	1.55E−03	1.02E+00	1.03E+00	6.17E+00	3.92E−01	1.49E−04
	Std	9.85E−17	7.35E−01	2.99E−01	1.90E+04	2.41E−04	5.70E+00	1.74E+01	2.24E−01	1.92E+02	2.65E−04
F_3	Best	9.98E−01	2.31E+00	3.75E+00	1.03E+00	1.41E+00	9.98E−01	9.98E−01	2.99E+00	9.98E−01	6.57E−06
	Avg	8.49E+00	9.14E+01	2.46E+01	6.87E+01	1.58E+01	1.62E+01	4.06E+00	1.20E+00	4.67E+00	2.07E−02
	Std	0.00E+00	2.19E+00	3.30E+00	1.81E−01	5.54E−01	7.15E−07	3.63E−16	3.42E+00	8.07E−01	1.17E−02
F_4	Best	3.42E−04	1.08E−03	4.25E−03	1.30E−03	7.24E−04	4.35E−04	5.33E−03	1.71E−03	8.92E−04	9.68E−06
	Avg	1.42E−01	1.97E+00	4.68E−01	1.27E+00	3.52E−01	5.56E−01	8.40E+00	2.07E−01	9.34E−01	3.23E−02
	Std	1.67E−04	4.81E−04	7.33E−03	4.44E−04	5.82E−05	9.62E−05	8.08E−03	5.08E−03	3.76E−04	1.74E−02
F_5	Best	4.25E−04	2.68E−03	3.88E−03	1.18E−04	1.74E−02	5.77E−03	3.74E−02	2.65E−01	1.60E−03	8.51E−05
	Avg	5.70E+00	5.70E+00	3.00E+00	3.00E+00	3.00E+00	1.48E+00	1.48E+00	2.25E+00	1.16E−01	4.50E−03
	Std	5.66E−04	1.46E+07	1.11E+03	7.07E+04	3.47E+02	3.08E+03	3.88E+02	8.72E−01	2.97E+02	6.38E−02
F_6	Best	1.05E−10	3.30E−02	1.37E−01	8.56E−02	2.69E−03	1.72E−02	1.45E−02	4.38E−03	4.42E−06	1.03E−07
	Avg	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	1.88E−14	1.62E+00	5.63E−01	2.21E−04	4.90E−07
	Std	3.76E−15	1.81E+03	2.20E+00	4.29E+02	8.98E−04	5.30E+01	1.83E+00	3.09E−01	4.59E−01	2.55E−05
F_7	Best	6.81E−05	1.74E+00	1.01E−01	2.49E−01	4.56E−03	1.36E−01	1.42E−02	6.54E−04	3.00E−05	3.99E−09
	Avg	7.24E−02	4.35E+01	5.33E+00	1.71E+00	8.92E−01	5.82E+00	9.62E+00	8.08E−01	5.08E−03	2.90E−04
	Std	8.69E−05	3.51E+00	8.11E+00	1.57E−01	1.74E−03	6.65E−02	6.85E−03	4.22E−04	3.40E−02	7.66E−02
F_8	Best	0.00E+00	1.55E−06	2.58E−07	1.49E−12	0.00E+00	1.16E−12	8.69E−09	2.87E−11	0.00E+00	0.00E+00
	Avg	1.22E−85	3.53E−01	5.44E−03	6.98E−04	1.05E−52	1.74E−03	2.24E−04	1.92E−02	3.41E−125	0.00E+00
	Std	0.00E+00	3.83E−01	2.23E−01	3.20E−01	9.23E−14	2.09E−01	2.89E−01	1.57E+00	0.00E+00	0.00E+00
F_9	Best	8.88E−14	1.37E−01	4.75E+00	7.83E+00	1.32E−16	9.49E+00	3.50E+00	1.39E−14	1.42E−14	1.01E−14
	Avg	1.87E−04	3.07E+00	6.13E+01	1.02E+01	1.96E−06	3.57E+01	5.03E+01	1.58E−04	5.77E−06	2.00E−08
	Std	0.00E+00	8.38E+00	2.56E+00	1.37E+00	2.00E−03	1.35E+00	1.11E+00	2.69E+00	2.49E+00	1.10E−09
F_{10}	Best	0.00E+00	1.20E−21	5.53E−12	6.07E−17	0.00E+00	2.50E−11	5.39E−09	1.62E−13	0.00E+00	0.00E+00
	Avg	1.48E−120	5.59E−02	4.48E−03	1.01E−01	1.67E−82	5.54E−02	4.65E−02	9.99E−03	2.43E−131	0.00E+00
	Std	0.00E+00	3.12E−01	6.28E−02	2.80E+00	0.00E+00	5.16E−01	1.27E−01	4.32E−03	0.00E+00	0.00E+00
F_{11}	Best	1.28E−02	9.36E−06	9.50E−01	1.73E−02	2.91E−02	6.06E−02	1.42E−01	2.00E−01	0.00E+00	0.00E+00
	Avg	1.03E+00	1.03E+00	1.03E+00	1.03E+00	1.03E+00	1.85E+00	3.90E+00	2.14E+00	7.01E−15	0.00E+00
	Std	3.40E−02	1.35E−06	9.61E+01	5.59E+01	1.12E+03	3.15E+02	1.49E+00	4.07E+03	0.00E+00	0.00E+00
F_{12}	Best	5.32E−06	2.60E−12	6.96E+00	2.20E−01	1.19E−01	1.24E+01	1.67E+01	3.20E+01	2.43E−16	8.55E−20
	Avg	4.72E+00	9.65E+00	3.20E+00	8.48E+00	1.05E+00	2.70E+00	1.55E+00	2.04E+00	2.90E−06	2.05E−07
	Std	8.50E−06	2.83E−12	1.71E+00	7.90E+00	6.88E+00	5.77E+00	1.22E+00	2.15E+01	1.43E−07	0.00E+00
F_{13}	Best	3.69E+01	1.82E−17	1.56E+00	3.26E+00	6.44E+01	7.33E−01	5.20E−01	1.52E−01	0.00E+00	0.00E+00
	Avg	3.27E+00	3.30E−02	2.89E+01	3.24E+01	3.25E+00	6.04E+00	4.90E+00	3.32E+00	5.74E−12	0.00E+00
	Std	4.63E+03	6.04E−17	9.16E+02	1.01E+02	3.02E+02	5.49E+03	1.17E+01	1.05E+04	0.00E+00	0.00E+00
F_{14}	Best	3.17E−02	6.05E−01	1.71E+00	1.91E−01	5.66E−02	1.63E−01	9.11E+00	3.29E−16	1.37E−18	1.07E−22
	Avg	3.80E+00	3.86E+00	3.85E+01	3.86E+00	3.86E+00	2.09E+00	1.65E+01	2.17E−03	1.36E−12	1.21E−18
	Std	4.29E+00	1.15E+01	2.26E−01	7.20E+00	1.37E−02	2.20E+00	3.62E+00	4.14E−01	9.57E−04	2.46E−11
F_{15}	Best	1.03E−02	3.25E−02	5.66E−02	6.33E−03	8.21E−02	6.01E−02	4.43E−03	6.87E−02	3.00E−03	2.99E−07
	Avg	2.71E+00	2.39E+00	6.83E−01	2.69E+00	2.80E+00	4.90E+00	1.86E+01	5.96E−01	1.22E−01	2.99E−03
	Std	6.78E−02	6.78E−02	3.94E−01	9.32E−02	1.44E−01	1.94E−01	1.85E−01	3.90E−01	2.14E−02	6.38E−03
F_{16}	Best	3.98E−01	6.03E−01	2.21E−02	7.21E−01	8.01E−01	1.15E+00	7.91E−01	3.62E−02	1.47E−04	8.01E−10
	Avg	1.75E+00	5.56E+00	9.88E−01	1.09E+00	4.59E+00	5.92E+01	3.71E+00	4.19E−01	4.22E−02	1.35E−06
	Std	7.30E+00	2.04E+00	2.72E−01	5.42E−01	3.35E+00	3.01E+00	1.88E−01	1.62E−01	5.63E−01	1.97E−06

Table 7 (continued)

