Mathematical Biosciences and Engineering

MBE, 19(8): 7756–7804.
DOI: 10.3934/mbe.2022364
Received: 12 April 2022
Revised: 28 April 2022
Accepted: 16 May 2022
Published: 25 May 2022

http://www.aimspress.com/journal/MBE

Research article

Velocity clamping-assisted adaptive salp swarm algorithm: balance analysis and case studies

Hongwei Ding[1,2], Xingguo Cao[1,2], Zongshan Wang[1,2,*], Gaurav Dhiman[3,4,5], Peng Hou[6,*], Jie Wang[7], Aishan Li[8], and Xiang Hu[9]

[1] School of Information Science and Engineering, Yunnan University, Kunming 650500, China
[2] University Key Laboratory of Internet of Things Technology and Application, Yunnan Province, Kunming 650500, China
[3] Department of Computer Science, Government Bikram College of Commerce, Patiala, India
[4] University Centre for Research and Development, Department of Computer Science and Engineering, Chandigarh University, Gharuan, Mohali, India
[5] Department of Computer Science and Engineering, Graphic Era Deemed to be University, Dehradun, India
[6] School of Computer Science, Fudan University, Shanghai 200433, China
[7] School of Mechanical and Power Engineering, Zhengzhou University, Zhengzhou 450000, China
[8] Rackham Graduate School, University of Michigan, Ann Arbor, USA
[9] College of Information, Shanghai Ocean University, Shanghai, China

* **Correspondence:** Email: wzs694513739@163.com.

Abstract: Salp swarm algorithm (SSA) is a recently proposed, powerful swarm-intelligence based optimizer, which is inspired by the unique foraging style of salps in oceans. However, the original SSA suffers from some limitations including immature balance between exploitation and exploration operators, slow convergence and local optimal stagnation. To alleviate these deficiencies, a modified SSA (called VC-SSA) with velocity clamping strategy, reduction factor tactic, and adaptive weight mechanism is developed. Firstly, a novel velocity clamping mechanism is designed to boost the exploitation ability and the solution accuracy. Next, a reduction factor is arranged to bolster the exploration capability and accelerate the convergence speed. Finally, a novel position update equation is designed by injecting an inertia weight to catch a better balance between local and global search. 23 classical benchmark test problems, 30 complex optimization tasks from CEC 2017, and five engineering design problems are employed to authenticate the effectiveness of the developed VC-SSA. The experimental results of VC-SSA are compared with a series of cutting-edge metaheuristics. The comparisons reveal that VC-SSA provides better performance against the canonical SSA, SSA variants, and other well-established metaheuristic paradigms. In addition, VC-SSA is utilized to handle a mobile robot path planning task. The results show that VC-SSA can provide the best results compared to the competitors and it can serve as an auxiliary tool for mobile robot path planning.

Keywords: salp swarm algorithm; swarm intelligence; numerical optimization; engineering design optimization; robot path planning

1. Introduction

Global optimization problems exist widely in scientific research and engineering applications. In the past, optimization problems were solved using methods such as Newton's method, Lagrange multiplier method, quasi-Newton, and other accurate calculation methods. Although the effectiveness of these approaches has long been proven, due to the increasing of data complexity and dimensionality, all information in the feature space is unknown in various fields such as machine learning and engineering design, so the optimal solutions to all optimization problems cannot be obtained using exact mathematical models. Finding optimal solution has become a challenge in the field of optimization. To solve this problem, researchers have created metaheuristic algorithms. It is a stochastic optimization technique that does not require any prior knowledge and can obtain approximate optimal solutions through established steps 1. Swarm intelligence-based algorithms are an essential branch of nature inspired algorithms and have become a research hot topic in the metaheuristic community, which include but are not limited to: particle swarm algorithm (PSO) 2, marine predators algorithm (MPA) 3, ant colony optimization (ACO) 4, earthworm optimization algorithm (EWA) 5, artificial bee colony (ABC) algorithm [6,7, moth flame optimizer (MFO) 8, firefly algorithm (FA) 910, whale optimization algorithm (WOA) 11, cuckoo search (CS) algorithm 12, grey wolf optimization (GWO) algorithm 13, krill herd (KH) algorithm 14, harris hawks optimizer (HHO) 15, monarch butterfly optimization (MBO) 16, grasshopper optimization algorithm (GOA) 17, elephant herding optimization (EHO) 18, slime mould algorithm (SMA) 19, hunger games search (HGS) 20, Runge Kutta method (RUN) 21, colony predation algorithm (CPA) 22, weighted mean of vectors (INFO) 23.

In this work, we focus on a relatively novel and well-established population intelligence-based approach, called salp swarm optimizer (SSA) 24, which was developed by Mirjalili et al. in 2017. Compared with other well-known metaheuristics, SSA has several merits, including few control parameters, straightforward framework, and distinctive search pattern. Numerous studies conducted on SSA have shown that the optimizer demonstrates competitive performance on function optimization and real-world engineering optimization problems. Consequently, SSA is extensively adopted to tackle various global optimization tasks. In 25, Wang et al. developed a dynamic version of SSA by introducing a boosted orthogonal opposition-based learning component and ranking-based learning tactic to the basic SSA. The boosted orthogonal opposition-based learning component helps the leader to search for unknown promising regions to improve the convergence speed. The dynamic learning mechanism enhances the dynamic nature of the follower's position update pattern, thus helping the algorithm avoid getting trapped in local optima. A series of experiments, including numerical function optimization, engineering design problems, and parameter estimation of PV model, validate that this SSA variant outperforms the basic SSA, well-known SSA variants, and frontier metaheuristic techniques. In addition, the developed SSA-based algorithm is employed to solve the mobile robot path planning problem and its performance is tested on five environmental maps. Simulation results show that the proposed path planning approach has competitive performance. In 26, Ewees et al. proposed an enhanced SSA algorithm based on the operators of FA, called SSAFA. In SSAFA, the operators of FA are used to improve the exploitation ability of the basic SSA. The performance of this SSA variant was verified on the unrelated parallel machine scheduling problem. In 27, Tu et al. developed a quantum-behaved SSA algorithm. The proposed method introduces quantum mechanics theories and trajectory analysis in the position update mechanism to enhance the local search ability of followers. The algorithm is applied to solve the localization problem in anisotropic wireless sensor networks. Experimental results show that this method outperforms its competitors in terms of accuracy and robustness against network anisotropy. In 28, Tubishat et al. suggested a novel SSA variant that includes two modifications. The first refinement includes the use of opposition-based learning mechanism during the population initialization period to increase the population diversity and to speed up the convergence of the algorithm. The second enhancement includes the introduction of a new local search mechanism to boost the local search ability. The performance of the developed method was validated using the feature selection problem. The experimental results indicate that this approach achieves better performance in terms of fitness values, accuracy, convergence curves, and feature reduction on most of the employed 18 datasets from UCI repository compared with the other four baseline approach.

Although SSA has demonstrated more competitive performance than some well-known metaheuristic approaches in solving numerical and engineering optimization problems, it still has some drawbacks, including unbalanced exploitation and exploration properties, poor convergence performance, and prone to fall into local optimal. To mitigate the above-mentioned limitations, many improved SSA-based algorithms have been developed. In 29, a novel SSA variant incorporating three perturbation mechanism was developed. Gaussian mutation was utilized to enhance neighborhood-informed ability. Cauchy mutation was employed to improve the global search capability. Levy-flight mechanism was introduced to boost randomness. The performance of the method was investigated using 23 classical benchmark functions. The experimental results show that the three mutation schemes can effectively augment the exploration and exploitation abilities. In 30, a hybrid improved whale optimization salp swarm algorithm was introduced. First, the linear relationships in WOA are replaced with exponential relationships to improve its search ability. Second, the improved WOA algorithm is fused with the basic SSA, and the combined algorithm is more flexible and has a faster convergence rate and higher convergence accuracy. The performance of the algorithm is verified using 23 classical benchmark functions and compared with 8 swarm intelligent algorithms. The comparison results demonstrate that the algorithm has good performance. In addition, the method was used to tune the adaptive PID controller. From the simulation results, the enhancement of the algorithm is demonstrated by different performance metrics. In 31, an improved SSA injected with random opposition-based learning, multiple leadership, and simulated annealing mechanisms was developed. The random opposition-based learning strategy facilitates the expansion of the population search. The multi-leader mechanism can improve the exploration ability. The simulated annealing component enhances the local search ability and is beneficial to improve the solution accuracy. The performance of the algorithm was tested on CEC2015 benchmark function. The algorithm was also used to train feedforward neural networks. The experimental results show that the developed approach has excellent performance on both numerical optimization problems and training feedforward neural network problems. In 32, the SSA algorithm was enhanced by adding three simple but effective strategies. Firstly, the control parameter in SSA responsible for equilibrating exploration and exploitation is chaotically altered to catch a better tradeoff between global search and local search. Secondly, a mutualistic strategy is used to augment the neighborhood interaction between leaders. Finally, a stochastic technique is adopted to enrich the diversity of the population to expand the search range of the followers. To verify the effectiveness of this algorithm, numerical optimization problems and practical engineering design cases are applied. Experimental results demonstrate that the algorithm outperforms its competitors. In 33, an improved SSA algorithm with Gaussian Barebone mechanism and Stochastic Fractal Search strategy, called GBSFSSA, was introduced. In GBSFSSA, Gaussian Barebone mechanism and Stochastic Fractal Search strategy effectively enhance the balance between exploration and exploitation of the basic SSA algorithm. Experimental results on CEC2017 functions show that GBSFSSA has competitive performance. Moreover, this approach is applied to solve image segmentation problems and the effectiveness of the proposed methodology is tested on COVID-19 CT images. The experimental results show that GBSFSSA is a trustworthy image segmentation tool. In 34, differential evolution and chaotic structure were embedded in the basic SSA to improve its convergence performance. Chaotic initialization can enrich the population diversity of the initial population and thus speed up the convergence rate. The differential evolution mechanism is introduced in the search process to make a more solid balance between exploration and exploitation. The improved SSA algorithm is tested using the CEC2014 benchmark functions and five classical engineering design problems, and the results show that the advocated method outperforms the basic SSA and some popular swarm intelligent algorithms. In addition, this SSA variant was used to solve the feature selection problem and obtained desirable results. In 35, the location update mechanism of the basic SSA was revised and an elite gray wolf domination strategy was inserted in the final stage of population location update. The modified location update pattern enhances the local search ability of the algorithm, and the elite domination strategy improves the convergence speed of the basic SSA. A series of experiments including numerical optimization problems, engineering design cases, and feature selection tasks verified the superior performance of the proposed algorithm. In 36, a novel annealing salp swarm algorithm named SASSA was proposed. In SASSA, the simulated annealing strategy is introduced in the follower position update phase, random walk in simulated annealing is implemented using levy flight, and the number of leaders is increased. With the help of these three mechanisms, the SSA algorithm achieves a better and solid balance between global search and local search. Experimental results on 30 CEC2017 test functions show that the SASSA algorithm outperforms the standard SSA and 23 peer algorithms. Additionally, the SASSA method is used to solve five engineering design problems and the fertilizer effect function problem, and the experimental results indicate that SASSA has strong competitiveness. In 37, a levy flight-based population position update mechanism was proposed to replace the location update pattern of the basic SSA. The new location update pattern improves the population diversity. Then, the search technique of HHO was introduced for improving the global search capability of the algorithm. The results of comparison experiments on CEC2014 test function set validate the superiority of the proposed method. This algorithm is also used to segment breast cancer microscopic images and the experimental results demonstrate its superior performance.

The existing SSA variants mainly focus on achieving a delicate balance between cohesion and alignment patterns. Despite the development of many effective SSA variants, a research gap still exists in this field. While the existing SSA variants improve the overall performance of the basic SSA, unbalanced exploitation and exploration operators still exist. This is mainly because the existing SSA variants mainly emphasize on improving the exploration or development capability of the algorithm in expectation of achieving a balance between the two, but there is no algorithm that can smoothly switch between development and exploration. Therefore, in this study, a novel SSA variant, called VC-SSA, is reported to intensify the overall performance of the standard SSA. The main contributions of this paper are summarized as follows:

1) A novel version of SSA named VC-SSA is proposed, which embeds three effective but simple mechanisms, namely, velocity clamping mechanism, reduction factor tactic, and adaptive weight strategy. The ablation study in subsection 4.5 verifies the role of the three components in VC-SSA.
2) To verify the effectiveness of the proposed VC-SSA algorithm, comparison experiments with a number of state-of-the-art swarm intelligent algorithms and SSA variants are carried out using 23 classical benchmark functions and 30 complex optimization problems from IEEE CEC 2017. The results are statistically analyzed by the Friedman test and the Wilcox sign-rank test.
3) To further prove the superiority of VC-SSA, it is employed to five engineering problems and the mobile robot path planning task. The experimental results demonstrate that VC-SSA can achieve satisfactory solutions for the above-mentioned cases.
4) The remainder of this study is arranged as follows: The fundamental principle of the basic SSA is explained in Section 2. Three proposed modified mechanisms are described and the developed VC-SSA algorithm is depicted in Section 3. Section 4 investigates the validity of VC-SSA on twenty-three well-known benchmark problems. Section 5 focus on a VC-SSA-based mobile robot path planning approach. Finally, the conclusions and future directions of our paper are summarized in Section 6. The detailed flow of the work is illustrated in Figure 1.

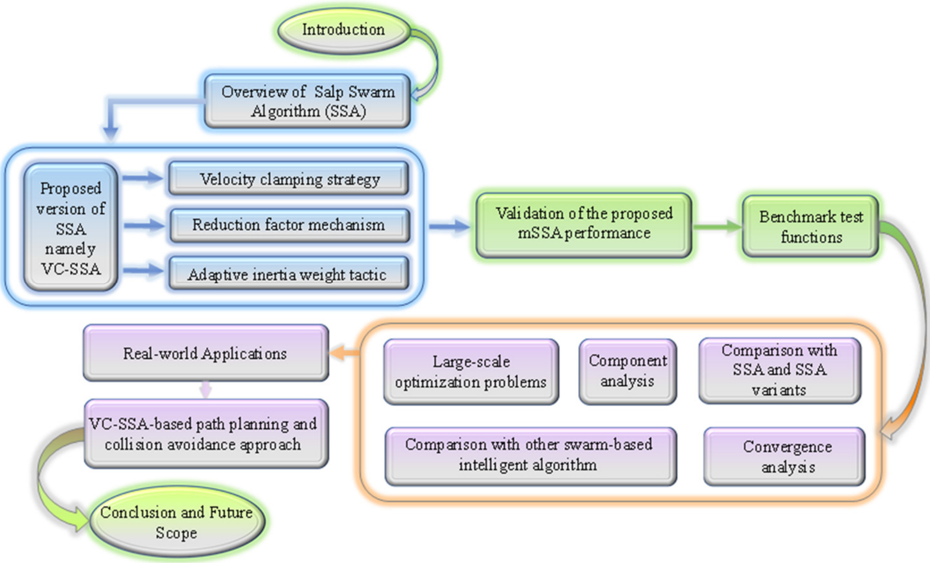

Figure 1. Overview of the paper.

2. Salp swarm algorithm

SSA is a swarm-intelligence based algorithm with a minimalist style developed by Mirjalili et al. [24] in 2017, which is motivated by the unique swarming behavior of the salps found in oceans. In SSA, the search agents performing optimization are divided into leading salps and followers according to their position. The individual at the top of the salp group is considered as the leader, and those behind it are classified as followers. The location updating formula for the leading salp is as follows:

$$x_j^1 = \begin{cases} F_j + c_1((ub_j - lb_j)c_2 + lb_j) & c_3 \geq 0.5 \\ F_j - c_1((ub_j - lb_j)c_2 + lb_j) & c_3 < 0.5 \end{cases} \tag{1}$$

where x_j^1 represent the location of the leading salp, F_j is the parameter that depicts the jth dimension of the food source, ub_j denotes the upper limit of the search space and lb_j shows the lower limit of the search landscape, c_2, and c_3 are random numbers between [0, 1], c_1 is an essential parameter in SSA, which is responsible for maintaining the balance between global search and local exploitation during the search process, and its value is updated by the following mathematical model.

$$c_1 = 2e^{-(4t/T)} \tag{2}$$

where t and T represent the current and the maximum iterations, respectively.

In follower position update phase, the mathematical model is formulated as follows:

$$x_j^i = \frac{1}{2}(x_j^i + x_j^{i-1}) \tag{3}$$

where x_j^i indicates the jth dimension in the position of the ith search agent.

3. Structure of the proposed VC-SSA

The SSA is a population-intelligence based global optimizer with straightforward framework and relatively outstanding performance. However, the strong minimalist style of SSA destabilizes its inherent tradeoff between convergence and diversity and slows down the convergence rate. Therefore, a modified SSA called VC-SSA is proposed for further augmenting the overall performance of the standard SSA in this section. In addition, the NFL theorem 38 demonstrates that the average performance of any optimization technique on all optimization cases is equal, which is the main motivation for us to conduct this study. It is important to state that VC-SSA improves the performance of SSA by introducing new operators while keeping its original framework intact. Consecutive subsections will elaborate on each of the components in detail.

3.1. Velocity clamping strategy

For swarm-intelligence based techniques, the exploration-exploitation trade-off is a core imperative that directly affects the convergence rate and the solution precision. In the standard SSA, the parameter c_1 acting on the leader position update phase is responsible for this aspect. Its value decreases nonlinearly over the course of iterations with the hope of helping the salp chain to develop the solution landscape during the early iterations and to mine the rough position of the global optimal after the lapse of iterations. From Eq (1), the term assisted by c_1, i.e., the vector $((ub_j\text{-}lb_j)c_2\text{+}lb_j)$, can essentially be understood as the leader's velocity. Based on this analysis, Eq (1) can be reformulated as

$$x_j^1 = \begin{cases} F_j + c_1 V_j & c_3 \geq 0.5 \\ F_j - c_1 V_j & c_3 < 0.5 \end{cases} \tag{4}$$

$$V_j = ((ub_j - lb_j)c_2 + lb_j) \tag{5}$$

where V_j means the jth dimension in the velocity of the leading salp.

From Eq (4), during the search process, the leader generates a step size depending on the velocity V and moves around the food source at that step. Nevertheless, according to Eq (5), the step size stemming from the velocity is completely random since the only parameter involved, c_2, is an arbitrary number between [0, 1]. This would lead to the velocity V escaping the constrain of c_1, thus defeating the expectation that a smaller step size should be used during the early search process, while the step size should reduce as the search progresses. To alleviate this problem, this paper proposes a velocity clamping strategy. After the leader generates the velocity, it is adjusted according to the following rule and the position is updated in accordance with the modified velocity.

$$V_j = \begin{cases} V_{j\max} & if\ V_j > V_{\max} \\ V_{j\min} & if\ V_j < V_{\min} \\ V_j & otherwise \end{cases} \tag{6}$$

where $V_{j\max}$ and $V_{j\min}$ are the jth dimension in the maximum and minimum legal velocity of the leader, respectively. To initialize them, Eqs (7) and (8) are employed.

$$V_{j\max} = \delta(ub_j - lb_j) \tag{7}$$

$$V_{j\min} = \delta(lb_j - ub_j) \tag{8}$$

where δ is a constant number called clamping factor.

Overall, with the help of the velocity clamping strategy, the role of the parameter c_1 is enhanced, so that the leader can adequately search the solution space using lager step sizes in the early search phase, while moving with smaller step sizes to improve the explored area in the later search stage.

3.2. Reduction factor strategy

The performance of the basic SSA is improved with the addition of the velocity clamping strategy, but it also brings a pitfall. If the velocity in each iteration is always equal to V_{max} or V_{min}, the search agent may be committed to searching on the boundaries of the search region, or it may continue to hunt on the boundaries of the already located current global optimal area, but unable to refine this promising zone. There are many alternatives to address this challenge. In the current study, we have tackled it by inserting a reduction coefficient into the leader position update equation. The mathematical expression for this new procedure is presented below.

$$x_j^1 = \begin{cases} H(t)\left[F_j + c_1 V_j\right] & c_3 \geq 0.5 \\ H(t)\left[F_j - c_1 V_j\right] & c_3 < 0.5 \end{cases} \tag{9}$$

where $H(t)$ is the reduction factor, and this attenuation is expressed using Eq (10).

$$H(t) = e^{-st} \tag{10}$$

Here, the value of s is defined by Eq (11)

$$s = 2 - c_1 \tag{11}$$

where c_1 is the conversion parameter of SSA.

From Eq (11), s decreases nonlinearly during the iterations, and consequently, H increases dynamically. As a result, the initial H is favorable for global exploration, while after the lapse of iterations, it is preferable for local exploitation. Consequently, the above notion can help the algorithm to obtain a subtle tradeoff between the diversification and intensification operators.

3.3. Adaptive inertia weight mechanism

In the basic SSA, the follower position update pattern adopts a minimalist style, which makes the algorithm easy to accomplish, but also poses some limitations. From Eq (3), the followers update their states only pursuant to the neighboring followers, and this procedure does not have any adaptivity, and the interaction between followers is relatively inflexible. Once a neighboring individual is trapped in the sub-optimal region, all followers behind it will inevitably be lured to that locally optimal region, eventually leading to premature convergence of the optimizer.

To tackle this limitation, we focus on modifying the position update formula of followers and propose an adaptive position update pattern to replace Eq (3). The new mathematical expression is:

$$x_j^i = \frac{1}{2}(x_j^i + \omega x_j^{i-1}) \tag{12}$$

where ω is an inertia weight.

The concept of inertia weights was first proposed for controlling the velocity of particles in PSO and obtained satisfactory results. Inspired by this idea, researchers have introduced this operator to other nature-inspired metaheuristic techniques. For example, Jena et al. 39 designed a sigmoid-adaptive inertia weight and introduced it into the social group optimization (SGO) 40 to improve its overall performance. Sun et al. 41 employed the nonlinear inertia weight factor to reinforce the equilibrium between exploration and exploitation properties of the atom search optimization (ASO) algorithm 42. Ma et al. 43 amended the position update pattern of MFO by utilizing the inertia weights. Li et al. 44 adopted the dynamic inertia weight to amend the position equation of sine cosine algorithm (SCA) 45. In this work, we design a new adaptively inertia weight coefficient and the mathematical formulation is formulated as below:

$$\omega = (\omega_{\max} - \omega_{\min}) \cdot \frac{e^{10-\mu t}}{e^{10-\mu t} + \lambda} + \omega_{\min} \tag{13}$$

where $\omega_{\max}$ and $\omega_{\min}$ represent the maximum and minimum weight values, respectively, λ and μ are constant number.

