Journal of Computational Design and Engineering, 2024, **11**, 73–98
DOI: 10.1093/jcde/qwae001
Advance access publication date: 3 January 2024
Research Article

A new approach for solving global optimization and engineering problems based on modified sea horse optimizer

Fatma A. Hashim[1], Reham R. Mostafa[2,3], Ruba Abu Khurma[4], Raneem Qaddoura[5] and Pedro A. Castillo[6,*]

[1]Faculty of Engineering, Helwan University, Cairo 4262027, Egypt
[2]Research Institute of Sciences and Engineering (RISE), University of Sharjah, Sharjah 27272, United Arab Emirates
[3]Department of Information System, Faculty of Computers and Information Sciences, Mansoura University, Mansoura 35516, Egypt
[4]MEU Research Unit, Faculty of Information Technology, Middle East University, Amman 11831, Jordan
[5]School of Computing and Informatics, Al Hussein Technical University, Amman 11831, Jordan
[6]Department of Computer Engineering, Automatics and Robotics, University of Granada, Granada 18071, Spain
*Correspondence: pacv@ugr.es

Abstract

Sea horse optimizer (SHO) is a noteworthy metaheuristic algorithm that emulates various intelligent behaviors exhibited by sea horses, encompassing feeding patterns, male reproductive strategies, and intricate movement patterns. To mimic the nuanced locomotion of sea horses, SHO integrates the logarithmic helical equation and Levy flight, effectively incorporating both random movements with substantial step sizes and refined local exploitation. Additionally, the utilization of Brownian motion facilitates a more comprehensive exploration of the search space. This study introduces a robust and high-performance variant of the SHO algorithm named modified sea horse optimizer (mSHO). The enhancement primarily focuses on bolstering SHO's exploitation capabilities by replacing its original method with an innovative local search strategy encompassing three distinct steps: a neighborhood-based local search, a global non-neighbor-based search, and a method involving circumnavigation of the existing search region. These techniques improve mSHO algorithm's search capabilities, allowing it to navigate the search space and converge toward optimal solutions efficiently. To evaluate the efficacy of the mSHO algorithm, comprehensive assessments are conducted across both the CEC2020 benchmark functions and nine distinct engineering problems. A meticulous comparison is drawn against nine metaheuristic algorithms to validate the achieved outcomes. Statistical tests, including Wilcoxon's rank-sum and Friedman's tests, are aptly applied to discern noteworthy differences among the compared algorithms. Empirical findings consistently underscore the exceptional performance of mSHO across diverse benchmark functions, reinforcing its prowess in solving complex optimization problems. Furthermore, the robustness of mSHO endures even as the dimensions of optimization challenges expand, signifying its unwavering efficacy in navigating complex search spaces. The comprehensive results distinctly establish the supremacy and efficiency of the mSHO method as an exemplary tool for tackling an array of optimization quandaries. The results show that the proposed mSHO algorithm has a total rank of 1 for CEC2020 test functions. In contrast, the mSHO achieved the best value for the engineering problems, recording a value of 0.012 665, 2993.634, 0.01 266, 1.724 967, 263.8915, 0.032 255, 58 507.14, 1.339 956, and 0.23 524 for the pressure vessel design, speed reducer design, tension/compression spring, welded beam design, three-bar truss engineering design, industrial refrigeration system, multi-product batch plant, cantilever beam problem, and multiple disc clutch brake problems, respectively. Source codes of mSHO are publicly available at https://www.mathworks.com/matlabcentral/fileexchange/135882-improved-sea-horse-algorithm.

Keywords: engineering problem, global optimization, metaheuristics, sea horse optimizer

1. Introduction

Optimization is the process of reaching the minimum or maximum value of some real function under a limited range of values. Using mathematical notations, the optimization function can be expressed as $f : D \to \mathbb{R}$ from some set D to the real numbers $\mathbb{R}$. In a minimization optimization problem, a member $x_0 \in D$, $f(x_0) \leq f(x) \forall x \in A$ whereas in a maximization optimization problem, $f(x_0) \geq f(x) \forall x \in A$ (Pierre, 1986; Smith, 1978). Traditional gradient-based optimization methods rely on finding the derivative of functions, but they have limitations, particularly when dealing with complex optimization problems that lack derivatives or involve many local minima in the search space surface (Sun *et al.*, 2019).

The scientific community has progressively adopted computational intelligence algorithms, such as metaheuristics, to optimize both discrete and continuous problems (Khurma *et al.*, 2020c). Metaheuristic algorithms offer advantages over traditional mathematical algorithms owing to their gradient-free nature, which makes them well-suited for tackling undifferentiated problems and yielding promising near-optimal solutions. Although these solutions may not be optimal, they provide valuable approximations (Hussien *et al.*, 2022). Furthermore, metaheuristics demonstrate polynomial time complexity, rendering them more efficient than conventional methods with exponential time complexity (Osaba *et al.*, 2021).

Received: April 22, 2023. **Revised:** November 13, 2023. **Accepted:** November 13, 2023

Metaheuristic algorithms have gained prominence in optimizing various problems, primarily due to their derivative-free nature, satisfactory performance metrics, simplicity, efficiency, and robustness (Morales-Castañeda *et al.*, 2020). In the realm of combinatorial optimization, there has been a proliferation of "novel" metaheuristic techniques, many of which draw inspiration from artificial or natural processes. Metaheuristics can be categorized into four main groups, each based on distinct concepts and sources of inspiration: evolutionary algorithms (EAs), physics-based algorithms (PhAs), swarm-based algorithms (SAs), and human-based algorithms (HAs). These categories encompass a wide range of optimization approaches, each with its unique principles and techniques:

(i) EAs, like genetic algorithms (GA, Katoch *et al.*, 2021), draw inspiration from biological evolution and natural selection. These algorithms emulate genetic variation, selection, and reproduction processes. GA, for instance, employs a population of potential solutions that evolve over generations using selection, crossover, and mutation operations. This iterative process gradually improves the population's fitness, guiding the optimization procedure by exploring the search space.

(ii) PhAs are inspired by fundamental principles and natural phenomena, using simulations of physical processes to optimize solutions. Simulated annealing (SimAnn) is a well-known example, mimicking the annealing process in metallurgy. SimAnn begins with high "temperature" to encourage exploration and then gradually reduces it to guide optimization toward better solutions. Other PhAs include the weIghted meaN oF vectOrs (Ahmadianfar *et al.*, 2022), rime optimization algorithm (Su *et al.*, 2023), Runge-Kutta method (RUN, Ahmadianfar *et al.*, 2021), and Fick's law algorithm (FLA, Hashim *et al.*, 2023). These algorithms draw insights from physics to improve optimization strategies.

(iii) SAs, inspired by the collective behaviors of natural swarms like bird flocks and ant colonies, prioritize communication, cooperation, and decentralized decision-making among swarm individuals. A notable example is particle swarm optimization (PSO, Kennedy & Eberhart, 1995), which simulates particle movement and information sharing within a swarm to guide the search process. PSO maintains a balance between exploration and exploitation by adjusting particle velocities based on individual and global best positions, harnessing the collective intelligence of the swarm to explore and exploit the search space for optimal solutions effectively. Other SAs, such as snake optimizers (Hashim & Hussien, 2022), spotted hyena optimizer (Dhiman & Kumar, 2017), slime mould algorithm (Li *et al.*, 2020), and colony predation algorithm (Tu *et al.*, 2021), similarly draw inspiration from natural swarming behaviors to enhance optimization techniques.

(iv) HAs belong to the category of metaheuristic algorithms that draw inspiration from human intelligence and problem-solving methods. These algorithms simulate or replicate human decision-making processes and learning mechanisms. A well-known example is teaching–learning-based optimization (TLBO, Rao *et al.*, 2011), which models the interaction between a teacher and students to facilitate knowledge transfer and solution improvement. By harnessing human-inspired approaches, these algorithms aim to enhance optimization and discover effective solutions for complex problems. Other HAs include the mother optimization algorithm (Matoušová *et al.*, 2023) and human mental search (Mousavirad & Ebrahimpour-Komleh (2017), each seeking to improve optimization using principles inspired by human behavior and cognition.

SAs are mathematical methodologies inspired by the collaborative behaviors observed in natural animal groups. These algorithms translate the survival and foraging behaviors of swarm members into mathematical equations. In response to the no-free lunch (NFL) theorem, researchers have developed and refined numerous SAs by drawing inspiration from different aspects of nature or introducing new variants to address their limitations. These variants involve proposing novel operators or techniques that are integrated with the original algorithm. Some enhancement approaches for SAs include incorporating chaotic maps (Khurma *et al.*, 2020a), introducing local search (Ahmed *et al.*, 2023), applying opposition-based learning (OBL, Khurma *et al.*, 2022; Mostafa *et al.*, 2023), utilizing EA selection operators (Khurma *et al.*, 2021), enhancing EA crossover and mutation operators (Alweshah *et al.*, 2022; Awadallah *et al.*, 2022), leveraging Levy flight behavior (Ewees *et al.*, 2022; Mostafa *et al.*, 2022), using Gaussian operators (Zhang *et al.*, 2020), and applying rank-based methods (Khurma *et al.*, 2020b). These efforts contribute to the ongoing evolution and advancement of SAs for optimization.

Various original and enhanced SAs have found applications in diverse fields. For instance, the mCOOT (modified coot optimization algorithm, COOT) algorithm was improved and employed in estimating unmeasured battery parameters, resulting in enhanced accuracy and reduced error rates (Houssein *et al.*, 2022a). Similarly, the electrostatically charged particles algorithm was improved and tested on IEEE CEC2017 test functions, demonstrating its effectiveness in estimating parameters for photovoltaic models (Kamel *et al.*, 2022). In the context of electrical distribution networks, a modified robust optimization method was proposed to optimize the distribution of distributed generators (DGs), leading to reduced energy losses (Tolba *et al.*, 2022). In another study, an improved version of the artificial ecosystem optimization algorithm, i.e., "artificial ecosystem optimization with opposition-based learning" was developed to determine the optimal distribution of DGs in radial distribution networks (Khasanov *et al.*, 2023). This approach considers the stochastic nature of renewable energy sources, like wind turbines and photovoltaic power generation, using appropriate probability models. The loss sensitivity index is utilized to identify suitable buses for integrating DG modules into the network. Additionally, an improved algorithm called "Lévy flight distribution with opposition-based learning" was proposed to address the limitations of the original Lévy flight distribution algorithm. This improved version was applied to optimize the parameters of a three-diode photovoltaic model and demonstrated superior performance (Houssein *et al.*, 2022b). These studies highlight the effectiveness and versatility of enhanced SAs in addressing complex optimization problems across different domains.

Metaheuristics have emerged as powerful tools in medical applications, offering innovative solutions in diverse areas. In the context of feature selection, metaheuristic approaches have been harnessed to efficiently identify relevant features from complex medical datasets, aiding in disease diagnosis, prognosis, and treatment planning. These algorithms navigate through high-dimensional data spaces to extract essential information, enhancing the accuracy of predictive models and reducing computational overhead. For example, Piri and Mohapatra (2021) introduced a novel approach called "multi-objective quadratic binary Harris hawks optimization", which utilizes the *K*-nearest neighbor method as a wrapper classifier. This technique aims to extract optimal feature subsets from medical data for enhanced performance. Thawkar *et al.* (2021) introduced a hybrid feature selection approach by combining the butterfly optimization algorithm (BOA) and the ant lion optimizer to create the hybrid BOAALO method. This method effectively selects an optimal subset of features, which is then employed to predict the benign or malignant status of breast tissue.

Additionally, metaheuristics find utility in multi-level threshold segmentation of medical images, facilitating the precise delineation of anatomical structures or pathological regions. By optimizing threshold values, these algorithms enable accurate segmentation, which is vital for quantitative analysis, disease quantification, and treatment evaluation. The versatility of metaheuristics in handling intricate and often noisy medical data underscores their potential to drive advancements in medical imaging, diagnosis, and patient care. For example, Chakraborty *et al.*, (2021a) focuses on developing a computational tool to quickly and accurately assess illness severity using COVID-19 chest X-ray images. It introduces a modified whale optimization algorithm (WOA), named modified whale optimization algorithm with population reduction (mWOAPR), that enhances diagnostic precision by integrating random population initialization during global search and optimizing parameter settings for improved exploration–exploitation balance. Xing *et al.*, (2023) introduced an enhanced WOA, termed quasi-opposition-based WOA (QGBWOA), tailored for COVID-19 applications. QGBWOA integrates quasi-opposition-based learning for improved solution search and a Gaussian barebone mechanism to enhance solution space diversity. This refinement holds promise for precise feature selection and multi-threshold image segmentation in COVID-19-related tasks.

In late 2022, a team of researchers introduced the sea horse optimizer (SHO), drawing inspiration from sea horses' locomotion, predation, and reproductive behaviors (Zhao *et al.*, 2022b). Sea horses exhibit distinctive locomotion, such as jumping and wrapping their tails around algae or leaves, often influenced by marine eddies, leading to spiral movement. They can also exhibit Brownian motion by turning upside down. Moreover, sea horses employ their uniquely shaped heads to stealthily approach and capture prey, achieving a remarkable success rate of up to 90%. Additionally, the random mating of male and female sea horses contributes to a new generation inheriting advantageous traits from their parents.

The observed sea horse behaviors have endowed the SHO algorithm with the capacity to effectively manage the exploration and exploitation phases when seeking optimal solutions. This ability enables SHO to strike a balance between thorough exploration of the solution space and efficient exploitation of promising areas. Furthermore, the incorporation of these behaviors promotes increased diversity among solutions within the SHO community. Consequently, SHO exhibits enhanced performance by mitigating premature convergence and avoiding getting trapped in local minima. SHO's advantageous attributes have been demonstrated through successful applications in diverse domains. Notably, it has proven effective in tasks such as fine-tuning power system stability and optimizing parameters (Aribowo, 2023). Additionally, SHO has been applied to reduce exhaust pollutants from diesel engines, showcasing its versatility and practical utility (Alahmer *et al.*, 2023).