Function	Metric	Comparative algorithms									
		ABC	BA	MFO	DE	GA	HS	KH	GWO	SSA	HSSA
F_{17}	Best	3.00E+00	4.85E−01	1.12E−02	5.23E+00	1.09E+00	6.36E+00	7.89E−01	1.12E−02	1.12E−02	3.54E−06
	Avg	2.37E+01	6.91E+00	1.34E+00	6.96E+01	4.55E+01	2.09E+02	9.54E+00	8.35E−01	1.59E−01	1.13E−04
	Std	2.18E+00	2.19E−01	2.69E−02	9.93E+00	1.36E+00	5.13E+00	1.48E+01	4.33E+01	2.25E−02	9.08E−03
F_{18}	Best	3.86E−01	7.55E+00	2.46E−03	5.79E+00	8.24E−02	6.42E−01	7.68E−06	2.99E−01	1.08E−06	3.02E−08
	Avg	3.40E+00	2.86E+01	2.06E−01	4.82E+01	4.44E+00	4.35E+00	2.73E−01	3.60E+00	2.60E−03	1.56E−05
	Std	2.65E−01	2.71E+00	1.02E−02	4.00E+00	3.67E−01	1.42E−01	2.09E−02	1.65E−01	2.17E−02	9.09E−04
F_{19}	Best	3.27E+00	3.23E+00	2.90E+00	3.27E+00	3.30E+00	3.32E+00	3.27E+00	3.30E+00	1.89E+00	6.82E−06
	Avg	3.30E+02	7.39E+01	1.51E+01	1.44E+01	9.51E+01	2.50E+02	1.01E+01	1.24E+01	8.24E+00	2.32E−02
	Std	6.03E−02	5.63E−02	1.20E−01	7.60E−02	4.84E−02	3.39E−05	6.04E−02	4.90E−02	3.32E−01	1.05E−02
F_{20}	Best	4.59E−01	2.60E−09	1.01E−02	7.66E−02	8.79E−06	1.86E−12	1.84E−01	3.45E−11	2.71E−12	1.39E−14
	Avg	1.02E+01	6.81E−02	5.14E+00	7.78E+00	8.66E−01	1.02E−01	4.72E+00	9.65E−02	3.20E−06	2.39E−08
	Std	6.96E+00	3.50E+00	2.76E+00	2.56E+00	3.04E+00	1.05E−02	2.70E+00	1.55E+00	2.04E−02	3.06E−06
F_{21}	Best	4.42E+00	1.05E−08	1.10E−02	2.34E−01	7.50E+00	4.52E−12	1.27E−02	3.10E−11	1.61E−12	3.33E−14
	Avg	1.04E+01	7.45E−01	6.19E+00	8.34E+00	1.01E+01	1.04E−03	6.34E+00	1.02E−01	4.60E−05	1.47E−08
	Std	1.19E+00	3.30E+00	3.16E+00	2.79E+00	1.40E+00	1.66E−04	3.69E+00	9.70E−01	1.38E−02	1.27E−08
F_{22}	Best	1.05E−01	8.48E+00	5.24E+00	8.30E+00	1.05E−01	1.05E−01	6.94E+00	1.00E−01	3.82E−01	1.79E−06
	Avg	1.60E+00	1.13E+02	2.68E+01	3.62E+02	2.87E+01	4.43E+01	1.34E+02	6.38E+00	1.55E+00	8.89E−03
	Std	1.75E+00	3.26E+00	2.99E+00	2.81E+00	7.38E+00	8.24E−01	3.94E+00	1.98E+00	1.69E+00	8.59E−03
F_{23}	Best	9.29E+00	9.69E+00	1.58E+00	3.73E+00	1.04E+01	4.86E+00	1.04E+01	1.04E+01	1.58E+00	4.04E−06
	Avg	9.57E+02	2.37E+01	2.40E+01	3.12E+01	3.29E+02	3.30E+01	3.46E+01	3.76E+02	1.18E+01	4.82E−03
	Std	2.58E+00	1.84E+00	3.02E+00	8.49E−01	1.07E+00	1.21E+00	7.00E+01	6.46E+00	4.24E+00	2.74E−03
F_{24}	Best	9.22E+00	8.99E+00	9.73E+00	3.65E+00	1.05E−01	5.28E+00	1.05E−01	1.05E−01	1.05E−01	9.81E−03
	Avg	1.37E+01	5.79E+01	3.19E+02	7.99E+01	8.80E+00	1.96E+01	4.42E+01	2.28E+01	2.87E+00	5.11E−03
	Std	2.74E+00	2.93E+00	2.14E+00	7.74E+01	3.17E−01	1.25E+00	9.32E+00	6.92E+01	5.45E−01	2.38E−02
F_{25}	Best	3.86E+00	3.85E−02	5.54E+00	2.91E−01	6.61E−01	4.44E+00	8.63E−01	7.99E−01	2.01E−02	3.55E−06
	Avg	4.63E+03	8.26E+01	8.13E+03	7.95E+00	6.16E+00	4.04E+03	6.29E+01	2.57E+01	5.01E−01	3.68E−04
	Std	1.36E+00	2.17E+00	6.63E+00	2.73E+00	2.71E+00	8.17E+00	2.26E+00	3.82E+00	1.18E−01	1.29E−04
F_{26}	Best	3.24E+00	3.25E+00	3.26E+00	1.73E+00	3.32E+00	2.83E+00	3.32E+00	3.29E+00	1.73E+00	2.12E−02
	Avg	3.69E+03	1.11E+02	1.47E+04	2.41E+01	7.32E+02	2.48E+02	3.67E+02	1.22E+01	3.14E+01	8.23E−01
	Std	5.74E−02	8.74E−02	6.05E−02	4.12E−01	7.73E−04	2.25E−01	1.23E−03	1.31E−02	2.47E−02	4.81E−01
F_{27}	Best	8.50E−01	7.02E−01	2.77E−16	7.69E−02	1.18E−05	1.30E−08	4.42E−01	5.79E−07	8.18E−13	3.25E−15
	Avg	8.48E+00	9.22E+00	7.63E−03	3.92E+00	1.02E−01	4.83E−02	1.02E+01	1.01E−02	1.01E−06	3.37E−08
	Std	2.90E+00	2.15E+00	2.81E+00	7.89E−01	1.18E−02	1.35E+00	4.91E−03	6.97E−03	2.62E−05	5.00E−06
F_{28}	Best	1.03E+00	1.03E+00	1.03E+00	1.72E−01	1.03E+00	1.03E+00	1.03E+00	1.03E+00	1.72E−01	3.16E−03
	Avg	5.32E+01	6.01E+01	6.60E+01	3.33E+00	2.64E+02	4.40E+01	7.93E+01	1.11E+01	3.38E+00	1.61E−02
	Std	7.01E+00	8.16E+00	8.62E+00	1.62E+00	6.78E+00	1.16E+00	5.98E+00	1.03E+00	2.04E+00	5.67E−01
F_{29}	Best	3.40E−02	2.22E−09	5.10E−02	2.03E−09	2.00E−12	1.70E−01	5.16E−11	4.55E−03	0.00E+00	0.00E+00
	Avg	3.98E+00	3.98E−01	3.98E−01	1.37E−02	3.98E−03	3.98E+00	3.98E−03	3.98E−01	0.00E+00	0.00E+00
	Std	2.21E−02	4.58E−07	2.46E−02	9.41E−04	4.93E−02	5.42E−01	2.48E−03	5.33E−01	0.00E+00	0.00E+00
F_{30}	Best	6.98E−02	1.05E−01	5.05E−02	5.78E−01	2.38E−12	2.14E−01	6.39E−11	9.64E−02	7.20E−16	2.38E−18
	Avg	1.28E−01	9.21E+00	5.80E+00	2.31E+00	2.32E−03	3.82E+00	1.22E−02	4.29E−01	8.30E−05	9.60E−07
	Std	3.41E−01	9.04E−02	3.49E−02	8.29E−02	4.63E−12	3.89E+00	8.99E−11	8.47E−01	1.61E−05	3.45E−06

Table 8 Best, average (Avg), and standard deviation (Std) for comparing the HSSA with the selection schemes (THSSA, PHSSA, LHSSA, EHSSA, GHSSA, and TrHSSA)

Function	Metric	Selection schemes						
		HSSA	THSSA	PHSSA	LHSSA	EHSSA	GHSSA	TrHSSA
F_1	Best	3.28E−08	4.58E−10	**1.26E−10**	3.98E−10	2.02E−09	4.77E−10	1.11E−09
	Avg	1.17E−06	4.32E−07	1.03E−07	5.24E−07	3.12E−07	3.36E−07	3.10E−07
	Std	1.80E−06	1.46E−06	4.57E−07	1.75E−07	3.21E−07	4.20E−07	3.05E−07
F_2	Best	2.08E−06	5.14E−07	**7.29E−08**	1.05E−07	5.66E−07	4.31E−06	1.79E−07
	Avg	1.49E−04	7.63E−05	6.40E−05	5.37E−05	3.60E−05	1.95E−05	4.43E−05
	Std	2.65E−04	1.51E−04	1.05E−04	1.03E−04	1.03E−04	2.97E−05	5.00E−05
F_3	Best	6.57E−06	4.50E−06	4.61E−07	**3.06E−07**	5.99E−07	4.02E−06	1.09E−06
	Avg	2.07E−02	3.05E−04	4.20E−04	3.30E−04	2.89E−04	2.83E−04	3.22E−04
	Std	1.17E−02	4.24E−04	5.02E−04	5.21E−04	3.83E−04	4.04E−04	4.04E−04
F_4	Best	9.68E−06	**1.00E−09**	1.70E−08	2.48E−06	2.20E−06	1.00E−07	9.63E−06
	Avg	3.23E−02	1.11E−06	3.67E−05	8.30E−03	7.52E−03	7.80E−03	8.00E−03
	Std	1.74E−02	5.87E−06	1.76E−05	1.56E−03	2.18E−03	2.10E−03	4.38E−02
F_5	Best	8.51E−05	1.10E−06	**2.60E−08**	4.34E−08	6.17E−07	6.24E−08	5.72E−07
	Avg	4.50E−03	3.02E−04	4.55E−03	7.60E−05	1.09E−04	7.39E−06	5.22E−04
	Std	6.38E−02	1.89E−06	5.15E−02	3.98E−02	6.15E−02	7.52E−02	5.78E−02
F_6	Best	1.03E−07	2.55E−09	7.16E−11	3.54E−10	1.54E−11	**7.02E−13**	4.82E−10
	Avg	4.90E−07	7.24E−08	5.32E−08	1.81E−09	4.26E−11	4.37E−08	8.29E−09
	Std	2.55E−05	3.99E−08	2.53E−06	1.08E−06	1.75E−05	2.07E−06	4.33E−06
F_7	Best	3.99E−09	2.91E−10	**2.77E−11**	2.81E−11	2.77E−09	2.11E−10	5.72E−11
	Avg	2.90E−04	3.01E−06	3.02E−06	3.02E−08	3.02E−05	3.09E−07	3.03E−06
	Std	7.66E−02	3.84E−02	4.43E−03	3.52E−02	3.87E−04	2.50E−02	3.65E−02
F_8	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Avg	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_9	Best	1.01E−14	2.08E−15	5.41E−17	5.78E−17	**5.22E−17**	8.85E−17	4.42E−15
	Avg	2.00E−08	7.14E−12	3.11E−10	2.47E−11	6.06E−14	7.14E−12	5.25E−11
	Std	1.10E−09	1.08E−12	4.35E−10	4.29E−11	2.14E−14	4.70E−12	2.91E−12
F_{10}	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Avg	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_{11}	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Avg	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_{12}	Best	8.55E−20	6.42E−26	**9.68E−27**	3.17E−25	4.16E−22	8.38E−21	4.43E−22
	Avg	2.05E−07	2.14E−09	2.17E−11	1.52E−10	1.62E−09	2.04E−09	1.64E−08
	Std	1.43E−07	1.45E−08	1.30E−11	8.98E−09	8.63E−09	1.11E−07	1.04E−07
F_{13}	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Avg	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_{14}	Best	1.07E−22	9.89E−25	**0.00E+00**	6.60E−27	8.35E−25	8.24E−26	8.56E−24
	Avg	1.21E−18	6.61E−20	0.00E+00	5.21E−22	9.98E−20	7.59E−18	5.61E−20
	Std	2.46E−11	3.57E−18	0.00E+00	4.68E−15	5.79E−17	1.24E−19	3.86E−17
F_{15}	Best	2.99E−07	4.30E−10	**1.81E−12**	2.88E−12	5.31E−12	1.16E−10	3.22E−09
	Avg	2.99E−03	3.30E−04	3.00E−04	3.81E−04	3.31E−04	2.55E−04	3.22E−04
	Std	6.38E−03	4.58E−04	5.12E−04	5.02E−04	4.99E−04	3.83E−04	4.04E−04
F_{16}	Best	8.01E−10	7.47E−12	**1.07E−16**	6.95E−16	1.82E−12	2.15E−13	1.21E−13
	Avg	1.35E−06	4.10E−08	5.84E−09	9.91E−09	6.58E−11	3.82E−10	1.17E−10
	Std	1.97E−06	5.81E−06	1.38E−07	1.79E−07	1.19E−06	7.83E−06	2.02E−06