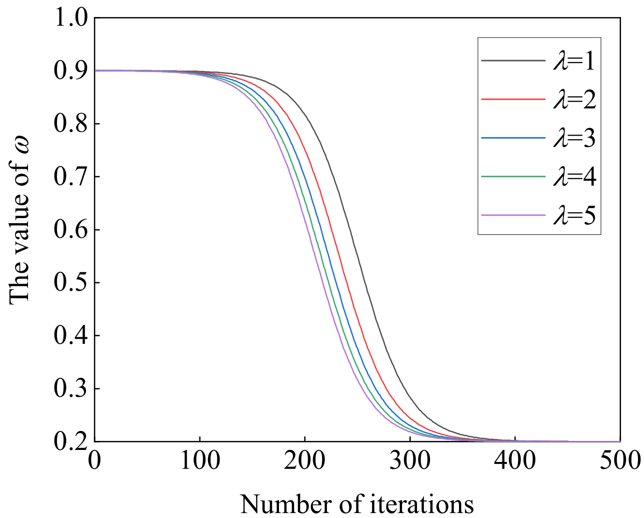

Figure 2. Visualization of proposed inertia weight ω.

Figure 2 draws the evolution of the proposed inertia weight over 500 iterations. The figure indicates that the value of ω decreases nonlinearly during the search process, which is advantageous for the algorithm to explore in the early phase of evolution and switch smoothly to exploit after the iterations have elapsed. Accordingly, the follower can adaptively update the state to balance diversification and intensification.

3.4. VC-SSA algorithm

Based on the above analysis, our modifications to the SSA algorithm are mainly based on the three components introduced. For a more systematic understanding of the proposed VC-SSA, its pseudo-code is given in Algorithm 1, and Figure 3 displays the flowchart of the proposed VC-SSA.

To analyze the calculation complexity of VC-SSA, the steps to calculate the complexity are as follows:

1) The computational complexity of population initialization is O(N), where N is the population scale.
2) The computational complexity of assessing the fitness values of the initial population is O(N).
3) The computational complexity of sorting the initial population according to fitness values and finding the food source position is O(N^2).
4) In the iterative search, the position of each salp is changed and their fitness values are computed. The computational complexity of this part is O(2N).
5) In the final stage of the iteration, the population is sorted and the food source is updated. The computational complexity of this operation if O(N^2).

Therefore, the total time complexity of the proposed VC-SSA is as follows:

$$O(2N) + O(N^2) + t \times (O(2N) + O(N^2)) = (t+1) \times (O(2N) + O(N^2)).$$

The generalized time complexity of the VC-SSA on the problem with a d-dimensional search space is given by

$$O(\text{VC-SSA}) = d \times t \times (O(2N) + O(N^2))$$

According to the above analysis, the computational complexity of VC-SSA does not increase compared to the basic SSA. Analyzed from another perspective, the VC-SSA algorithm focuses on adjusting the position update mechanism of SSA and does not include any additional search phase, so its time complexity is the same as that of the basic SSA.

Algorithm 1: Pseudo-code of VC-SSA

```
Initialize the salp population X_i (i = 1,2,…,N) randomly consider ub and lb
Obtain food source position based on the fitness
while (t < T) do
  Update c_1 by Eq (2)
  for i = 1 to N
     if i == 1 (leader)
        Calculate the velocity by Eq (5)
        Amend the velocity according to Eq (6)
        Calculate the reduction factor H(t) by Eq (10)
        Calculate the new location of the search agent by Eq (9)
     else if (follower)
        Calculate the new location of the search agent by Eq (12)
     end if
  end for
  Amend the search agent based on the boundaries
  Estimate the salp population with respect to the fitness
  Update the food source location
  t = t + 1
  end while
  Output the food source position
```

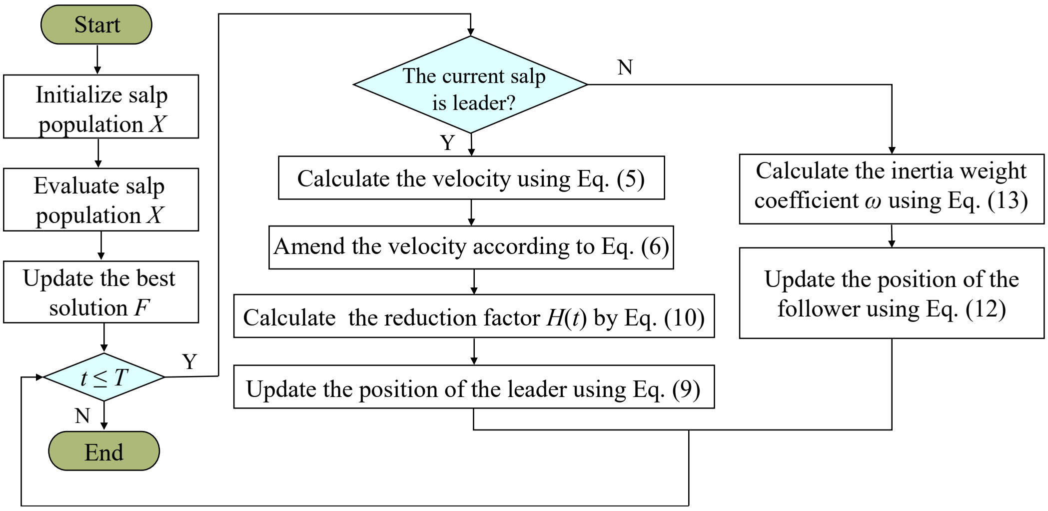

Figure 3. The flowchart of VC-SSA.

4. Simulation and comparisons

To strictly investigate the effectiveness of the developed VC-SSA, a series of experiments are performed to tackle various benchmark functions. Any involved algorithm is coded on MATLAB R2016b software under Windows 10 operating system with Intel(R) Xeon(R) W-2102 CPU @ 2.90 GHz with 8 GB RAM.

4.1. Benchmark test functions

For comparison purposes, a recognized benchmark set including 11 unimodal and 12 multi-modal functions is selected. Details of the covered classical baseline functions are reported in Table 1.

In Table 1, these 23 classical benchmarks have different characteristics, and by testing on them, the global search and partial exploration abilities of the optimizer can be comprehensively investigated. The unimodal problems possess only one global optimal solution and are appropriate for testing the approach's capability to exploit in a narrow region. On the contrary, the multimodal cases have multiple local minima and are appropriate for examining the global exploration ability of the optimizer 46.

4.2. Compared against SSA and SSA variants

To evaluate the effectiveness of the developed VC-SSA approach, we conducted tests on 23 classical benchmark functions with 100 dimensions, the details are presented in Table 1. The VC-SSA approach was compared with 11 SSA variants, including the basic SSA 24, ISSA 28, GSSA 29, IWOSSA 30, ASSA 47, RDSSA 48, CSSA 49, ESSA 50, LSSA 51, OBSSA 52, ASSO 53. The general parameters of all the mentioned approaches identical, i.e., the population size and the maximum evaluations were 30 and 500, respectively. The specific parameters of the involved approaches were extracted from the respective source work. The details are reported in Table 2. In the proposed VC-SSA algorithm, $\delta = 0.003$, $\mu = 0.04$, $\omega_{max} = 0.9$, $\omega_{min} = 0.2$, $\lambda = 3$, where the value of δ is set using test and trail. Each algorithm was executed 30 trials and the average minimum (Mean) and standard deviation (Std) were employed as metrics to evaluate the effectiveness of the algorithms. To strengthen the belief in the VC-SSA algorithm, Friedman's rank test was employed. In addition, to observed the statistical differences in outcomes of VC-SSA and its competitors, the Wilcoxon signed rank test with significance level 5% has been considered. The comparison results of VC-SSA with its competitors and the average ranking of each method are reported in Table 3. The *p*-value from Wilcoxon signed rank test are provided in Table 4. The symbols "+/-/=" in Table 4 indicate that the VC-SSA performs significantly superior, inferior, or similar to its peers.

From Table 3, VC-SSA finds theoretical optimal solutions on all benchmarks except f_7 and f_{13}. Compared with SSA, IWOSSA, CSSA, ISSA, ASSA, and LSSA, VC-SSA provides superior values on all test cases. Concerning GSSA, VC-SSA finds superior and similar scores on 21 and two cases, respectively. VC-SSA outperforms ESSA and OBSSA on 17 test functions, and for f_5, f_{12}~f_{14}, f_{17}, and f_{21}, they achieve similar results. With respect to ASSO and RDSSA, VC-SSA shows better and similar performance on 18 and 5 test cases, respectively. Moreover, the average ranking presented at Table 3 indicates that VC-SSA gets the highest ranking while the basic SSA has the lowest ranking, which demonstrates that all SSA variants intensify the performance of the standard SSA, with VC-SSA enhancing it the most. As shown in Table 4, there is almost no *p*-value lager than 0.05, which proves that the differences between VC-SSA and the comparison methods are statistically significant.

In addition, Figure 4 illustrates the average ranking of the involved approaches on the selected 23 tested functions in the form of a radar plot. The radar chart shows that the VC-SSA algorithm is ahead of the comparison algorithms on all benchmark problems.

Table 1. The characteristics of the classical benchmark functions.

Function type	Function formulation	Search range	$f_{\min}$
Unimodal	$f_1(x)=\sum_{i=1}^{D}x_i^2$	[-100, 100]	0
	$f_2(x)=\sum_{i=1}^{D}ix_i^2$	[-10, 10]	0
	$f_3(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}x_j\right)^2$	[-100, 100]	0
	$f_4(x)=\max_i\left\{\lvert x_i\rvert,1\le x_i\le D\right\}$	[-100, 100]	0
	$f_5(x)=\sum_{i=1}^{D}\left(\lfloor x_i+0.5\rfloor\right)^2$	[-100, 100]	0
	$f_6(x)=\sum_{i=1}^{D}ix_i^4$	[-1.28, 1.28]	0
	$f_7(x)=\sum_{i=1}^{D}ix_i^4+random[0,1)$	[-1.28, 1.28]	0
	$f_8(x)=\sum_{i=1}^{D}\lvert x_i\rvert^{(i+1)}$	[-1, 1]	0
	$f_9(x)=\sum_{i=1}^{D}(10^6)^{(i-1)/(D-1)}x_i^2$	[-100, 100]	0
	$f_{10}(x)=x_1^2+10^6\cdot\sum_{i=2}^{D}x_i^6$	[-100, 100]	0
	$f_{11}(x)=10\cdot x_1^2+\sum_{i=2}^{D}x_i^6$	[-1, 1]	0
Multimodal	$f_{12}(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	[-5.12, 5.12]	0
	$f_{13}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)+20+e$	[-32, 32]	0
	$f_{14}(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600, 600]	0
	$f_{15}(x)=\sum_{i=1}^{D}\lvert x_i\cdot\sin(x_i)+0.1x_i\rvert$	[-10, 10]	0
	$f_{16}(x)=\sin^2(\pi x_1)+\sum_{i=1}^{D-1}[x_i^2\cdot(1+10\sin^2(\pi x_1))+(x_i-1)^2-\sin^2(2\pi x_i)]$	[-10, 10]	0
	$f_{17}(x)=0.1D-\left(0.1\sum_{i=1}^{D}\cos(5\pi x_i)-\sum_{i=1}^{D}x_i^2\right)$	[-1, 1]	0
	$f_{18}(x)=\sum_{i=1}^{D}x_i^2+\left(\sum_{i=1}^{D}0.5x_i\right)^2+\left(\sum_{i=1}^{D}0.5x_i\right)^4$	[-5, 10]	0

Continued on next page

$f_{19}(x)=\sum_{i=1}^{D}(0.2x_i^2+0.1x_i^2\cdot\sin(2x_i))$	[-10, 10]	0
$f_{20}(x)=\left[\frac{1}{D-1}\sum_{i=1}^{D}\left(\sqrt{x_i}(\sin(50.0x_i^{0.2})+1)\right)\right]^2$	[-100, 100]	0
$f_{21}(x)=\sum_{i=1}^{D-1}[x_i^2+2x_{i+1}^2-0.3\cos(3\pi x_i)-0.4\cos(4\pi x_{i+1})+0.7]$	[-15, 15]	0
$f_{21}(x)=\sum_{i=1}^{D-1}(x_i^2+2x_{i+1}^2)^{0.25}\cdot((\sin 50(x_i^2+x_{i+1}^2)^{0.1})^2+1)$	[-10, 10]	0
$f_{23}(x)=\sum_{i=1}^{D-1}x_i^6\cdot\left(2+\sin\frac{1}{x_i}\right)$	[-1, 1]	0

Table 2. Parameter settings of SSA variants used for comparative analysis.

Algorithm	Parameters
SSA	c_1, c_2, c_3
ISSA	c_1, c_2, c_3
GSSA	c_1, c_2, c_3, $\mu = 0$, $\sigma = 1$
IWOSSA	c_1, c_2, c_3, r_1 = random [0, 1], r_2 = random [0, 1], a = exponentially decreased from 2 to 0, l = random [-1.73, 1]
ASSA	c_1, c_2, c_3, adaptive population
RDSSA	c_1, c_2, c_3
CSSA	c_1, c_2, c_3, tent chaotic map
ESSA	c_1, c_2, c_3, r_1 = integer random [0, 1]; r_2 = random [0, 1]
LSSA	c_1, c_2, c_3, $\lambda_1 = 4.0$, $\lambda_2 = 0.44$, $\lambda_3 = 1.1$
OBSSA	c_1, c_2, c_3
ASSO	c_1, c_2, c_3, $\omega_{max} = 2$, $\omega_{min} = 0$, $a = 1/\pi^2$

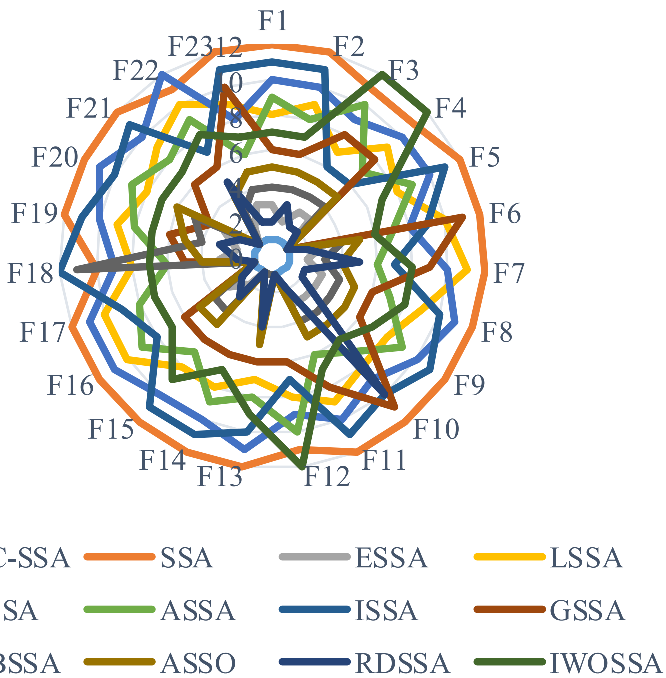

Figure 4. Radar chart for consolidated ranks of 23 benchmarks with the VC-SSA and the involved SSA variants.

Table 3. Comparisons of twelve optimizers on 23 benchmarks with 100 dimensions.

Function	Results	SSA	ESSA	LSSA	CSSA	ASSA	ISSA	GSSA	OBSSA	ASSO	RDSSA	IWOSSA	VC-SSA
f_1	Mean	1.42E+03.	2.03E-43	3.50E-03	1.5878	0.0398	103.4197	2.69E-14	4.63E-32	2.02E-26	7.70E-55	1.45E-06	0
	Std	4.75E+02	1.11E-42	6.10E-03	1.8430	0.0290	23.3033	7.14E-14	5.99E-32	2.21E-27	4.22E-54	7.61E-07	0
	f-rank	12	3	8	10	9	11	6	4	5	2	7	1
f_2	Mean	908.4570	1.98E-40	0.1554	0.7546	0.0178	50.8115	1.83E-16	3.01E-32	1.04E-26	7.48E-40	5.53E-07	0
	Std	226.5479	1.08E-39	0.1022	0.6672	0.0106	13.1040	4.57E-16	2.86E-32	1.37E-27	4.10E-39	4.85E-07	0
	f-rank	12	2	9	10	8	11	6	4	5	3	7	1
f_3	Mean	5.44E-04	6.73E-36	7.33E+03	1.80E+04	1.90E+04	2.13E+03	1.44E+04	5.09E-30	1.84E-25	6.89E-40	1.11E+05	0
	Std	3.08E+04	3.69E-35	4.93E+03	8.79E+03	1.20E+04	751.5218	7.91E+03	9.47E-30	8.37E-26	3.77E-39	2.33E+04	0
	f-rank	11	3	7	9	10	6	8	4	5	2	12	1
f_4	Mean	26.8536	5.76E-23	23.8694	24.3837	10.0738	5.6898	23.0690	4.95E-17	3.49E-14	5.87E-30	42.7411	0
	Std	2.9965	2.79E-22	5.5154	2.4304	2.6223	0.6011	3.0543	2.67E-17	3.31E-15	3.01E-29	7.7522	0
	f-rank	11	3	9	10	7	6	8	4	5	2	12	1
f_5	Mean	2.70E+03	0	9.2000	128.5000	13.9333	464.3333	0	0	0	0	1.3333	0
	Std	578.4251	0	4.9578	34.1081	6.0739	99.3289	0	0	0	0	2.6566	0
	f-rank	12	1	8	10	9	11	1	1	1	1	7	1
f_6	Mean	0.2735	1.83E-88	0.0052	3.35E-06	1.09E-07	7.43E-04	0.0307	4.46E-70	1.60E-59	7.30E-129	2.44E-12	0
	Std	0.1192	7.52E-88	0.0040	4.63E-06	1.90E-07	2.64E-04	0.0330	1.07E-69	4.61E-60	4.00E-128	5.77E-12	0
	f-rank	12	3	10	8	7	9	11	4	5	2	6	1
f_7	Mean	2.6063	8.30E-05	0.5879	0.2971	0.0917	0.1049	0.1946	8.38E-05	1.08E-04	6.72E-04	0.1073	5.58E-05
	Std	0.5632	9.57E-05	0.2509	0.0720	0.0209	0.0367	0.3215	7.01E-05	1.23E-04	7.01E-04	0.0679	5.72E-05
	f-rank	12	2	11	10	6	7	9	3	4	5	8	1
f_8	Mean	2.65E-06	7.81E-51	1.23E-10	2.31E-06	2.54E-26	1.22E-07	6.49E-35	4.98E-39	5.32E-36	1.97E-87	2.37E-15	0
	Std	1.90E-06	4.26E-50	3.71E-10	2.63E-06	1.20E-25	1.70E-07	3.56E-34	1.08E-38	9.77E-36	1.08E-86	1.29E-14	0
	f-rank	12	3	9	11	7	10	6	4	5	2	8	1
f_9	Mean	9.40E+07	7.71E-41	1.0180	8.58E+05	131.2321	3.49E+06	8.01E-15	4.29E-27	1.31E-21	5.39E-57	0.0120	0
	Std	4.48E+07	2.82E-40	1.1596	3.64E+05	94.4958	1.63E+06	2.26E-14	4.56E-27	4.13E-22	2.77E-56	0.0093	0
	f-rank	12	3	8	10	9	11	6	4	5	2	7	1
f_{10}	Mean	5.92E+13	1.35E-46	8.32E+08	2.17E+09	8.57E+05	9.56E+09	3.25E+13	3.85E-35	4.99E-32	6.69E-81	3.38E+05	0
	Std	3.75E+13	7.39E-46	2.26E+09	2.45E+09	9.80E+05	7.25E+09	4.42E+13	7.78E-35	7.48E-32	3.67E-80	8.64E+05	0
	f-rank	12	3	8	9	7	10	11	4	5	10	6	1
f_{11}	Mean	1.17170	3.01E-44	2.08E-04	0.2968	1.33E-10	0.3814	2.13E-05	3.73E-33	5.34E-30	2.60E-80	3.21E-09	0
	Std	1.9917	1.33E-43	5.19E-04	0.8082	5.45E-10	0.4976	8.68E-05	7.24E-33	7.95E-30	1.42E-79	3.83E-09	0
	f-rank	12	3	9	10	6	11	8	4	5	2	7	1
f_{12}	Mean	237.5810	0	161.4550	178.0876	207.0592	63.6131	7.81E-10	0	0	0	333.8639	0
	Std	41.3978	0	85.5654	28.2649	41.6477	13.6057	3.48E-09	0	0	0	130.4643	0
	f-rank	11	1	8	9	10	7	6	1	1	1	12	1
f_{13}	Mean	10.1023	8.88E-16	0.0264	4.5661	0.0268	3.8197	9.05E-10	8.88E-16	1.84E-14	4.44E-15	0.6653	8.88E-16
	Std	1.0767	0	0.0196	0.6729	0.0081	0.2982	1.44E-09	0	2.46E-15	0	3.6432	0
	f-rank	12	1	7	11	8	10	6	1	5	4	9	1
f_{14}	Mean	13.8946	0	0.0414	0.4276	0.0558	1.8593	1.95E-14	0	0	0	0.0057	0
	Std	3.9399	0	0.0715	0.2683	0.0412	0.1729	7.06E-14	0	0	0	0.0126	0
	f-rank	12	1	8	10	9	11	6	1	1	1	7	1
f_{15}	Mean	28.6555	6.03E-24	13.1498	18.7772	8.3729	23.8194	1.44E-14	1.32E-17	1.12E-14	4.01E-22	17.8688	0
	Std	5.6915	3.24E-23	11.1280	4.2692	4.5133	7.4723	2.45E-14	5.66E-18	8.08E-16	2.19E-21	14.6364	0
	f-rank	12	2	8	10	7	11	6	4	5	3	9	1
f_{16}	Mean	348.7569	1.03E-36	141.1216	325.2005	43.9365	32.5928	4.91E-08	2.00E-32	8.28E-27	9.76E-49	14.7505	0
	Std	51.0819	5.62E-36	125.2993	90.5840	27.4721	10.8309	1.79E-07	2.49E-32	8.41E-28	5.34E-48	30.6389	0
	f-rank	12	3	10	11	9	8	6	4	5	2	7	1
f_{17}	Mean	4.5875	0	1.2547	3.0950	1.47E-04	0.1476	0	0	0	0	1.82E-09	0
	Std	0.6527	0	0.7694	0.6764	3.34E-04	0.0337	0	0	0	0	1.02E-09	0
	f-rank	12	1	10	11	8	9	1	1	1	1	7	1
f_{18}	Mean	69.0543	2.08E-43	0.0283	0.1697	0.0091	1.75E+03	9.74E-09	211.9922	1.74E-28	3.41E-46	0.0178	0
	Std	16.5279	1.14E-42	0.0326	0.1576	0.0067	4.37E+03	3.09E-08	37.5769	3.73E-29	1.36E-45	0.0266	0
	f-rank	10	3	8	9	6	12	5	11	4	2	7	1
f_{19}	Mean	17.4862	1.04E-48	0.0097	7.6192	1.65E-04	10.7372	1.04E-16	9.66E-35	3.90E-29	2.96E-46	2.30E-09	0
	Std	4.2696	5.70E-48	0.0374	3.7965	1.31E-04	4.6346	3.98E-16	7.83E-35	5.45E-30	1.62E-45	1.39E-09	0
	f-rank	12	2	9	10	8	11	6	4	5	3	7	1
f_{20}	Mean	5.0798	1.82E-12	0.2658	4.0590	1.7211	2.2862	4.21E-10	4.79E-09	1.30E-07	2.37E-14	0.0855	0
	Std	0.1582	9.52E-12	0.1518	0.3076	0.4478	0.0946	4.02E-10	1.53E-09	4.27E-09	1.28E-13	0.2302	0
	f-rank	12	3	8	11	9	10	4	5	6	2	7	1
f_{21}	Mean	194.4929	0	10.4383	65.8560	0.5684	71.4322	4.14E-16	0	0	0	1.29E-06	0
	Std	30.9901	0	4.5683	8.4227	0.6392	8.4400	1.47E-15	0	0	0	1.19E-06	0
	f-rank	12	1	9	10	8	11	6	1	1	1	7	1
f_{22}	Mean	4.2550	4.99E-12	4.2451	5.1620	4.1830	1.9226	1.1235	4.35E-09	1.20E-07	0.0168	4.0824	0
	Std	0.1876	1.64E-11	0.2626	0.4098	0.2032	0.1101	0.5455	1.13E-09	4.11E-09	0.0231	1.1409	0
	f-rank	11	2	10	12	9	7	6	3	4	5	8	1
f_{23}	Mean	2.66E-04	5.04E-126	8.82E-06	1.20E-06	7.52E-11	8.61E-05	7.34E-05	1.02E-108	1.85E-92	6.39E-127	3.82E-09	0
	Std	1.90E-04	2.76E-125	6.42E-06	1.03E-06	1.03E-10	4.94E-05	5.20E-05	4.27E-108	8.95E-93	3.50E-126	1.36E-08	0
	f-rank	12	3	9	8	6	11	10	4	5	2	7	1
	Average f-rank	11.73913	2.26087	8.695652	9.956522	7.913043	9.608696	6.4347	3.478261	4.043478	2.608696	7.869565	1
	Overall f-rank	12	2	9	11	8	10	6	4	5	3	7	1

Table 4. Statistical conclusions of the multiple-problem Wilcoxon's test on benchmark cases with 100 dimensions.