The unique characteristics and accomplishments of the SHO algorithm have motivated us to extend its application to address a diverse array of engineering challenges. Nevertheless, a notable drawback of SHO lies in its exploitation strategy, which depends on selecting a neighboring individual at random during local searches within the search space. Recognizing this limitation, we have undertaken the task of enhancing the local search procedure of SHO. To rectify this deficiency, we propose novel methods aimed at optimizing the local search process within the algorithm. These methods are designed to enhance SHO's performance by effectively circumventing the risk of becoming trapped in local minima and facilitating the convergence toward optimal solutions. Our study's primary contributions encompass the following key points:

- (i) Proposing a robust, high-performance variant of SHO, named the modified sea horse optimizer (mSHO) method, enhances the SHO exploitation strategy. This is done by replacing the original method with a new local search strategy, which is done in three steps:
  - (a) Neighborhood-based local search strategy,
  - (b) A global non-neighbor-based search strategy, and
  - (c) Walk around the existing search strategy.
- (ii) The mSHO method is compared with the original SHO and eight different optimizers in ten CEC2020 test positions.
- (iii) mSHO is used to solve nine real-world engineering problems, namely: welded beam design problem, three-bar truss design problem, tension/compression spring design, speed reducer design, industrial refrigeration system, pressure vessel design, cantilever beam design, multi-disc clutch brake, and multi-product batch plant.
- (iv) Results of mSHO outperformed other algorithms in both constrained and unconstrained problems.

The rest of the paper is structured as follows. Section 2 presents some recent literature in which researchers proposed improvements to SAs for solving complicated engineering problems. Section 3 describes the SHO algorithm's inspiration and mathematical methodology in detail. Section 4 discusses the proposed mSHO in detail. Section 5 provides the results and in-depth discussion of mSHO and other competitive algorithms on CEC2020 test functions. The mSHO's performance on various engineering problems is presented in Section 6. Section 7 summarizes the results and discusses the limitations of the work. Section 8 concludes the paper and offers some potential research directions that can help improve SHO performance as well.

2. Related Works

This section provides an overview of recent studies that have proposed methods to enhance the performance of specific metaheuristic algorithms for solving global and engineering problems over the past 3 year. Table 1 provides a summary of the studies mentioned, considering four key criteria: the publication year, the metaheuristic algorithm employed, the enhancement approach taken, whether the IEEE CEC suite was used, the number of benchmark test functions examined, and the quantity and nature of the engineering problems used for evaluation.

Hongwei *et al.*, (2019) utilized chaos theory to introduce an enhanced variant of moth flame optimization (MFO) called CMFO. Chaotic functions were employed for initializing individuals, managing overrides, and adjusting the distance parameter. CMFO underwent testing on three standard function groups and two real-world engineering problems. The statistical findings demonstrated that incorporating an appropriate chaotic map (Singer's map) into the relevant component of MFO significantly improved its performance. Nevertheless, the study did not explore the application of other chaotic maps to the MFO algorithm. Sheikhi

Table 1: Summary of related literature on improved metaheuristics for global and engineering optimization problems.

Ref.	Year	Metaheuristics	Improved MA	Improvement	IEEE CEC suite	#Benchmark functions	#Engineering problems	Name of engineering problems
(Hongwei *et al.*, 2019)	2019	MFO	CMFO	Chaotic maps	CEC2005	18	2	Welded beam design, tensional/compressional spring design
(Sheikhi Azqandi *et al.*, 2020)	2020	TEO	ETEO	Population clustering, memory-based	Benchmark functions	54	5	Three-bar truss design, pressure vessel design, speed reducer design, tension/compression spring design, welded beam design
(Chen *et al.*, 2020)	2020	WOA	OBCWOA	Chaos, quasi-opposition	Benchmark functions	20	1	Tension/compression string design
(Fan *et al.*, 2020)	2020	WOA	ESSAWOA	SSA, LOBL	Benchmark functions	23	3	Tension/compression spring design, pressure vessel design, welded beam design
(Nadimi-Shahraki *et al.*, 2021)	2021	GWO	IGWO	DLH	CEC2018	30	4	Pressure vessel design, welded beam design, optimal power flow (IEEE 118-bus, IEEE 30-bus)
(Wang *et al.*, 2021)	2021	BOA	MBFPA	FPA, mutation	Benchmark functions	49	5	Three-bar truss design, speed reducer, multi-plate disc clutch brake design, pressure vessel design, welded beam design
(Chakraborty *et al.*, 2021b)	2021	WOA	WOAmM	SOS	CEC2019	36	6	Three-bar truss design problem, gas transmission compressor design, pressure vessel design, cantilever beam design, gear train design
(Zhang *et al.*, 2021)	2021	JAYA	EJAYA	Local exploitation, global exploration	CEC2014, CEC2015	45	7	Welded beam design, tension/compression spring design, pressure vessel design, speed reducer design, rolling element bearing design, hydrostatic thrust bearing, car side impact design
(Yıldız *et al.*, 2022)	2022	RUN	CRUN	Chaotic maps			6	Gear train design, coupling with a bolted rim, pressure vessel design, Belleville spring, vehicle brake-pedal
(Sharma *et al.*, 2022)	2022	BOA	mLBOA	Self-adaptive parameter setting, Lagrange interpolation, local search, Levy flight	CEC2017	15	3	Spread spectrum radar polyphase design, three-bar truss design, gas transmission compressor design
(Saha, 2022)	2022	SCA	MAMSCA	Dividing the population, modified mutualism	CEC2019	50	5	Gear train design, gas transmission compressor design, car side impact design, cantilever beam design, three-bar truss design
(Chakraborty *et al.*, 2023)	2023	WOA	m-SDWOA	SOS, DE	CEC2019	42	4	Gear train design, gas transmission compressor design, welded beam design, weight minimization of a speed reduce

Azqandi *et al.* introduced an enhanced variant of temporal evolutionary optimization (TEO) called ETEO in their study (Sheikhi Azqandi *et al.*, 2020). This enhancement strategy incorporated a temporal evolution factor and population clustering. A memory was utilized to store some of the best designs. ETEO underwent evaluation across a range of constrained and unconstrained problems, as well as engineering design problems. ETEO aimed to improve TEO's performance, address its weaknesses, and enhance search capabilities during both exploration and exploitation phases. By employing population clustering, enhancing the environmental factor, and incorporating a memory to preserve some of the best design variables, ETEO demonstrated competitiveness with other metaheuristic algorithms in terms of statistical outcomes, particularly concerning the best objective function and the number of function evaluations performed during optimization.

Chen *et al.*, (2020) introduced chaos mechanism based on quasi-opposition WOA (OBCWOA), an enhanced variant of the WOA, which incorporated chaos and quasi-opposition strategies for global optimization problems. OBCWOA demonstrated robustness in solving global optimization problems, excelling in convergence accuracy, speed, high-dimensional search capability, and stability. It was also effective in addressing real engineering design problems. However, OBCWOA had some drawbacks, including the need for adjusting more parameters than the original WOA, longer running times compared with some metaheuristic algorithms, and limited improvement in the correctness of certain test functions. Nonetheless, OBCWOA remained a valuable tool for complex practical problems and remained competitive among state-of-the-art algorithms. In Fan *et al.,* (2020), Enhanced Whale Optimization Algorithm integrated with Salp Swarm Algorithm (ESSAWOA) was developed by integrating WOA with Salp Swarm Algorithm (SSA) and a lens OBL (Lens Opposition-based Learning, LOBL) strategy for global optimization. The exploitative power of the SSA leader's strategy was used to update personnel attitudes before WOA operations were implemented. Subsequently, the non-linear parameter of SSA in the prey encircling and attacking phases was incorporated into WOA to enhance the convergence behavior. The LOBL strategy was adopted to increase population diversity. ESSAWOA was evaluated using 23 standard functions and three classical engineering design problems. The findings show that ESSAWOA can swiftly and efficiently find a promising solution to these optimization issues. ESSAWOA performs much better than the fundamental WOA, SSA, and other metaheuristic algorithms.

Nadimi-Shahraki *et al.,* (2021), proposed an improved gray wolf (IGWO) optimizer for engineering problems. IGWO adopted a new movement strategy called learning-based hunting (DLH) inspired by wolves' natural hunting behavior. DLH implemented a different method of neighborhood identification for each individual so that neighborhood information could be shared between individuals. The performance of the IGWO algorithm was evaluated on a set of CEC2018 standards and four engineering problems. IGWO is compared across all tests to six more cutting-edge metaheuristics. Friedman and mean absolute error (MAE) statistical tests are also used to assess the results. In comparison with the algorithms employed in the studies, the IGWO algorithm is very competitive and frequently superior, as shown by the experimental findings and statistical testing. The suggested algorithm's performance and applicability on engineering design challenges are shown by the findings.

Wang *et al.,* (2021) proposed a new variant of the BOA called butterfly optimization algorithm and flower pollination base (MBFPA) by hybridizing it with the flower pollination and symbiosis mechanism of global optimization problems. Flower pollination and symbiotic organisms support exploration and exploitation capacity, respectively. Moreover, the possibility of alternating exploration and adaptive exploitation improves the balance between these two phases. The MFPPA is tested on 49 standardized test functions and five classic engineering problems. The findings demonstrate the viability of the suggested method and demonstrate its competitiveness and high application prospects. Chakraborty *et al.,* (2021b) proposed enhanced WOA (WOAmM) using the mutualism phase from symbiotic organisms search (SOS). The proposed WOAmM method was tested on 36 benchmark functions and IEEE CEC2019 function suite. In addition, six real-world engineering optimization problems were solved by the proposed method. In comparison with other competing approaches, the results show that the suggested SSC algorithm (security service chain) is resilient, effective, efficient, and convergence analysis.

Zhang *et al.,* (2021) improved the global search phase of Jaya algorithm (JAYA) and implemented the enhanced JAYA (EJAYA) for global optimization. EJAYA had many distinguished features such as local exploitation, which defined upper and lower local attractors. Furthermore, the global exploration was guided by historical population, and it did not make any adjustments for initial parameters. The EJAYA was verified by testing it on 45 test functions from IEEE CEC2014 and IEEE CEC2015 test suites. Furthermore, EJAYA was implemented to solve seven real-world engineering design optimization problems. The effectiveness of the newly proposed improved techniques to JAYA and the strong ability of EJAYA to escape from the local optimum for tackling difficult optimization issues are supported by experimental results. In Yıldız *et al.,* (2022), Yildiz proposed a chaotic RUN (CRUN). In this study, 10 different chaotic maps were integrated into the RUN algorithm to boost its performance, and it was tested on some design engineering problems. The results showed that CRUN was the best compared with the most recent algorithms in the literature. The proposed CRUN method can also uncover advantageous features in a variety of managerial implications, including supply chain management, business models, fuzzy circuits, and management models.

Sharma *et al.*, (2022) proposed a new variant of BOA, namely modified butterfly optimization algorithm (mLBOA). He integrated the self-adaptive parameter setting, Lagrange interpolation formula, a new local search strategy, and levy flight operators with BOA. The IEEE CEC2017 benchmark suite and three real-world engineering design problems were used to evaluate the mLBOA. The outcomes were contrasted with six cutting-edge algorithms and five BOA variations. Additionally, a number of statistical tests have been carried out to support the rank, significance, and complexity of the proposed mLBOA, including the Friedman rank test, Wilcoxon's rank test, convergence analysis, and complexity analysis. The mLBOA has also been used to resolve three actual engineering design issues. According to all of the analyses, the suggested mLBOA algorithm is competitive with other well-known state-of-the-art algorithms and BOA variants.

Saha (2022) introduced an enhanced version of the sine cosine algorithm (SCA) called multi-population-based adaptive sine cosine algorithm (MAMSCA). The enhancement involved dividing the SCA's population into two halves and applying either a sine or cosine method to update each half. Furthermore, a modified mutualism phase was incorporated into the algorithm. MAMSCA was applied to standard benchmark functions, IEEE CEC2019 functions, and five engineering design problems. The results demonstrated significant improvements in addressing real-world problems. A comprehensive assessment of the algorithm, including a statistical analysis, evaluation of time complexity, and solution generation speed, underscored its enhanced performance and suitability for practical applications. Chakraborty *et al.*, (2023) introduced a novel variant of WOA called m-SDWOA, which integrates WOA with the modified mutualism phase of SOS, the mutation strategy of differential evolution (DE), and the commensalism phase of SOS. The algorithm incorporates a new parameter, denoted as Y, to determine whether to apply the global or local phases. The efficiency of the algorithm was assessed using 42 benchmark functions, an IEEE CEC2019 test suite, and four engineering design problems. These evaluations consistently demonstrated the superior performance of the proposed algorithm compared with the methods it was benchmarked against.

In the aforementioned studies, researchers introduced various operators and techniques to address limitations commonly associated with metaheuristic algorithms, including early convergence, bias toward local minima, and imbalances in exploration and exploitation. These innovations have the potential to enhance the optimizer's performance, stability, and robustness, leading to more dependable and effective results. Building upon the foundations of the NFL theory, this paper extends this research trajectory by harnessing the recently developed SHO algorithm. Additionally, the paper integrates specific local search strategies into SHO to further enhance its effectiveness in tackling global optimization and engineering problems.

3. Background

SHO draws inspiration from the predation, movement, and breeding behaviors of sea horses, which enable them to adapt to their environment and survive. Sea horses, small fish found in warm waters, have a head resembling that of a horse. In terms of movement, sea horses exhibit a spiral motion when wrapping their tails around a stem (or leaf) of algae. Their unique head shape aids in stealthy predation. Furthermore, sea horses engage in random mating between females and males to produce offspring in their breeding behavior. The algorithm encompasses four phases: initialization, movement behavior, predation behavior, and breeding behavior.