Table 8 (continued)

Function	Metric	Selection schemes						
		HSSA	THSSA	PHSSA	LHSSA	EHSSA	GHSSA	TrHSSA
F_{17}	Best	3.54E−06	1.53E−09	2.84E−09	**1.13E−10**	5.43E−09	9.03E−09	4.32E−08
	Avg	1.13E−04	9.86E−06	7.54E−06	2.91E−06	1.03E−04	1.17E−06	1.07E−06
	Std	9.08E−03	1.11E−06	7.41E−06	3.57E−06	5.87E−05	9.90E−04	9.84E−04
F_{18}	Best	3.02E−08	2.42E−12	**2.33E−16**	4.48E−16	2.41E−15	2.92E−15	2.27E−14
	Avg	1.56E−05	5.83E−09	6.73E−09	5.23E−11	5.91E−09	8.06E−08	7.28E−09
	Std	9.09E−04	7.63E−06	1.70E−08	2.77E−06	7.67E−07	4.55E−06	2.65E−06
F_{19}	Best	6.82E−06	2.00E−11	**1.93E−12**	2.06E−12	2.02E−10	1.98E−10	2.01E−10
	Avg	2.32E−02	4.50E−06	2.54E−09	7.02E−07	4.22E−07	3.20E−06	6.33E−06
	Std	1.05E−02	4.79E−04	4.94E−04	4.25E−03	5.60E−04	6.28E−04	4.54E−04
F_{20}	Best	1.39E−14	1.02E−18	**7.96E−22**	7.97E−20	7.96E−19	4.07E−20	1.38E−20
	Avg	2.39E−08	1.10E−11	1.80E−14	1.80E−10	1.80E−09	9.59E−12	1.14E−10
	Std	3.06E−06	2.68E−08	6.94E−07	5.38E−09	6.99E−08	1.70E−09	3.08E−08
F_{21}	Best	3.33E−14	4.23E−19	1.82E−25	**2.77E−27**	5.98E−26	3.18E−24	3.07E−24
	Avg	1.47E−08	1.35E−12	1.77E−11	4.96E−13	3.01E−10	4.49E−10	2.62E−10
	Std	1.27E−08	5.65E−10	2.49E−11	2.33E−10	3.52E−11	2.42E−10	2.47E−10
F_{22}	Best	1.79E−06	3.21E−09	**4.49E−10**	4.68E−09	2.84E−08	3.26E−08	2.25E−08
	Avg	8.89E−03	8.18E−05	1.11E−05	7.93E−07	1.34E−06	1.23E−05	1.21E−05
	Std	8.59E−03	7.46E−04	1.15E−05	6.70E−05	5.37E−05	1.27E−04	9.09E−04
F_{23}	Best	4.04E−06	**4.27E−10**	7.27E−10	8.56E−09	4.05E−08	7.14E−08	3.60E−08
	Avg	4.82E−03	7.10E−06	9.45E−06	7.81E−07	6.71E−08	6.54E−07	8.06E−07
	Std	2.74E−03	9.88E−05	1.18E−05	1.21E−05	8.77E−05	9.08E−05	9.76E−05
F_{24}	Best	9.81E−03	1.45E−08	**1.06E−09**	3.62E−08	8.53E−07	8.74E−08	1.66E−09
	Avg	5.11E−03	7.96E−07	5.04E−08	2.16E−08	3.50E−07	4.15E−08	8.65E−08
	Std	2.38E−02	8.50E−05	5.36E−05	6.37E−05	1.48E−04	2.64E−05	3.80E−05
F_{25}	Best	3.55E−06	2.16E−11	**1.53E−12**	8.32E−11	4.62E−09	6.73E−09	5.82E−09
	Avg	3.68E−04	4.45E−07	4.46E−07	4.47E−07	4.44E−07	4.46E−07	4.47E−07
	Std	1.29E−04	5.26E−05	6.21E−06	3.57E−05	2.12E−05	6.19E−05	2.99E−05
F_{26}	Best	2.12E−02	1.32E−04	1.31E−04	1.31E−04	1.32E−04	1.32E−04	**1.28E−04**
	Avg	8.23E−01	2.00E−03	1.20E−02	2.37E−02	1.76E−02	1.75E−02	1.39E−02
	Std	4.81E−01	2.26E−02	1.18E−01	1.35E−01	3.05E−02	3.13E−02	3.40E−02
F_{27}	Best	3.25E−15	9.18E−19	**1.36E−21**	2.39E−20	8.47E−20	2.83E−19	1.55E−19
	Avg	3.37E−08	1.04E−10	1.48E−12	2.51E−11	9.67E−11	2.95E−11	1.67E−10
	Std	5.00E−06	1.47E−09	3.50E−10	4.53E−09	1.98E−09	5.12E−09	3.01E−09
F_{28}	Best	3.16E−03	2.93E−05	2.99E−07	3.01E−05	**2.92E−07**	2.96E−07	2.74E−06
	Avg	1.61E−02	6.73E−04	1.84E−04	1.06E−03	7.80E−03	3.88E−03	2.57E−04
	Std	5.67E−01	9.92E−03	2.20E−02	3.59E−02	9.98E−03	1.36E−02	3.45E−03
F_{29}	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Avg	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_{30}	Best	2.38E−18	8.50E−21	**2.64E−21**	6.37E−21	1.48E−20	5.36E−21	3.80E−21
	Avg	9.60E−07	2.46E−10	4.02E−10	4.02E−11	4.02E−10	2.14E−10	2.54E−10
	Std	3.45E−06	5.98E−08	5.67E−09	4.08E−08	5.67E−09	3.78E−09	6.85E−09