Function	SSA	ESSA	LSSA	CSSA	ASSA	ISSA	GSSA	OBSSA	ASSO	RDSSA	IWOSSA
	p-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value
f_1	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_2	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_3	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_4	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_5	1.21E-12	N/A	1.15E-12	1.20E-12	1.17E-12	1.21E-12	N/A	N/A	N/A	N/A	6.60E-04
f_6	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_7	3.02E-11	0.6100	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	0.06	0.04	2.61E-10	3.02E-11
f_8	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	4.57E-12	1.21E-12
f_9	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{10}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{11}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{12}	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	0.0028	N/A	N/A	N/A	1.21E-12
f_{13}	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	N/A	6.21E-13	1.69E-14	1.21E-12
f_{14}	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	8.86E-07	N/A	N/A	N/A	1.21E-12
f_{15}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{16}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{17}	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	N/A	N/A	N/A	N/A	1.21E-12
f_{18}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{19}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{20}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{21}	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	0.0028	N/A	N/A	N/A	1.21E-12
f_{22}	8.26E-13	1.21E-12	1.21E-12	9.47E-13	1.21E-12	1.17E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{23}	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
+/=/-	23/0/0	16/7/0	23/0/0	23/0/0	23/0/0	23/0/0	21/2/0	17/6/0	18/5/0	18/5/0	23/0/0

4.3. Scalability test

In this subsection, the effect of dimensionality variation on the performance of the algorithm is discussed through scalability test. For further experiments, VC-SSA is employed to tackle the 23 benchmark problems with 10000 dimensions in Table 1. The parameter settings are identical to those in Subsection 4.2. The VC-SSA approach is performed 30 times on per test case, and the optimal (Best), worst (Worst), average minimum (Mean), and standard deviation (Std) of the objective function values are reported in Table 5. Moreover, the success rate indicator was introduced to estimate the performance of the VC-SSA algorithm in dealing with high-dimensional problems. For this purpose, the following flag is designed to distinguish whether a solution is successful or not.

$$\begin{cases} \frac{|S_A - S_T|}{S} \leq 10^{-5}, S_T \neq 0 \\ |S_A - S_T| \leq 10^{-5}, S_T = 0 \end{cases} \tag{14}$$

where S_A indicates the result achieved by the optimizer on the benchmark problem, and S_T represents the theoretical optimal solution of the corresponding case. If the solution provided by the proposed approach on the test case satisfies the above equation, it means that this solution is successful. Note that the success rate listed in Table 5 is the proportion of the successful times out of 30 experiments.

From Table 5, the VC-SSA algorithm obtains global optimal solutions on 21 benchmarks; for f_7 and f_{13}, although VC-SSA fails to converge to the theoretical optimum, the solutions obtained are still satisfactory. From another perspective, comparing the results in Table 5 with those reported in Table 2, we can see that the suggested approach exhibits similar performance on large-scale problems as on 100-dimensional functions, which represents that the impact caused by dimensionality variation on the VC-SSA algorithm is negligible. Also, in terms of SR%, VC-SSA can achieve 100% success rate on 22 tested functions; for f_7, the VC-SSA algorithm has 80% success rate. According to the above analysis, the proposed optimizer has excellent scalability and is available as as an auxiliary tool for tackling large-scale optimization cases.

Table 5. Results obtained by VC-SSA on 10000-dimensional functions.

Function	VC-SSA				
	Best	Worst	Mean	Std	SR%
f_1	0	0	0	0	100
f_2	0	0	0	0	100
f_3	0	0	0	0	100
f_4	0	0	0	0	100
f_5	0	0	0	0	100
f_6	0	0	0	0	100
f_7	6.78E-08	2.08E-04	5.71E-05	5.02E-05	80
f_8	0	0	0	0	100
f_9	0	0	0	0	100
f_{10}	0	0	0	0	100
f_{11}	0	0	0	0	100
f_{12}	0	0	0	0	100
f_{13}	8.88E-16	8.88E-16	8.88E-16	0	100
f_{14}	0	0	0	0	100
f_{15}	0	0	0	0	100
f_{16}	0	0	0	0	100
f_{17}	0	0	0	0	100
f_{18}	0	0	0	0	100
f_{19}	0	0	0	0	100
f_{20}	0	0	0	0	100
f_{21}	0	0	0	0	100
f_{22}	0	0	0	0	100
f_{23}	0	0	0	0	100

4.4. Comparison against other meta-heuristic algorithms

To further validate the effectiveness of the anticipated VC-SSA algorithm, 11 cutting-edge metaheuristic techniques, including TSA 54, SOGWO 55, MPA 3, HGS 20, EO 46, ArOA 57, AOA 58, IGWO 59, WEMFO 60, DMMFO 43, and OGWO 61, were utilized as the comparison approaches.

The general parameters of the approaches involved are consistent with those mentioned in Subsection 4.2, and each method was performed 30 times to mitigate the effect of random values. The specific parameter settings of the competitors were adopted from the respective original work, as detailed in Table 6. The detailed results with 100 dimensions, including the average minimum (Mean), standard deviation (Std), and the average ranking are listed in Table 7. The Wilcoxon signed-rank test with significance level 5% are presented in Table 8 to check the significant variances between the VC-SSA and its competitors on the benchmark functions.

Table 6. Parameter settings of cutting-edge algorithms used for comparative analysis.

Algorithm	Parameters
TSA	$P_{\max} = 4$, $P_{\min} = 1$
SOGWO	$d = 2$
MPA	Fish aggregating devices = 0.2, $P = 0.5$, R = random [0, 1]
HGS	$l = 0.03$, $LH = 100$
EO	Generation probability = 0.5, $a_1 = 2$, $a_2 = 1$
ArOA	$\alpha = 5$, $\mu = 0.5$
AOA	$C_1 = 2$, $C_2 = 6$, $C_3 = 2$, $C_4 = 0.5$
IGWO	a was linearly decreased from 2 to 0
WEMFO	ω_1 and ω_2 were nonlinearly reduced from 1 to 0 and from 2 to 0, respectively.
DMMFO	Constant coefficient $b = 1$
OGWO	a was nonlinearly decreased from 2 to 0

From Table 7, the performance of VC-SSA is better than TSA and DMMFO on all test functions. Compared to IGWO and OGWO, VC-SSA provides superior and similar outcomes on 22 and one problems, respectively. Regarding EO, AOA and WEMFO, the suggested method presents better and similar performance on 19 and four benchmarks, respectively. Compared with MPA, VC-SSA gets the better results on 18 cases; for f_5, f_{12}, f_{14}, f_{17} and f_{21}, they successfully find the global optimal solution. Concerning HGS, VC-SSA provides better results on 17 functions; for the other seven problems, two optimizers converge to the theoretical optimum. VC-SSA outperforms ArOA on 16 cases; for the other seven benchmarks, two algorithms find the theoretical optimum. Moreover, according to the average ranking results of the 12 methods, VC-SSA obtained the first ranking, TSA ranked last, and the rest of the approaches ranked second to eleventh. The above analysis reveals that the overall performance of the proposed VC-SSA algorithm outperformed its peers.

From Table 8, the reported p-values are less than 0.05 except for six pairwise comparisons (ArOA versus VC-SSA on f_2, ArOA versus VC-SSA on f_7, ArOA versus VC-SSA on f_{21}, HGS versus VC-SSA on f_6, EO versus VC-SSA on f_{14}, and AOA versus VC-SSA on f_{14}). This implies that VC-SSA outperforms its peers and that the difference between VC-SSA and the comparison algorithms is statistically significant.

Figure 5 draws a radar plot to highlight the rank variation of VC-SSA and other state-of-the-art metaheuristic techniques over 23 benchmark problems. From the radar diagram, the developed approach achieves better rankings on all cases compared to the competitors, which represents that the suggested VC-SSA outperforms all the competitors.

Table 7. Comparisons of twelve algorithms on 23 test functions with 100 dimensions.

Function	Results	TSA	SOGWO	MPA	HGS	EO	ArOA	AOA	IGWO	WEMFO	DMMFO	OGWO	VC-SSA
f_1	Mean	3.56E-10	2.96E-12	1.73E-19	1.01E-134	3.03E-29	0.0272	5.94E-80	3.21E-12	2.89E-22	3.20E+04	2.83E-15	0
	Std	4.67E-10	3.18E-12	1.51E-19	5.54E-134	4.04E-29	0.0106	2.13E-79	2.33E-12	8.58E-22	8.14E+03	3.22E-15	0
	f-rank	10	8	6	2	4	11	3	9	5	12	7	1
f_2	Mean	3.05E-10	5.27E-13	8.70E-20	4.32E-141	2.38E-29	6.43E-86	2.20E-78	8.45E-13	1.76E-24	1.41E+04	7.27E-16	0
	Std	4.94E-10	4.94E-13	1.12E-19	2.37E-140	5.50E-29	3.52E-85	1.13E-77	9.05E-13	3.89E-24	2.81E+03	8.51E-16	0
	f-rank	11	9	7	2	5	3	4	10	6	12	8	1
f_3	Mean	1.39E+04	1.28E+03	9.6562	1.82E-22	22.9242	0.7175	1.94E-57	5.69E+03	3.64E-11	2.34E+05	1.37E+03	0
	Std	6.76E+03	1.25E+03	14.2133	9.98E-22	62.6396	0.5256	1.06E-56	2.12E+03	1.30E-10	4.18E+04	1.95E+03	0
	f-rank	11	8	6	3	7	5	2	10	4	12	9	1
f_4	Mean	55.0716	0.7470	2.31E-07	2.86E-64	4.75E-04	0.0920	9.37E-39	3.7153	3.17E-10	88.3981	2.0555	0
	Std	10.2327	0.4040	1.07E-07	1.34E-63	5.60E-04	0.0122	4.82E-38	1.4176	8.52E-10	2.2765	2.3980	0
	f-rank	11	8	5	2	6	7	3	10	4	12	9	1
f_5	Mean	17.2000	0	0	0	0	0	0	0	0	3.38E+04	0	0
	Std	11.3939	0	0	0	0	0	0	0	0	5.69E+03	0	0
	f-rank	11	1	1	1	1	1	1	1	1	12	1	1
f_6	Mean	8.56E-19	3.05E-25	6.78E-41	1.03E-317	4.10E-50	0	3.45E-160	1.86E-22	1.40E-47	101.2232	9.21E-29	0
	Std	2.22E-18	4.24E-25	1.33E-40	0	1.89E-49	0	1.87E-159	4.97E-22	7.56E-47	16.5895	1.77E-28	0
	f-rank	11	9	7	3	5	1	4	10	6	12	8	1
f_7	Mean	0.0496	0.0072	0.0019	0.0011	0.0020	6.40E-05	5.44E-04	0.0134	0.0015	98.5698	0.0026	5.52E-05
	Std	0.0162	0.0027	0.0011	0.0021	8.40E-04	7.06E-05	4.61E-04	0.0046	9.74E-04	25.9483	0.0028	4.56E-05
	f-rank	11	9	6	4	7	2	3	10	5	12	8	1
f_8	Mean	3.56E-33	3.71E-61	1.50E-59	6.70E-76	2.87E-125	0	7.44E-189	7.73E-61	1.58E-82	0.0018	2.40E-51	0
	Std	1.95E-32	2.02E-60	7.50E-59	3.67E-75	1.44E-124	0	0	4.12E-60	8.00E-82	0.0016	1.25E-50	0
	f-rank	11	7	9	6	4	1	3	8	5	12	10	1
f_9	Mean	6.27E-07	5.04E-09	9.74E-16	3.51E-170	1.81E-25	117.8736	3.02E-73	3.35E-09	2.30E-18	2.01E+08	6.56E-12	0
	Std	1.04E-06	4.31E-09	1.22E-15	0	3.41E-25	140.8922	1.29E-72	2.63E-09	9.09E-18	6.83E+07	7.29E-12	0
	f-rank	10	9	6	2	4	11	3	8	5	12	7	1
f_{10}	Mean	0.0576	3.82E-16	3.04E-39	2.21E-58	5.19E-48	0.6707	6.91E-169	2.59E-10	3.22E-45	2.93E+17	1.64E-20	0
	Std	0.2656	7.61E-16	5.27E-39	9.73E-58	2.65E-47	0.3197	0	7.01E-10	1.76E-44	9.95E+16	5.47E-20	0
	f-rank	10	8	6	3	4	11	2	9	5	12	7	1
f_{11}	Mean	3.66E-20	4.13E-35	7.94E-53	2.74E-42	1.28E-65	0	1.27E-180	1.11E-28	2.28E-58	0.9418	5.01E-39	0
	Std	1.41E-19	7.37E-35	1.83E-52	1.41E-41	5.83E-65	0	0	2.31E-28	1.24E-57	0.4020	1.78E-38	0
	f-rank	11	9	6	7	4	1	3	10	5	12	8	1
f_{12}	Mean	991.6601	9.0738	0	0	0	0	0	143.7451	431.8780	830.9407	1.0877	0
	Std	91.9241	6.1775	0	0	0	0	0	47.4518	330.5842	67.3292	2.3463	0
	f-rank	12	8	1	1	1	1	1	9	10	11	7	1
f_{13}	Mean	1.20E-05	1.42E-07	4.90E-11	8.88E-16	3.56E-14	6.01E-04	19.9667	1.65E-07	0.0034	19.7348	5.99E-09	8.88E-16
	Std	2.08E-05	5.83E-08	2.94E-11	0	8.43E-15	0.0011	1.21E-04	6.24E-08	0.0162	0.1648	2.71E-09	0
	f-rank	8	6	4	1	3	9	12	7	10	11	5	1
f_{14}	Mean	0.0091	0.0059	0	0	9.06E-04	639.2493	0.0023	0.0015	0	320.3774	0.0022	0
	Std	0.0169	0.0122	0	0	0.0050	185.3638	0.0124	0.0049	0	60.0949	0.0070	0
	f-rank	10	9	1	1	5	12	8	6	1	11	7	1
f_{15}	Mean	153.2401	0.0039	3.83E-12	1.50E-69	3.99E-18	1.15E-61	1.46E-41	0.0046	57.0003	57.1239	3.79E-04	0
	Std	24.2002	0.0019	3.04E-12	8.24E-69	2.48E-18	6.32E-61	7.10E-41	0.0022	26.7735	9.5240	7.96E-04	0
	f-rank	12	8	6	2	5	3	4	9	10	11	7	1
f_{16}	Mean	139.6205	0.9104	3.29E-15	3.22E-150	3.16E-26	1.14E-64	5.04E-71	9.4036	4.89E-22	1.53E+03	6.39E-21	0
	Std	116.3671	1.7319	1.49E-14	1.76E-149	6.35E-26	6.22E-64	2.04E-70	4.8555	2.67E-21	340.2941	2.91E-20	0
	f-rank	11	9	8	2	5	4	3	10	6	12	7	1
f_{17}	Mean	3.6610	2.75E-14	0	0	0	0	0	2.22E-14	0	12.8129	4.74E-15	0
	Std	2.7571	5.93E-15	0	0	0	0	0	8.39E-15	0	1.7195	3.07E-15	0
	f-rank	11	10	1	1	1	1	1	9	1	12	8	1
f_{18}	Mean	1.05E-11	2.11E-13	4.35E-19	1.16E-94	6.62E-28	1.32E+03	1.51E-71	2.05E-12	1.09E-24	491.1379	1.36E-15	0
	Std	1.05E-11	1.68E-13	4.56E-19	6.37E-94	1.87E-27	104.7894	8.21E-71	1.54E-12	3.21E-24	83.6894	1.01E-15	0
	f-rank	10	8	6	2	4	12	3	9	5	11	7	1
f_{19}	Mean	2.13E-04	4.58E-15	3.73E-22	1.69E-154	1.16E-31	1.24E-82	8.25E-80	1.13E-14	2.14E-25	174.8451	3.02E-25	0
	Std	0.0012	3.65E-15	3.19E-22	9.27E-154	3.25E-31	5.67E-82	3.77E-79	9.38E-15	6.55E-25	28.0411	1.19E-24	0
	f-rank	11	9	8	2	5	3	4	10	6	12	7	1
f_{20}	Mean	2.8095	0.0107	5.85E-07	2.36E-34	1.44E-09	0.0018	4.20E-22	0.0109	8.04E-09	8.5093	9.13E-05	0
	Std	1.3209	0.0027	8.29E-07	1.29E-33	5.96E-10	0.0059	1.74E-21	0.0041	8.03E-09	0.3460	5.09E-05	0
	f-rank	11	9	6	2	4	8	3	10	5	12	7	1
f_{21}	Mean	4.3723	1.29E-12	0	0	0	7.40E-12	0	2.06E-12	0	2.19E+03	1.17E-15	0
	Std	9.9305	1.71E-12	0	0	0	2.94E-11	0	1.99E-12	0	360.5419	1.55E-15	0
	f-rank	11	8	1	1	1	10	1	9	1	12	7	1
f_{22}	Mean	6.9978	1.5147	0.6369	6.06E-40	0.2623	0.0175	0.0158	2.9821	0.0039	7.4149	0.8292	0
	Std	0.5640	0.2502	0.0815	2.89E-39	0.0613	0.0256	0.0110	0.3593	0.0055	0.2655	0.2141	0
	f-rank	11	9	7	2	6	5	4	10	3	12	8	1
f_{23}	Mean	2.37E-14	1.51E-32	4.82E-55	7.4968	3.07E-64	0	4.83E-239	1.39E-25	2.55E-67	0.5292	8.76E-59	0
	Std	7.38E-14	5.13E-32	2.07E-54	8.8417	1.59E-63	0	0	5.29E-25	1.39E-66	0.1852	0.1852	0
	f-rank	10	8	7	12	5	1	3	9	4	11	6	1
	Average f-rank	10.69565	8.086957	5.26087	2.782609	4.193913	5.347826	3.391304	8.782609	4.913043	11.73913	7.173913	1
	Overall f-rank	11	9	6	2	4	7	3	10	5	12	8	1

Table 8. Statistical conclusions of the multiple-problem Wilcoxon's test on benchmark cases with 100 dimensions.