In the initialization phase, the algorithm generates the initial population of sea horses, as represented by equation (1), where *Dim* represents the dimension and *pop* is the population size (Zhao *et al.*, 2022b):

$$Sea\ horses = \begin{bmatrix} x_1^1 & \dots & x_1^{Dim} \\ \dots & \dots & \dots \\ x_{pop}^1 & \dots & x_{pop}^{Dim} \end{bmatrix}. \tag{1}$$

Each individual is represented by equation (2), with each value in the list calculated using equation (3), where *rand* is a random value in the range of [0, 1]. Here, x_i^j represents the *j*th dimension of the *i*th individual, and LB^j and UB^j denote the lower and upper bounds of the *j*th dimension:

$$X_i = [x_i^1, x_i^2 ... x_i^{Dim}] \tag{2}$$

$$x_i^j = rand \times (UB^j - LB^j) + LB^j. \tag{3}$$

The individual with the lowest fitness function value is referred to as the elite individual X_{elite}, which is calculated using equation (4) (Zhao *et al.*, 2022b):

$$X_{elite} = argmin(fitness(X_i)). \tag{4}$$

In the movement behavior phase, sea horses exhibit two types of movements: the spiral motion and the Brownian motion. When engaged in spiral motion, the new position of a sea horse is determined using equation (5), where the values of x, y, and z are computed as shown in equations (6), (7), and (8). Here, $\rho = u \times e^{\theta v}$ represents the length of the stems defined by the logarithmic spiral constants u and v, which are set to 0.05. θ is a random value within the range $[0, 2\pi]$. The Levy distribution function, $Levy(\lambda)$, is calculated using equation (9), where λ is a random number in the range [0, 2], and *s* is fixed at 0.01. The variables w and k are random numbers in the range [0, 1], and σ is determined by equation (10) (Zhao *et al.*, 2022b):

$$X_{new}^1(t+1) = X_i(t) + Levy(\lambda)((X_{elite}(t) - X_i(t)) \times x \times y \times z + X_{elite}(t)) \tag{5}$$

$$x = \rho \times cos(\theta) \tag{6}$$

$$y = \rho \times sin(\theta) \tag{7}$$

$$z = \rho \times \theta \tag{8}$$

$$Levy(\lambda) = s \times \frac{w \times \sigma}{|k|^{\frac{1}{\lambda}}} \tag{9}$$

$$\sigma = \left(\frac{\Gamma(1+\lambda) \times sin(\frac{\pi\lambda}{2})}{\Gamma(\frac{1+\lambda}{2}) \times \lambda \times 2^{\frac{\lambda-1}{2}}} \right). \tag{10}$$

Conversely, in the case of Brownian motion, the new position of a sea horse is determined using equation (11), where l is a constant coefficient. The value of β_t is calculated according to equation (12) (Zhao *et al.*, 2022b):

$$X_{new}^1(t+1) = X_i(t) + rand * l * \beta_t * (X_i(t) - \beta_t * X_{elite}) \tag{11}$$

$$\beta_t = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{x^2}{2}\right). \tag{12}$$

To summarize the calculations, equation (13) encompasses the calculations of the new positions, with r_1 denoting a random number (Zhao *et al.*, 2022b):

$$X_{new}^1(t+1) = \begin{cases} X_i(t) + Levy(\lambda)((X_{elite}(t) - X_i(t)) \times x \times y \times z + X_{elite}(t)) & r_1 > 0 \\ X_i(t) + rand * l * \beta_t * (X_i(t) - \beta_t * X_{elite}) & r_1 \le 0. \end{cases} \tag{13}$$

The predation behavior is calculated using equation (14), where α is determined as shown in equation (15), and r_2 represents a random number within the range [0, 1] (Zhao *et al.*, 2022b):

$$X_{new}^2(t+1) = \begin{cases} \alpha * (X_{elite} - rand * X_{new}^1(t)) + (1-\alpha) * X_{elite} & r_2 > 0.1 \\ (1-\alpha) * (X_{new}^1(t) - rand * X_{elite}) + \alpha * X_{new}^1(t) & r_2 \le 0.1 \end{cases} \tag{14}$$

$$\alpha = \left(1 - \frac{t}{T}\right)^{\frac{2t}{T}}. \tag{15}$$

The breeding behavior is determined by assigning roles to mother and father sea horses, as depicted in equations (16) and (17), where X_{sort}^2 signifies all X_{new}^2 sorted in ascending order of their fitness values (Zhao *et al.*, 2022b). The actual mating process to produce new offspring is described in equation (18), where r_3 is a random number within the range [0, 1], *i* is a positive integer within the range [1, *pop*/2], and X_i^{father} and X_i^{mother} represent randomly selected father and mother individuals (Zhao *et al.*, 2022b):

$$fathers = X_{sort}^2(1 : pop/2) \tag{16}$$

$$mothers = X^{2}_{sort}(pop/2 + 1 : pop) \tag{17}$$

$$X^{offspring}_{i} = r_3 X^{father}_{i} + (1 - r_3) X^{mother}_{i}. \tag{18}$$

4. Proposed Method

The original SHO algorithm exhibits certain shortcomings, particularly in achieving a harmonious balance between global and local search behaviors during the movement phase. This issue arises from the random selection of the search strategy, whether it is spiral or Brownian motion, based solely on a random number r_1. Furthermore, the fixed values assigned to parameters u and v, which dictate the length of the stems, remain constants throughout the optimization process, potentially impeding the algorithm's ability to guide solutions effectively to new positions. To address these limitations, this paper introduces an improved version of SHO, named mSHO, aimed at enhancing the algorithm's performance and addressing its main limitations.

In this section, we delve into the proposed mSHO method, which brings about significant changes in the movement behavior phase. Instead of the traditional approach, the mSHO method incorporates the following three distinct steps:

(i) Neighborhood-based local search strategy,
(ii) Non-neighborhood-based global search strategy, and
(iii) Wandering around-based search strategy.

Neighborhood-based local search strategy leverages an individual's conscious neighborhood to enhance the quality of exploitation within that neighborhood. Specifically, a random neighbor, denoted as c_{local}, is chosen from within the individual's local neighborhood, and another neighbor, termed c_{global}, is selected from outside the local neighborhood but possessing the lowest fitness function value. Subsequently, if the fitness value of c_{local} is found to be lower than that of c_{global}, the individual adjusts its position toward that of c_{local}, as calculated by equation (19) (Zamani *et al.*, 2019):

$$X_i(t+1) = X_i(t) + r_i \times fl_i(t) \times (m_{local}(t) - X_i(t)) \tag{19}$$

where $fl_i(t)$ is the flight length of the individual in iteration t, r_i is a random number in the range of [0, 1], and $m_{local}(t)$ is the hiding position of c_{local} for iteration t.

In contrast, the non-neighborhood-based global search strategy is activated when the fitness value of c_{local} exceeds or equals the fitness value of c_{global}. In this situation, the individual moves toward the position of c_{global}, represented as $X_{ij}(t+1)$, and this relocation is determined using equation (20) (Zamani *et al.*, 2019):

$$X_{ij}(t+1) = r_i \times fl_i(t) \times (m_{global\,j}(t) - X_{ij}(t)) \tag{20}$$

where j is the dimension value, $m_{globalj}(t)$ is the hiding position of c_{global} for iteration t and dimension j.

The neighborhood-based local search strategy and the non-neighborhood-based global search strategy both include a validation step to ensure that the new position falls within the problem space's defined range. If it does not, the strategy randomly adjusts the dimensions that have exceeded this range, bringing them back into the problem space's boundaries.

On the other hand, the wandering around-based search strategy is employed when the previous two strategies fail to improve an individual's fitness value. It operates by analyzing the surrounding environment and maneuvering the individual to a potentially more favorable position with a lower fitness value. Equation (21) calculates this new position, where $m_{gbestj}(t)$ represents the best hiding position in the entire population for dimension j, and $X_{rj}(t)$ corresponds to a randomly selected individual in the jth dimension (Zamani *et al.*, 2019):

$$X_{ij}(t+1) = m_{gbest\,j}(t) + r_i \times fl_i(t) \times (X_{rj}(t) - X_{ij}(t)). \tag{21}$$

Figure 1 illustrates the step-by-step process of the proposed mSHO algorithm. The algorithm commences by generating the initial population of sea horses, following the principles outlined in equations (1), (2), and (3). In each iteration, the algorithm proceeds to evaluate the fitness of each individual and updates the elite individual using equation (4). Subsequently, for each individual, two neighboring sea horses, denoted as c_{local} and c_{global}, are selected. Their fitness values are then compared, and the individual adopts the position of the sea horse with the lower fitness value. This position update is determined by equations (19) and (20). Following this, another fitness comparison is conducted, this time between the individual's fitness at the new position and its fitness at the previous position. If the fitness at the new position is not lower than the previous one, the individual's position is modified using equation (21). The predation and breeding behaviors are calculated according to equations (14). This flowchart provides a comprehensive overview of the mSHO algorithm's operation.

5. Assessment of mSHO on CEC2020 Test Functions

To prove the efficiency of mSHO, several tests and experiments have been conducted. This study covers two major tests: global optimization problems using 10 functions from CEC2020 and nine engineering problems. All experiments were run using MATLAB 2022b on an Intel® Core™ i7 (3.40 GHz) CPU with RAM 16GB running Microsoft Windows 11.

Several metaheuristics were evaluated and compared with the proposed mSHO in this experiment to ensure a fair assessment. The selected metaheuristics include dandelion optimizer (DO, Zhao *et al.*, 2022a), covariance matrix adaptation evolution strategy (CMA-ES, Hansen & Ostermeier, 2001), hunger games search (HGS, Yang *et al.*, 2021), smell agent optimization (SAO, Salawudeen *et al.*, 2021), Harris hawks optimization (HHO, Heidari *et al.*, 2019), PSO (Kennedy & Eberhart, 1995), and stochastic paint optimizer (SPO, Kaveh *et al.*, 2020). All algorithms were evaluated under the same conditions with 30 search agents and a maximum of 1000 iterations. To eliminate the impact of random initialization, 30 independent runs were performed, and the algorithms' performance was evaluated using the average fitness and standard deviation metrics. The parameters of the other algorithms are mentioned in Table 2.

5.1. Experimental series 1: CEC2020

In this section, we analyze the outcomes of our experiments on the CEC2020 functions, categorizing our findings into four distinct segments: statistical analysis, boxplot representation, convergence assessment, and the Wilcoxon's rank test. The CEC2020 dataset encompasses 10 distinct functions, as outlined in Table 3. These functions are classified into four categories: uni-modal, multi-modal shifted and rotated functions, hybrid, and composition functions. Each function, denoted as *Fi*, is associated with an optimal value that serves as our objective. For instance, *F1*'s optimal value is set at 100, with the optimizer striving to identify a solution that closely approximates this value.

5.1.1. *Statistical analysis on CEC2020 test suite*

The fitness function values for the different CEC2020 functions with different competitive algorithms are displayed in Table 4. Each function's mean, standard deviation, and rank are calculated

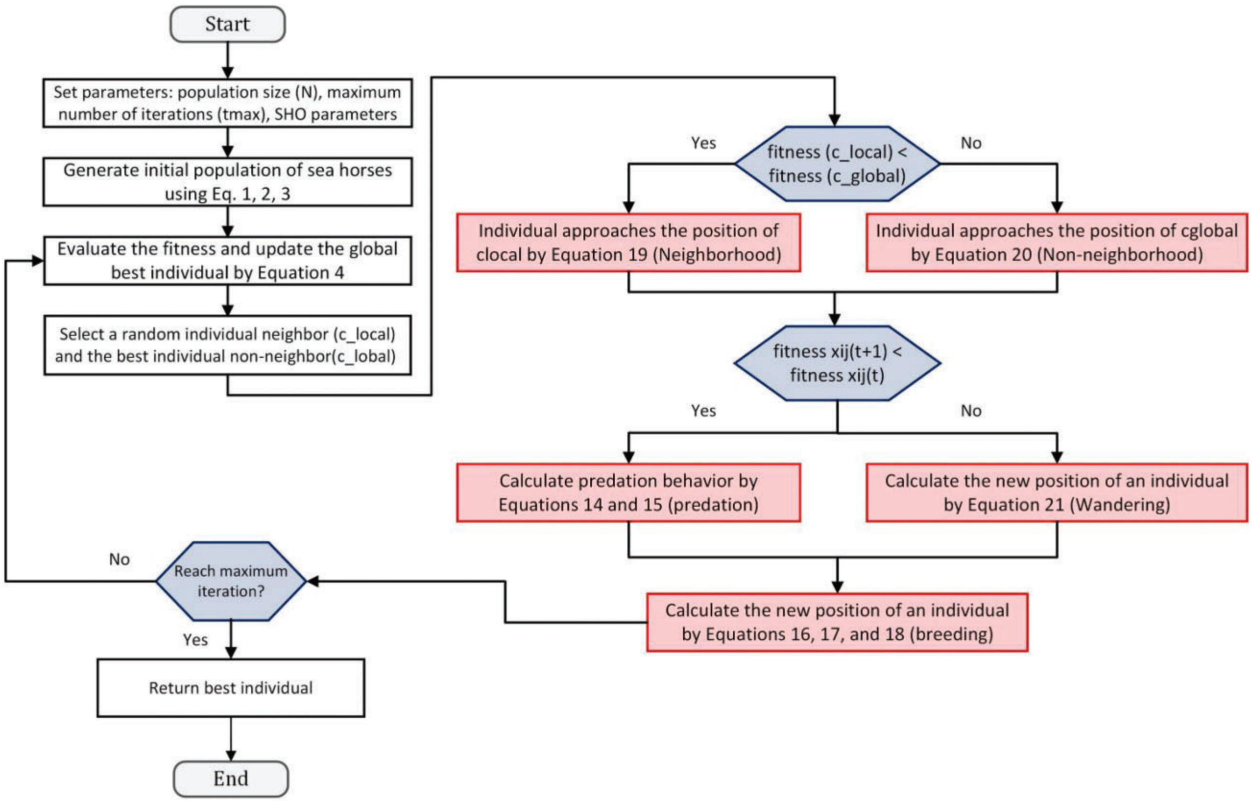


Figure 1: Flowchart of the proposed mSHO.