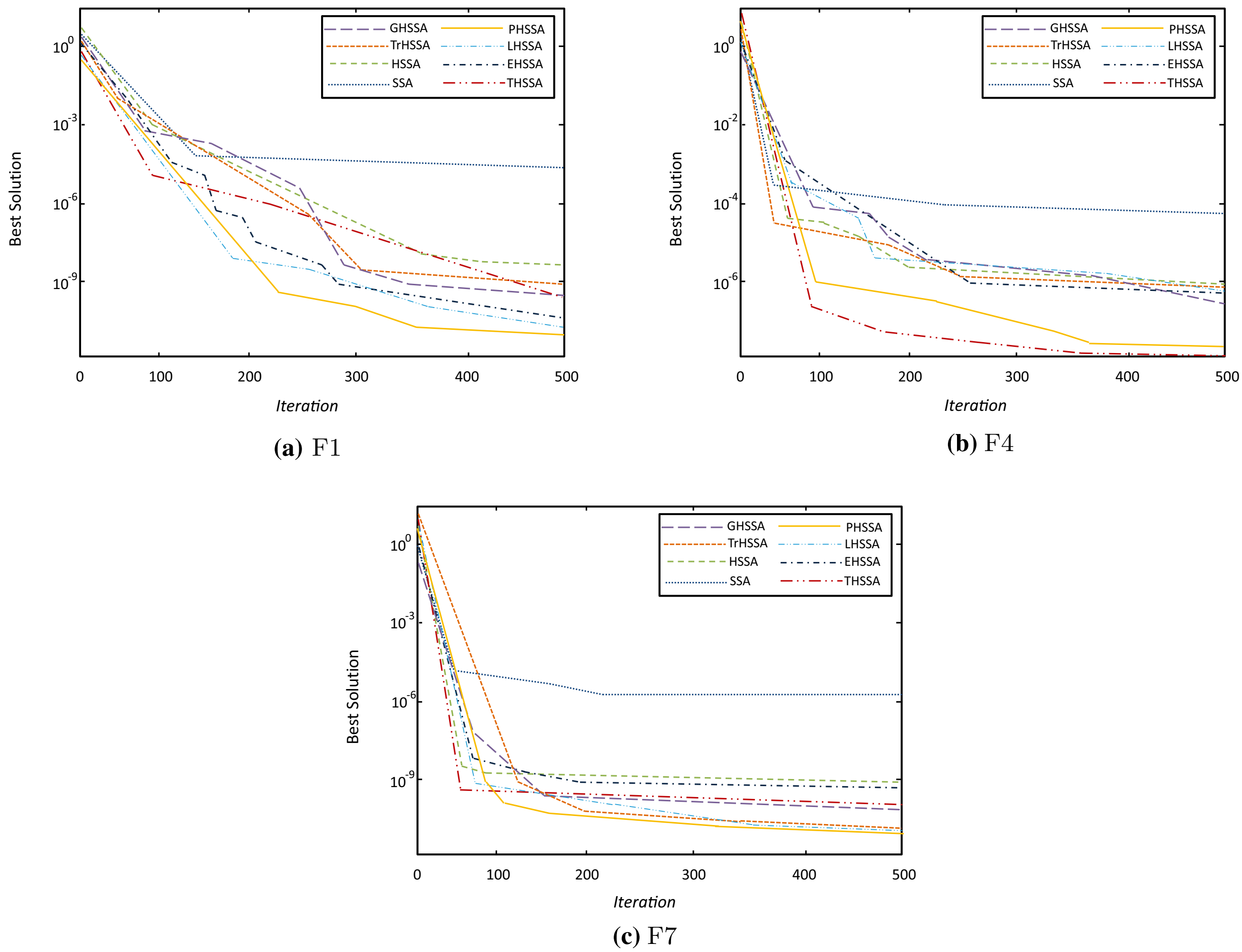


Fig. 2 Convergence graphs of the unimodal benchmark functions

4.1.2 Performance measures

Normalization measure is the process of regularizing data with respect to the difference in values between samples. In the experiments, the effects of different values of the dimensions and the search agents are compared with one another. This procedure is difficult due to the wide gap between solutions. Therefore, normalization improves data integrity [46]. In this work, normalization is calculated based on the following equation:

$$z_i = \frac{x_i - \mu}{S} \tag{15}$$

where is $x = (x_1, \dots, x_n)$, n denotes the total number of data, z_i denotes the normalized data for element ith, μ is the mean and S is the standard deviation. Finally, the minimum element of the data will be 1 in the normalization results.

The best measure is utilized to calculate the best-obtained value by the algorithm to be evaluated for several predefined numbers of runs, which can be measured as follows:

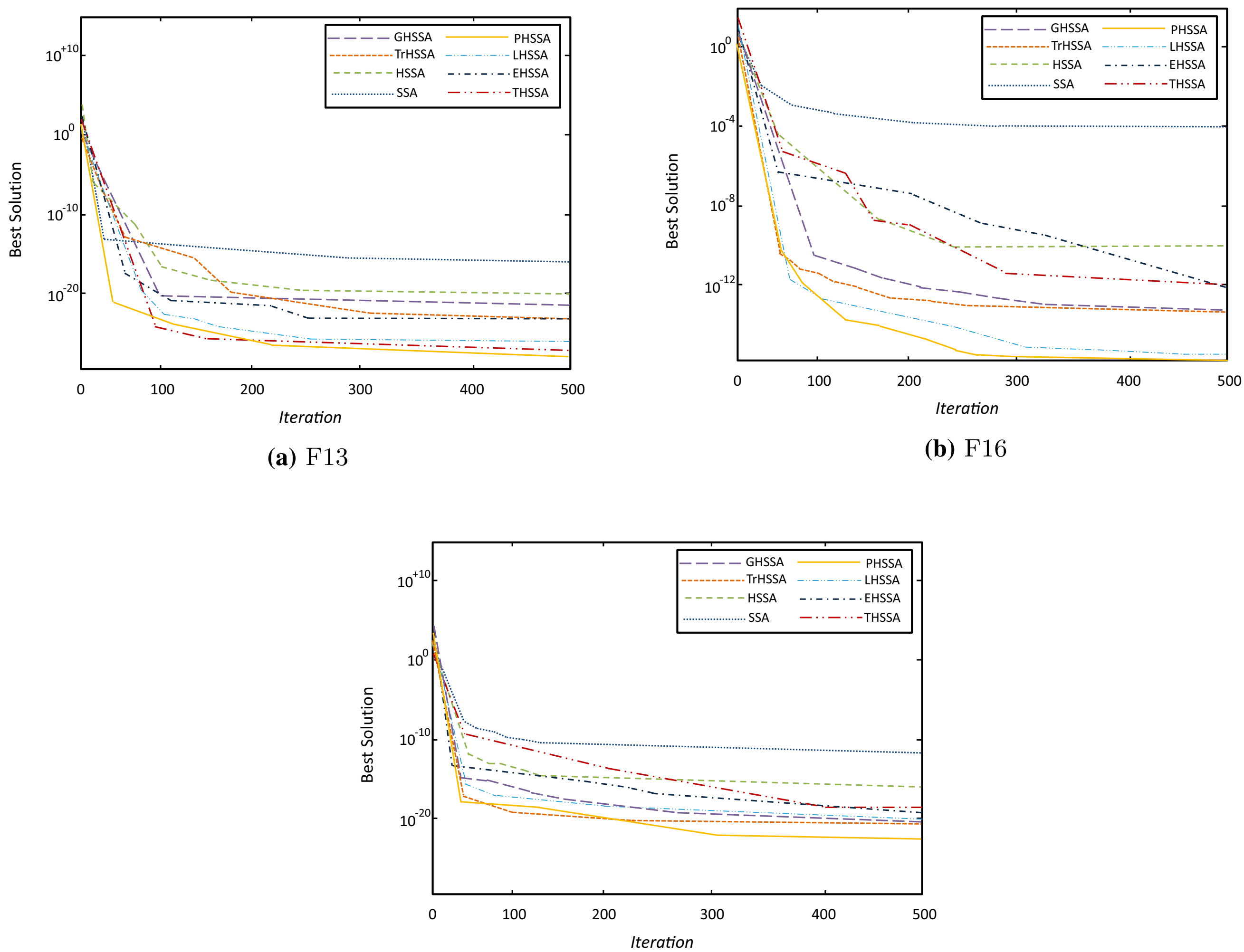
$$\text{Best} = \min_{1 \le i \le N_r} F_i^* \tag{16}$$

where N_r denoted to the number of various runs and F_i^* denoted to the best-obtained value.

The average measure (avg) is utilized to calculate the mean of the best-obtained values by the algorithm to be evaluated for several predefined numbers of runs, which can be measured as follows:

$$\mu_F = \frac{1}{N_r} \sum_{i=1}^{N_r} F_i^* \tag{17}$$

The standard deviation (std) is a measure utilized to test if the algorithm to be evaluated can obtain the same best value



(a) F13 **(b)** F16 **(c)** F20

Fig. 3 Convergence graphs of the multimodal benchmark functions

in several various runs and examine the repeatability test of the algorithm results, which can be measured as follows:

$$\mathrm{STD}_F = \sqrt{\frac{1}{N_r - 1}\sum_{i=1}^{N_r}(F_i - \mu_F)^2} \tag{18}$$

Also, convergence trajectories are shown to display the behavior of the comparative algorithms in order to give the optimal value.

4.1.3 Sensitivity analysis

In this section, comprehensive investigations are distributed into two experimental series; in the first series, a set of experiments are conducted to evaluate the influence of the dimensional spaces on the results of the SSA. In the second series, a set of experiments are conducted to evaluate the influence of the population sizes.

Experiment series 1: influence of the dimensional spaces

In this part, to analyze the influence of the problem dimensional spaces, experiments are produced for several potential dimensional spaces (i.e., $D = 5, 10, 15, 20, 25, 30, 35, 40, 45,$ and 50) as reported in the literature using the utilized 30 benchmark functions. The results for 30 functions are illustrated in Table 4 using the best normalized values. The best results have been written in bold font.

As shown in Table 4, the SSA obtained the overall best results when $D = 5$; it got the best results on 17 cases. Furthermore, for the10 scalable unimodal functions, the SSA got the most of the best results when $D = 5$ and 30; it got 7 out of 10 best cases in both dimensions. For 12 scalable multimodal functions, the SSA got the most of the best results when $D = 12$; it got 7 out of 12 best cases. For the 8 fixed-dimension multimodal functions, the SSA got the