Function	TSA	SOGWO	MPA	HGS	EO	ArOA	AOA	IGWO	WEMFO	DMMFO	OGWO
	p-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value	*p*-value
f_1	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_2	1.21E-12	1.21E-12	1.21E-12	6.64E-05	4.57E-12	**0.1608**	1.66E-11	1.21E-12	1.21E-12	1.21E-12	5.77E-11
f_3	1.21E-12	1.21E-12	1.21E-12	1.31E-07	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_4	1.21E-12	1.21E-12	1.21E-12	8.87E-07	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_5	1.20E-12	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1.21E-12	N/A
f_6	1.21E-12	1.21E-12	1.21E-12	**0.1608**	1.21E-12	**N/A**	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_7	3.02E-11	3.02E-11	3.02E-11	5.46E-06	3.02E-11	**0.8073**	7.83E-10	3.02E-11	5.49E-11	3.02E-11	4.50E-11
f_8	1.21E-12	1.21E-12	1.21E-12	2.21E-06	1.21E-12	N/A	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_9	1.21E-12	1.21E-12	1.21E-12	1.46E-04	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{10}	1.21E-12	1.21E-12	1.21E-12	1.27E-05	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{11}	1.21E-12	1.21E-12	1.21E-12	8.87E-07	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{12}	1.21E-12	1.21E-12	N/A	N/A	N/A	N/A	N/A	1.21E-12	5.85E-09	1.21E-12	1.21E-12
f_{13}	1.21E-12	1.21E-12	1.21E-12	N/A	8.67E-13	3.13E-04	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{14}	1.21E-12	1.21E-12	N/A	N/A	**0.3337**	1.21E-12	**0.3337**	1.21E-12	N/A	1.21E-12	1.21E-12
f_{15}	1.21E-12	1.21E-12	1.21E-12	1.27E-05	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{16}	1.21E-12	1.21E-12	1.21E-12	1.46E-04	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{17}	1.21E-12	1.04E-12	N/A	N/A	N/A	N/A	N/A	1.11E-12	N/A	1.21E-12	5.25E-10
f_{18}	1.21E-12	1.21E-12	1.21E-12	6.61E-05	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{19}	1.21E-12	1.21E-12	1.21E-12	6.61E-05	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{20}	1.21E-12	1.21E-12	1.21E-12	6.61E-05	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
f_{21}	1.21E-12	1.21E-12	N/A	N/A	N/A	**0.1608**	N/A	1.21E-12	N/A	1.21E-12	1.20E-12
f_{22}	1.21E-12	1.21E-12	1.11E-12	1.27E-05	1.14E-12	1.21E-12	1.21E-12	1.21E-12	1.19E-12	7.72E-13	1.21E-12
f_{23}	1.28E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	N/A	1.66E-11	1.21E-12	1.21E-12	1.21E-12	1.21E-12
+/=/-	23/0/0	22/1/0	18/5/0	16/7/0	18/5/0	14/9/0	18/5/0	22/1/0	19/4/0	23/0/0	22/0/0

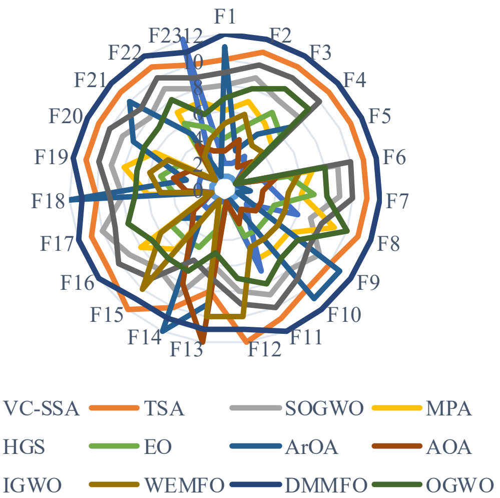

Figure 5. Radar chart for consolidated ranks of 23 benchmarks with the VC-SSA and the cutting-edge meta-heuristic methods.

4.5. Component analysis

The developed VC-SSA algorithm enhances the overall performance of the standard SSA by introducing three novel components, namely, velocity clamping strategy, reduction factor tactic, and adaptive inertia weight mechanism. In this subsection, we perform several suites of comparison experiments to check the validity of the modifications. For this purpose, 23 classical numerical functions from Table 1 were used, and the dimensionality was set to 100. The parameter settings were identical to those in Subsection 4.2. The naming rules for the SSA-based approaches equipped with a single improvement strategy are: the enhanced SSA using only the velocity clamping strategy is called VSSA; the improved SSA adopting only the reduction factor is noted as RSSA; and the boosted SSA applying only the adaptive inertia weight mechanism is known as ISSA. Each method is executed 30 times independently for each benchmark problem, and the corresponding average minimum (Mean) and standard deviation (Std) as well as average rankings are selected as performance indicators and recorded in Table 9.

From Table 9, ISSA, RSSA and VSSA outperform the standard SSA on all benchmark functions, which proves that the three developed components are effective. Moreover, according to the pairwise comparison between the single-strategy algorithm and the VC-SSA, VC-SSA obtains better outcomes than VSSA on all benchmark functions. With respect to ISSA, VC-SSA shows better performance on 17 benchmark functions, and for the other 6 problems, both algorithms get theoretically optimal solutions. VC-SSA and RSSA obtain similar results on most benchmark functions, while for f_7, VC-SSA gets better results than RSSA, which indicates that the reduction factor can improve the performance of the basic SSA, but the exploitation capability is still insufficient. It is necessary to use velocity clamping and adaptive inertia weighting strategies to further improve the ability to refine the already explored region. Finally, the average ranking derived from the Friedman's rank test indicates that the VC-SSA algorithm ranks first, followed by RSSA, ISSA, and VSSA, which further validates that the proposed strategies are efficient and can maximize the performance improvement when the three introduced components collaborate to assist SSA.

Table 9. Comparisons of SSA, VSSA, RSSA, ISSA and VC-SSA on 23 benchmark functions with 100 dimensions.

Function	Results	SSA	ISSA	RSSA	VSSA	VC-SSA
f_1	Mean	101.9661	6.51E-30	0	9.6847	0
	Std	23.6750	8.99E-31	0	2.7464	0
	f-rank	5	3	1	4	1
f_2	Mean	52.3696	3.26E-30	0	10.5935	0
	Std	11.1902	5.27E-31	0	2.5934	0
	f-rank	5	3	1	4	1
f_3	Mean	2.11E+03	1.71E-28	0	669.6495	0
	Std	970.9828	1.52E-28	0	204.1917	0
	f-rank	5	3	1	4	1
f_4	Mean	5.7138	6.99E-16	0	4.4588	0
	Std	0.6158	1.01E-16	0	0.9816	0
	f-rank	5	3	1	4	1
f_5	Mean	460.6333	0	0	273.8333	0
	Std	117.4976	0	0	99.7013	0
	f-rank	5	1	1	4	1

Continued on next page

f_6	Mean	7.61E-04	1.80E-66	0	2.48E-05	0
	Std	3.37E-04	6.83E-67	0	1.13E-05	0
	f-rank	5	3	1	4	1
f_7	Mean	0.1001	1.07E-04	8.33E-05	0.0452	6.71E-05
	Std	0.0411	1.06E-04	7.86E-05	0.0204	5.91E-05
	f-rank	5	3	2	4	1
f_8	Mean	9.64E-08	1.39E-36	0	8.61E-09	0
	Std	1.50E-07	4.83E-36	0	1.04E-08	0
	f-rank	5	3	1	4	1
f_9	Mean	3.36E+06	4.98E-25	0	1.23E+06	0
	Std	1.32E+06	1.53E-25	0	4.47E+05	0
	f-rank	5	3	1	4	1
f_{10}	Mean	8.42E+09	1.69E-33	0	1.46E+08	0
	Std	5.41E+09	5.64E-33	0	1.38E+08	0
	f-rank	5	3	1	4	1
f_{11}	Mean	0.5521	1.28E-31	0	0.0158	0
	Std	0.6718	3.77E-31	0	0.0431	0
	f-rank	5	3	1	4	1
f_{12}	Mean	61.5251	0	0	5.5763	0
	Std	12.1160	0	0	2.1549	0
	f-rank	5	1	1	4	1
f_{13}	Mean	3.7691	8.88E-16	8.88E-16	3.1683	8.88E-16
	Std	0.2472	0	0	0.3735	0
	f-rank	5	1	1	4	1
f_{14}	Mean	1.8677	0	0	1.0855	0
	Std	0.1991	0	0	0.0189	0
	f-rank	5	1	1	4	1
f_{15}	Mean	23.0309	2.04E-16	0	7.1201	0
	Std	7.5508	1.42E-17	0	5.9783	0
	f-rank	5	3	1	4	1
f_{16}	Mean	34.3256	2.59E-30	0	23.3460	0
	Std	10.6110	3.49E-31	0	5.2071	0
	f-rank	5	3	1	4	1
f_{17}	Mean	0.1305	0	0	0.0129	0
	Std	0.0360	0	0	0.0032	0
	f-rank	5	1	1	4	1
f_{18}	Mean	71.0173	6.89E-32	0	3.8100	0
	Std	14.8584	2.72E-32	0	1.0623	0
	f-rank	5	3	1	4	1
f_{19}	Mean	11.6823	1.26E-32	0	3.7714	0
	Std	5.3239	1.86E-33	0	3.1951	0
	f-rank	5	3	1	4	1
f_{20}	Mean	2.2905	1.74E-08	0	2.0389	0
	Std	0.1032	6.81E-10	0	0.1163	0
	f-rank	5	3	1	4	1
f_{21}	Mean	73.6870	0	0	56.6792	0
	Std	10.4842	0	0	6.5283	0
	f-rank	5	1	1	4	1

Continued on next page

f_{22}	Mean	1.9538	1.61E-08	0	1.7078	0
	Std	0.1127	6.33E-10	0	0.1515	0
	f-rank	5	3	1	4	1
f_{23}	Mean	1.20E-04	7.82E-103	0	1.34E-06	0
	Std	1.51E-04	3.86E-103	0	9.75E-07	0
	f-rank	5	3	1	4	1
	Average f-rank	5	2.4783	1.0435	4	1
	Overall f-rank	5	3	2	4	1

4.6. Convergence analysis

The proposed VC-SSA algorithm aims to expedite the convergence speed and convergence precision of the standard SSA, thus providing a well-performance optimization tool for global optimization tasks. In this subsection, the convergence performance of the VC-SSA approach will be investigated. In Figure 6, the convergence curves of VC-SSA and SSA-based algorithms on 15 representative functions in Table 1 are plotted. In Figure 7, the convergence graphs of VC-SSA and 11 frontier methods on 15 typical functions in Table 1 are drawn. In Figure 8, the convergence curves of VC-SSA and SSA using different optimization mechanisms are plotted. In all experiments, the dimensionality of the function is set to 100, the algorithm terminates after 500 iterations, and the specific parameters of all algorithms are set as before.

From Figure 6, the VC-SSA algorithm progressively converges during the early evaluation process and becomes faster over the course of iterations. The main reason is that a well-established swarm intelligence algorithm needs to allocate a large number of steps in the initial stage to explore the search domain and find the rough location of the global optimum, and in the later phase, it needs to shift from global search to local exploitation and precisely search the already explored region precisely to improve the solution accuracy. Therefore, as shown in Figure 6, the comparison algorithms fall into search stagnation after several iterations, while VC-SSA continues to converge, and the overall convergence rate and convergence precision are better than the involved peer approaches. From Figure 7, the suggested algorithm converges sluggishly during the early search phase. After the lapse of few iterations, the converge speed becomes faster. Compared with other frontier algorithms, VC-SSA shows a greater advantage in terms of convergence rate and solution accuracy, which is due to that the developed method can maintain a proper equilibrium between exploration and exploitation. According to Figure 8, the convergence performance of all three SSA algorithm using different optimization mechanisms outperforms the basic SSA, which proves that the proposed optimization mechanisms are effective. In addition, the VC-SSA algorithm outperforms the SSA using a single strategy in terms of convergence speed and convergence accuracy, which indicates that the three strategies can maximize the performance of SSA when they help SSA collaboratively. According to the convergence graph, RSSA facilitates to accelerate the convergence speed of the algorithm in the later iteration, VSSA contributes to improve the search ability of the algorithm in the early iteration stage, and ISSA solidifies the balance between exploitation and exploration of the algorithm. Overall, the convergence performance of VC-SSA outperforms the other SSA variants and the compared cutting-edge methods.

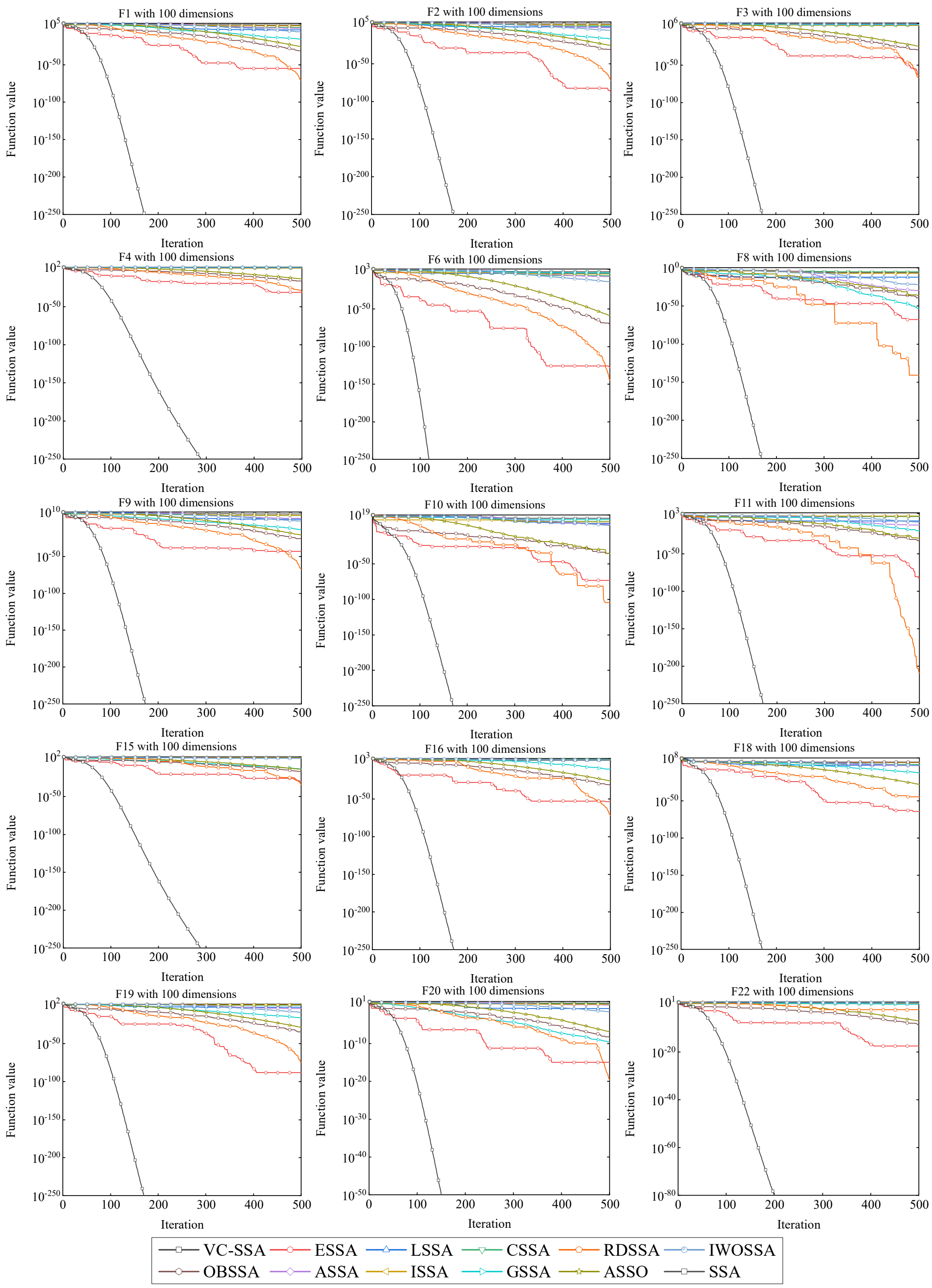

Figure 6. Convergence curves of SSA-based algorithms on some selected functions.

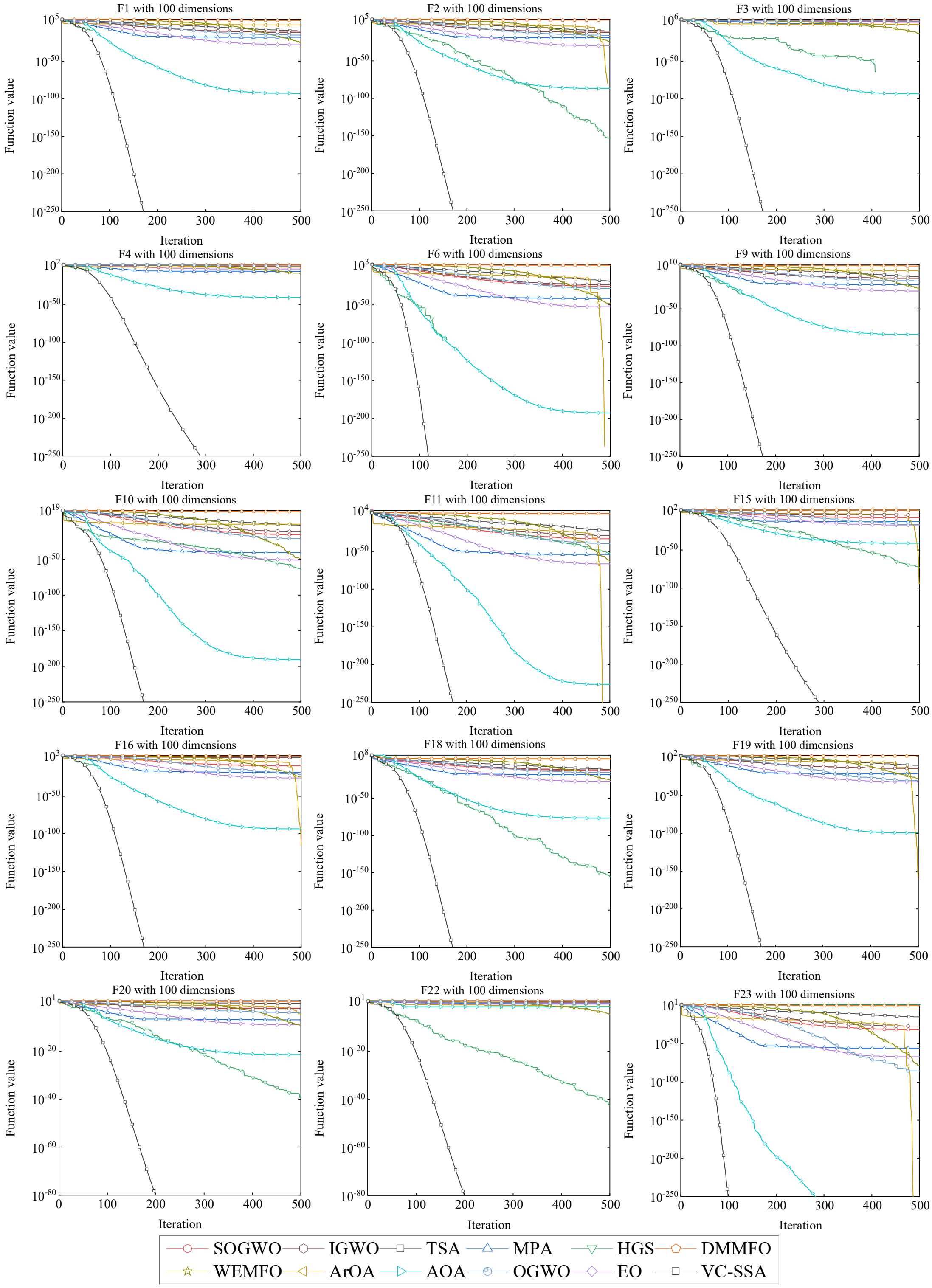

Figure 7. Convergence curves of VC-SSA and eleven cutting-edge algorithms on some selected functions.

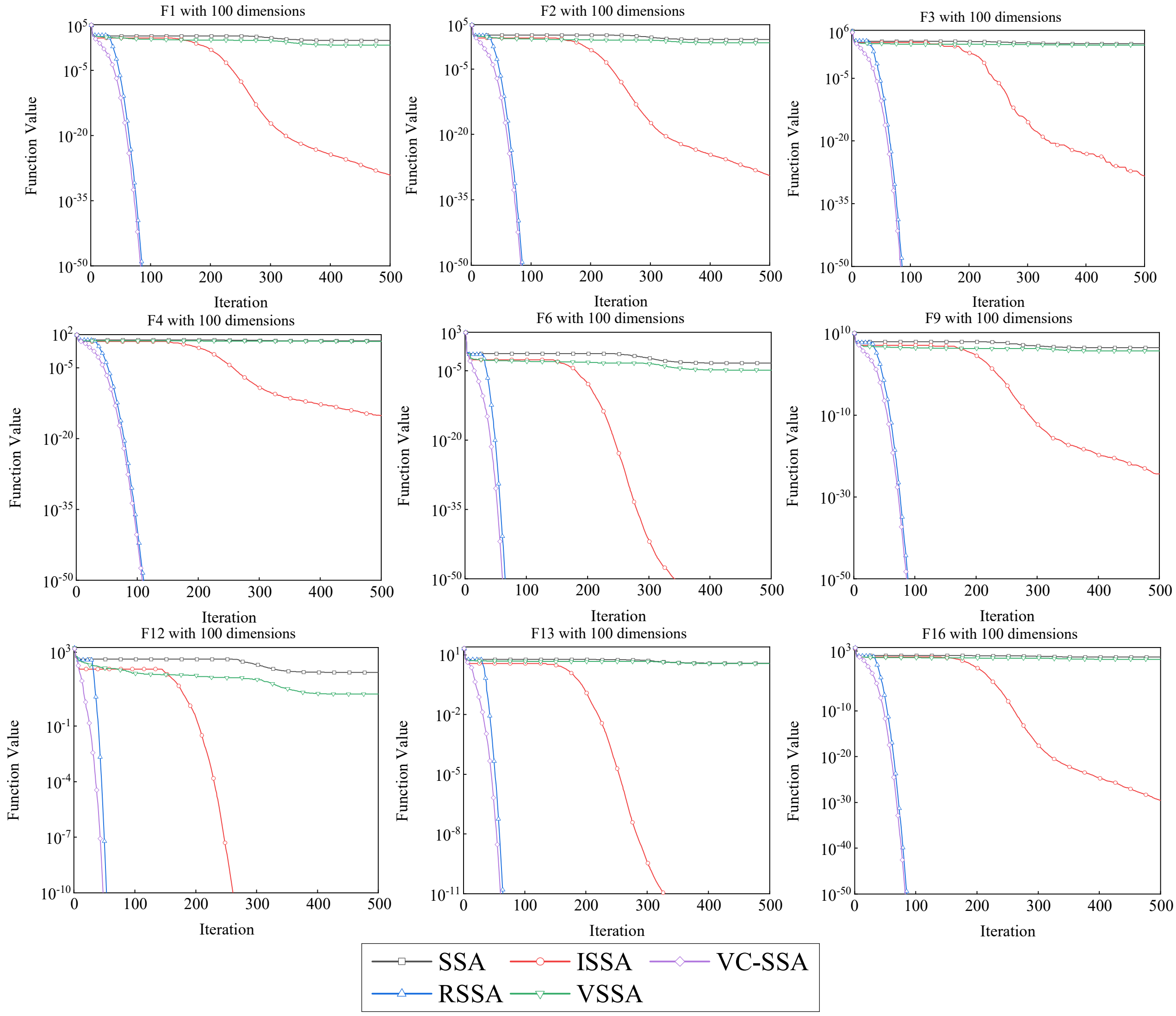

Figure 8. Convergence curves of VC-SSA and SSA using different optimization mechanisms on some selected functions.