Table 2: Parameter settings.

	Parameter	Value
Population size (N)	CEC2020 problem	30
	Engineering problem	30
Maximum iterations	CEC2020	1000
	Engineering problem	1000
Problem dimensions (D)	CEC2020 problem	10
	Engineering problem	Dimension of problem
PSO	Cognitive component (c1)	2
	Social component (c2)	2
	Inertia weight	0.2–0.9
DO	Adaptive parameters (α, k)	[0,1], [0,1]
HGS	k	0.3
	$r_1, r_2, r_3, r_4, r_5, r_6$	$rand[0, 1]$
SAO	olf	0.75
	SL	0.9
HHO	β	1.5
AOA	μ	0.499
	α	5

Table 3: CEC2020 test suite description.

No.	Function specification	Fi*
Uni-modal function		
F1	Shifted and Rotated Bent Cigar Function	100
	Multi-modal shifted and rotated functions	
F2	Shifted and Rotated Schwefel's Function	1100
F3	Shifted and Rotated Lunacek bi-Rastrigin Function	700
F4	Expanded Rosenbrock's plus Griewangk's Function	1900
	Hybrid functions	
F5	N = 3	1700
F6	N = 4	1600
F7	N = 5	2100
	Composition functions	
F8	N = 3	2200
F9	N = 4	2400
F10	N = 5	2500

across the different optimization algorithms. The mean value is useful because it provides a comparative estimate of the total of many runs. Standard deviation, on the other hand, provides variability of the different runs. The rank is given based on the mean value, where the rank of 1 is provided to the lowest value.

It is observed from the table that the proposed mSHO has very competitive fitness values with the rank of 1 for all of the functions except the *F*6 and *F*10. The HGS and CMA-ES algorithms have better values for *F*6 and *F*10, respectively. In addition, the HGS algorithm has an overall competitive rank of 2 for many functions, while the proposed mSHO algorithm has a total rank of 1. On the other hand, the Friedman test shows that the proposed mSHO displays the lowest value of 1.3, followed by HGS and PSO, whereas SAO has the worst value.

5.1.2. *Boxplot behavior analysis*

The boxplots of the 30 runs over the different algorithms are presented in Fig. 2. The boxplot shows the maximum, minimum, median, and interquartile range (Qaddoura *et al.*, 2021). The pro-

Table 4: The mean and STD of fitness values for 30 runs obtained by the competitor algorithms on the CEC2020 test suite with $Dim = 10$.

Function	Measures	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
F1	Mean	**2167.666**	6.09E+09	2960.56	2906.272	2933.422	4.09E+09	6813.764	2982.666	2958.932	3001.223
	Std	2818.008	3.05E+09	39.30 731	**0.010 848**	32.80 232	2.5E+09	1204.266	25.08 729	31.72 322	64.70 297
	Rank	1	10	5	2	3	6	9	7	4	8
F2	Mean	**1339.177**	2817.837	2217.5	5457.625	1806.443	3325.647	5650.254	2898.974	2677.729	3653.926
	Std	**149.4172**	443.9491	325.2414	279.64	281.3332	406.6865	518.6831	463.5156	464.5185	1202.01
	Rank	1	6	3	9	2	4	10	7	5	8
F3	Mean	**731.9907**	824.6887	798.2153	810.0422	753.2998	887.7204	1095.175	899.9842	758.5414	782.7039
	Std	**5.659 652**	28.792	26.43 054	7.270 733	13.70 135	30.08 877	33.57 846	30.11 809	11.17 006	35.64 884
	Rank	1	7	5	6	2	8	10	9	3	4
F4	Mean	**1901.451**	5076.296	1908.204	1906.697	1903.654	2234.568	1111 983	1923.178	1902.609	3297.193
	Std	**0.373 742**	1953.856	4.256 352	1.816 287	1.436 275	772.5102	930 956	5.257 264	0.885 894	4944.59
	Rank	1	9	6	5	3	4	10	7	2	8
F5	Mean	**54 240.62**	848 010.9	172 393.5	1292 352	397 062.9	253 090.3	5219 985	581 330.9	99 603.21	1696 510
	Std	**29 805.33**	656 376.1	136 582.4	822 183.4	307 375.9	152 575.9	5142 683	447 915.4	60 581.07	5204 490
	Rank	1	7	4	8	5	2	10	6	3	9
F6	Mean	1601.524	1814.781	3080.036	2079.857	**753.2998**	2222.706	3517.05	2161.23	1931.721	1985.644
	Std	**0.416 483**	177.8612	3665.986	220.6828	13.70 135	201.3467	335.3659	211.6085	131.3297	256.4341
	Rank	2	3	9	7	1	4	10	8	5	6
F7	Mean	**21 683.19**	233 325.4	139 959.4	551 833.7	215 977	109 998.4	4322 030	242 057.3	89 234.49	118 786
	Std	**10 175.81**	205 427.5	143 246.4	260 253.9	259 901.8	42 789.46	4895 366	174 732.6	79 457.84	151 227.1
	Rank	1	7	5	9	6	3	10	8	2	4
F8	Mean	**2300.408**	3364.95	3408.826	6943.684	2834.071	4651.941	6598.57	4060.821	3197.561	3478.461
	Std	**0.588 443**	642.2368	1395.141	290.1084	946.9044	2117.116	976.2632	1592.76	1327.532	1708.349
	Rank	1	4	5	10	2	8	9	7	3	6
F9	Mean	**2836.961**	3012.995	2938.716	2859.627	2909.554	3006.273	3501.35	3109.96	2864.636	3050.984
	Std	**11.55 151**	40.23 653	36.40 562	40.10 379	51.90 887	42.96 402	139.0932	92.04 638	25.1554	135.1523
	Rank	1	7	6	2	4	5	10	9	3	8
F10	Mean	2948.073	3122.153	2960.56	**2906.272**	2933.422	3081.367	6813.764	2982.666	2958.932	3001.223
	Std	37.95 805	83.68 379	39.30 731	**0.010 848**	32.80 232	76.68 404	1204.266	25.08 729	31.72 322	64.70 297
	Rank	3	9	5	1	2	6	10	7	4	8
Friedman's mean rank		1.3	6.8	5	5.7	3	6.8	9.7	7	3.1	6.6
Rank		1	7	4	5	2	7	9	8	3	6

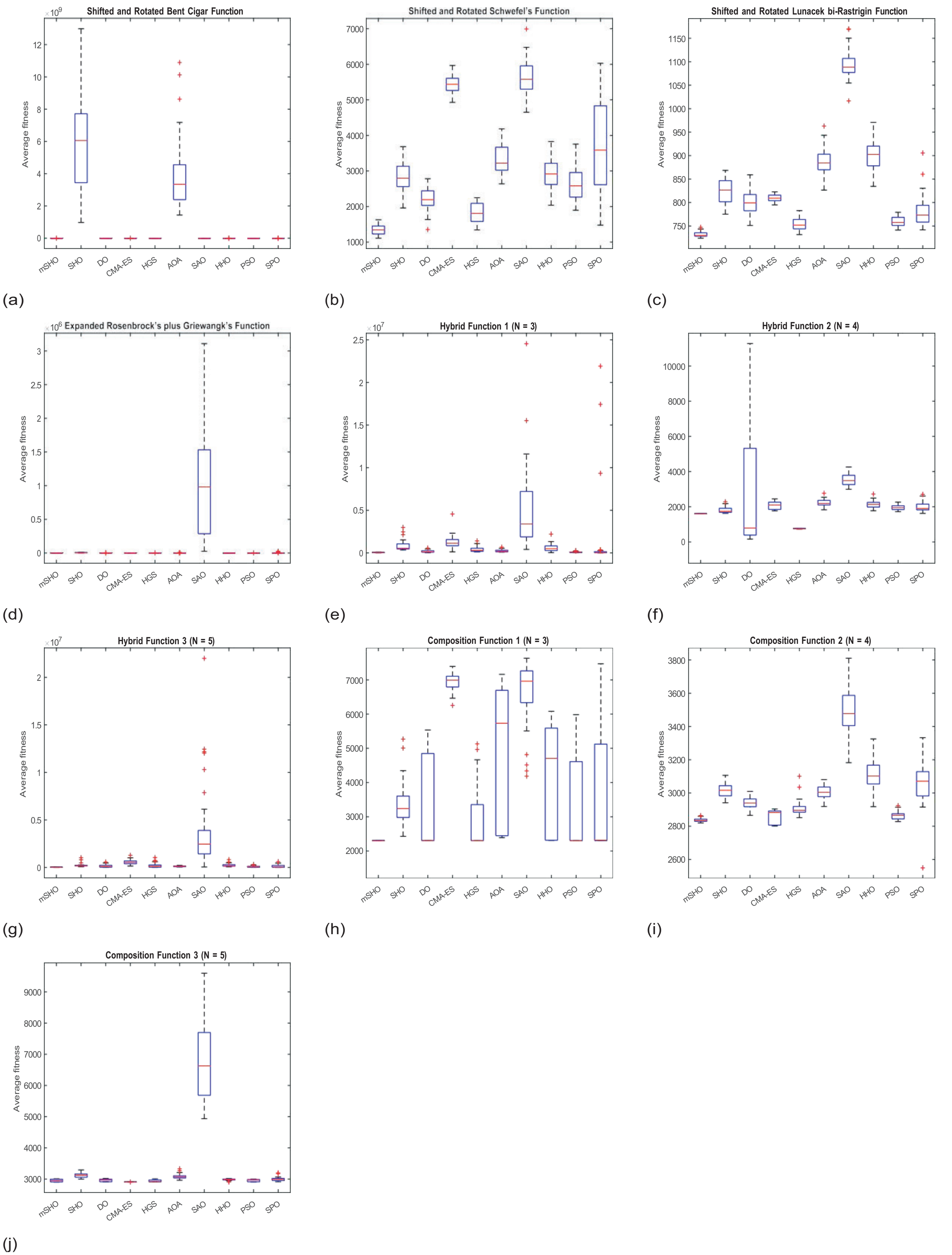


Figure 2: The boxplot curves of the proposed mSHO and the other approaches obtained over CEC2020 test suite with $Dim = 10$.

posed mSHO shows compacted box distribution compared with the other algorithms, indicating a stable algorithm.

The proposed mSHO also has the lowest minimum and maximum values for all the functions except for *F*6. Overall, the proposed mSHO confirms the consistency of the algorithm. Some other observations can be concluded from the figure. SHO has a high standard deviation compared with the other algorithms for *F*1, while SAO has a high standard deviation for *F*10 as well as DO for *F*6 and SOA for *F*5. *F*8 shows an interesting pattern since all the algorithms show large values for the standard deviation except for the mSHO, which indicates a very stable algorithm.

5.1.3. *Convergence performance analysis*

The convergence curve represents the values of the fitness function across the different iterations. This is presented in Fig. 3 for the experimented results. The convergence curve is important since it shows that the value of the fitness function decreases when progressing through the iterations. It also shows that after some iterations, the value of the fitness function stays as is, indicating that the algorithm cannot explore better solutions and that the best solution resulting from the algorithm is the closest solution to the correct one.

The convergence curves show advanced values for most functions across all the iterations except for *F*1 and *F*6, while *F*1 obtained the lowest fitness value at the final iterations. This proves the effectiveness of the optimization task for the proposed mSHO algorithm by converging toward minimum values. On the other hand, *F*2, *F*5, *F*7, and *F*8 show that mSHO is finding better solutions than the other algorithms, but it has similar behavior to the other algorithms for *F*3, *F*6, and *F*10.

5.1.4. *Wilcoxon's rank test analysis*

The *P*-values of the Wilcoxon's rank-sum test for each competitive algorithm with the proposed mSHO are represented in Table 5. Wilcoxon's rank-sum test is a non-parametric test to find the significance of the results. It is proposed by Wilcoxon (1992) with a 5% significant level. It is observed from the table that the proposed mSHO wins in all comparisons except when compared with DO for *F*6, HGS for *F*10, PSO for *F*8/*F*10, and SPO for *F*5.

6. Performance of mSHO on Engineering Design Problems

This section evaluates the mSHO algorithm's performance in real-world engineering applications such as:

(i) Pressure vessel design problem,
(ii) Speed beam design problem,
(iii) Tension/compression spring design,
(iv) Welded beam design problem,
(v) Three-bar truss engineering design problem,
(vi) Industrial refrigeration system problem,
(vii) Multi-product batch plant problem,
(viii) Cantilever beam problem, and
(ix) Multi-disc clutch brake problem.

Those problems have been addressed using mSHO and comparing results against those of other competing algorithms. The mSHO and competing algorithms were run 30 separate times with total 1000 iterations to arrive at a fair comparison.

6.1. Pressure vessel design problem

One of the most common engineering design problems is pressure vessel design problem, with the aim of finding cost of the pressure vessel. This problem has four different types of variables: head thickness (T_h), shell thickness (T_s), length of cylindrical unit (L), and the inner radius (R). The mathematical structure of the pressure vessel design problem and the four types of constraints applied to the problem design is presented in equation (22). This engineering problem (tensile design/compressed spring) is solved using the proposed mSHO and other competitive algorithms as shown in Table 6. The obtained statistical results are presented in Table 7. Table 7 shows that the optimal value of the function is 0.012 665, which was achieved using the mSHO algorithm. Results reveal that FLA is superior to all other competing algorithms.