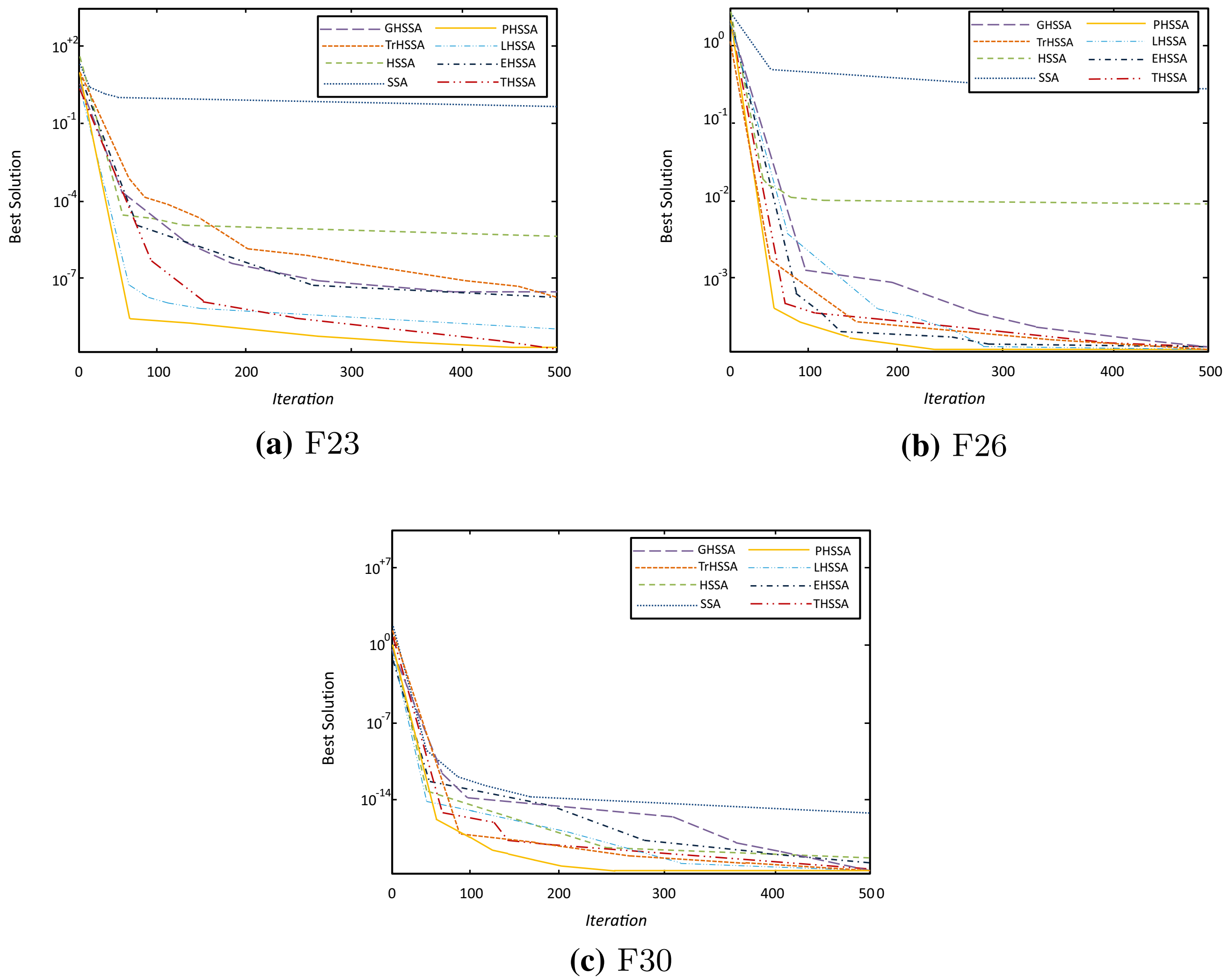


Fig. 4 Convergence graphs of the fixed-dimension multimodal benchmark functions

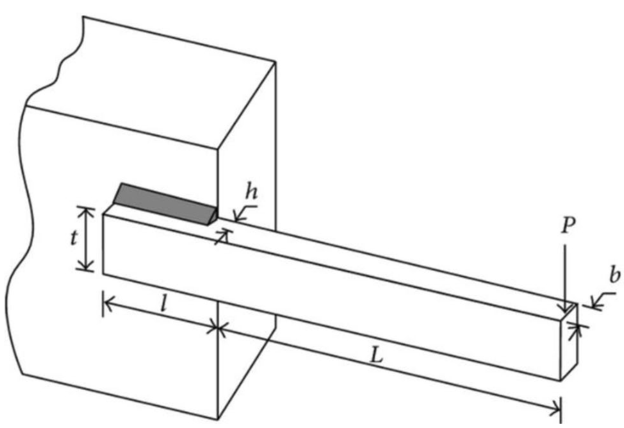


Fig. 5 Welded beam design problem: **a** schematic of the weld; **b** stress pattern evaluated at the optimum design; **c** displacement appearance at the optimum design

most of the best results when $D = 5$; it got 7 out of 8 best cases. From these results, we concluded that increasing the overall performance of SSA is observed by increasing the problem dimensional space. Usually, the SSA is unable to solve the problem before getting the maximum number of iterations. However, as seen, SSA gives better results for high-dimensional problems.

Experiment series 2: influence of the population sizes

In order to demonstrate the influence of the population sizes, the experiments are produced using several values for population sizes (i.e., $P = 5, 10, 15, 20, 50, 100, 250$, and 500) for the utilized 30 benchmark functions. Table 5 shows the results for different population sizes.

As shown in Table 5, we can see that the best-normalized results for SSA with population sizes. The SSA obtained the best results (24 times) when the population size is equal to 20. Furthermore, for the10 scalable unimodal functions, the SSA got the most of the best results when $P = 20$; it got

Table 9 The algorithms results for solving the welded beam design problem

Algorithm	Optimal values				Optimal cost
	h	*l*	*t*	*b*	
SIMPLEX [55]	0.2792	5.6256	7.7512	0.2796	2.5307
DAVID [55]	0.2434	6.2552	8.2915	0.2444	2.3841
APPROX [55]	0.2444	6.2189	8.2915	0.2444	2.3815
GA [56]	0.2489	6.1730	8.1789	0.2533	2.4300
HS [57]	0.2442	6.2231	8.2915	0.2400	2.3807
CSCA [58]	0.203137	3.542998	9.033498	0.206179	1.733461
CPSO [59]	0.202369	3.544214	9.04821	0.205723	1.72802
RO [60]	0.203687	3.528467	9.004233	0.207241	1.735344
WOA [61]	0.205396	3.484293	9.037426	0.206276	1.730499
GSA [62]	0.182129	3.856979	10.000	0.202376	1.87995
MVO [62]	0.205463	3.473193	9.044502	0.205695	1.72645
OBSCA [63]	0.230824	3.069152	8.988479	0.208795	1.722315
PHSSA	0.202369	3.544214	9.04821	0.205723	1.72802

8 out of 10 best cases. For 12 scalable multimodal functions, the SSA got the most of the best results when $P = 20$; it got 10 out of 12 best cases. For the 8 fixed-dimension multimodal functions, the SSA got the most of the best results when $P = 20$; it got 6 out of 8 best cases. It is clearly observed that when the population size is equal to 20, it is the more suitable size for all benchmark test functions

4.1.4 Comparative study

In this section, deeply comprehensive investigations are conducted to check the performance of the proposed methods by two experiments series: in the first series, a set of experiments are conducted to show the performance of the proposed HSSA compared to SSA and other similar optimization algorithms. In the second series, the influence of the proposed six selection schemes is investigated by conducting other experiments. Note, the parameters settings of the comparative algorithms are shown in Table 6.

Experiment series 1: comparisons between HSSA, SSA, and the other algorithms

For a clear comparison, as shown in Table 7, the proposed HSSA is compared with the basic SSA [7] and other similar nine optimization algorithms, namely Ant Bee Colony (ABC) Algorithm [47], Bat-inspired Algorithm (BA) [48], Moth Flame Optimization (MFO) Algorithm [49], Dragonfly Algorithm (DE) [50], Genetic Algorithm (GA) [51], Harmony Search (HS) Algorithm [52], Krill Herd (KH) Algorithm [53], and Grey Wolf Optimizer (GWO) Algorithm [54]. Table 7 shows the best, average (Avg), the standard division (Std) of fitness values obtained by all comparative algorithms over 30 runs, respectively.

As shown in Table 7, the basic SSA has some weakness (weak local search) in achieving excellent results in unimodal functions (i.e., F1, F2, F4, F5, F6, and F9). Consequently, the hybrid SSA with HC is proposed to improve the exploitation searchability of SSA. Thus, functions F1–F10 are scalable unimodal benchmarks since they have just one global optimum. These functions support assessing the exploitation ability of the examined optimization algorithms. It can be seen from Table 7 that HSSA is a very competitive algorithm compared to other similar algorithms. Mainly, it was the most effective algorithm for functions F1 and F10 in most test problems. The proposed HSSA hence provides perfect exploitation. HSSA got better results in solving unimodal functions compared to the proposed HSSA, where

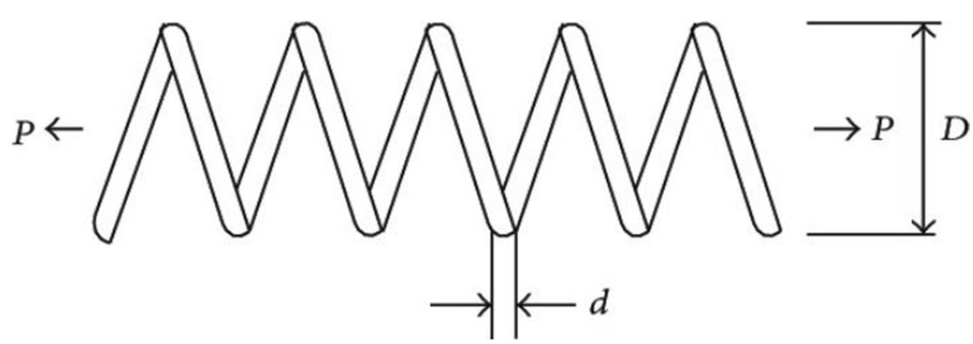


Fig. 6 Tension/compression spring design problem: **a** schematic of the spring; **b** stress pattern evaluated at the optimum design; and **c** displacement pattern evaluated at the optimum design

Table 10 The algorithms results for solving the tension/compression spring design problem

Algorithm	Optimal values			Optimal weight
	d	*D*	*N*	
CC [64]	70.050000	0.315900	14.250000	0.0128334
GA [65]	0.051480	0.351661	11.632201	0.01270478
HS [66]	0.051154	0.349871	12.076432	0.0126706
CSCA [58]	0.051609	0.354714	11.410831	0.0126702
PSO [59]	0.051728	0.357644	11.244543	0.0126747
CPSO [59]	0.051728	0.357644	11.244543	0.0126747
ES [67]	0.051643	0.355360	11.397926	0.012698
RO [60]	0.051370	0.349096	11.76279	0.0126788
WOA [61]	0.051207	0.345215	12.004032	0.0126763
GSA [62]	0.050276	0.323680	13.525410	0.0127022
MVO [62]	0.05251	0.37602	10.33513	0.012790
OBSCA [63]	0.05230	0.31728	12.54854	0.012625
PHSSA	0.331680	12.834269	0.0501910	0.0123947

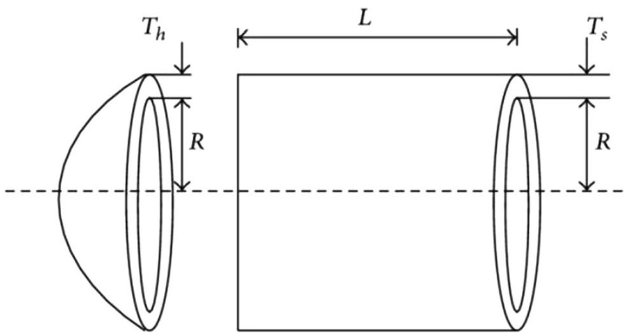


Fig. 7 Pressure vessel design problem: **a** schematic of the vessel; **b** stress pattern assessed at the optimum design; and **c** displacement pattern assessed at the optimum design

it almost obtained all best results in unimodal functions as well in other test functions (i.e., multimodal F11–F22 and fixed-dimension multimodal F23–F30). Finally, although the results indicate that HSSA also has excellent exploration searchability, it is possible to further improve the exploration search to make a balance between exploitation and exploration search. Moreover, performance, diversity, and the convergence rate of HSSA can be enhanced.