4.7. Experiment on CEC2017 benchmark functions

In this section, the effectiveness of VC-SSA is verified using the CEC 2017 test function set. The performance of VC-SSA is compared with the state-of-the-art swarm intelligent algorithms, including SSA, MFO 8, MVO 72, SCA 45, GWO 13, HGS 20, WOA 11, and HHO 15. In this experiment, the population size is set to 30 and the maximum number of evaluations is set to $D \times 10{,}000$, where D denotes the dimension of the problem. The parameter settings of the compared algorithms are the same as those recommended in the respective original literature. The parameters of the VC-SSA algorithm are the same as those used in Subsection 4.2. All algorithms are run 30 times independently on each function, and the mean (Mean) and standard deviation (Std) for all fitness are recorded. In addition, the results of the Friedman's rank test are also provided. To investigate the effect of the problem dimensions on the algorithms, the algorithms are tested using test functions with different dimensions. The statistical results are shown in Table 10 when D is set to 30, and the statistical results are reported in Table 11 when D is set to 50.

As shown in Table 10, VC-SSA outperforms the SCA, WOA, and HHO algorithms for all test functions. Compared with SSA, MFO, and GWO algorithms, VC-SSA achieves better and inferior results on 28 and one cases, respectively. VC-SSA beats MVO on 27 benchmarks and is inferior to MVO on two cases. With respect to HGS, VC-SSA achieves superiority on 23 problems and is worse than HGS on six functions. According to the average ranking, the proposed VC-SSA received the top rank, followed by MVO, HGS, GWO, SSA, MFO, HHO, WOA, and SCA. From Table 11, VC-SSA outperforms SSA on 23 test functions and underperforms SSA on the remaining six problems. Compared to MFO and HHO, VC-SSA wins on 28 test cases and loses to the competitor on only one function. With respect to MVO, VC-SSA shows better performance on 26 benchmark problems and slightly worse performance on three functions. The proposed VC-SSA beats SCA and WOA on all test functions. Compared to GWO, VC-SSA achieves superiority on 21 test functions and shows inferiority on eight problems. VC-SSA beat HGS on 12 test problems and inferior to HGS on 17 benchmarks.

Table 10. Comparison results of nine algorithms on CEC 2017 benchmark functions with 30 dimensions.

Function	Results	SSA	MFO	MVO	SCA	GWO	HGS	WOA	HHO	VC-SSA
F1	Mean	5.20E+03	4.95E+09	5.88E+05	1.62E+10	9.09E+08	7.44E+05	5.17E+08	2.46E+07	4.03E+03
	Std	5.79E+03	3.64E+09	1.50E+05	2.89E+09	8.11E+08	3.72E+06	2.31E+08	7.68E+06	7.27E+03
	f-rank	2	8	3	9	7	4	6	5	1
F3	Mean	2.26E+04	1.29E+05	4.58E+02	6.29E+04	4.12E+04	1.57E+04	2.32E+05	2.72E+04	3.97E+02
	Std	6.41E+03	4.77E+04	9.92E+01	1.20E+04	1.15E+04	1.07E+04	6.20E+04	5.53E+03	1.38E+02
	f-rank	4	8	2	7	6	3	9	5	1
F4	Mean	5.08E+02	7.55E+02	4.96E+02	1.99E+03	5.49E+02	4.97E+02	7.09E+02	5.59E+02	4.91E+02
	Std	2.47E+01	2.56E+02	1.05E+01	4.15E+02	3.39E+02	3.28E+01	9.31E+01	4.28E+01	6.0525
	f-rank	4	8	2	9	5	3	7	6	1
F5	Mean	6.53E+02	6.81E+02	5.97E+02	8.08E+02	5.98E+02	6.26E+02	7.83E+02	7.43E+02	5.68E+02
	Std	5.05E+01	3.64E+01	2.28E+01	2.03E+01	2.49E+01	2.51E+01	5.72E+01	3.18E+01	2.04E+01
	f-rank	5	6	2	9	3	4	8	7	1
F6	Mean	6.47E+02	6.24E+02	6.19E+02	6.55E+02	6.06E+02	6.03E+02	6.75E+02	6.61E+02	6.00E+02
	Std	1.31E+01	8.4954	1.22E+01	7.1518	2.1146	2.6879	1.27E+01	8.4097	7.09E-01
	f-rank	6	5	4	7	3	2	9	8	1
F7	Mean	8.82E+02	9.87E+02	8.56E+02	1.18E+03	8.53E+02	8.85E+02	1.24E+03	1.24E+03	8.10E+02
	Std	5.19E+01	1.22E+02	4.77E+01	5.06E+01	3.09E+01	3.24E+01	7.93E+01	6.19E+01	2.27E+02
	f-rank	4	6	3	7	2	5	8	9	1
F8	Mean	9.38E+02	9.82E+02	9.15E+02	1.08E+03	8.82E+02	9.25E+02	1.02E+03	9.68E+02	8.69E+02
	Std	4.84E+01	4.44E+02	3.61E+01	2.42E+01	3.13E+01	3.17E+01	4.71E+01	2.04E+01	1.83E+01
	f-rank	5	7	3	9	2	4	8	6	1
F9	Mean	4.66E+03	6.06E+03	3.65E+03	6.82E+03	1.53E+03	3.86E+03	1.03E+04	7.29E+03	9.91E+02
	Std	1.52E+03	1.83E+03	3.04E+03	1.03E+03	5.74E+02	1.16E+03	3.47E+03	1.11E+03	3.41E+02
	f-rank	5	6	3	7	2	4	9	8	1

Continued on next page

F10	Mean	5.01E+03	5.39E+03	4.48E+03	8.56E+03	4.44E+03	4.29E+03	6.70E+03	5.82E+03	4.11E+03
	Std	6.84E+02	8.19E+02	7.49E+02	3.19E+02	9.64E+02	6.19E+02	8.45E+02	7.04E+02	5.57E+02
	f-rank	5	6	4	9	3	2	8	7	1
F11	Mean	1.30E+03	2.55E+03	1.33E+03	3.04E+03	1.71E+03	1.23E+03	4.28E+03	1.28E+03	1.27E+03
	Std	6.21E+01	2.25E+03	5.74E+01	6.49E+02	7.33E+02	4.41E+01	1.63E+03	4.33E+01	5.73E+01
	f-rank	4	7	5	8	6	1	9	3	2
F12	Mean	2.49E+07	7.28E+07	1.13E+07	1.80E+09	5.18E+07	2.83E+06	1.16E+08	1.83E+07	4.21E+06
	Std	2.29E+07	1.46E+08	9.82E+06	4.85E+08	4.81E+07	1.70E+06	8.83E+07	1.33E+07	4.31E+06
	f-rank	5	7	3	7	6	1	8	4	2
F13	Mean	1.43E+05	5.30E+06	1.37E+05	6.76E+08	8.88E+05	3.95E+04	5.65E+05	4.89E+05	2.11E+04
	Std	7.45E+04	1.81E+07	1.09E+05	2.05E+08	3.09E+06	2.75E+04	4.81E+05	2.71E+05	1.81E+04
	f-rank	4	8	3	9	7	2	6	5	1
F14	Mean	3.43E+04	1.96E+05	2.53E+04	3.58E+05	2.80E+05	1.13E+05	1.82E+06	3.85E+05	1.19E+04
	Std	3.35E+04	2.12E+05	2.23E+04	2.64E+05	4.47E+05	1.05E+05	2.21E+06	4.32E+05	9.35E+03
	f-rank	3	5	2	7	6	4	9	8	1
F15	Mean	5.90E+04	3.90E+04	7.94E+04	3.63E+07	1.52E+06	1.86E+04	1.59E+05	6.98E+04	2.08E+04
	Std	4.91E+04	1.88E+04	5.98E+04	2.68E+07	5.62E+06	1.49E+04	8.31E+04	3.64E+04	1.44E+04
	f-rank	4	3	6	9	8	1	7	5	2
F16	Mean	2.82E+03	2.87E+03	2.63E+03	3.89E+03	2.47E+03	2.78E+03	3.75E+03	3.35E+03	2.18E+03
	Std	3.04E+02	3.91E+02	3.35E+02	2.03E+02	3.03E+02	2.97E+02	4.63E+02	4.14E+02	2.57E+02
	f-rank	5	6	3	9	2	4	8	7	1
F17	Mean	2.24E+03	2.35E+03	2.05E+03	2.64E+03	2.01E+03	2.37E+03	2.61E+02	2.59E+03	1.99E+03
	Std	1.97E+02	2.77E+02	1.80E+02	2.14E+02	1.91E+02	1.89E+02	2.58E+02	2.99E+02	1.32E+02
	f-rank	4	5	3	9	2	6	8	7	1
F18	Mean	1.00E+06	4.63E+06	3.18E+05	8.26E+06	1.44E+06	1.63E+06	4.78E+06	2.04E+06	2.98E+05
	Std	9.96E+05	1.03E+07	2.54E+05	5.39E+06	2.03E+06	1.96E+06	6.09E+06	3.01E+06	2.57E+05
	f-rank	3	7	2	9	4	5	8	6	1
F19	Mean	2.52E+06	5.49E+06	1.56E+06	5.91E+07	9.12E+05	1.77E+04	8.83E+06	4.37E+05	1.56E+05
	Std	1.43E+06	2.03E+07	9.06E+05	3.27E+07	1.94E+06	1.81E+04	7.83E+06	3.39E+05	1.90E+05
	f-rank	6	7	5	9	4	1	8	3	2
F20	Mean	2.51E+03	2.53E+03	2.44E+02	2.78E+03	2.45E+03	2.62E+03	2.79E+03	2.87E+03	2.28E+03
	Std	1.85E+02	1.94E+02	1.38E+02	1.38E+02	1.56E+02	2.39E+02	1.87E+02	1.98E+02	1.34E+02
	f-rank	4	5	2	8	3	6	9	7	1
F21	Mean	2.43E+03	2.47E+03	2.40E+03	2.58E+03	2.38E+03	2.44E+03	2.59E+03	2.56E+03	2.37E+03
	Std	5.32E+01	4.48E+01	3.02E+01	2.56E+01	2.78E+01	3.79E+01	4.21E+01	5.67E+01	2.13E+01
	f-rank	4	6	3	8	2	5	9	7	1

Continued on next page

F22	Mean	3.74E+03	5.87E+03	4.97E+03	9.02E+03	4.74E+03	4.97E+03	6.81E+03	6.02E+03	5.25E+03
	Std	2.13E+03	1.99E+03	1.47E+03	2.12E+03	2.04E+03	1.57E+03	2.44E+03	2.04E+03	1.35E+03
	f-rank	1	6	4	9	2	3	8	7	5
F23	Mean	2.76E+03	2.80E+03	2.75E+03	3.04E+03	2.75E+03	2.78E+03	3.09E+03	3.14E+03	2.69E+03
	Std	3.08E+01	3.59E+01	3.50E+01	4.08E+01	3.19E+01	2.62E+01	9.67E+01	1.22E+02	1.67E+02
	f-rank	4	6	3	7	2	5	8	9	1
F24	Mean	2.92E+03	2.97E+03	2.91E+03	3.22E+03	2.93E+03	2.99E+03	3.21E+01	3.37E+03	2.87E+03
	Std	3.39E+01	2.82E+01	2.63E+01	3.27E+01	6.24E+01	5.03E+01	8.67E+01	1.16E+02	1.59E+01
	f-rank	3	5	2	8	4	6	7	9	1
F25	Mean	2.92E+03	3.24E+03	2.89E+03	3.36E+03	2.97E+03	2.89E+03	3.06E+03	2.95E+03	2.89E+03
	Std	2.02E+01	2.84E+02	1.48E+01	1.32E+02	3.66E+01	1.49E+01	4.51E+01	2.39E+01	4.9219
	f-rank	4	8	3	9	6	2	7	5	1
F26	Mean	4.72E+03	5.50E+03	4.66E+03	7.44E+02	4.72E+03	4.88E+03	8.14E+03	7.79E+03	4.46E+03
	Std	1.04E+03	5.19E+02	5.73E+02	4.53E+02	3.82E+02	6.27E+02	1.15E+03	8.04E+02	1.91E+02
	f-rank	3	6	2	7	4	5	9	8	1
F27	Mean	3.24E+03	3.23E+03	3.22E+03	3.47E+03	3.23E+03	3.23E+03	3.43E+03	3.44E+03	3.21E+03
	Std	2.25E+01	1.02E+01	1.67E+01	4.22E+01	1.73E+01	1.36E+01	1.10E+02	1.19E+02	9.5398
	f-rank	6	4	3	9	5	2	7	8	1
F28	Mean	3.27E+03	3.80E+03	3.24E+03	4.14E+03	3.37E+03	3.26E+03	3.26E+03	3.46E+03	3.24E+03
	Std	4.91E+01	6.05E+02	4.34E+01	2.98E+02	5.95E+01	5.95E+01	4.12E+01	6.75E+01	4.57E+01
	f-rank	4	8	1	9	6	3	7	5	2
F29	Mean	4.18E+03	3.97E+03	3.78E+03	4.97E+03	3.75E+03	3.87E+03	5.26E+03	4.62E+03	3.63E+03
	Std	2.56E+02	2.54E+02	1.76E+02	2.57E+02	1.51E+02	1.96E+02	5.18E+02	4.09E+02	1.59E+02
	f-rank	6	5	3	9	2	4	8	7	1
F30	Mean	6.98E+06	3.22E+05	4.12E+06	1.25E+08	8.28E+06	9.55E+04	2.03E+07	3.74E+06	5.39E+05
	Std	6.84E+06	3.71E+05	2.41E+06	4.91E+07	7.29E+06	1.39E+05	1.67E+07	2.37E+06	5.28E+05
	f-rank	6	2	5	8	7	1	9	4	3
	Average f-rank	4.2414	6.0689	3.0689	8.2759	4.1724	3.3793	7.9655	6.3793	1.3793
	Overall f-rank	5	6	2	9	4	3	8	7	1

Table 11. Comparison results of nine algorithms on CEC 2017 benchmark functions with 50 dimensions.

Function	Results	SSA	MFO	MVO	SCA	GWO	HGS	WOA	HHO	VC-SSA
F1	Mean	8.05E+03	3.41E+10	5.17E+06	5.62E+10	5.37E+09	1.81E+07	4.17E+09	3.11E+08	5.17E+06
	Std	9.85E+03	1.32E+10	1.17E+06	5.96E+09	2.33E+09	1.35E+07	1.74E+09	8.67E+07	8.51E+03
	f-rank	1	8	3	9	7	4	6	5	2
F3	Mean	1.20E+05	3.24E+05	2.09E+04	1.60E+05	1.11E+05	6.94E+04	2.12E+05	1.01E+05	2.08E+04
	Std	3.58E+04	5.98E+04	5.72E+03	2.24E+04	2.12E+04	1.77E+04	6.94E+04	1.80E+04	8.17E+03
	f-rank	6	9	2	7	5	3	8	4	1
F4	Mean	6.28E+02	3.84E+03	5.79E+02	9.80E+03	9.37E+02	6.04E+02	1.59E+03	8.71E+02	5.78E+02
	Std	4.94E+01	2.18E+03	5.05E+01	2.14E+03	2.91E+02	6.40E+01	3.08E+02	1.13E+02	4.59E+01
	f-rank	4	8	2	9	6	3	7	5	1
F5	Mean	8.32E+02	9.21E+02	7.30E+02	1.09E+03	6.98E+02	7.72E+02	1.03E+03	9.04E+02	7.30E+02
	Std	5.49E+01	7.22E+01	5.12E+01	3.37E+01	3.31E+01	3.86E+01	9.87E+01	3.83E+01	4.99E+01
	f-rank	5	7	3	9	1	4	8	6	2
F6	Mean	6.58E+02	6.47E+02	6.35E+02	6.78E+02	6.14E+02	6.16E+02	6.87E+02	6.73E+02	6.35E+02
	Std	1.19E+01	1.01E+01	1.48E+01	5.9721	3.78E+02	6.2744	1.31E+01	5.4802	3.4388
	f-rank	6	5	4	8	1	2	9	7	3
F7	Mean	1.15E+03	1.69E+03	1.08E+03	1.75E+03	1.05E+03	1.17E+03	1.76E+03	1.84E+03	1.08E+03
	Std	1.18E+02	3.56E+02	8.28E+01	9.00E+01	6.26E+01	7.89E+01	1.16E+02	9.54E+01	4.03E+01
	f-rank	4	6	3	7	1	5	8	9	2
F8	Mean	1.12E+03	1.23E+03	1.03E+03	1.40E+03	1.03E+03	1.06E+03	1.31E+03	1.19E+03	1.03E+03
	Std	5.40E+01	7.21E+01	4.13E+01	2.88E+01	5.75E+01	5.07E+01	9.11E+01	3.61E+01	3.84E+01
	f-rank	5	7	2	9	3	4	8	6	1
F9	Mean	1.39E+04	1.79E+04	1.42E+04	2.78E+04	6.58E+03	1.23E+04	3.22E+04	2.48E+04	1.43E+04
	Std	2.61E+03	5.18E+03	6.27E+03	4.06E+03	3.42E+03	3.02E+03	9.50E+03	3.40E+03	1.79E+03
	f-rank	3	6	4	8	1	2	9	7	5
F10	Mean	7.69E+03	8.32E+03	7.18E+03	1.52E+04	7.32E+03	6.66E+03	1.19E+04	9.23E+03	7.18E+03
	Std	1.16E+03	1.22E+03	1.05E+03	4.68E+02	1.65E+03	7.45E+02	1.46E+03	9.60E+02	1.01E+03
	f-rank	5	6	3	9	4	1	8	7	2
F11	Mean	1.65E+03	9.65E+03	1.51E+03	9.86E+03	3.95E+03	1.38E+03	3.39E+03	1.65E+03	1.51E+03
	Std	1.12E+02	6.52E+03	8.69E+01	2.15E+03	1.54E+03	6.36E+01	5.96E+02	1.40E+02	7.17E+01
	f-rank	5	8	3	9	7	1	6	5	2
F12	Mean	1.43E+08	2.26E+09	7.90E+07	1.67E+10	4.82E+08	3.63E+07	9.73E+08	2.47E+08	4.47E+07
	Std	1.05E+08	2.09E+09	3.68E+07	2.83E+09	3.94E+08	3.17E+07	4.80E+08	1.52E+08	3.59E+07
	f-rank	4	8	3	9	6	1	7	5	2

Continued on next page

F13	Mean	1.70E+05	1.97E+08	2.27E+05	5.22E+09	9.48E+07	2.70E+04	3.61E+07	4.66E+06	2.27E+05
	Std	1.68E+05	5.29E+08	1.16E+05	2.09E+09	1.48E+08	1.45E+04	3.47E+07	3.74E+06	2.83E+04
	f-rank	2	8	4	9	7	1	6	5	3
F14	Mean	2.43E+05	1.51E+06	1.98E+05	4.49E+06	6.97E+05	4.54E+05	2.49E+06	1.31E+06	1.98E+05
	Std	1.59E+05	1.83E+06	1.15E+06	2.28E+06	5.70E+05	2.96E+05	1.48E+06	1.12E+06	9.32E+04
	f-rank	3	7	2	9	5	4	8	6	1
F15	Mean	7.59E+04	1.19E+07	1.04E+05	7.34E+08	1.46E+07	1.98E+04	4.11E+06	6.29E+05	1.04E+05
	Std	5.86E+04	2.69E+07	4.40E+04	255E+08	2.02E+07	8.19E+03	7.65E+06	2.25E+05	1.89E+04
	f-rank	2	7	4	9	8	1	6	5	3
F16	Mean	3.87E+03	4.17E+03	3.18E+03	5.93E+03	3.15E+03	3.97E+03	5.61E+03	4.61E+03	3.18E+03
	Std	4.47E+02	4.96E+02	3.82E+02	3.10E+02	5.07E+02	4.73E+02	8.91E+02	7.06E+02	3.41E+02
	f-rank	4	6	3	9	1	5	8	7	2
F17	Mean	3.42E+03	3.89E+03	3.25E+03	4.86E+03	2.99E+03	3.34E+03	4.27E+03	3.77E+03	3.25E+03
	Std	3.59E+02	4.80E+02	3.37E+02	3.04E+02	4.09E+02	3.84E+02	5.39E+02	3.47E+02	2.64E+02
	f-rank	5	7	3	9	1	4	8	6	2
F18	Mean	3.52E+06	9.07E+06	2.20E+06	2.79E+07	5.13E+06	4.32E+06	1.87E+07	3.75E+06	2.19E+06
	Std	3.10E+06	1.02E+07	1.50E+06	1.18E+07	4.46E+06	3.05E+06	1.28E+07	2.32E+06	7.36E+05
	f-rank	3	7	2	9	6	5	8	4	1
F19	Mean	5.85E+06	7.22E+06	3.95E+06	4.44E+08	4.16E+06	2.12E+04	5.89E+06	1.27E+06	3.95E+06
	Std	3.67E+06	3.26E+07	2.84E+06	1.90E+08	9.44E+06	1.81E+04	7.34E+06	9.71E+05	6.12E+05
	f-rank	6	8	4	9	5	1	7	2	3
F20	Mean	3.24E+03	3.48E+03	3.11E+03	4.20E+03	2.84E+03	3.24E+03	3.85E+03	3.45E+03	3.11E+03
	Std	3.95E+02	3.63E+02	3.39E+02	2.18E+02	3.09E+02	3.30E+02	3.31E+02	3.06E+02	2.68E+02
	f-rank	5	7	3	9	1	5	8	6	2
F21	Mean	2.65E+03	2.69E+03	2.52E+03	2.93E+03	2.52E+03	2.56E+03	2.99E+03	2.82E+03	2.52E+03
	Std	7.45E+01	6.89E+01	4.43E+01	4.88E+01	6.85E+01	4.91E+01	1.06E+02	8.57E+01	3.38E+01
	f-rank	5	6	2	8	3	4	9	7	1
F22	Mean	9.54E+03	1.07E+04	9.00E+03	1.68E+04	8.86E+03	8.78E+03	1.32E+04	1.16E+04	9.01E+03
	Std	1.03E+03	1.74E+03	1.02E+03	2.99E+02	8.68E+02	1.05E+03	9.62E+02	8.29E+02	9.93E+02
	f-rank	5	6	3	9	2	1	8	7	4
F23	Mean	3.03E+03	3.10E+03	2.95E+03	3.63E+03	2.96E+03	3.00E+03	3.73E+03	3.82E+03	2.95E+03
	Std	8.01E+01	6.84E+01	5.68E+01	8.54E+01	6.77E+01	5.26E+01	1.77E+02	1.83E+02	3.12E+01
	f-rank	5	6	2	7	3	4	8	9	1
F24	Mean	3.14E+03	3.19E+03	3.11E+03	3.78E+03	3.12E+03	3.27E+03	3.81E+03	4.15E+03	3.11E+03
	Std	6.31E+01	5.17E+01	5.67E+01	6.76E+01	7.24E+01	1.00E+02	1.62E+02	1.65E+02	2.83E+01
	f-rank	4	5	2	7	3	6	8	9	1