In addition, as shown in Fig. 4, the convergence curves and boxplot for the mSHO algorithm and other compared methods for the pressure vessel design problem are presented. The convergence curve plots the average best values against the number of iterations for the mSHO algorithm and other compared algorithms after running 1000 times. The results indicate that the mSHO algorithm converges faster than the other algorithms and can typically reach a near-optimal solution more quickly. While the other algorithms demonstrate competitive performance, the SAO and CMA-ES exhibit the lowest performance. Furthermore, the boxplot results illustrate the stability of the proposed mSHO algorithm, followed by the AOA and PSO algorithms. These findings demonstrate the effectiveness and stability of the proposed mSHO algorithm in tackling the pressure vessel design problem.

$$\text{Consider } \vec{x} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} = \begin{bmatrix} T_s & T_h & R & L \end{bmatrix},$$

$$\text{Minimize } 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3,$$

$$\begin{aligned} \text{Subject to } \; & g_1(\vec{x}) = -x_1 + 0.0193x_3 \leqslant 0, \\ & g_2(\vec{x}) = -x_2 + 0.00\,954x_3 \leqslant 0 \\ & g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296\,000 \leqslant 0, \\ & g_4(\vec{x}) = x_4 - 240 \leqslant 0 \end{aligned} \quad (22)$$

$$\begin{aligned} \text{Variables range } \; & 0 \leqslant x_1 \leqslant 99, \\ & 0 \leqslant x_2 \leqslant 99 \\ & 10 \leqslant x_3 \leqslant 200 \\ & 10 \leqslant x_4 \leqslant 200. \end{aligned}$$

6.2. Speed reducer design problem

One of the most significant engineering design problems is the speed reducer, as described in the study by Sadollah *et al.*, (2013). The primary goal of this problem is to minimize the weight of the speed reducer by optimizing seven variables while also accounting for limitations on the curvature stress of gear teeth, transverse deflections of the shafts, stresses in the shafts, and surface stress. The mathematical model for this problem is presented below:

$$\begin{aligned} \text{Minimize} f(\vec{x}) = \; & 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) \\ & - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) \end{aligned}$$

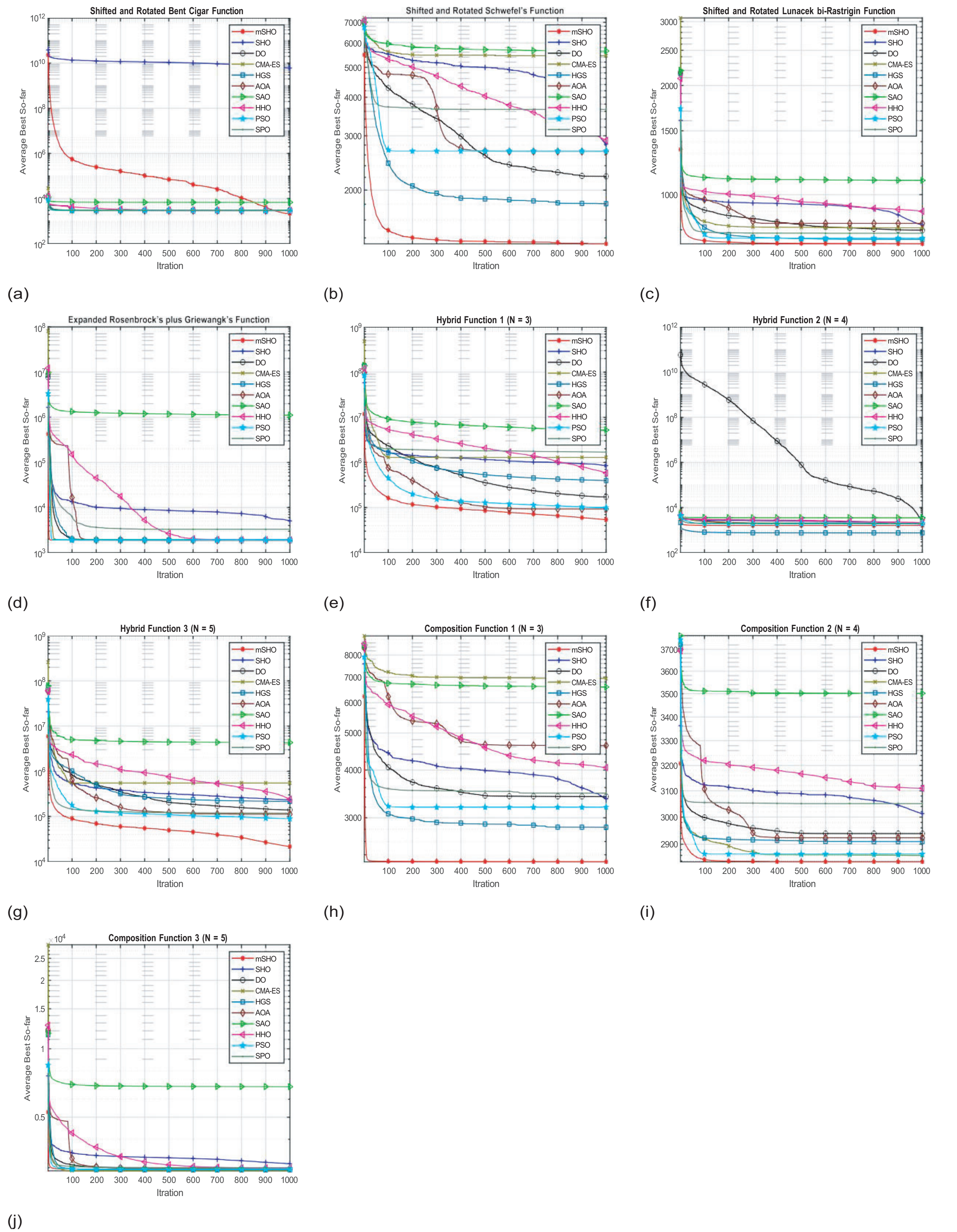


Figure 3: The convergence curves of the proposed mSHO and the competitor algorithms obtained on CEC2020 test suite with $Dim = 10$.

Table 5: mSHO versus other metaheuristics algorithms for CEC2020 ($D = 10$) in terms of *P*-values of the Wilcoxon's rank-sum test.

mSHO versus	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
F1	3.02E−11	0.007 959	0.0079 312	0.007 958 996	3.01 986E−11	5.53 286E−08	0.007 959	0.007 959	0.007 959
F2	3.02E−11	1.09E−10	3.02E−11	1.3111E−08	3.01 986E−11	3.01 986E−11	3.02E−11	3.02E−11	4.5E−11
F3	3.02E−11	3.02E−11	3.02E−11	4.57257E−09	3.01 986E−11	3.01 986E−11	3.02E−11	9.92E−11	4.5E−11
F4	3.02E−11	3.02E−11	4.077E−11	3.15 889E−10	3.01 986E−11	3.01 986E−11	3.02E−11	3.65E−08	3.02E−11
F5	3.02E−11	1.53E−05	3.02E−11	4.50 432E−11	1.32 885E−10	3.01 986E−11	3.5E−09	0.00073	0.982 307
F6	3.02E−11	0.379 036	3.02E−11	3.01 986E−11	3.01 986E−11	3.01 986E−11	3.02E−11	3.02E−11	3.02E−11
F7	3.02E−11	1.61E−06	3.02E−11	7.59 915E−07	3.01 986E−11	3.01 986E−11	3.02E−11	0.000 132	0.002 755
F8	3.02E−11	9.26E−09	3.02E−11	0.000 117 472	3.01 986E−11	3.01 986E−11	3.02E−11	0.137 241	6.07E−11
F9	3.02E−11	3.02E−11	0.0292 054	6.69 552E−11	3.01 986E−11	3.01 986E−11	3.02E−11	2.32E−06	5.57E−10
F10	3.69E−11	0.030 317	6.715E−05	0.673 495 053	1.95 678E−10	3.01 986E−11	0.000 318	0.122 353	0.000 125

$$
\begin{aligned}
\text{Subject to}\quad & g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leqslant 0\\
& g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leqslant 0\\
& g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leqslant 0\\
& g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leqslant 0\\
& g_5(\vec{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times 10^6}}{110.0x_6^3} - 1 \leqslant 0\\
& g_6(\vec{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 157.5\times 10^6}}{85.0x_6^3} - 1 \leqslant 0\\
& g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \leqslant 0\\
& g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \leqslant 0\\
& g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \leqslant 0\\
& g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leqslant 0\\
& g_{11}(\vec{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leqslant 0\\
\text{Variables range}\quad & 2.6 \leqslant x1 \leqslant 3.6\\
& 0.7 \leqslant x2 \leqslant 0.8\\
& 17 \leqslant x3 \leqslant 28\\
& 7.3 \leqslant x4 \leqslant 8.3\\
& 7.8 \leqslant x5 \leqslant 8.3\\
& 2.9 \leqslant x6 \leqslant 3.9\\
& 5.0 \leqslant x7 \leqslant 5.5.
\end{aligned} \tag{23}
$$

The speed reducer engineering problem was tackled using the proposed mSHO algorithm and other competitive algorithms, as depicted in Table 8. The obtained statistical results are presented in Table 9. As demonstrated in Table 9, the optimal value of the function is 2993.634, which was achieved using the mSHO, HGS, AOA, and PSO algorithms. The results reveal that the mSHO algorithm produces promising outcomes compared with the other algorithms and has the potential to achieve minimal total weight for the speed reducer in this problem.

Figure 5 depicts the convergence curves and boxplot for the mSHO algorithm and other compared methods for the speed reducer design problem. As shown in the figure, the mSHO algorithm converges faster than the other algorithms and can usually obtain the near-optimal solution more rapidly. Although the other algorithms also showed competitive performance, the HHO and SAO had the lowest performance. On the other hand, the boxplot results illustrate the stability of the proposed mSHO algorithm, followed by the HGS and PSO algorithms. Overall, the experiment's findings reveal the efficacy and stability of the proposed mSHO algorithm in tackling the speed reducer design problem.

6.3. Tension/compression spring problem

The tension/compression spring design optimization problem is a mechanical engineering problem that aims to minimize the weight of the spring while ensuring that certain constraints are satisfied (Bhadoria & Kamboj, 2019). The problem involves selecting the optimal values for parameters such as wire diameter (*d*), number of active coils (*N*), and mean coil diameter (*D*). Constraints are placed on the surge frequency, minimum deflection, and shear stress. The goal is to find the optimal combination of parameters

Table 6: Best solution obtained from the comparative algorithms for solving the pressure vessel design problem.

Algorithm	x1	x2	x3	x4	Cost
mSHO	0.774 555	0.383 203	40.31 962	200	5870.12 409
SHO	0.783 011	0.395 825	40.70 835	194.6584	5908.30 062
DO	0.774 533	0.383 229	40.31 962	200	5870.12 562
CMA-ES	0.859 051	0.473 962	43.62 445	180.7057	6879.70 268
HGS	0.774 549	0.383 204	40.31 962	200	5870.12 398
AOA	0.774 549	0.383 204	40.31 962	200	5870.12 398
SAO	2.527 902	5.801 292	1.559 886	5.133 647	7710.96 822
HHO	0.835 827	0.430 123	43.61 014	158.7634	6046.19 065
PSO	0.778 476	0.385 079	40.52 191	197.203	5876.94 102
SPO	0.774 574	0.383 204	40.31 962	200	5870.1246

Table 7: Results obtained from competitor algorithms for pressure vessel design problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	5870.1241	5908.3006	5870.1256	6879.703	5870.124	5870.124	7710.968	6046.191	5876.941	5870.125
Max	5870.1342	7271.3656	7156.128	8758.232	7301.196	6666.339	23 390.18	7464.369	6898.222	21 829.09
Mean	5870.1266	6196.8232	6199.5342	6942.32	6353.867	6145.893	13 266.08	6530.491	6242.955	7035.835
Std	0.0029 181	368.8253	418.47 407	342.971	563.4305	219.693	3334.069	399.0299	248.7349	2850.707
Rank	1	3	4	9	6	2	10	7	5	8

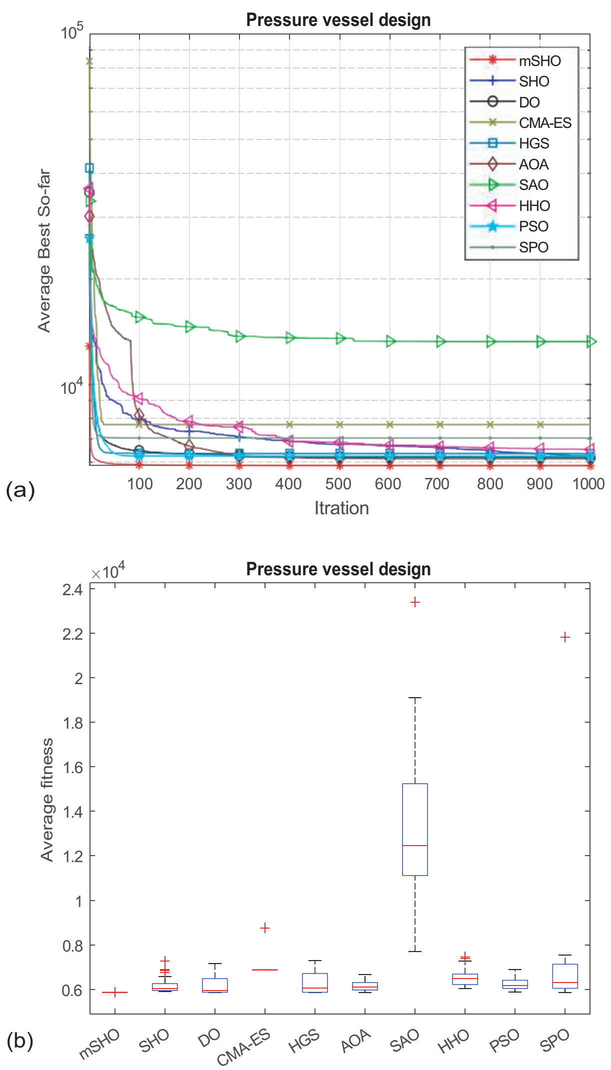


Figure 4: Convergence curve and boxplot for mSHO against other competitors – pressure vessel design problem.

that satisfies all constraints while minimizing the weight of the spring. The following equations present the mathematical model for this particular engineering design problem:

$$\text{Consider } \vec{x} = [x_1\ x_2\ x_3] = [d\ D\ N]$$

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

$$\begin{aligned} \text{Subject to } \ & g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71\,785 x_1^4} \leqslant 0 \\ & g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12\,566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} \leqslant 0 \\ & g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leqslant 0 \\ & g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leqslant 0 \end{aligned} \tag{24}$$

$$\begin{aligned} \text{Variables range } \ & 0.05 \leqslant x_1 \leqslant 2 \\ & 0.25 \leqslant x_2 \leqslant 1.30 \\ & 2.00 \leqslant x_3 \leqslant 15. \end{aligned}$$

The proposed mSHO algorithm and other competitive algorithms were employed to solve this engineering problem as presented in Table 10. The statistical results obtained from the experiments are provided in Table 11. It is evident from Table 11 that the mSHO algorithm achieved the optimal value of the function, which was 0.012 665. The results indicate that the mSHO algorithm performed significantly better than the other algorithms in achieving a minimal weight of the tension spring in this problem.