Experiment series 2: comparison between selection schemes

In this part, as shown in the previous section that the HSSA can further improve its exploration search abilities, new experiments series are conducted to investigate the skills of the selection schemes in enhancing the global search abilities. Various selection scheme mechanisms (tournament selection scheme (THSSA), proportional selection scheme (PHSSA), linear ranking selection scheme (LHSSA), exponential ranking selection scheme (EHSSA), greedy-based selection scheme (GHSSA), and truncation selection scheme (TrHSSA)) have been tested on the HSSA to improve its exploration search abilities.

Contrary to unimodal functions, multimodal functions cover many local optima, whose number grows exponentially with the number of decision variables (problem size). Accordingly, this kind of benchmark functions becomes very beneficial if the objective is to evaluate the exploration searchability of an optimization algorithm.

Optimization of benchmark functions is a very challenging job because just a precise balance between exploration and exploitation supports local optima to be evaded. Optimization results listed in Table 8 show that the proposed hybrid SSA with HC using proportional selection scheme (PHSSA) is almost the best optimizer in all test problems and overcomes other similar comparative algorithms. It is definitely demonstrated that the proposed PHSSA support exploration and exploitation phases to be balanced. Moreover, the results indicate that PHSSA also has excellent exploration searchability. However, the proposed PHSSA always is the most useful algorithm in the majority of function problems.

Figure 2 shows the convergence graphs of the unimodal benchmark functions (F_1, F_4, and F_7). The convergence graphs are plotted between the best solutions of each algorithm and the number of iterations based on the results acquired through 30 independent runs. It is observed from the convergence graphs of the unimodal functions that the HSSA overcomes the weaknesses of the basic SSH. Also, PHSSA achieved good convergence performance in F_1 and F_7 in comparison with the other proposed algorithms, while the convergence performance of THSSA is the best at F_4 followed by PHSSA and the other proposed algorithms.

The convergence graphs of the multimodal functions (F_{13}, F_{16}, and F_{20}) are illustrated in Fig. 3. Similar to the mentioned above (i.e., unimodal functions) HSSA outperformed the basic SSA, clearly. On the other hand, the convergence performance of the PHSSA achieved best results in all functions [i.e., (F_{13}, F_{16}, and F_{20})]. It is worth to mention that in F_{13} the PHSSA is the fastest method for finding the best solutions in the first part (i.e., until iteration 100) and in the last part (i.e, after iteration 200), where the THSSA achieved the best solutions between 100 and 200 iterations. While in F_{16}, the results of TrHSSA, LHSSA, and PHSSA are very close together until the iteration 100, after that the PHSSA outperformed until the end. Almost the same results were repeated in F_{20}.

Figure 4 illustrates that the results of the hybrid functions (F_{23}, F_{26}, and F_{30}) which are nearly similar to the results shown in Fig. 3, for instance, the superiority of the HSSA over the basic SSA in determining the best solution. In addition, the results in F_{26} and F_{30} for most of the proposed algorithms are very close together with preference to PHSSA, while the results in F_{23} shows the preference for the

Table 11 The algorithms results for solving the pressure vessel design problem

Algorithm	Optimal values				Optimal cost
	T_s	T_h	R	L	
Branch-bound [68]	1.125	0.625	48.97	106.72	7982.5
GA [65]	0.81250	0.43750	42.097398	176.65405	6059.94634
HS [66]	1.125000	0.625000	58.29015	43.69268	7197.730
CSCA [58]	0.8125	0.4375	42.098411	176.63769	6059.7340
PSO-SCA [69]	0.8125	0.4375	42.098446	176.6366	6059.71433
CPSO [59]	0.8125	0.4375	42.091266	176.7465	6061.0777
HPSO [70]	0.8125	0.4375	42.0984	176.6366	6059.7143
ES [67]	0.8125	0.4375	42.098087	176.640518	6059.74560
ACO [71]	0.812500	0.437500	42.098353	176.637751	6059.7258
WOA [61]	0.812500	0.437500	42.0982699	176.638998	6059.7410
GSA [72]	1.125	0.625	55.9886598	84.4542025	8538.8359
MVO [62]	0.8125	0.4375	42.090738	176.73869	6060.8066
OBSCA [63]	1.2500	0.0625	59.1593	70.8437	5833.9892
PHSSA	0.815200	0.426501	42.091254	176.7423141	6043.9861

THSSA. The PHSSA was outperformed the other proposed algorithms.

4.2 The experiments using engineering optimization problems

In this section, we test the proposed algorithm with proportional selection scheme (called PHSSA) using four engineering optimization problems: the welded beam design problem, tension/compression spring design problem, three-bar truss design problem, and pressure vessel design problem. The number of solutions in the experiments is 30 and the maximum number of iterations is 500 to address these problems. The following subsections show the results of the proposed PHSSA compared with the results of the state-of-the-art methods.

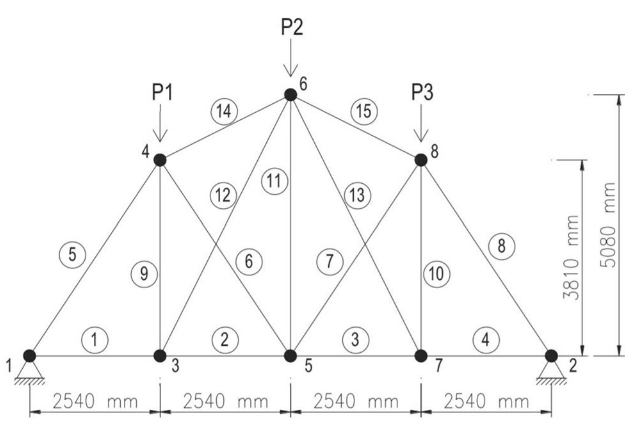


Fig. 8 Construction of a 15-bar truss

4.2.1 Welded beam design problem

The main objective of the welded beam design problem is to find the minimum fabrication cost by defining the optimal value of the given variables (four optimization variables as shown in Fig. 5), namely length of attached part of bar (l), thickness of weld (h), the height of the bar (t), and thickness of the bar (b). The given variables need to be satisfied with seven constraints. The mathematical representation of this problem is described as follows:

Consider $\vec{x} = [x_1 x_2 x_3 x_4] = [hltp]$,

Minimize $f(\vec{x}) = 1.10471x_1^2 x_1 + 0.04811x_3x_4(14.0 + x_2)$

Subject to $g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0$,

$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0$,

$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0$,

$g_4(\vec{x}) = x_1 - x_4 \le 0$,

$g_5(\vec{x}) = p - p_c(\vec{x}) \le 0$,

$g_6(\vec{x}) = 0.125 - x_1 \le 0$,

$g_7(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 0.5 \le 0$,

Variables range $(0.1 \le x_1, x_4 \le 2)$, $(0.1 \le x_2, x_3 \le 10)$,

where

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')}, \tau' = \frac{P}{\sqrt{2x_1x_2}},$$

$$\tau'' = \frac{\mathrm{MR}}{J}, M = P\left(L + \frac{x_2}{2}\right), R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2},$$

$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{0}}}{L^2}, \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),$$

$$J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_x^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\}, \qquad \sigma(\vec{x}) = \frac{6PL}{x_4x_3^2},$$

$$\delta(\vec{x}) = \frac{6PL^3}{Ex_4x_3^2},$$

Note that, $P = 6000$ lb, $L = 14$ in, $\delta_{\max} = 0.25$ in, $E = 30 \times 1^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{\max} = 13{,}600$ psi, and $\sigma_{\max} = 30{,}000$ psi.

Table 12 The algorithms results for solving the 15-bar truss design problem

Algorithm	Optimal values															Optimal weight (kg)
	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	
MSAA [73]	133.2	133.2	133.2	133.2	736.7	133.2	133.2	736.7	133.2	133.2	133.2	133.2	133.2	334.3	334.3	105.740
ACA [74]	133.2	133.2	133.2	133.2	736.7	133.2	133.2	736.7	133.2	133.2	133.2	133.2	133.2	334.3	334.3	105.740
TLBO [75]	133.2	133.2	133.2	133.2	736.7	133.2	133.2	736.7	133.2	133.2	133.2	133.2	133.2	334.3	334.3	105.740
HGA [76]	308.6	174.9	338.2	143.2	736.7	185.9	265.9	507.6	143.2	507.6	279.1	174.9	297.1	235.9	265.9	142.117
HHS [77]	133.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	106.157
PSO [81]	185.9	113.2	143.2	113.2	736.7	143.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	108.84
PSOPC [81]	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	185.9	334.3	334.3	108.96
HPSO [81]	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	105.73
MBA [79]	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	105.735
SOS [80]	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	105.735
WOA [61]	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	105.735
PHSSA	113.2	113.2	113.2	113.2	736.7	113.2	113.2	736.7	113.2	113.2	113.2	113.2	113.2	334.3	334.3	105.735

The proposed algorithm (PHSSA) is applied for solving this engineering problem (welded beam design) and compared it with several optimization algorithms, which are published in the literature; these works are Simplex method (SIMPLEX) [55], Davidon–Fletcher–Powell (DAVID) [55], Griffith and Stewarts successive linear approximation (APPROX) [55], Genetic Algorithm (GA) [56], Harmony Search (HS) [57], Co-evolutionary Differential Evolution (CSCA) [58], Co-evolutionary Particle Swarm Optimization (CPSO) [59], Ray Optimization (RO) [60], Whale Optimization Algorithm (WOA) [61], Gravitational Search Algorithm (GSA) [62], Multi-verse Optimizer (MVO) [62], and Opposition-Based Sine Cosine Algorithm (OBSCA) [63], as shown in Table 9. From Table 9, we concluded that the results of the proposed algorithm are better than all other comparative algorithms. Hence, it can be declared that the proposed PHSSA can find the best possible optimal solution (design) for solving this problem (i.e., welded beam design).