Continued on next page

F25	Mean	3.10E+03	4.69E+03	3.05E+03	7.89E+03	3.47E+03	3.09E+03	3.66E+03	3.28E+03	3.05E+03
	Std	4.19E+01	1.46E+03	3.95E+01	1.06E+03	2.25E+02	3.39E+01	2.23E+02	6.34E+01	2.46E+01
	f-rank	4	8	2	9	6	3	7	5	1
F26	Mean	5.05E+03	7.89E+03	6.05E+03	1.32E+04	6.16E+03	5.52E+03	1.39E+04	1.07E+04	6.04E+03
	Std	2.09E+03	7.26E+02	5.53E+02	8.52E+02	5.81E+02	2.19E+03	1.50E+03	2.03E+03	3.73E+02
	f-rank	1	6	4	8	5	2	9	7	3
F27	Mean	3.55E+03	3.54E+03	3.35E+04	4.73E+03	3.57E+03	3.42E+03	4.67E+03	4.50E+03	3.41E+03
	Std	1.22E+02	7.71E+01	4.99E+01	1.96E+02	8.32E+01	6.52E+01	5.52E+02	4.47E+02	6.44E+02
	f-rank	4	3	9	8	5	2	7	6	1
F28	Mean	3.39E+03	7.73E+03	3.30E+03	7.91E+03	4.01E+03	3.41E+03	4.55E+03	3.87E+03	3.76E+03
	Std	5.01E+01	1.07E+03	2.33E+01	8.57E+02	2.41E+02	8.23E+01	3.59E+02	1.53E+02	1.33E+03
	f-rank	2	8	1	9	6	3	7	5	4
F29	Mean	5.21E+03	5.02E+03	4.86E+03	8.02E+03	4.53E+03	4.44E+03	8.02E+03	6.42E+03	4.33E+03
	Std	4.64E+02	4.60E+02	3.28E+02	7.27E+02	2.57E+02	3.19E+02	1.03E+03	7.41E+02	3.84E+02
	f-rank	6	5	4	9	3	2	8	7	1
F30	Mean	1.23E+08	2.39E+07	6.49E+07	9.76E+08	1.08E+08	1.99E+06	2.19E+08	5.62E+07	2.99E+07
	Std	3.21E+07	5.65E+07	1.77E+07	3.59E+08	3.29E+07	7.80E+05	1.11E+08	2.48E+07	1.04E+07
	f-rank	7	2	5	9	6	1	8	4	3
	Average f-rank	4.1724	6.5517	3.1379	8.5517	4.0690	2.8966	7.6552	5.9655	2.0689
	Overall f-rank	5	7	3	9	4	2	8	6	1

5. Application to engineering disciplines

In this section, five restricted practical engineering problems are chosen to testify the applicability of VC-SSA on real applications. These five problems namely, car side impact design (CSID) 62, pressure vessel design (PVD) 57, cantilever beam design (CBD) 63, speed reducer design (SRD), and three-bar truss design (TBTD) 57, which have been extensively studied to evaluate the ability of the metaheuristic approach to address real-life problems. The applied practical design cases have different constrains, and it is indispensable to use adequate methods to deal with them to successfully obtain feasible solutions. In this experiment, we use the penalty function approach 64 to cope with the optimization constrains. The parameter settings are as consistent with Subsection 4.2.

5.1. CSID problem

The CSID problem was originally introduced by Gu et al. 65. For this case, the vehicle finite element model illustrated in Figure 9 is adopted to investigate the crashworthiness of the vehicle. The object of the problem is to optimize the gross weight of the vehicle by identifying eleven hybrid variables. All the decision variables are continuous except for the eighth and ninth variables. This challenging problem is restricted by ten unequal constrains.

The mathematical model of the CSID problem can be described as follows:
Objective function:

$$f(x) = 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 2.73x_7 \tag{15}$$

Subject to:

$$\begin{aligned}
g_1(x) &= 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10} \le 1, \\
g_2(x) &= 0.261 - 0.0159x_1x_2 - 0.188x_1x_8 - 0.019x_2x_7 + 0.0144x_3x_5 \\
&\quad + 0.0008757x_5x_{10} + 0.080405x_6x_9 + 0.00139x_8x_{11} + 0.00001575x_{10}x_{11} \le 0.32, \\
g_3(x) &= 0.214 + 0.00817x_5 - 0.131x_1x_8 - 0.0704x_1x_9 - 0.018x_2x_7 \\
&\quad + 0.0208x_3x_8 + 0.121x_3x_9 - 0.00364x_5x_6 + 0.0007715x_5x_{10} \\
&\quad - 0.0005354x_6x_{10} + 0.00121x_8x_{11} \le 0.32, \\
g_4(x) &= 0.074 - 0.061x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2 \le 0.32, \\
g_5(x) &= 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 \\
&\quad - 7.77x_7x_8 + 0.32x_9x_{10} \le 32, \\
g_6(x) &= 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11.0x_2x_8 \\
&\quad - 0.0215x_5x_{10} - 9.98x_7x_8 + 22.0x_8x_9 \le 32, \\
g_7(x) &= 46.36 - 9.9x_2 - 12.9x_1x_8 + 0.1107x_3x_{10} \le 32, \\
g_8(x) &= 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 \le 4, \\
g_9(x) &= 10.58 - 0.674x_1x_2 - 1.95x_2x_8 + 0.02054x_3x_{10} \\
&\quad - 0.0198x_4x_{10} + 0.028x_6x_{10} \le 9.9, \\
g_{10}(x) &= 16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} \\
&\quad - 0.000786x_{11}^2 \le 15.7,
\end{aligned} \tag{16}$$

where $0.5 \le x_1$-$x_7 \le 1.5$, x_8, $x_9 \in (0.192, 0.345)$ and $-30 \le x_{10}$, $x_{11} \le 30$.

The VC-SSA is utilized to address this challenging problem, and compared with eight well performed algorithms, namely, GA 66, PSO 2, FA 9, ABC 6, GWO 13, WCA 67, SAFA 68, and SSA 24. The optimal results achieved by the approaches for the problem are reported in Table 12.

From Table 12, VC-SSA obtains the best value among the methods. This indicates that the total weight of the vehicle can be 22.8412 when the eleven parameters are set as 0.5017, 1.1172, 0.5000, 1.2959, 0.5001, 1.5000, 0.5035, 0.3450, 0.1920, -18.6991, and 0.0379. Therefore, VC-SSA can be used as an effective tool for solving the optimal vehicle crash design problem.

Figure 9. Model of the CSID.

Table 12. Results of VC-SSA versus other works for CSID.

Algorithm	VC-SSA	WCA	SAFA	GWO	ABC	PSO	FA	GA	SSA
x_1	0.5017	0.5000	0.5000	0.5000	0.5000	0.5003	0.5000	0.5000	0.5000
x_2	1.1172	1.1156	1.1196	1.1153	1.1216	1.1211	1.1157	1.2802	1.1093
x_3	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5004	0.5000	0.5000
x_4	1.2959	1.3034	1.2971	1.3043	1.2942	1.2952	1.3051	1.0330	1.3148
x_5	0.5001	0.5000	0.5000	0.5000	0.5000	0.5003	0.5002	0.5000	0.5000
x_6	1.5000	1.5000	1.5000	1.5000	1.4999	1.4998	1.4993	0.5000	1.5000
x_7	0.5035	0.5000	0.5000	0.5000	0.5003	0.5002	0.5006	0.5000	0.5000
x_8	0.3450	0.3450	0.3450	0.3450	0.3450	0.3450	0.3450	0.3450	0.3450
x_9	0.1920	0.1920	0.1920	0.3450	0.3450	0.3450	0.1920	0.1920	0.1920
x_{10}	-18.6991	-19.6996	-18.9941	-19.7485	-18.6349	-18.725	-19.6744	10.3119	-20.8217
x_{11}	0.0379	-0.0239	0.1949	0.7174	-0.3053	0.3862	-1.4541	0.0017	0.4413
Optimal weight	22.8412	22.8436	22.8438	22.8448	22.8463	22.849	22.8542	22.8565	23.0422

5.2. PVD problem

The second engineering design problem that we study in this work is PVD. The goal of this case is to minimize the overall cost, which is strictly correlated with material, forming and welding, as illustrated in Figure 10. In the PVD project, there are four decision variables to be authorized, including the thickness of the shell (T_s), the thickness of the head (T_h), the inner radius (R), and the cylinder section length (L). The mathematical expression can be formulated as shown below:

Consider:

$$x = (x_1, x_2, x_3, x_4) = (T_s, T_h, R, L) \tag{17}$$

Objective function:

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 19.84x_1^2x_3 + 3.1661x_1^2x_4 \tag{18}$$

Subject to:

$$\begin{aligned} &g_1(x) = 0.0193x_3 - x_1 \le 0, \\ &g_2(x) = 0.00954x_3 - x_2 \le 0, \\ &g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0, \\ &g_4(x) = x_4 - 240 \le 0. \end{aligned} \tag{19}$$

where $0 \le x_1 \le 99$, $0 \le x_2 \le 99$, $10 \le x_3 \le 200$, $10 \le x_4 \le 200$.

This task has been addressed by the advocated VC-SSA approach, and compared with nine classical methods, namely, GSA 69, ABC 6, BA [70], MFO 8, HHO 15, GWO 13, WOA 11, AOS 71, and SSA 24. The best-obtained solutions from all involved optimizers are reported in Table 13. From Table 13, VC-SSA can get design a pressure vessel with the optimal total cost, i.e., 5886.9631. In addition, MBA and ASO can provide competitive results, which are 5889.3216 and 5888.4579, respectively.

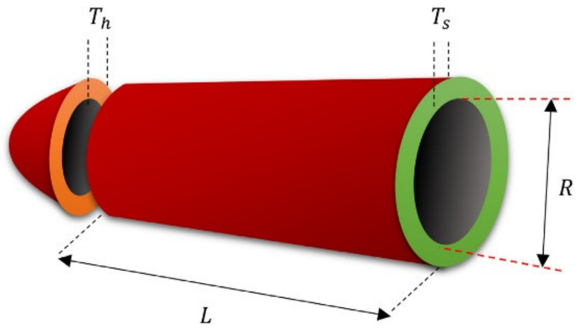

Figure 10. PVD problem.

Table 13. Results of VC-SSA versus other works for PVD.

Algorithm	Optimal values for variables				Optimal cost
	T_s	T_h	R	L	
VC-SSA	0.7812	0.3848	40.4734	197.8583	5886.9631
ASO	0.7786744	0.3853217	40.3408906	199.7215178	5888.4579
HHO	0.8175838	0.4072927	42.09174576	176.7196352	6000.4625
SSA	0.790678	0.390834	40.96773875	195.91822	6012.1885
GWO	0.8125	0.4345	42.098181	176.75873	6051.5639
BA	0.8125	0.4375	42.0984456	176.6365958	6059.7143
MFO	0.8125	0.4375	42.098445	176.636596	6059.7143
ABC	0.8125	0.4375	42.098446	176.636576	6059.7143
WOA	0.812500	0.437500	42.0982699	176.638998	6059.7410
GSA	1.125	0.625	55.9886598	84.4542025	8538.8359

5.3. CBD problem

The third real-world engineering problem that we investigate is CBD. In this problem, the objective function is to minimize the weight of the cantilever beam. As illustrated in Figure 11, this issue has five structure parameters. There is one more vertical displacement constraint that requires be considered. The CBD problem can be modelled as follows:

Objective function:

$$f(x) = 0.0624 \times (x_1 + x_2 + x_3 + x_4 + x_5) \tag{20}$$

Subject to:

$$g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0 \tag{21}$$

where $0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100$.

This problem has been tackled by MFO 8, MVO 72, SOS 73, CS 12, ALO 74, MMA 75, and SSA 24. The optimal results obtained by VC-SSA and other algorithms on this issue are presented in Table 14. From Table 14, VC-SSA outperforms all competitors. Therefore, the developed algorithm can provide strong aid for the CBD problem.

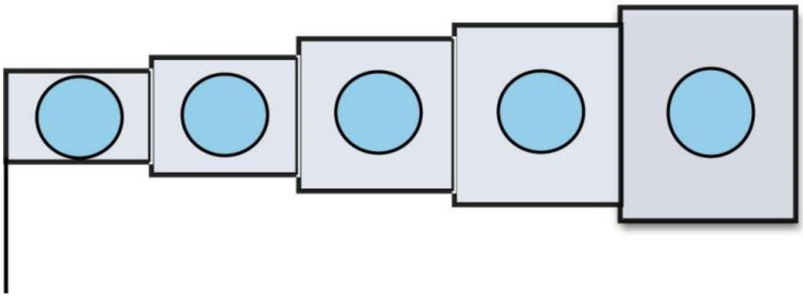

Figure 11. CBD problem.

Table 14. Results of VC-SSA versus other works for CBD problem.

Algorithm	Optimal value for variables					Optimal cost
	x_1	x_2	x_3	x_4	x_5	
VC-SSA	6.5612	5.4789	4.1526	3.1172	2.0084	1.32685
ALO	6.01812	5.31142	4.48836	3.49751	2.158329	1.33995
SSA	6.0151345	5.3093046	4.4950067	3.5014262	2.1527879	1.33995639
MVO	6.0239402	5.30601123	4.49501132	3.49602232	2.15272617	1.3399595
SOS	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
MFO	5.9848717	5.3167269	4.4973325	3.5136164	2.1616202	1.339988085
CS	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
MMA	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400

5.4. SRD problem

In this subsection, the SRD issue is considered, whose main objective is to devise a speed reducer with the minimum weight. Seven structure variables $x = (b, m, z, l_1, l_2, d_1, d_2)$ are concluded in this problem, where b is the surface width, m is the module of gear teeth, z represents the quantity of teeth on pinion, l_1 and l_2 indicate the lengths of the first and second shafts between bearings, respectively, and d_1 and d_2 denote the diameters of the first and second shafts, respectively. For solving this problem, four design constrains needed to be considered. The configuration of this problem is illustrated in Figure 12, and the mathematical expression can be expressed as follows:

Consider:

$$x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (b, m, z, l_1, l_2, d_1, d_2) \tag{22}$$

Objective function:

$$\begin{aligned} f(x) = {} & 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934) - 1.508 x_1 (x_6^2 + x_7^2) \\ & + 7.4777 (x_6^3 + x_7^3) + 0.7854 (x_4 x_6^2 + x_5 x_7^2) \end{aligned} \tag{23}$$

Subject to:

$$
\begin{aligned}
& g_1(x) = -x_1 + 0.0193x_3 \le 0,\ g_2(x) = \frac{397.5}{x_1 x_2^2 x_3} - 1 \le 0, \\
& g_3(x) = \frac{1.93x_4^3}{x_2 x_6^4 x_3} - 1 \le 0,\ g_4(x) = \frac{1.93x_5^3}{x_2 x_7^5 x_3} - 1 \le 0, \\
& g_5(x) = \frac{\left[(745x_4/x_2x_3)^2 + 16.9\times 10^6\right]^{\frac{1}{2}}}{110x_6^3} - 1 \le 0, \\
& g_6(x) = \frac{\left[(745x_5/x_2x_3)^2 + 157.5\times 10^6\right]^{\frac{1}{2}}}{85x_7^3} - 1 \le 0, \\
& g_7(x) = \frac{x_2 x_3}{40} - 1 \le 0,\ g_8(x) = \frac{5x_2}{x_1} - 1 \le 0, \\
& g_9(x) = \frac{x_1}{12x_2} - 1 \le 0,\ g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0, \\
& g_{11}(x) = \frac{1.1x_7 + 1.7}{x_5} - 1 \le 0.
\end{aligned} \tag{24}
$$

where $2.6 \le x_1 \le 3.6$, $0.7 \le x_1 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.3 \le x_5 \le 8.3$, $3.9 \le x_6 \le 3.9$, $5.0 \le x_7 \le 5.5$.

This problem has been addressed by several nature-inspired metaheuristics including ABC 6, CS 15, SCA 45, MBFPA 76, WWO 77, and PVS 78. Table 15 tabulates the generated optimal solution by VC-SSA and its peer algorithms. From Table 15, VC-SSA can obtain the best weight, i.e., 2841.600123. The other compared algorithms achieve similar results.

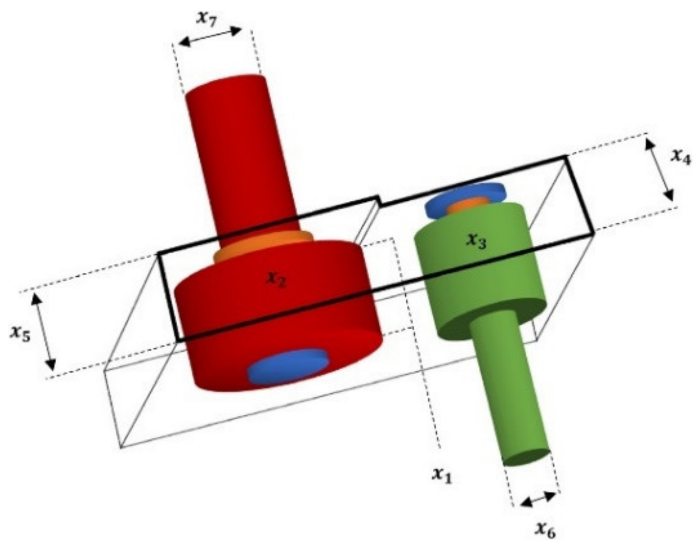

Figure 12. Construction of a speed reducer.

Table 15. Results of VC-SSA versus other works for SRD problem.

Algorithm	Optimal values for variables							Optimal cost
	b	m	z	l_1	l_2	d_1	d_2	
VC-SSA	3.60000	0.7000	17	7.3930	7.3617	3.2615	5.0141	2841.600123
MBFPA	3.5	0.7	17	7.3	7.7153199122	3.35021466	5.28665446	2994.341315
CS	3.5	0.7	17	7.3	7.715319	3.350214	5.286654	2994.471066
ABC	3.5	0.7	17	7.3	7.715319	3.350214	5.286654	2994.471066
WWO	3.5	0.7	17	7.3	7.715319	3.350214	5.286654	2994.471066
PVS	3.49999	0.6999	17	7.3	7.8	3.3502	5.2866	2996.3481
SCA	0.350001	0.7	17	7.300156	7.800027	3.350221	5.286685	2996.356689

5.5. TBTD problem

The last constrained engineering design problem, named TSTD, has two decision variables with the target function of minimize the weight. In this problem, there are three constraints that need to be considered. The schematic of this problem is plotted in Figure 13. The corresponding expression of this project is as follows:

Consider:

$$x = (x_1, x_2) = (A_1 A_2) \tag{25}$$

Objective function:

$$f(x) = (2\sqrt{2}x_1 + x_2) \times l \tag{26}$$

Subjective to:

$$\begin{aligned} g_1(x) &= \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0, \\ g_2(x) &= \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0, \\ g_3(x) &= \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \le 0. \end{aligned} \tag{27}$$

where $0 \le x_1, x_2 \le 1$, $L = 100\text{cm}$, $P = 2\text{km/cm}^2$, $\sigma = 2\text{km/cm}^2$.

The algorithms HHO 15, MVO 72, GOA 17, MFO 8, SSA 24, ALO 74, CS 12 and WWO 77 are respectively employed to solve the TBTD problem. The outcomes gained by these algorithms and VC-SSA are shown in Table 16. With respect to the comparison results, VC-SSA can offer the optimal results, i.e., 263. 89584337713. The outstanding outputs proves that the proposed approach can effectively solve the TBTD problem.

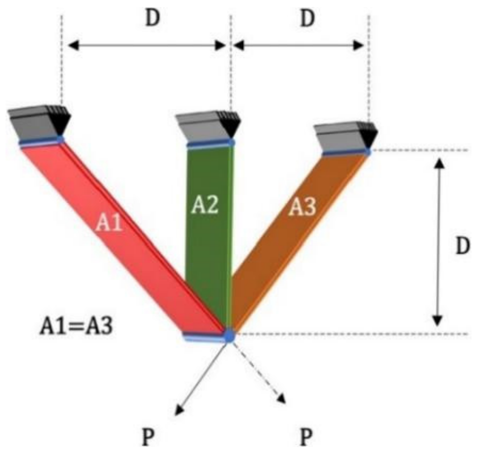

Figure 13. TBTD problem.

Table 16. Results of VC-SSA versus other works for TBTD problem.

Algorithm	Optimal values for variables		Optimal weight
	x_1	x_2	
VC-SSA	0.8958433771383	0.408245590095633	263.895843
WWO	0.788651	0.408316	263.895843
HHO	0.788662816	0.408283133832900	263.8958434
SSA	0.788665414258065	0.408275784444547	263.8958434
ALO	0.788662816000317	0.408283133832901	263.8958434
MVO	0.78860276	0.40845307	263.8958499
GOA	0.788897555578973	0.407619570115153	263.895881496069
MFO	0.788244770931922	0.409466905784741	263.895979682
CS	0.78867	0.40902	263.9716

6. Robot path planning

Mobile robot path planning (MRPP) is a key problem in the field of robotics, where the main goal is to generate a collision-free path from the starting position to the destination for an autonomous mobile robot (AMR). Since the inception of mobile robots, researchers have been conducting corresponding research with the hope of designing efficient MRPP methods. Topologically, the MRPP problem is closely related to the problem of finding the shortest path between two points in the plane 79. In recent years, nature-inspired swarm intelligence metaheuristic techniques have become increasingly popular in MRPP tasks due to their strong stochasticity, robust search capability, and ease of implementation 80. In this section, we propose a VC-SSA-based MRPP approach for AMRs to help robots move along the shortest collision-avoiding path from one position to another.