Figure 6 presents the convergence curves and boxplot for the tension/compression spring design problem solved using the mSHO algorithm and other competitive methods. The results show that the proposed mSHO algorithm outperforms the other algorithms, achieving the near-optimal solution faster. The SAO and HGS algorithms exhibit the lowest performance, while the AOA and SHO algorithms perform competitively. The boxplot analysis further confirms the stability of the mSHO algorithm, followed by AOA and SHO. These results demonstrate the efficiency and stability of the mSHO algorithm in solving the tension/compression spring design problem.

6.4. Welded beam design problem

The welded beam design problem is another important engineering design problem that has been considered in previous research

Table 8: Best solution obtained from the comparative algorithms for solving speed reducer design problem.

Algorithm	x1	x2	x3	x4	x5	x6	x7	Cost
mSHO	3.497 599	0.7	17	7.3	7.713 535	3.350 056	5.285 631	2993.634
SHO	3.498 576	0.7	17	7.3	7.70 853	3.349 319	5.28 455	2994.77
DO	3.497 563	0.7	17	7.300 001	7.713 536	3.350 059	5.285 605	2993.635
CMA-ES	3.6	0.7	17	7.3	8.072 148	3.402 879	5.312 241	3071.526
HGS	3.497 599	0.7	17	7.3	7.713 535	3.350 056	5.285 631	2993.634
AOA	3.497 599	0.7	17	7.3	7.713 535	3.350 056	5.285 631	2993.634
SAO	3.6	2.6	3.565 453	2.84 373	2.951 027	3.314 187	2.715 053	3230.902
HHO	3.49 797	0.7	17	7.3	7.732 423	3.350 521	5.285 312	2994.197
PSO	3.497 599	0.7	17	7.3	7.713 535	3.350 056	5.285 631	2993.634
SPO	3.497 613	0.7	17	7.3	7.713 331	3.350 886	5.285 453	2993.836

Table 9: Results obtained from competitor algorithms for speed reducer engineering problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	2993.6343	2994.7697	2993.63 462	3071.526	2993.634	2993.634	3230.902	2994.197	2993.634	2993.836
Max	2993.6343	3821.162	3001.20 628	3072.334	2993.717	3006.627	8071.254	4307.676	2993.634	3359.51
Mean	2993.6343	3139.9316	2994.73 651	3072.307	2993.637	2994.509	4153.676	3146.587	2993.634	3118.58
Std	2.67E−13	200.19 962	1.80 047 494	0.147 413	0.015 028	2.925 884	956.0172	259.7566	2.67E−13	107.8919
Rank	1	8	5	7	3	4	10	9	2	6

(Sadollah *et al.*, 2013). The main objective of this problem is to minimize the cost of fabricating a welded beam by optimizing four variables: bar thickness (b), bar length including attached parts (l), weld thickness (h), and bar height (h). The problem is subject to four constraints, including buckling constraints of the bar (P_c), side constraints, end deflection of the beam (d), bending stress of the beam (h), and shear stress. The mathematical model for this problem is given as follows:

Consider $\vec{x} = [x_1 x_2 x_3 x_4] = [\ \mathrm{hlttbb}\ ]$

Minimize $f(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2)$

Subject to

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{max} \leqslant 0$$
$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{max} \leqslant 0$$
$$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{max} \leqslant 0$$
$$g_4(\vec{x}) = x_1 - x_4 \leqslant 0$$
$$g_5(\vec{x}) = P - P_c(\vec{x}) \leqslant 0$$
$$g_6(\vec{x}) = 0.125 - x_1 \leqslant 0$$
$$g_7(\vec{x}) = 1.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leqslant 0$$

Variables range

$$0.1 \leqslant x_1 \leqslant 2$$
$$0.1 \leqslant x_2 \leqslant 10$$
$$0.1 \leqslant x_3 \leqslant 10$$
$$0.1 \leqslant x_4 \leqslant 2$$

where

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \qquad (25)$$
$$\tau' = \frac{p}{\sqrt{2x_1x_2}}, \tau'' = \frac{MR}{J}$$
$$M = P\left(L + \frac{x_2}{2}\right)$$
$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2}$$
$$J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\}$$
$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \quad \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4}$$
$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)}{L^2}($$
$$P = 6000\ \mathrm{lb}, L = 14\ \mathrm{in.}, \quad \delta_{max} = 0.25\ \mathrm{in.}$$
$$E = 30 \times 1^6\ \mathrm{psi}, \quad G = 12 \times 10^6\ \mathrm{psi}$$
$$\tau_{max} = 13600\ \mathrm{psi}, \quad \sigma_{max} = 30000\ \mathrm{psi}.$$

This engineering problem is solved using the proposed mSHO and other competitive algorithms as shown in Table 12. The obtained statistical results are presented in Table 13. Table 13 shows that the optimal value of the function is 1.724 967, which was achieved using the mSHO algorithm. As the results show, mSHO produces promising results in comparison with the other algorithms and has a good ability for achieving minimal fabrication's cost in this problem.

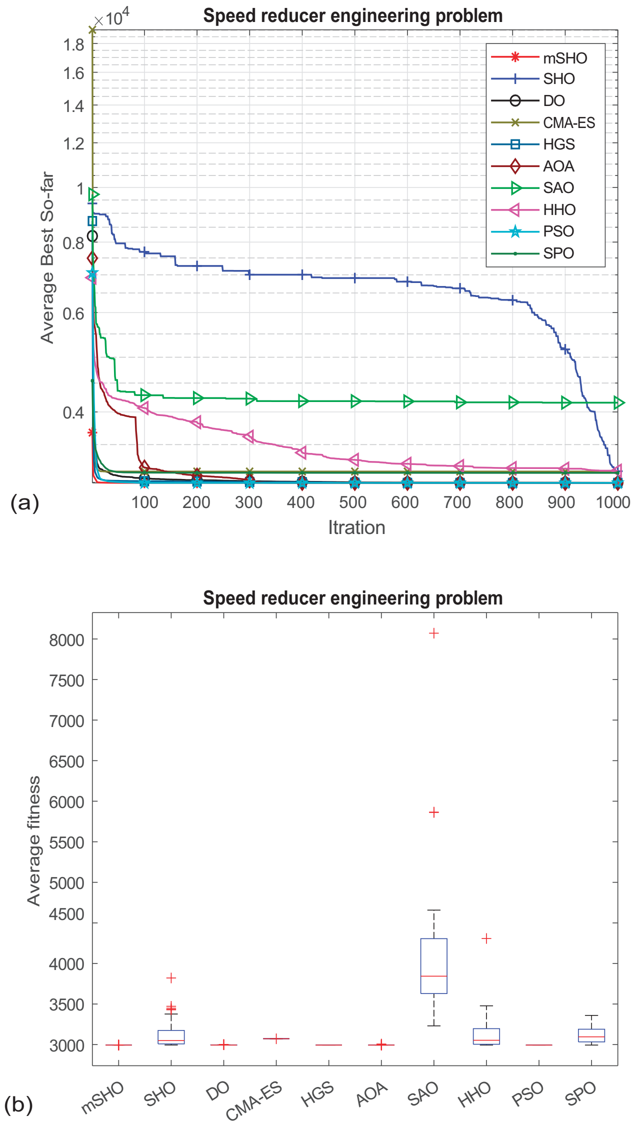


Figure 5: Convergence curve and boxplot for mSHO against other competitors – speed reducer engineering problem.

Table 10: Best solution obtained from the comparative algorithms for solving tension/compression spring problem.

Algorithm	x1	x2	x3	Cost
mSHO	0.051 687	0.356 672	11.29 167	0.012 665
SHO	0.050 987	0.340 068	12.33 627	0.012 674
DO	0.051 983	0.363 837	10.88 351	0.012667
CMA-ES	0.05	0.311 342	15	0.013 232
HGS	0.05 251	0.376 782	10.20 369	0.012 678
AOA	0.051 803	0.359 477	11.12 903	0.012 665
SAO	1.227 483	1.068 724	0.631 804	0.013 213
HHO	0.052 513	0.376 854	10.19 919	0.012 677
PSO	0.051 671	0.356 288	11.31 423	0.012 665
SPO	0.051 436	0.350 646	11.65 453	0.012 667

The performance of the mSHO algorithm and other competitive algorithms in solving the welded beam design problem is depicted in Fig. 7. The results show that the mSHO algorithm converges faster than the other algorithms and achieves near-optimal solutions quicker. Although the other algorithms also perform competitively, the SAO and CMA-ES algorithms show the lowest performance. Furthermore, the boxplot results demonstrate the stability of the mSHO algorithm, followed by the DO and AOA algorithms. These results indicate the efficiency and stability of the mSHO algorithm in solving the welded beam design problem.

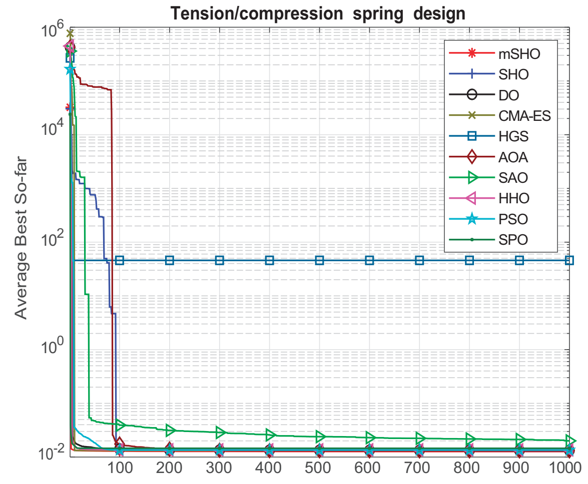
6.5. Three-bar truss engineering design problem

The aim of this engineering design problem is to minimize the weight of a truss by optimizing two parameters that represent the cross-sectional areas (x_1 and x_2), subject to the bounds constraints of $0 \leq x_1, x_2 \leq 1$. Additionally, three inequality constraints are related to buckling, deflection, and stress. The mathematical representation of this problem is as follows:

$$\begin{aligned}
&\text{Consider } \vec{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} A_1 & A_2 \end{bmatrix} \\
&\text{Minimize } f(\vec{x}) = \left(2\sqrt{2x_1} + x_2\right) * l, \\
&\text{Subject to } g_1(\vec{x}) = \frac{\sqrt{2x_1} + x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leqslant 0 \\
&g_2(\vec{x}) = \frac{x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \leqslant 0 \\
&g_3(\vec{x}) = \frac{1}{\sqrt{2x_2} + x_1} P - \sigma \leqslant 0 \\
&\text{Variables range } 0 \leqslant x_1, x_2 \leqslant 1 \\
&\text{where } l = 100 \text{ cm}, P = \frac{2KN}{cm^2}, \sigma = \frac{2KN}{cm^2}.
\end{aligned} \tag{26}$$

The engineering problem is tackled using the proposed mSHO algorithm and other competitive algorithms, as shown in Table 14. The statistical results obtained are presented in Table 15, which indicates that the mSHO algorithm achieved the optimal value of the function, 263.8915. The results demonstrate that mSHO produces better results than the other algorithms and has a strong ability to minimize the weight of the truss in this problem.

In addition, Fig. 8 displays the convergence curves and boxplot for the mSHO and other compared algorithms in solving the three-bar truss design problem. The figure demonstrates that the mSHO algorithm converges faster than the other methods and can generally achieve near-optimal solutions more quickly. Although the other algorithms also exhibit competitive performance, the SAO and HGS show the lowest performance. Moreover, the boxplot results indicate the stability of the mSHO algorithm, followed by the PSO and AOA algorithms. These results suggest that the proposed



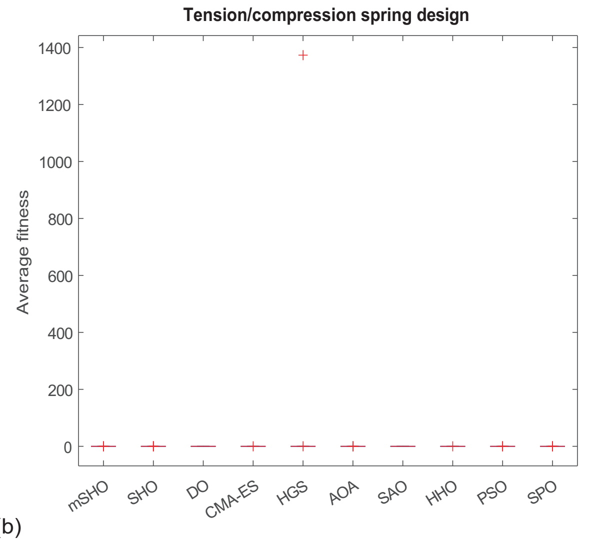


Figure 6: Convergence curve and boxplot for mSHO against other competitors – tension/compression spring problem.