4.2.2 Tension/compression spring design problem

The main objective of the tension/compression spring design problem is to find the minimum weight of the tension/compression spring to satisfy its design constraints: shear stress, surge frequency, and deflection as shown in Fig. 6. Three design variables need to be taken into account: wire diameter (d), mean coil diameter (D), and the number of active coils (N). The mathematical representation of this problem is described as follows:

Consider $\vec{x} = [x_1 x_2 x_3] = [dDN]$,

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

Subject to $g_1(\vec{x}) = 1 - \frac{x_3 x_2^3}{71785 x_1^4} \le 0$,

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

$g_3 = (\vec{x}) = 1 - \frac{140.54 x_1}{x_2^2 x_3} \le 0$,

$g_4 = (\vec{x}) = \frac{x_1 x_2}{1.5} - 1 \le 0$,

Variables range $(0.05 \le x_1 \le 2)$, $(0.25 \le x_2 \le 1.30)$, $(2.00 \le x_3 \le 15)$,

The proposed PHSSA is applied for solving this engineering problem (tension/compression spring design) and compared it with a mathematical technique and optimization algorithms, which are published in the literature; these works are Constraints Correction (CC) [64], Genetic Algorithm (GA) [65], Improved Harmony Search (HS) [66], Co-evolutionary Differential Evolution (CSCA) [58], Particle Swarm Optimization (PSO) [59], Co-evolutionary Particle Swarm Optimization (CPSO) [59], Evolution Strategy (ES) [67], Ray Optimization (RO) [60], Whale Optimization Algorithm (WOA) [61], Gravitational Search Algorithm (GSA) [62], Multi-verse Optimizer (MVO) [62], and Opposition-Based Sine Cosine Algorithm (OBSCA) [63], as shown in Table 10. The obtained results of the proposed PHSSA are compared with the literature in Table 10. It can be observed that PHSSA outperforms all other algorithms except OBSCA.

4.2.3 Pressure vessel design problem

The main objective of the pressure vessel design problem is to find the overall cost of the cylindrical pressure vessel to satisfy its design constraints: forming, material, and welding as shown in Fig. 7. Both edges of the vessel are capped while the top has a hemispherical shape. Four design variables need to be taken into account in the optimization operations to satisfy its four constraints: the inner radius (R), the thickness of the head (T_{h}), thickness of the shell (T_{s}), and the length of the cylindrical part without examining the head (L). The mathematical representation of this problem is described as follows:

Consider $\vec{x} = [x_1 x_2 x_3 x_4] = [T_{\text{s}} T_{\text{h}} R L]$,

Minimize $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$,

Subject to $g_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0$,

$g_2(\vec{x}) = -x_3 + 0.000954x_3 \leq 0$,

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

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

Variables range $(0 \leq x_1, x_2 \leq 99)$, $(10 \leq x_3, x_4 \leq 200)$,

The obtained results by the proposed HHSA for solving this problem (pressure vessel design problem) are compared with other several optimization algorithms, which are published in the literature; these works are Nonlinear Integer and Discrete Programming (Branch-bound) [68], Genetic Algorithm (GA) [65], Improved Harmony Search (HS) [66], Co-evolutionary Differential Evolution (CSCA) [58], Hybridizing Particle Swarm Optimization with Differential Evolution (PSO-SCA) [69], Co-evolutionary Particle Swarm Optimization (CPSO) [59], Hybrid Particle Swarm Optimization with a Feasibility-based Rule (HPSO) [70], Evolution Strategy (ES) [67], Improved Ant Colony Optimization (ACO) [71], Whale Optimization Algorithm (WOA) [61], Gravitational Search Algorithm (GSA) [72], Multi-verse Optimizer (MVO) [62], and Opposition-Based Sine Cosine Algorithm (OBSCA) [63], as shown in Table 11. From this table, we concluded that the results of PHSSA are better than almost all other comparative algorithms. It can be observed that PHSSA outperforms all other algorithms except OBSCA.

4.2.4 15-Bar truss design problem

This main objective of this 15-bar truss design problem (it is considered as a discrete problem) is to find the minimum weight of a 15-bar truss to satisfy its 46 design constraints:15 compression, 15 tension, and 16 displacement constraints. There are also eight nodes and fifteen bars as shown in Fig. 8. Consequently, there are 15 variables. It also may be observed in this figure that three loads are connected to the nodes P1, P2, and P3. The mathematical representation of this problem is described as follows:

- $\rho = 7800\,\text{kg/m}^3$
- $E = 200\,\text{GPa}$
- Stress limitation = 120 MPa
- Maximum stress = 115.37 MPa
- Displacement limitation = 10 mm
- Maximum displacement = 4.24 mm

Design variable set = $\left\{\begin{array}{l} 113.2,\ 143.2,\ 145.9,\ 174.9,\ 185.9,\ 235.9,\ 265.9,\ 297.1 \\ 308.6,\ 334.3,\ 338.2,\ 497.8,\ 507.6,\ 736.7,\ 791.2,\ 1063.7 \end{array}\right\}$,

Three different sets of cases (loads) have been used in solving this problem as stated in the literature, which is as follows:

- Case 1: $P_1 = 35\,\text{kN}$, $P_2 = 35\,\text{kN}$, $P_3 = 35\,\text{kN}$
- Case 2: $P_1 = 35\,\text{kN}$, $P_2 = 0\,\text{kN}$, $P_3 = 35\,\text{kN}$
- Case 3: $P_1 = 35\,\text{kN}$, $P_2 = 0\,\text{kN}$, $P_3 = 0\,\text{kN}$

The obtained results by the proposed HHSA for solving this problem (15-bar truss design problem) are compared with other several optimization algorithms, which are published in the literature; these works are Modified Simulated Annealing Algorithm (MSAA) [73], Ant Colony Algorithm (ACA) [74], Teaching-Learning-Based Optimization (TLBO) [75], Improved Hybrid Genetic Algorithm (HGA) [76], Hybrid Harmony Search algorithm (HHS) [77], Particle Swarm Optimizer (PSO) [78], Particle Swarm Optimizer with Passive Congregation (PSOPC) [78], Heuristic Particle Swarm Optimizer (HPSO) [78], Mine Blast Algorithm (MBA) [79], Symbiotic Organisms Search (SOS) [80], and Whale Optimization Algorithm (WOA) [61], as shown in Table 12. As this problem is considered as a discrete problem, the search solutions of PHSSA were rounded to the nearest integer number through the optimization processes. In Table 12, we concluded that the results of PHSSA are better than almost all comparative algorithms. PHSSA can give very competitive results in addressing this problem as well as its results are similar to HPSO, MBA, SOS, and WOA. The proposed method got the advantages of SSA and HC, which increase the efficiency of the proposed method. The hybrid method got just the benefits of both combined algorithms with no disadvantage according to the effectiveness measures.

For further observation, the results of HC-based selection schemes improved the candidate solutions of the SSA. The obtained results showed how the SSA got promising results by applying the HC-based selection schemes to maximize capacity for both strategies. These results demonstrated the merits of the proposed hybrid method in solving complicated problems with wide search spaces. Hence, this robust optimization method is offered as a mechanism for determining the optimal solutions of optimization problems in various areas of study.

5 Conclusions and future directions

We presented a new two-stage variant of the Salp Swarm Algorithm (SSA). In the first stage, the basic SSA is hybridized with hill climbing (HC) local search to improve its exploitation search, while in the second stage, a selection scheme is applied to enhance the exploration capabilities of the algorithm. Six selection schemes were considered, and the proportional was selected as it yielded the best performance.

Experiments were conducted using thirty benchmark functions and four engineering design problems. We compared our algorithm to a number of similar algorithms published in the literature. The effectiveness of each algorithm was evaluated based on three measures, the best, average, and standard deviation of the fitness values. The results showed that the proposed hybrid SSA method using the proportional selection scheme (PHSSA) was the best optimizer on almost all test problems. In summary, by providing an appropriate balance between exploration and exploitation and by maintaining the diversity of solutions, our proposed PHSSA algorithm was able to demonstrate results on the engineering design problems that were at least comparable and in many cases superior to SSA and similar algorithms in the literature.

In future work, we will consider other algorithms to create new hybrid versions, and we will apply them to different optimization problems, including multi-objective problems.