6.1. Robot path planning problem description

The MRPP problem can be treated as a constrained optimization task, where the trajectory between two locations is the objective to be optimized, and the threatening region is the constraint to be considered. Nature-inspired swarm intelligence metaheuristic approaches are effective tools for solving optimization problems by optimizing the objective function to acquire the feasible solution to the issue. The results obtained using the metaheuristic technique for MRPP tasks are influenced by two factors, namely, the performance of the metaheuristic method and the effectiveness of the goal function. According to the previous experimental results, the VC-SSA algorithm has outstanding optimization performance; therefore, the main factor that affects the effectiveness of the VC-SSA-based MRPP algorithm is the objective function. We need to establish an efficient objective function according to the characteristics of the MRPP problem, and the VC-SSA algorithm completes the path planning task by evaluating and optimizing the established objective function. Based on the above analysis, an objective function is designed considering the trail length and obstacle-avoiding for evaluating the quality of the paths produced by the VC-SSA-based MRPP approach. The proposed objective function is as follows:

$$F = L(1 + \varpi \cdot \eta) \tag{28}$$

where ϖ is the scaling factor used to motivate conflict-free paths and prevent routes containing threatening regions, and L denotes the route length. To calculate L, the following formula is applied.

$$L = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{29}$$

where (x_i, y_i) represents the coordinates of the ith interpolation point.

In Eq (28), η is a marker variable that is used to evaluate the security condition of the current route. To calculate it, the following mathematical formula is utilized.

$$\eta = \sum_{k=1}^{nobs} \sum_{j=1}^{m} max\left(1 - \frac{d_{j,k}}{robs_k}, 0\right) \tag{30}$$

where $nobs$ is the number of barriers, m represents the quantity of interpolant points in the path, $d_{j,k}$ denotes the distance between the interpolant spot j to barrier k, and $robs_k$ indicates the radius of the kth threatening region. It is clear from Eq (30) that if the current route includes no obstacles, then the calculated η-value is smaller, and vice versa.

6.2. Experiment and results

In this subsection, we implement the VC-SSA-based MRPP algorithm, aiming to plan a globally optimal collision-avoiding route with the shortest path length. To validate the feasibility and superiority of the developed approach, five maps provided by 81 are adopted. These environmental maps have different characteristics, such as different number and size of obstacles and different distances between the starting and target points, as detailed in Table 17. After study the feasibility of VC-SSA in MRPP problems, the routes planned by this approach were compared with those generated by other metaheuristics, namely SSA, GWO, PSO, FA and ABC. For a fair comparison, the general parameter settings of the algorithms are identical, while the specific parameter setting of each method are taken from the source paper. The path lengths generated by the six methods in different environments are presented in Table 18, and the respective trajectories are presented in Figures 14–18.

Table 17. Type of environment.

Terrain	No. of obstacles	Initial coordinates	Final coordinates	X axis	Y axis	Obstacle radius
Map 1	3	0, 0	4, 6	[1, 1.8, 4.5]	[1, 5.0, 0.9]	[0.8, 1.5, 1]
Map 2	6	0, 0	10, 10	[1.5, 8.5, 3.2, 6.0, 1.2, 7.0]	[4.5, 6.5, 2.5, 3.5, 1.5, 8.0]	[1.5, 0.9, 0.4, 0.6, 0.8, 0.6]
Map 3	13	3, 3	14, 14	[1.5, 4.0, 1.2, 5.2, 9.5, 6.5, 10.8, 5.9, 3.4, 8.6, 11.6, 3.3, 11.8]	[4.5, 3.0, 1.5, 3.7, 10.3, 7.3, 6.3, 9.9, 5.6, 8.2, 8.6, 11.5, 11.5]	[0.5, 0.4, 0.4, 0.8, 0.7, 0.7, 0.7, 0.7, 0.7, 0.7, 0.7, 0.7, 0.7]
Map 4	30	3, 3	14, 14	[10.1, 10.6, 11.1, 11.6, 12.1, 11.2, 11.7, 12.2, 12.7, 13.2, 11.4, 11.9, 12.4, 12.9, 13.4, 8, 8.5, 9, 9.5, 10, 9.3, 9.8, 10.3, 10.8, 11.3, 5.9, 6.4, 6.9, 7.4, 7.9]	[8.8, 8.8, 8.8, 8.8, 8.8, 11.7, 11.7, 11.7, 11.7, 11.7, 9.3, 9.3, 9.3, 9.3, 9.3, 5.3, 5.3, 5.3, 5.3, 5.3, 6.7, 6.7, 6.7, 6.7, 6.7, 8.4, 8.4, 8.4, 8.4, 8.4]	[0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4]
Map 5	45	0, 0	15, 15	[2, 2, 2, 2, 2, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 8, 8, 8, 8, 8, 8, 8, 8, 8, 10, 10, 10, 10, 10, 10, 10, 10, 10, 12, 12, 12, 12, 12, 14, 14, 14, 14]	[8, 8.5, 9, 9.5, 10, 10.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 11, 11.5, 12, 1, 1.5, 2, 2.5, 3, 3.4, 4, 4.5, 5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10, 10, 10.5, 11, 11.5, 12, 10, 10.5, 11, 11.5]	[0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4]

Table 18. The minimum route length comparison of VC-SSA-based MRPP method and comparison approaches under five environmental setups.

Terrain	PSO Path length	FA Path length	ABC Path length	GWO Path length	SSA Path length	VC-SSA Path length
Map 1	7.7328	7.5281	7.7527	7.7625	8.6017	**7.4796**
Map 2	14.5037	14.4288	14.8418	14.3565	15.2430	**14.3144**
Map 3	17.2219	16.1593	17.4818	17.3517	16.9979	**15.9858**
Map 4	16.1925	16.3031	16.2919	16.3281	16.8157	**15.8511**
Map 5	21.8520	21.9933	21.8792	22.3073	22.1842	**21.5750**

From Table 18, the lengths of the paths planned by VC-SSA for all five terrains are the shortest compared to the comparison algorithms. This proves that the proposed VC-SSA-based MRPP method is an effective path planning tool that can be applied in both simple and complex environmental setups. Next, we analyzed the simulation results in detail based on the specific trajectories planned for each type of environment by the six algorithms.

Optimal paths planned by all approaches for the first environmental setup is exhibited in Figure 14(a–f). From the figure, all the algorithms can plan a collision-free path for this map. In terms of route length, SSA takes the longest route from the initial point to destination, while VC-SSA follows the shortest obstacle-avoiding route. Regarding the generated trajectories, FA and SSA design similar paths, while PSO, GWO, ABC and VC-SSA create another type of straightforward paths. Overall, VC-SSA avoids the local optimum and designs the most reasonable obstacle-free path for the first environmental setup.

Optimal paths designed by all algorithms for the second environmental setup is shown in Figure 15(a–f). The comparison results reveal that VC-SSA outperforms the compared approaches for this environment setting with the shortest path length of 14.3120, followed by GWO, FA, PSO, ABC and SSA. With respect to the trajectory, FA, ABC, GWO, SSA and VC-SSA produce similar routes, while PSO provides a different style of route. Based on the above analysis, all algorithms generate a path without collisions, while VC-SSA circumvents the local optima and thus outperforms its peers.

A comparison of the best trajectories produced by all approaches for the third environment map is displayed in Figure 16(a–f). The figure depicts that all approaches can generate an unobstructed route. FA, GWO and SSA fall into the local optimum, which leads to the generated path having redundancy. Nevertheless, PSO and VC-SSA provide satisfactory routes for this map. Since VC-SSA plans a more sensible trajectory, it provides a shorter collision-avoiding path than PSO.

The best performance of all algorithms for the fourth environment map with 30 obstacles of different scales are displayed in Figure 17(a–f). From the figure, PSO, ABC, GWO and SSA try to take the same trajectory during the maneuver from the starting point to the terminal point to produce a collision avoidance optimal path, while FA and VC-SSA take another more subtle trajectory. Compared with the trajectories provided by PSO, ABC, GWO and SSA, the curved routes generated by FA and VCSSA are smoother. Whereas FA falls into a sub-optimal solution, which affects the quality of the generated paths, VC-SSA obtains the optimal path with the length of 15.8750.

The simulation results of optimal routes produced by all methods for the fifth landscape are shown in Figure 18. From the figure, PSO, FA, ABC and SSA attempt to take the same trajectory during the maneuver from the initial position to goal to produce an obstacle-free shortest path, while GWO and VC-SSA attend to take a more promising path. However, GWO is trapped in a local optimum, while VC-SSA avoids the local optimal solution and exhibits better performance than the competitors with the path length of 21.5506.

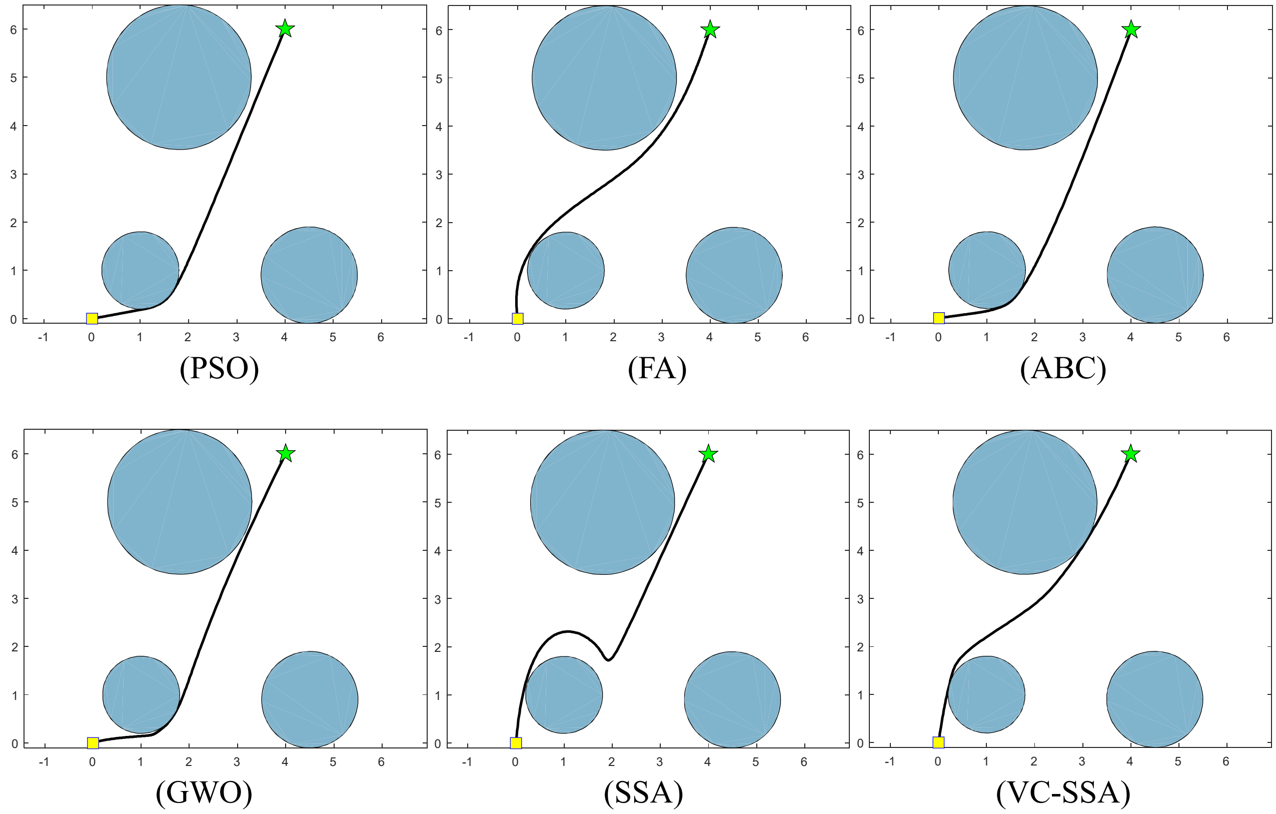

Figure 14. Map 1 (a) PSO, (b) FA, (c) ABC, (d) GWO, (e) SSA and (f) VC-SSA.

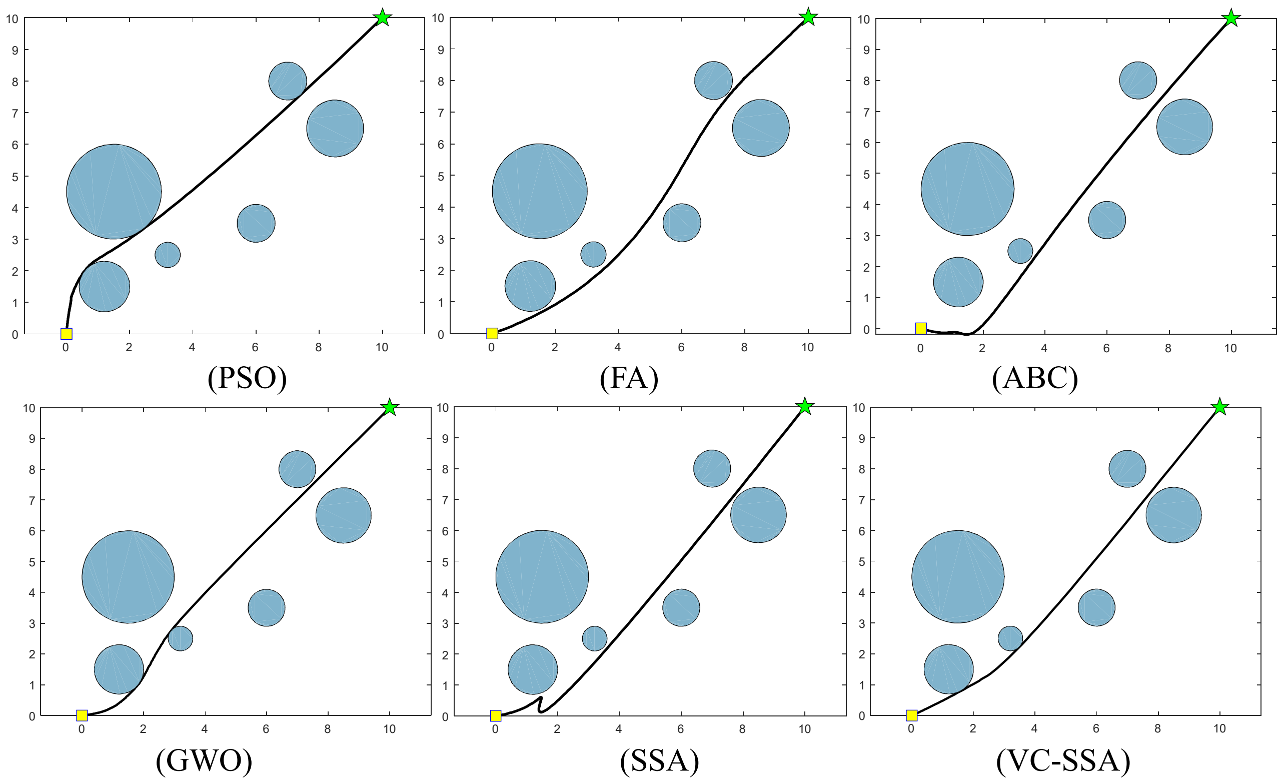

Figure 15. Map 2 (a) PSO, (b) FA, (c) ABC, (d) GWO, (e) SSA and (f) VC-SSA.

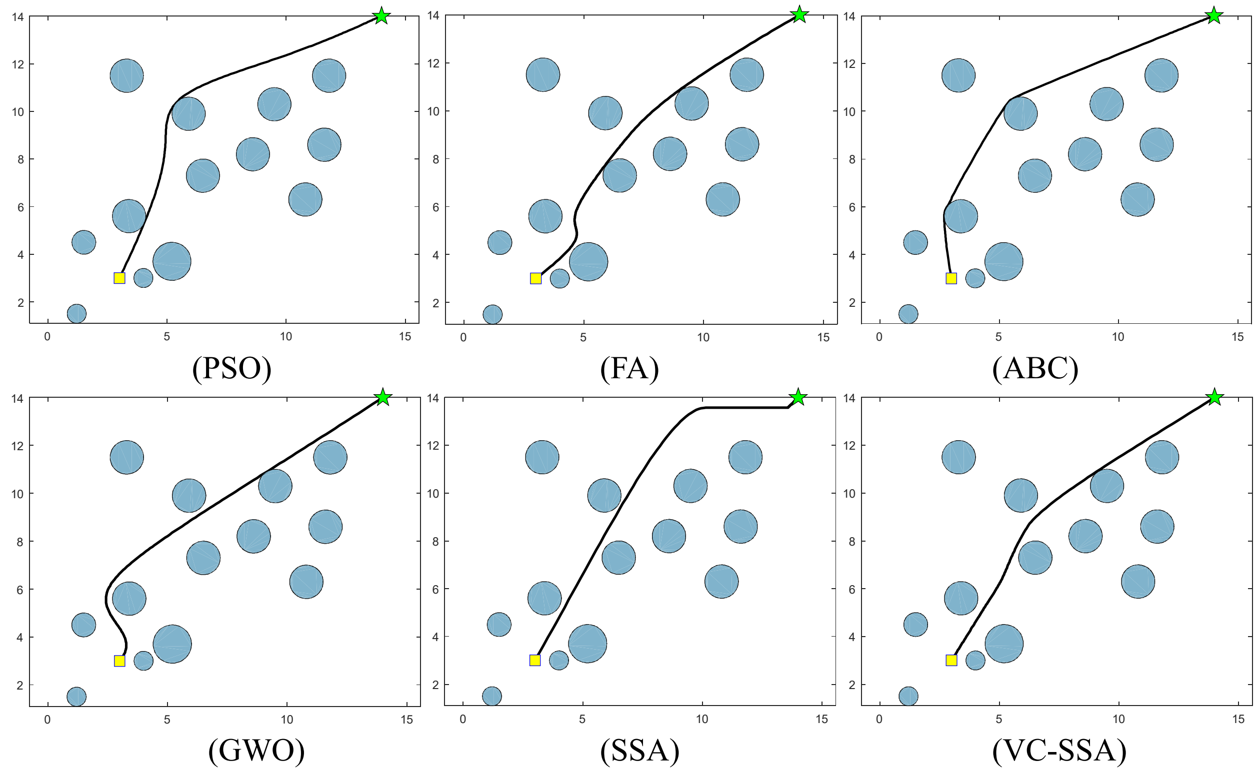

Figure 16. Map 3 (a) PSO, (b) FA, (c) ABC, (d) GWO, (e) SSA and (f) VC-SSA.

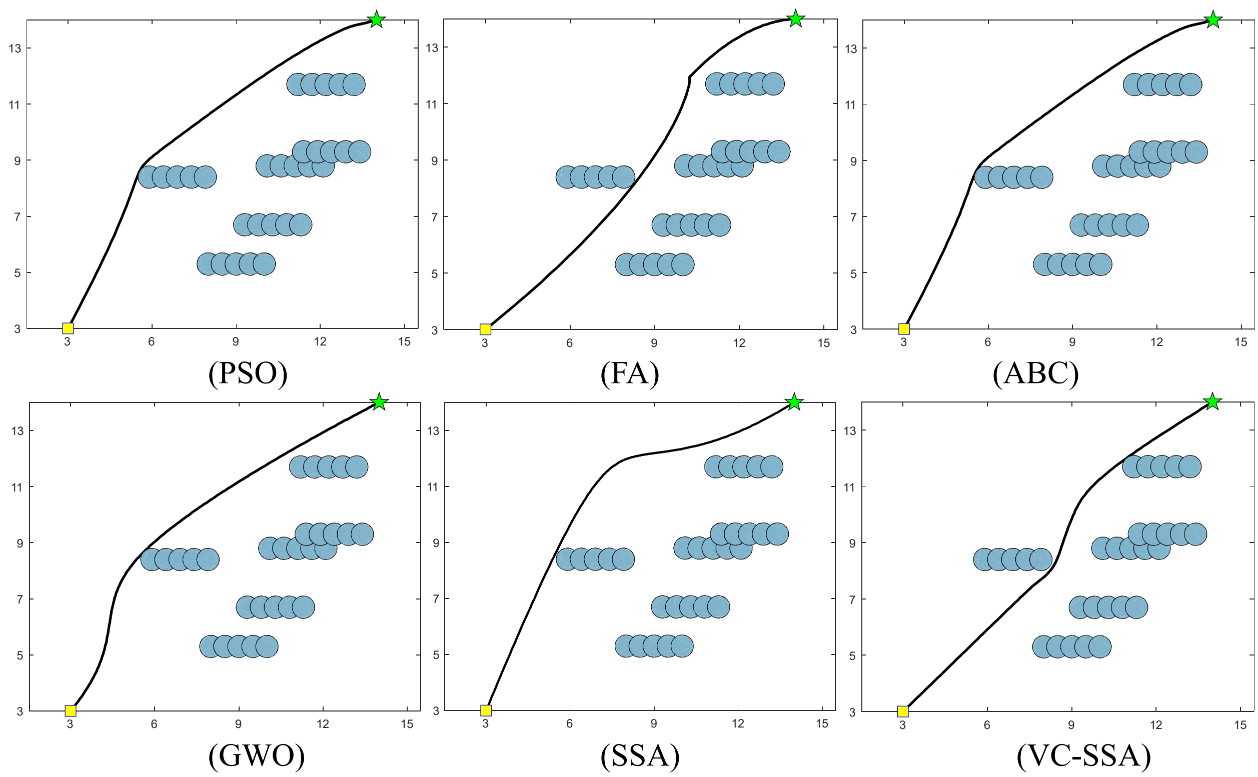

Figure 17. Map 4 (a) PSO, (b) FA, (c) ABC, (d) GWO, (e) SSA and (f) VC-SSA.