Table 11: Results obtained from competitor algorithms for tension/compression spring problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	0.012 665	0.012 674	0.012 667	0.013 232	0.012 678	0.012 665	0.013 213	0.012 677	0.012 665	0.012 667
Max	0.012 738	0.014 513	0.014 664	0.013 278	1373.172	0.014 318	0.029 041	0.017 286	0.016 907	0.030 455
Mean	0.012 672	0.012 965	0.013 254	0.013 277	45.79 453	0.013 044	0.020 018	0.013 592	0.013 508	0.014 547
Std	1.44E-05	0.000 415	0.000 563	8.44E-06	250.7016	0.000 507	0.003 995	0.000 955	0.001 161	0.004 475
Rank	1	2	4	5	10	3	9	7	6	8

Table 12: Best solution obtained from the comparative algorithms for solving welded beam design problem.

Algorithm	x1	x2	x3	x4	Cost
mSHO	0.20 573	3.470 471	9.036 627	0.20 573	1.724 852
SHO	0.192 302	3.778 244	9.052 142	0.205 653	1.746 601
DO	0.20 573	3.470 495	9.036 626	0.20 573	1.724 854
CMA-ES	0.205 259	3.492 761	9.043 531	0.205 725	1.728 297
HGS	0.205 736	3.470 414	9.036 457	0.205 737	1.724 881
AOA	0.20 573	3.470 473	9.036 624	0.20 573	1.724 852
SAO	2	0.932 642	1.496 869	0.644 415	1.792 207
HHO	0.173 239	4.308 508	9.096 589	0.205 631	1.790 459
PSO	0.20 573	3.470 475	9.036 624	0.20 573	1.724 852
SPO	0.20 573	3.470 484	9.036 623	0.20 573	1.724 852

Table 13: Results obtained from competitor algorithms for the welded beam problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	1.724 852	1.746 601	1.724 854	1.728 297	1.724 881	1.724 852	1.792 207	1.790 459	1.724 852	1.724 852
Max	1.7256	4.306 289	1.744 596	2.266 116	4.341 756	1.836 955	4.036 102	2.682 967	2.175 585	3.03 395
Mean	1.724 967	1.995 641	1.72 778	2.230 261	2.13 089	1.733 932	2.995 645	2.025 109	1.816 491	2.067 492
Std	0.000 192	0.477 618	0.00 427	0.136 449	0.631 101	0.024 171	0.596 869	0.210 771	0.13 514	0.387 053
Rank	1	5	2	9	8	3	10	6	4	7

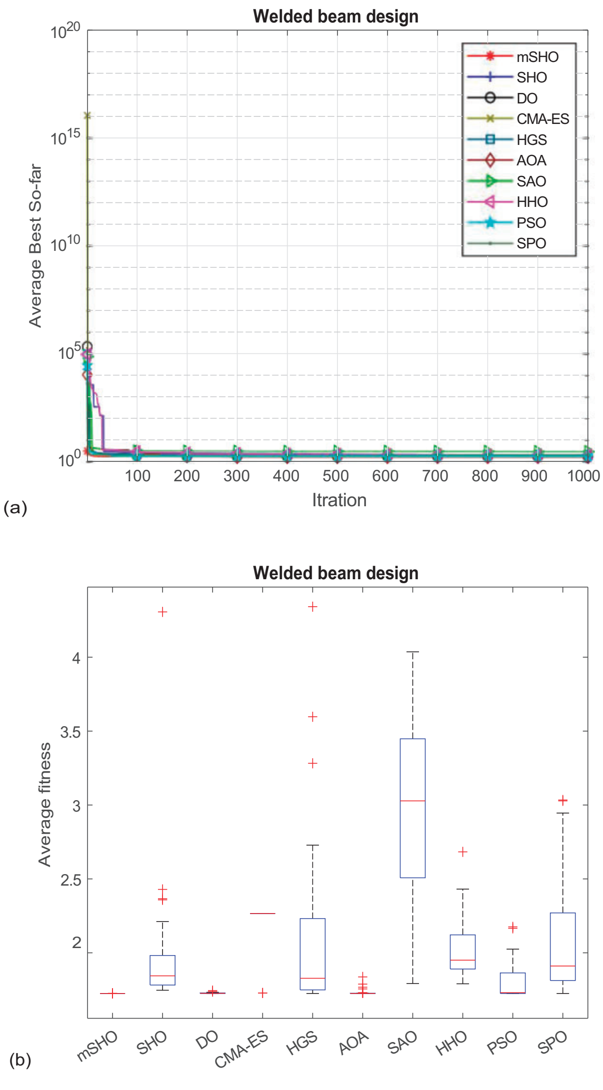


Figure 7: Convergence curve and boxplot for mSHO against other competitors – welded beam design problem.

mSHO algorithm is efficient and stable in solving the three-bar truss design problem.

6.6. Industrial refrigeration system problem

The objective of the industrial refrigeration system problem is to minimize the cost of the refrigeration system while optimizing the refrigerants, temperature levels, cycle configuration, and compression technology. This problem is described mathematically and has multiple variables and constraints. The details of the problem formulation can be found in Marechal and Kalitventzeff (2001). The problem can be mathematically formulated as follows:

$$
\begin{aligned}
\text{Minimize}\quad f(x) = {} & 63\,098.88x_2x_4x_{12} + 5441.5x_2^2x_{12} \\
& + 115\,055.5x_2^{1.664}x_6 + 6172.27x_2^2x_6 \\
& + 63\,098.88x_1x_3x_{11}5441.5x_1^2x_{11} \\
& + 115\,055.5x_1^{1.664}x_5 + 6172.27x_1^2x_5 \\
& + 140.53x_1x_{11} + 281.29x_3x_{111} \\
& + 70.26x_1^2 + 281.29 + 281.29x_3^2 \\
& + 14\,437x_8^{1.8812}x_{12}^{0.3424}x_{10}x_{14}^{-1}x_1^2x_7x_9^{-1} \\
& + 20\,470.2x_7^{2.893}s_{11}^{0.316}x_1^2x_1x_3
\end{aligned}
$$

$$
\begin{aligned}
\text{Subject to}\quad & g_1(x) = 1.524x_7^{-1} \le 1 \\
& g_2(x) = 1.524x_8^{-1} \le 1 \\
& g_3(x) = 0.07\,789x_1 - 2x_7^{-1}x_9 - 1 \le 0 \\
& g_4(x) = 7.05\,305x_9^{-1}x_1^2x_{10}x_8^{-1}x_2^{-1}x_{14}^{-1} - 1 \le 0, \\
& g_5(x) = 0.0833x_{13}^{-1}x_{14} - 1 \le 0, \\
& g_6(x) = 47.136x_2^{0.333}x_{10}^{-1}x_{12} - 1.333x_8x_{13}^{2.1195} \\
& \qquad + 62.08x_{13}^{2.1195}x_{12}^{-1}x_8^{0.2}x_{10}^{-1} - 1 \le 0 \\
& g_7(x) = 0.04\,771x_{10}x_8^{1.8812}x_{12}^{0.3424} - 1 \le 0 \\
& g_8(x) = 0.0488x_9x_7^{1.893}x_{11}^{0.316} - 1 \le 0 \\
& g_9(x) = 0.0099x_1x_3^{-1} - 1 \le 0, \\
& g_{10}(x) = 0.0193x_2x_4^{-1} - 1 \le 0 \\
& g_{11}(x) = 0.0298x_1x_5^{-1} - 1 \le 0, \\
& g_{12}(x) = 0.056x_2x_6^{-1} - 1 \le 0 \\
& g_{13}(x) = 2x_9^{-1} - 1 \le 0, \\
& g_{14}(x) = 2x_{10}^{-1} - 1 \le 0, \\
& g_{15}(x) = x_{12}x_{11}^{-1} - 1 \le 0
\end{aligned}
\qquad (27)
$$

$$\text{Variables range}\quad 0.001 \le x_i \le 5, i = 1, \cdots, 14.$$

The proposed mSHO algorithm and other competitive algorithms are used to solve this engineering problem, as shown in Table 16. The statistical results obtained are presented in Table 17, which indicates that the mSHO algorithm achieved the optimal value of the function at 0.032 255. The results demonstrate

Table 14: Best solution obtained from the comparative algorithms for solving three-bar truss engineering design problem.

Algorithm	x1	x2	Cost
mSHO	0.788 649	0.408 235	263.8915
SHO	0.787 638	0.411 102	263.8922
DO	0.788 649	0.408 234	263.8915
CMA-ES	0.756 483	0.508 411	264.8067
HGS	0.780 942	0.431 176	264.0014
AOA	0.788 649	0.408 235	263.8915
SAO	1	0.733 032	264.4989
HHO	0.788 486	0.408 697	263.8915
PSO	0.788 649	0.408 235	263.8915
SPO	0.788 651	0.408 229	263.8915

that mSHO yields promising outcomes compared with other algorithms and is proficient in achieving the minimum cost of the refrigeration system in this problem.

Figure 9 shows the convergence curves and boxplot for the industrial refrigeration system optimization problem using mSHO and the compared algorithms. The results indicate that the mSHO algorithm achieves faster convergence and usually obtains near-optimal solutions quicker than the other algorithms. Although the other algorithms also demonstrate competitive performance, the SAO and SPO algorithms show the lowest performance. Furthermore, the boxplot results show that the proposed mSHO algorithm exhibits stability in comparison with the DO and SHO algorithms. These findings demonstrate the effectiveness and stability of the proposed mSHO algorithm in solving the industrial refrigeration system optimization problem.

6.7. Multi-product batch plant problem

The objective of this model is to minimize the production cost of a multi-product batch process by optimizing the allocation of resources. The process consists of three stages that all products follow, and there are two different products being produced. The model has 10 decision variables: $N_1, N_2, N_3, V_1, V_2, V_3, T_1, T_2, B_1$, and B_2, represented by the shorthand notations x_1 through x_{10}. The mathematical formulation of the model, as presented in Kumar *et al.*, (2020), is as follows:

$$\text{Minimize } f(x) = \sum_{j=1}^{M} a_j N_j V_j^{b_j}$$

$$\begin{aligned} \text{Subject to } \; & g_1(x) = S_{ij} B_i - V_j \le 0 \\ & g_2(x) = -H + \textstyle\sum_{i=1}^{N} \frac{Q_i T_i}{B_i} \le 0 \\ & g_3(x) = t_{ij} - N_j T_i \le 0 \end{aligned} \tag{28}$$

$$\begin{aligned} \text{Variables range } \; & 1 \le N_i \le 3 \\ & 250 \le V_i \le 2500 \\ & \max\left(\frac{t_{ij}}{N_j^u}\right) \le T_i \le \max(t_{ij}) \\ & \frac{Q_i^* T_i}{H} \le B_i \le \min\left(Q_i, \min\left(\frac{V_j^u}{S_{ij}}\right)\right) \end{aligned}$$

where $N = 2$, $M = 3$, $a_j = 250$, $H = 6000$, $b_j = 0.6$, $S_{11} = 2$, $S_{12} = 3$, $S_{13} = 4$, $S_{21} = 4$, $S_{22} = 6$, $S_{23} = 3$, $t_{11} = 8$, $t_{12} = 20$, $t_{13} = 8$, $t_{21} = 16$, $t_{22} = 4$, and $t_{23} = 4$. i means the product, j means the stage of production, a_j is variable cost coefficient of stage j process equipment investment cost, N_j is the number of equipment at stage j, V_j is the size of equipment at stage j, T_i means the cycle time of product i, B_i means the batch size of product i, b_j is the fixed-cost charges for the investment cost of process equipment at stage j.

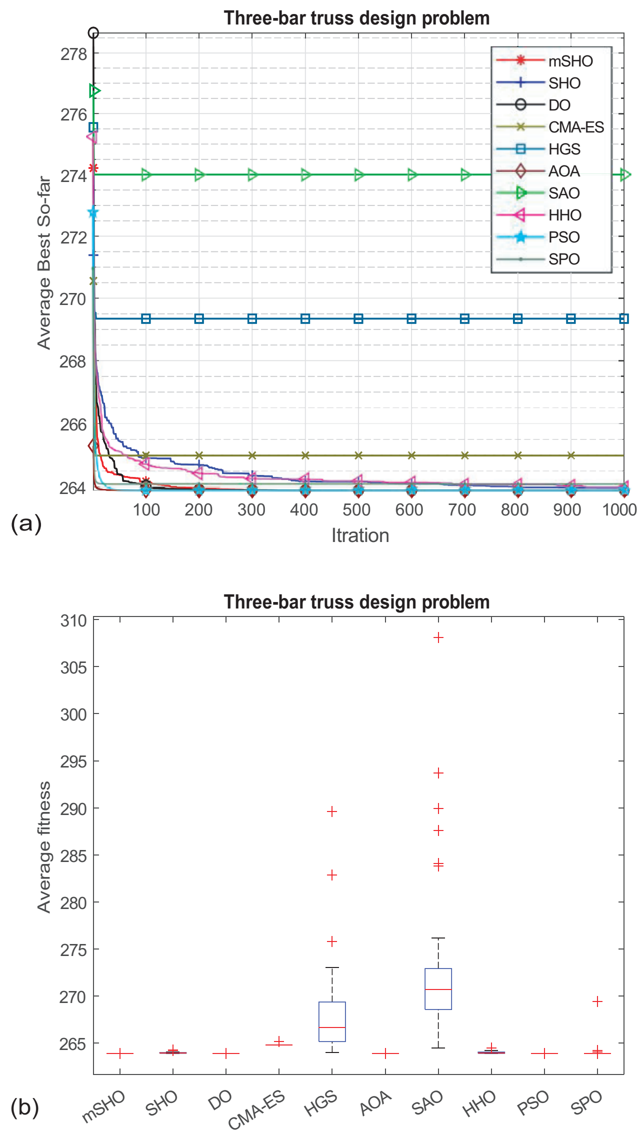


Figure 8: Convergence curve and boxplot for mSHO against other competitors – three-bar truss engineering design problem.