References

1. Ewees AA, Elaziz MA, Houssein EH (2018) Improved grasshopper optimization algorithm using opposition-based learning. Expert Syst Appl 112:156–172
2. Abualigah L (2020) Multi-verse optimizer algorithm: a comprehensive survey of its results, variants, and applications. Neural Comput Appl. https://doi.org/10.1007/s00521-020-04839-1
3. Xu Y, Chen H, Heidari AA, Luo J, Zhang Q, Zhao X, Li C (2019) An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks. Expert Syst Appl 129:135–155
4. Osman IH, Laporte G (1996) Metaheuristics: a bibliography. Springer, Berlin
5. Sörensen K (2015) Metaheuristics—the metaphor exposed. Int Trans Oper Res 22:3–18
6. Abualigah LM, Khader AT, Hanandeh ES (2019) Modified krill herd algorithm for global numerical optimization problems. In: Shandilya S, Shandilya S, Nagar A (eds) Advances in nature-inspired computing and applications. Springer, Berlin, pp 205–221
7. 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
8. Abualigah L, Diabat A (2020) A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Cluster Comput. https://doi.org/10.1007/s10586-020-03075-5
9. Shehab M, Alshawabkah H, Abualigah L, Nagham A-M (2020) Enhanced a hybrid moth-flame optimization algorithm using new selection schemes. Eng Comput. https://doi.org/10.1007/s00366-020-00971-7
10. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
11. Yıldız BS, Yıldız AR (2019) The Harris Hawks optimization algorithm, salp swarm algorithm, grasshopper optimization algorithm and dragonfly algorithm for structural design optimization of vehicle components. Mater Test 61:744–748
12. Abbassi R, Abbassi A, Heidari AA, Mirjalili S (2019) An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models. Energy Convers Manage 179:362–372
13. Qais MH, Hasanien HM, Alghuwainem S (2019) Enhanced salp swarm algorithm: application to variable speed wind generators. Eng Appl Artif Intell 80:82–96
14. Singh N, Chiclana F, Magnot J-P et al (2020) A new fusion of salp swarm with sine cosine for optimization of non-linear functions. Eng Comput 36:185–212
15. Rizk-Allah RM, Hassanien AE, Elhoseny M, Gunasekaran M (2019) A new binary salp swarm algorithm: development and application for optimization tasks. Neural Comput Appl 31:1641–1663
16. Kaur S, Awasthi LK, Sangal A (2020) HMOSHSSA: a hybrid meta-heuristic approach for solving constrained optimization problems. Eng Comput. https://doi.org/10.1007/s00366-020-00989-x
17. Mirjalili S, Hashim SZM, Sardroudi HM (2012) Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl Math Comput 218:11125–11137
18. Kao Y-T, Zahara E, Kao I-W (2008) A hybridized approach to data clustering. Expert Syst Appl 34:1754–1762
19. Abualigah LM, Khader AT, Hanandeh ES (2018) A combination of objective functions and hybrid krill herd algorithm for text document clustering analysis. Eng Appl Artif Intell 73:111–125
20. Abualigah LMQ (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Berlin
21. Abualigah LM, Khader AT, Hanandeh ES (2018) Hybrid clustering analysis using improved krill herd algorithm. Appl Intell 48:4047–4071
22. Bairathi D, Gopalani D (2018) Opposition based salp swarm algorithm for numerical optimization. In: International conference on intelligent systems design and applications. Springer, pp 821–831
23. Moghdani R, Abd Elaziz M, Mohammadi D, Neggaz N (2020) An improved volleyball premier league algorithm based on sine cosine algorithm for global optimization problem. Eng Comput. https://doi.org/10.1007/s00366-020-00962-8
24. Long W, Cai S, Jiao J, Tang M (2020) An efficient and robust grey wolf optimizer algorithm for large-scale numerical optimization. Soft Comput 24:997–1026
25. Chen D, Zou F, Li Z, Wang J, Li S (2015) An improved teaching–learning-based optimization algorithm for solving global optimization problem. Inf Sci 297:171–190

26. Yang X-S (2010) Nature-inspired metaheuristic algorithms. Luniver Press, Bristol
27. Abualigah L, Shehab M, Alshinwan M, Mirjalili S, Abd Elaziz M (2020) Ant lion optimizer: a comprehensive survey of its variants and applications. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-020-09420-6
28. McCauley DJ, Pinsky ML, Palumbi SR, Estes JA, Joyce FH, Warner RR (2015) Marine defaunation: animal loss in the global ocean. Science 347:1255641
29. Shehab M, Khader AT, Al-Betar M (2016) New selection schemes for particle swarm optimization. IEEJ Trans Electron Inf Syst 136:1706–1711
30. Abualigah L, Shehab M, Alshinwan M, Alabool H (2019) Salp swarm algorithm: a comprehensive survey. Neural Comput Appl. https://doi.org/10.1007/s00521-019-04629-4
31. Shehab M, Khader AT, Laouchedi M (2017) Modified cuckoo search algorithm for solving global optimization problems. In: International conference of reliable information and communication technology. Springer, pp 561–570
32. Shehab M, Khader AT, Laouchedi M, Alomari OA (2019) Hybridizing cuckoo search algorithm with bat algorithm for global numerical optimization. J Supercomput 75:2395–2422
33. Ibrahim RA, Ewees AA, Oliva D, Elaziz MA, Lu S (2019) Improved salp swarm algorithm based on particle swarm optimization for feature selection. J Ambient Intell Human Comput 10:3155–3169
34. Abusnaina AA, Ahmad S, Jarrar R, Mafarja M (2018) Training neural networks using salp swarm algorithm for pattern classification. In: Proceedings of the 2nd international conference on future networks and distributed systems. ACM, p 17
35. Grefenstette JJ (1989) How genetic algorithms work: a critical look at implicit parallelism. In: Genetic algorithm and their applications: proceedings of third international conference of genetic algorithm
36. Blickle T, Thiele L (1995) A mathematical analysis of tournament selection. In: ICGA, vol 95. Citeseer, pp 9–15
37. Mitchell M (1998) An introduction to genetic algorithms. MIT Press, Cambridge
38. Oladele R, Sadiku J (2013) Genetic algorithm performance with different selection methods in solving multi-objective network design problem. Int J Comput Appl 70:5–9
39. Sharma P, Wadhwa A (2014) Analysis of selection schemes for solving an optimization problem in genetic algorithm. Int J Comput Appl 93:1–3
40. Holland JH et al (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge
41. Razali NM, Geraghty J et al (2011) Genetic algorithm performance with different selection strategies in solving TSP. In: Proceedings of the world congress on engineering, vol 2. International Association of Engineers Hong Kong, pp 1–6
42. Hancock PJ (1994) An empirical comparison of selection methods in evolutionary algorithms. In: AISB workshop on evolutionary computing. Springer, pp 80–94
43. Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization. Swarm Intell 1(1):33–57
44. Schlierkamp-Voosen D, Mühlenbein H (1993) Predictive models for the breeder genetic algorithm. Evol Comput 1:25–49
45. Bäck T (1995) Generalized convergence models for tournament- and (μ, λ)-selection. In: Proceedings of the sixth international conference on genetic algorithms, pp 2–8
46. Volkovs M, Chiang F, Szlichta J, Miller RJ (2014) Continuous data cleaning. In: 2014 IEEE 30th international conference on data engineering. IEEE, pp 244–255
47. Karaboga D, Basturk B (2007) Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In: International fuzzy systems association world congress. Springer, pp 789–798
48. Yang X-S (2010) A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010). Springer, pp 65–74
49. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249
50. Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27:1053–1073
51. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3:95–99
52. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76:60–68
53. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17:4831–4845
54. Singh N (2018) A modified variant of grey wolf optimizer. Int J Sci Technol Sci Iran. https://doi.org/10.24200/sci.2018.50122.1523
55. Ragsdell K, Phillips D (1976) Optimal design of a class of welded structures using geometric programming. J Eng Ind 98:1021–1025
56. Deb K (1991) Optimal design of a welded beam via genetic algorithms. AIAA J 29:2013–2015
57. Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 194:3902–3933
58. Huang F-Z, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186:340–356
59. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20:89–99
60. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294
61. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
62. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27:495–513
63. Elaziz MA, Oliva D, Xiong S (2017) An improved opposition-based sine cosine algorithm for global optimization. Expert Syst Appl 90:484–500
64. Arora JS (2004) Introduction to optimum design. Elsevier, Amsterdam
65. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41:113–127
66. Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188:1567–1579
67. Mezura-Montes E, Coello CAC (2008) An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J Gen Syst 37:443–473
68. Sandgren E (1990) Nonlinear integer and discrete programming in mechanical design optimization. J Mech Des 112:223–229
69. Liu H, Cai Z, Wang Y (2010) Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. Appl Soft Comput 10:629–640
70. He Q, Wang L (2007) A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. Appl Math Comput 186:1407–1422
71. Kaveh A, Talatahari S (2010) An improved ant colony optimization for constrained engineering design problems. Eng Comput 27:155–182

72. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
73. Millán-Páramo C (2018) Modified simulated annealing algorithm for discrete sizing optimization of truss structure. Jordan J Civ Eng 12:683
74. Sabour M, Eskandar H, Salehi P (2011) Imperialist competitive ant colony algorithm for truss structures. World Appl Sci J 12:105–2011
75. Dede T (2014) Application of teaching-learning-based-optimization algorithm for the discrete optimization of truss structures. KSCE J Civ Eng 18:1759–1767
76. Zhang Y-N, Liu P, Liu B, Zhu C-Y, Li Y (2005) Application of improved hybrid genetic algorithm to optimized design of architecture structures. Huanan Ligong Daxue Xuebai (Ziran Kexue Ban)/J South China Univ Technol (Natural Science Edition) (China) 33:69–72
77. Cheng M-Y, Prayogo D, Wu Y-W, Lukito MM (2016) A hybrid harmony search algorithm for discrete sizing optimization of truss structure. Autom Constr 69:21–33
78. Li L, Huang Z, Liu F (2009) A heuristic particle swarm optimization method for truss structures with discrete variables. Comput Struct 87:435–443
79. Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2012) Mine blast algorithm for optimization of truss structures with discrete variables. Comput Struct 102:49–63
80. Cheng M-Y, Prayogo D (2014) Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput Struct 139:98–112
81. Oftadeh R, Mahjoob M, Shariatpanahi M (2010) A novel metaheuristic optimization algorithm inspired by group hunting of animals: hunting search. Comput Math Appl 60:2087–2098