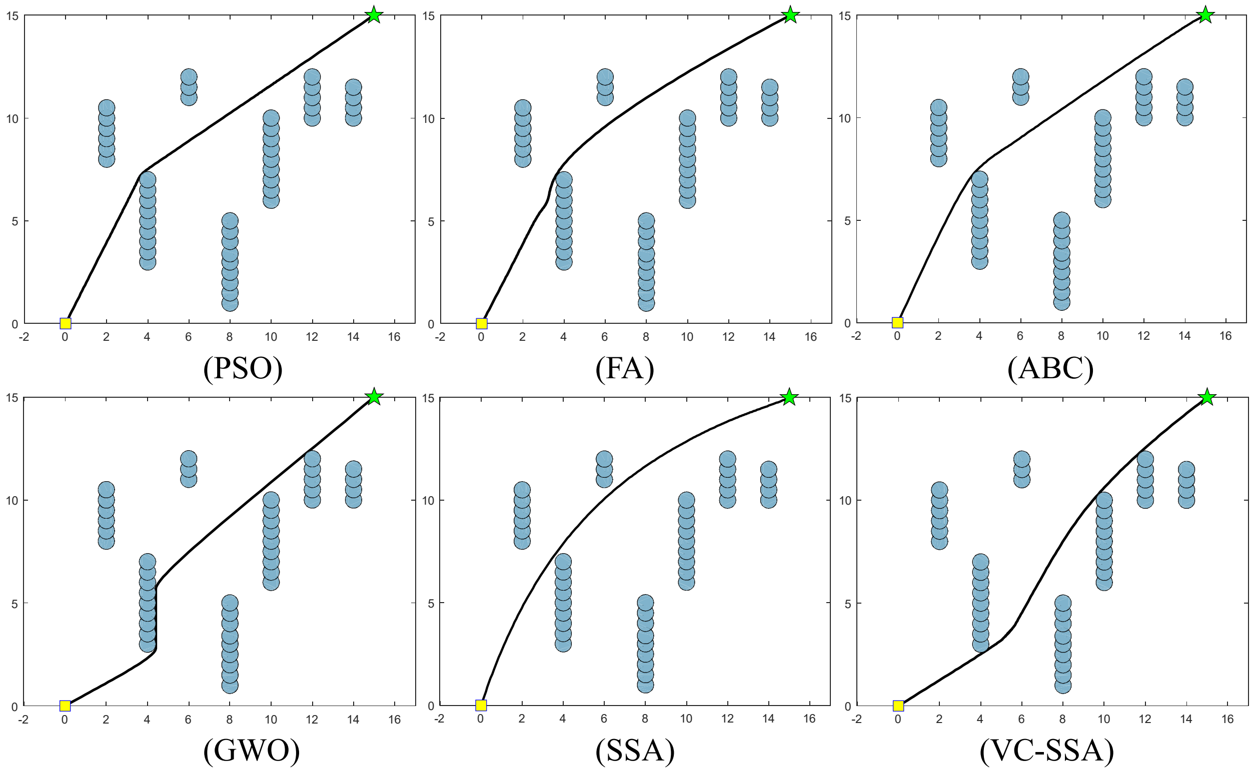

Figure 18. Map 5 (a) PSO, (b) FA, (c) ABC, (d) GWO, (e) SSA and (f) VC-SSA.

7. Conclusions and scope for future work

In this work, a novel SSA variant, called the VC-SSA approach, has been proposed with the hope of enhancing the solution quality and convergence rate of the standard salp swarm optimizer. We first introduce the velocity clamping strategy to boost the exploitation ability and the solution precious. Reduction factor mechanism is designed to expedite the convergence of the basic SSA. Finally, a novel position update equation is introduced by injecting an inertia weight to establish a counterbalance between exploration and exploitation, thus effectively expanding the landscape of global exploration during the early search, and refining the rough region of the global optima with the hope of advancing the solution accuracy after the lapse of iterations. According to the experimental results implemented in this paper, each of the introduced modifications can improve the performance of the basic SSA. More importantly, the components are novel, easy to accomplish, and do not destroy the minimalist style of the standard SSA. To evaluate the effectiveness and applicability of VC-SSA, we tested it on four classes of problems including 23 widely used benchmark functions, 30 latest benchmark problems from CEC 2017, five real-life engineering design problems, and a mobile robot path planning problem, and thoroughly compared it with the basic SSA, the well-performance SSA variants, and the frontier metaheuristic techniques. Based on the comparison results and non-parametric test results, our novel approach is able to provide superior, or at least competitive, performance on most test cases compared to its peer algorithms.

However, this study still has some limitations. For example, the value of the parameter δ was determined through a large number of experiments, and although the algorithm shows superior performance for $\delta = 0.003$, this way of taking the value needs to be improved. In future research, the optimal value of δ will be trained using reinforcement learning. In addition, this paper proposes a VC-SSA-based mobile robot path planning approach in autonomous mobile robots, and although the proposed method is able to plan a reasonable optimal path for the robot in different environmental settings, it cannot be applied to multi-robot systems. In future work, we will focus on developing this path planning approach to multi-robot systems, aiming to plan shortest collision-free paths for multiple robots acting in concert. Finally, we expect researchers to use our proposed VC-SSA algorithm to tackle optimization problems in different disciplines.

Acknowledgments

This work was supported by the National Natural Science Foundation of China [Grant Numbers 61461053, 61461054, and 61072079], Yunnan Provincial Education Department Scientific Research Fund Project [2022Y008].

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. F. Cicirelli, A. Forestiero, A Giordano, C. Mastroianni, Transparent and efficient parallelization of swarm algorithms, *ACM Trans. Auton. Adapt. Syst.*, **11** (2016), 1–26. https://doi.org/10.1145/2897373
2. A. M. Lal, S. M. Anouncia, Modernizing the multi-temporal multispectral remotely sensed image change detection for global maxima through binary particle swarm optimization, *J. King Saud Univ., Comput. Inf. Sci.*, **34** (2022), 95–103. https://doi.org/10.1016/j.jksuci.2018.10.010
3. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H. Gandomi, Marine predators algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl.*, **152** (2020), 113377. https://doi.org/10.1016/j.eswa.2020.113377
4. D. Sarkar, S. Choudhury, A. Majumder, Enhanced-Ant-AODV for optimal route selection in mobile ad-hoc network, *J. King Saud Univ., Comput. Inf. Sci.*, **33** (2021), 1186–1201. https://doi.org/10.1016/j.jksuci.2018.08.013
5. G. G. Wang, S. Deb, L. D. S. Coelho, Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems, *Int. J. Bio-Inspired Comput.*, **12** (2018), 1–22. https://doi.org/10.1504/IJBIC.2018.093328
6. D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *J. Global Optim.*, **39** (2007), 459–471. https://doi.org/10.1007/s10898-007-9149-x
7. Z. Wang, H. Ding, B. Li, L. Bao, Z. Yang, An energy efficient routing protocol based on improved artificial bee colony algorithm for wireless sensor networks, *IEEE Access*, **8** (2020), 133577–133596. https://doi.org/10.1109/ACCESS.2020.3010313
8. S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowl.-Based Syst.*, **89** (2015), 228–249. https://doi.org/10.1016/j.knosys.2015.07.006
9. X. S. Yang, Firefly algorithm, in *Engineering Optimization: An Introduction with Metaheuristic Applications*, (2010), 221–230. https://doi.org/10.1002/9780470640425.ch17
10. Z. Wang, H. Ding, B. Li, L. Bao, Z. Yang, Q. Liu, Energy efficient cluster based routing protocol for WSN using firefly algorithm and ant colony optimization, *Wireless Pers. Commun.*, 2022. https://doi.org/10.1007/s11277-022-09651-9

11. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
12. X. S. Yang, S. Deb, Cuckoo search: recent advances and applications, *Neural Comput. Appl.*, **24** (2014), 169–174. https://doi.org/10.1007/s00521-013-1367-1
13. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
14. K. P. B. Resma, M. S. Nair, Multilevel thresholding for image segmentation using Krill Herd Optimization algorithm, *J. King Saud Univ., Comput. Inf. Sci.*, **33** (2021), 528–541. https://doi.org/10.1016/j.jksuci.2018.04.007
15. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, *Future Gener. Comput. Syst.*, **97** (2019), 849–872. https://doi.org/10.1016/j.future.2019.02.028
16. G. G. Wang, S. Deb, Z. Cui, Monarch butterfly optimization, *Neural Comput. Appl.*, **31** (2019), 1995–2014. https://doi.org/10.1007/s00521-015-1923-y
17. S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: theory and application, *Adv. Eng. Software*, **105** (2017), 30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004
18. W. Li, G. G.Wang, A. H. Alavi, Learning-based elephant herding optimization algorithm for solving numerical optimization problems, *Knowl.-Based Syst.*, **195** (2020), 105675. https://doi.org/10.1016/j.knosys.2020.105675
19. S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst.*, **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055
20. Y. Yang, H. Chen, A. A. Heidari, A. H. Gandomi, Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts, *Expert Syst. Appl.*, **177** (2021), 114864. https://doi.org/10.1016/j.eswa.2021.114864
21. I. Ahmadianfar, A. A. Heidari, A. H. Gandomi, X. Chu, H. Chen, RUN beyond the metaphor: An efficient optimization algorithm based on Runge Kutta method, *Expert Syst. Appl.*, **181** (2021), 115079. https://doi.org/10.1016/j.eswa.2021.115079
22. J. Tu, H. Chen, M. Wang, A. H. Gandomi, The colony predation algorithm, *J. Bionic Eng.*, **18** (2021), 674–710. https://doi.org/10.1007/s42235-021-0050-y
23. I. Ahmadianfar, A. A. Heidari, S. Noshadian, H. Chen, A. H. Gandomi, INFO: An efficient optimization algorithm based on weighted mean of vectors, *Expert Syst. Appl.*, **195** (2022), 116516. https://doi.org/10.1016/j.eswa.2022.116516
24. S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, *Adv. Eng. Software*, **114** (2017), 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
25. Z. Wang, H. Ding, Z. Yang, B. Li, Z. Guan, L. Bao, Rank-driven salp swarm algorithm with orthogonal opposition-based learning for global optimization, *Appl. Intell.*, **52** (2022), 7922–7964. https://doi.org/10.1007/s10489-021-02776-7
26. A. A. Ewees, M. A. Al-qaness, M. Abd Elaziz, Enhanced salp swarm algorithm based on firefly algorithm for unrelated parallel machine scheduling with setup times, *Appl. Math. Modell.*, **94** (2021), 285–305. https://doi.org/10.1016/j.apm.2021.01.017
27. Q. Tu, Y. Liu, F. Han, X. Liu, Y. Xie, Range-free localization using reliable anchor pair selection and quantum-behaved salp swarm algorithm for anisotropic wireless sensor networks, *Ad Hoc Networks*, **113** (2021), 102406. https://doi.org/10.1016/j.adhoc.2020.102406

28. M. Tubishat, N. Idris, L. Shuib, M. A. Abushariah, S. Mirjalili, Improved salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection, *Expert Syst. Appl.*, **145** (2020), 113122. https://doi.org/10.1016/j.eswa.2019.113122
29. B. Nautiyal, R. Prakash, V. Vimal, G. Liang, H. Chen, Improved salp swarm algorithm with mutation schemes for solving global optimization and engineering problems, *Eng. Comput.*, 2021. https://doi.org/10.1007/s00366-020-01252-z
30. M. M. Saafan, E. M. El-Gendy, IWOSSA: An improved whale optimization salp swarm algorithm for solving optimization problems, *Expert Syst. Appl.*, **176** (2021), 114901. https://doi.org/10.1016/j.eswa.2021.114901
31. D. Bairathi, D. Gopalani, An improved salp swarm algorithm for complex multi-modal problems, *Soft Comput.*, **25** (2021), 10441–10465. https://doi.org/10.1007/s00500-021-05757-7
32. E. Çelik, N. Öztürk, Y. Arya, Advancement of the search process of salp swarm algorithm for global optimization problems, *Expert Syst. Appl.*, **182** (2021), 115292. https://doi.org/10.1016/j.eswa.2021.115292
33. Q. Zhang, Z. Wang, A. A. Heidari, W. Gui, Q. Shao, H. Chen, et al., Gaussian barebone salp swarm algorithm with stochastic fractal search for medical image segmentation: a COVID-19 case study, *Comput. Biol. Med.*, **139** (2021), 104941. https://doi.org/10.1016/j.compbiomed.2021.104941
34. H. Zhang, T. Liu, X. Ye, A. A. Heidari, G. Liang, H. Chen, et al., Differential evolution-assisted salp swarm algorithm with chaotic structure for real-world problems, *Eng. Comput.*, 2022. https://doi.org/10.1007/s00366-021-01545-x
35. S. Zhao, P. Wang, X. Zhao, H. Tuiabieh, M. Mafarja, H. Chen, Elite dominance scheme ingrained adaptive salp swarm algorithm: a comprehensive study, *Eng. Comput.*, 2021. https://doi.org/10.1007/s00366-021-01464-x
36. J. Song, C. Chen, A. A. Heidari, J. Liu, H. Yu, H. Chen, Performance optimization of annealing salp swarm algorithm: frameworks and applications for engineering design, *J. Comput. Des. Eng.*, **9** (2022), 633–669. https://doi.org/10.1093/jcde/qwac021
37. S. Zhao, P. Wang, A. A. Heidari, H. Chen, W. He, S. Xu, Performance optimization of salp swarm algorithm for multi-threshold image segmentation: Comprehensive study of breast cancer microscopy, *Comput. Biol. Med.*, **139** (2021), 105015. https://doi.org/10.1016/j.compbiomed.2021.105015
38. D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, *IEEE Trans. Evol. Comput.*, **1** (1997), 67–82. https://doi.org/10.1109/4235.585893
39. J. J. Jena, S. C. Satapathy, A new adaptive tuned Social Group Optimization (SGO) algorithm with sigmoid-adaptive inertia weight for solving engineering design problems, *Multimed Tools Appl.*, 2021. https://doi.org/10.1007/s11042-021-11266-4
40. A. Naik, S. C. Satapathy, A. S. Ashour, N. Dey, Social group optimization for global optimization of multimodal functions and data clustering problems, *Neural Comput. Appl.*, **30** (2018), 271–287. https://doi.org/10.1007/s00521-016-2686-9
41. P. Sun, H. Liu, Y. Zhang, L. Tu, Q. Meng, An intensify atom search optimization for engineering design problems, *Appl. Math. Modell.*, **89** (2021), 837–859. https://doi.org/10.1016/j.apm.2020.07.052
42. W. Zhao, L. Wang, Z. Zhang, Atom search optimization and its application to solve a hydrogeologic parameter estimation problem, *Knowl.-Based Syst.*, **163** (2019), 283–304. https://doi.org/10.1016/j.knosys.2018.08.030

43. L. Ma, C. Wang, N. Xie, M. Shi, Y. Ye, L. Wang, Moth-flame optimization algorithm based on diversity and mutation strategy, *Appl. Intell.*, **51** (2021), 5836–5872. https://doi.org/10.1007/s10489-020-02081-9
44. Y. Li, Y. Zhao, J. Liu, Dynamic sine cosine algorithm for large-scale global optimization problems, *Expert Syst. Appl.*, **177** (2021), 114950. https://doi.org/10.1016/j.eswa.2021.114950
45. S. Mirjalili, SCA: A sine cosine algorithm for solving optimization problems, *Knowl.-Based Syst.*, **96** (2016), 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
46. W. Long, J. Jiao, X. Liang, T. Wu, M. Xu, S. Cai, Pinhole-imaging-based learning butterfly optimization algorithm for global optimization and feature selection, Appl. Soft Comput., **103** (2021), 107146. https://doi.org/10.1016/j.asoc.2021.107146
47. R. Salgotra, U. Singh, S. Singh, G. Singh, N. Mittal, Self-adaptive salp swarm algorithm for engineering optimization problems, *Appl. Math. Modell.*, **89** (2021), 188–207. https://doi.org/10.1016/j.apm.2020.08.014
48. H. Ren, J. Li, H. Chen, C. Li, Stability of salp swarm algorithm with random replacement and double adaptive weighting, *Appl. Math. Modell.*, **95** (2021), 503–523. https://doi.org/10.1016/j.apm.2021.02.002
49. G. I. Sayed, G. Khoriba, M. H. Haggag, A novel chaotic salp swarm algorithm for global optimization and feature selection, *Appl. Intell.*, **48** (2018), 3462–3481. https://doi.org/10.1007/s10489-018-1158-6
50. M. H. Qais, H. M. Hasanien, S. Alghuwainem, Enhanced salp swarm algorithm: application to variable speed wind generators, *Eng. Appl. Artif. Intell.*, **80** (2019), 82–96. https://doi.org/10.1016/j.engappai.2019.01.011
51. M. Braik, A. Sheta, H. Turabieh, H. Alhiary, A novel lifetime scheme for enhancing the convergence performance of salp swarm algorithm, *Soft Comput.*, **25** (2021), 181–206. https://doi.org/10.1007/s00500-020-05130-0
52. A. G. Hussien, An enhanced opposition-based salp swarm algorithm for global optimization and engineering problems, *J. Ambient Intell. Humaniz. Comput.*, **13** (2022), 129–150. https://doi.org/10.1007/s12652-021-02892-9
53. F. A. Ozbay, B. Alatas, Adaptive salp swarm optimization algorithms with inertia weights for novel fake news detection model in online social media, *Multimedia Tools Appl.*, **80** (2021), 34333–34357. https://doi.org/10.1007/s11042-021-11006-8
54. S. Kaur, L. K. Awasthi, A. L. Sangal, G. Dhiman, Tunicate swarm algorithm: a new bio-inspired based metaheuristic paradigm for global optimization, *Eng. Appl. Artif. Intell.*, **90** (2020), 103541. https://doi.org/10.1016/j.engappai.2020.103541
55. S. Dhargupta, M. Ghosh, S. Mirjalili, R. Sarkar, Selective opposition based grey wolf optimization, *Expert Syst. Appl.*, **151** (2020), 113389. https://doi.org/10.1016/j.eswa.2020.113389
56. A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, *Knowl.-Based Syst.*, **191** (2020), 105190. https://doi.org/10.1016/j.knosys.2019.105190
57. L. Abualigah, A. Diabat, S. Mirjalili, M. A. Elaziz, A. H. Gandomi, The arithmetic optimization algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/j.cma.2020.113609
58. F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems, *Appl. Intell.*, **51** (2021), 1531–1551. https://doi.org/10.1007/s10489-020-01893-z

59. M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, *Expert Syst. Appl.*, **166** (2021), 113917. https://doi.org/10.1016/j.eswa.2020.113917
60. W. Shan, Z. Qiao, A. A. Heidari, H. Chen, H. Turabieh, Y. Teng, Double adaptive weights for stabilization of moth flame optimizer: balance analysis, engineering cases, and medical diagnosis, *Knowl.-Based Syst.*, **214** (2021), 106728. https://doi.org/10.1016/j.knosys.2020.106728
61. X. Yu, W. Xu, C. Li, Opposition-based learning grey wolf optimizer for global optimization, *Knowl.-Based Syst.*, **226** (2021), 107139. https://doi.org/10.1016/j.knosys.2021.107139
62. B. S. Yildiz, N. Pholdee, S. Bureerat, A. R. Yildiz, S. M. Sait, Enhanced grasshopper optimization algorithm using elite opposition-based learning for solving real-world engineering problems, *Eng. Comput.*, 2021. https://doi.org/10.1007/s00366-021-01368-w
63. I. Ahmadianfar, O. Bozorg-Haddad, X. Chu, Gradient-based optimizer: A new metaheuristic optimization algorithm, *Inf. Sci.*, **540** (2020), 131–159. https://doi.org/10.1016/j.ins.2020.06.037
64. P. R. Singh, M. A. Elaziz, S. Xiong, Modified spider monkey optimization based on Nelder-Mead method for global optimization, *Expert Syst. Appl.*, **110** (2018), 264–289. https://doi.org/10.1016/j.eswa.2018.05.040
65. L. Gu, R. J. Yang, C. H. Tho, M. Makowskit, O. Faruquet, Y. Li, Optimisation and robustness for crashworthiness of side impact, *Int. J. Veh. Des.*, **26** (2004), 348–360. https://doi.org/10.1504/IJVD.2001.005210
66. C. A. Coello, Use of a self-adaptive penalty approach for engineering optimization problems, *Comput. Ind.*, **41** (2000), 113–127. https://doi.org/10.1016/S0166-3615(99)00046-9
67. H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm–A novel metaheuristic optimization method for solving constrained engineering optimization problems, *Comput. Struct.*, **110–111** (2012), 151–166. https://doi.org/10.1016/j.compstruc.2012.07.010
68. A. H. Gandomi, X. S. Yang, A. H. Alavi, Mixed variable structural optimization using firefly algorithm, *Comput. Struct.*, **89** (2011), 2325–2336. https://doi.org/10.1016/j.compstruc.2011.08.002
69. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: A gravitational search algorithm, *Inf. Sci.*, **179** (2009), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
70. X. S. Yang, A new metaheuristic bat-inspired algorithm, in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, (2010), 65–74. https://doi.org/10.1007/978-3-642-12538-6_6
71. M. Azizi, Atomic orbital search: A novel metaheuristic algorithm, *Appl. Math. Modell.*, **93** (2021), 657–683. https://doi.org/10.1016/j.apm.2020.12.021
72. S. Mirjalili, S. M. Mirjalili, A. Hatamlou, Multi-verse optimizer: A nature-inspired algorithm for global optimization, *Neural Comput. Appl.*, **27** (2016), 495–513. https://doi.org/10.1007/s00521-015-1870-7
73. M. Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, *Comput Struct.*, **139** (2014), 98–112. https://doi.org/10.1016/j.compstruc.2014.03.007
74. S. Mirjalili, The ant lion optimizer, *Adv. Eng. Software*, **83** (2015), 80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010
75. H. Chickermane, H. Gea, Structural optimization using a new local approximation method, *Int. J. Numer. Methods Eng.*, **39** (1996), 829–846. https://doi.org/10.1002/(SICI)1097-0207(19960315)39:5<829::AID-NME884>3.0.CO;2-U
76. Z. Wang, Q. Luo, Y. Zhou, Hybrid metaheuristic algorithm using butterfly and flower pollination base on mutualism mechanism for global optimization problems, *Eng. Comput.*, **37** (2021), 3665–3698. https://doi.org/10.1007/s00366-020-01025-8

77. Y. J. Zheng, Water wave optimization: A new nature-inspired metaheuristic, *Comput. Oper. Res.*, **55** (2015), 1–11. https://doi.org/10.1016/j.cor.2014.10.008
78. P. Savsani, V. Savsani, Passing vehicle search (PVS): A novel metaheuristic algorithm, *Appl. Math. Modell.*, **40** (2016), 3951–3978. https://doi.org/10.1016/j.apm.2015.10.040
79. F. Gul, I. Mir, L. Abualigah, P. Sumari, A. Forestiero, A consolidated review of path planning and optimization techniques: technical perspectives and future directions, *Electronics*, **10** (2021), 2250. https://doi.org/10.3390/electronics10182250
80. Z. Wang, H. Ding, J. Yang, J. Wang, B. Li, Z. Yang, P. Hou, Advanced orthogonal opposition-based learning-driven dynamic salp swarm algorithm: framework and case studies, *IET Control Theory Appl.*, 2022. https://doi.org/10.1049/cth2.12277
81. D. Agarwal, P. S. Bharti, Implementing modified swarm intelligence algorithm based on slime moulds for path planning and obstacle avoidance problem in mobile robots, *Appl. Soft Comput.*, **107** (2021), 107372. https://doi.org/10.1016/j.asoc.2021.107372