The multi-product batch process model presented above aims to reduce production costs by optimizing the allocation of resources in product manufacturing. To solve this problem, the proposed mSHO algorithm and several other competitive algorithms were compared, and the results are presented in Table 18. From

Table 15: Results obtained from competitor algorithms for three-bar truss engineering design problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	263.8915	263.8922	263.8915	264.8067	264.0014	263.8915	264.4989	263.8915	263.8915	263.8915
Max	263.8915	264.2569	263.8919	265.1765	289.6074	263.8915	308.1042	264.4841	263.8915	269.4398
Mean	263.8915	263.9661	263.8915	264.819	269.3437	263.8915	274.0035	264.001	263.8915	264.0988
Std	2.59E−11	0.086 065	8.03E−05	0.067 512	6.670 943	7.45E−07	9.827 149	0.128 725	3.61E−07	1.011 145
Rank	1	5	4	8	9	3	10	6	2	7

Table 16: Best solution obtained from the comparative algorithms for solving industrial refrigeration system problem.

Algorithm	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	Cost
mSHO	0.001	0.001	0.001	0.001	0.001	0.001	1.524	1.524	4.999998	2	0.001	0.001	0.007 279	0.087 379	0.032 255
SHO	0.001	0.001	0.001	0.001	0.001	0.001	1.527 549	1.523 777	4.835144	2.0468	0.001	0.001	0.002 692	0.030 836	0.095 676
DO	0.001	0.001 001	0.001 038	0.001 015	0.001 001	0.001 001	1.524	1.524 002	4.999952	2.000 011	0.001	0.001	0.007 291	0.087 529	0.032 272
CMA-ES	0.001	0.001	3.896 117	5	0.001	0.001	2.622 885	5	5	2.652 281	0.001	0.001	0.001	0.001	4311.659
HGS	0.001	0.001	0.001	0.001	0.001	0.001	1.524	1.524	5	2	0.001	0.001	0.007 293	0.087 556	0.032 213
AOA	0.001 005	0.001	0.001 086	0.001 146	0.001 035	0.001	1.524	1.524 039	4.998429	2	0.001	0.001	0.00 728	0.087 399	0.032 711
SAO	4.056 261	3.897 615	3.83 318	0.341 573	0.159 483	4.162 318	3.263 213	3.984 312	0.914673	5	0.729 443	4.060 669	4.240 338	5	14 246 044
HHO	0.001	0.001	0.005 309	0.242 903	0.010 031	0.004 435	2.86 165	2.513 477	2.024121	2.000 021	0.001	0.001	0.006 787	0.079 804	0.364 219
PSO	0.001	0.001	0.001	0.001	0.001	0.001	1.524	1.524	5	2.000 022	0.001	0.001	0.007 293	0.087 557	0.032 213
SPO	0.001	0.001	0.001 176	0.002 806	0.001	0.001	1.524 087	1.524 001	5	5	0.001 136	0.001 117	0.008 457	0.101 529	0.058 815

Table 17: Results obtained from competitor algorithms for industrial refrigeration system problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	0.032 255	0.095 676	0.032 272	4311.659	0.032 213	0.032 711	14 246 044	0.364 219	0.032 213	0.058 815
Max	0.052 875	0.324 592	0.13 298	2660 163	936 189.4	936 189.4	4.76E+09	984 165.9	936 189.4	2.44E+08
Mean	0.038 305	0.189 575	0.056 888	92 840.04	312 063.2	31 206.37	7.88E+08	349 833.5	93 619	62 388 451
Std	0.005 233	0.041 015	0.023 447	484 889.9	448 868.4	170 924	9.26E+08	466 964.4	285 658.1	74 649 811
Rank	1	3	2	8	6	4	10	7	5	9

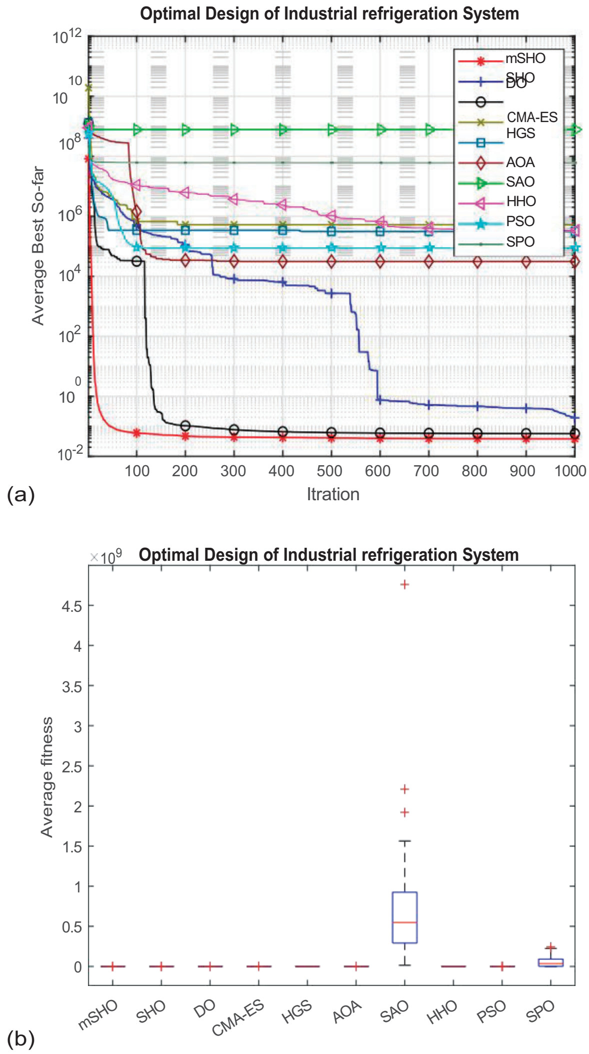


Figure 9: Convergence curve and boxplot for mSHO against other competitors – industrial refrigeration system problem.

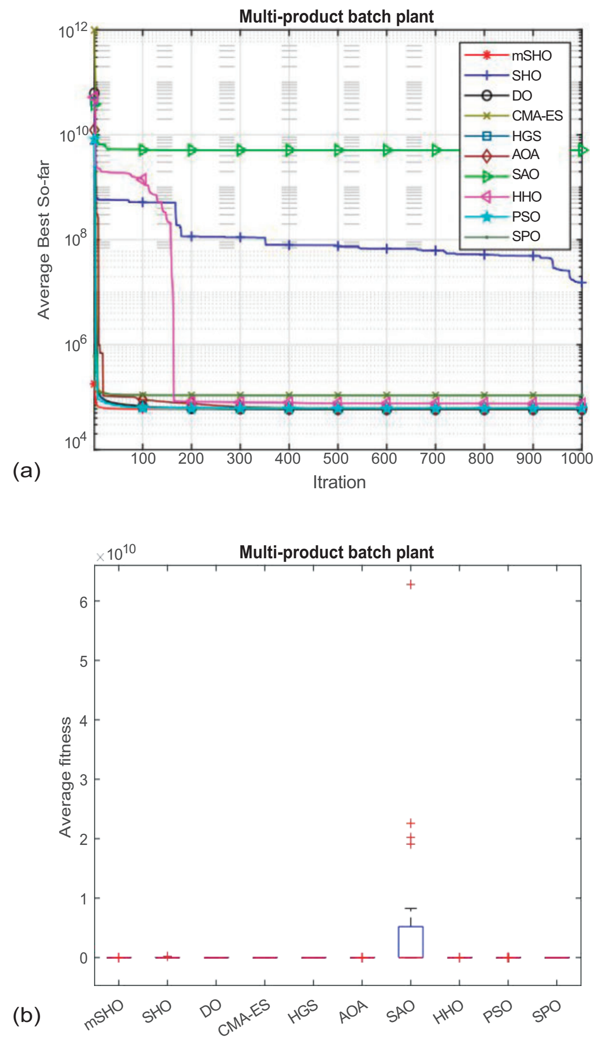


Figure 10: Convergence curve and boxplot for mSHO against other competitors – multi-product batch plant problem.

Table 18: Best solution obtained from the comparative algorithms for solving multi-product batch plant problem.

Algorithm	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	Cost
mSHO	1.525 762	1.508 002	0.674 961	479.9229	719.8871	660.2033	9.999 419	7.999 732	120.1043	59.92 858	58 507.14
SHO	1.519 752	1.855 192	0.628 977	531.0469	822.9263	705.5837	9.992 353	8.515 774	120.4098	71.54 331	62 676.08
DO	0.728 125	0.645 495	1.135 645	963.3442	1445.014	1309.242	19.99 983	15.9997	234.6931	123.4888	53 639.01
CMA-ES	1.780 084	2.350 703	0.549 313	483.1708	735.9209	709.1498	10.06 453	8.004 826	128.8223	56.37	59 471.57
HGS	0.51	0.730 582	0.771 096	980.4283	1470.642	1286.898	20	16	220.6308	134.7916	53 820.53
AOA	1.4302	1.377 871	0.610 608	959.619	1439.429	1321.696	20	16	240.7911	119.5092	53 663.04
SAO	2.324 455	2.900 101	1.435 999	2.209 186	1.393 715	0.792 087	3.341 121	1.503 093	3.419 913	2.639 791	80 120.89
HHO	1.713 052	1.587 087	1.188 423	524.3093	743.4804	1127.821	9.999 611	8.001 561	150.0532	48.29 276	64 778.06
PSO	1.847 141	1.97 878	0.697 938	479.3873	719.081	663.0222	9.999 897	7.999 933	121.3927	59.1505	58 506.03
SPO	0.51	0.51	0.51	1021.185	2088.131	1597.147	20	16	332.5038	89.04 394	61 399.49

Table 19: Results obtained from competitor algorithms for multi-product batch plant problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	58 507.14	62 676.08	53 639.01	59 471.57	53 820.53	53 663.04	80 120.89	64 778.06	58 506.03	61 399.49
Max	66 592.15	2.12E+08	66 651.86	163 121.8	73 667.78	66 772.51	6.28E+10	90 866.09	71 888.26	156 962.1
Mean	58 966.85	15 388 990	58 445.22	108 481.4	62 058.75	61358.21	5.12E+09	74008.67	61 126.71	108 824.3
Std	1564.308	38 309 104	4082.766	31 395.86	5715.205	2877.974	1.26E+10	6004.125	4792.113	28 836.15
Rank	2	9	1	7	5	4	10	6	3	8

Table 20: Best solution obtained from the comparative algorithms for solving cantilever beam problem.

Algorithm	x1	x2	x3	x4	x5	Cost
mSHO	6.015 906	5.308 734	4.495 939	3.500 899	2.152 182	1.339 956
SHO	6.06 771	5.389 926	4.417 899	3.450 059	2.155 511	1.340 484
DO	6.014 985	5.305 781	4.490 804	3.509 239	2.152 893	1.339 959
CMA-ES	6.018 345	5.329 495	4.474 012	3.518 808	2.133 878	1.340 011
HGS	6.019 407	5.297 822	4.482 162	3.517 852	2.1567	1.339 974
AOA	6.011 535	5.310 885	4.495 655	3.502 741	2.152 851	1.339 957
SAO	22.51 819	66.97 232	48.25 931	7.342 964	15.97 925	1.558 312
HHO	6.045 302	5.265 671	4.576 413	3.520 634	2.075 623	1.340 579
PSO	6.015 424	5.30 515	4.496 319	3.504 543	2.152 239	1.339 957
SPO	6.022 957	5.276 079	4.519 865	3.514 974	2.140 741	1.340 016

Table 21: Results obtained from competitor algorithms for cantilever beam problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	1.339 956	1.340 484	1.339 959	1.340 011	1.339 974	1.339 957	1.558 312	1.340 579	1.339 957	1.340 016
Max	1.339 967	1.351 724	1.340 016	1.340 075	1.34 073	1.340 113	10.84 034	1.346 869	1.340 026	3.082 532
Mean	1.339 957	1.344 196	1.339 972	1.340 013	1.340 229	1.339 991	6.036 222	1.342 705	1.339 974	1.552 187
Std	2.08E−06	0.003 087	1.39E−05	1.17E−05	0.000 202	4.44E−05	2.312 212	0.001 409	1.93E−05	0.319 524
Rank	1	8	2	5	6	4	10	7	3	9

the statistical results in Table 19, it can be seen that the mSHO algorithm achieved the optimal value of the function, which was 58 507.14. The comparison of the algorithms shows that mSHO outperforms the others and can achieve minimal production costs in this problem.

Furthermore, Fig. 10 displays the convergence curves and boxplot for the mSHO algorithm and other competitive methods in solving the multi-product batch plant problem. The graph shows that the proposed mSHO algorithm converges faster than the other algorithms and can typically obtain near-optimal solutions more quickly. Although the other algorithms also perform competitively, SAO and SPO demonstrate the poorest performance. Conversely, the boxplot results indicate the stability of the mSHO algorithm, followed by the DO and PSO algorithms. These findings indicate that the proposed mSHO algorithm is effective and stable in addressing the multi-product batch plant problem.

6.8. Cantilever beam problem

The cantilever beam problem belongs to the category of concrete engineering problems, as described in the study by Bhadoria and Kamboj (2019). The problem aims to minimize the overall weight of a cantilever beam by optimizing the parameters of a hollow square cross-section. The mathematical formulation of this problem is presented as follows:

$$\begin{aligned} &\text{Consider } \vec{x} = [x_1 x_2 x_3 x_4 x_5] \\ &\text{Minimize } f(\vec{x}) = 0.6224\,(x_1 + x_2 + x_3 + x_4 + x_5) \\ &\text{Subject to } g(\vec{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \le 1 \\ &\text{Variable range } 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100. \end{aligned} \tag{29}$$

The proposed mSHO algorithm and other competitive algorithms were employed to solve the cantilever beam problem, as shown in Table 20. The statistical results obtained are presented in Table 21,which shows that the optimal value of the function is 1.339 956, achieved by using the mSHO algorithm. These results demonstrate that mSHO produces promising outcomes in com-

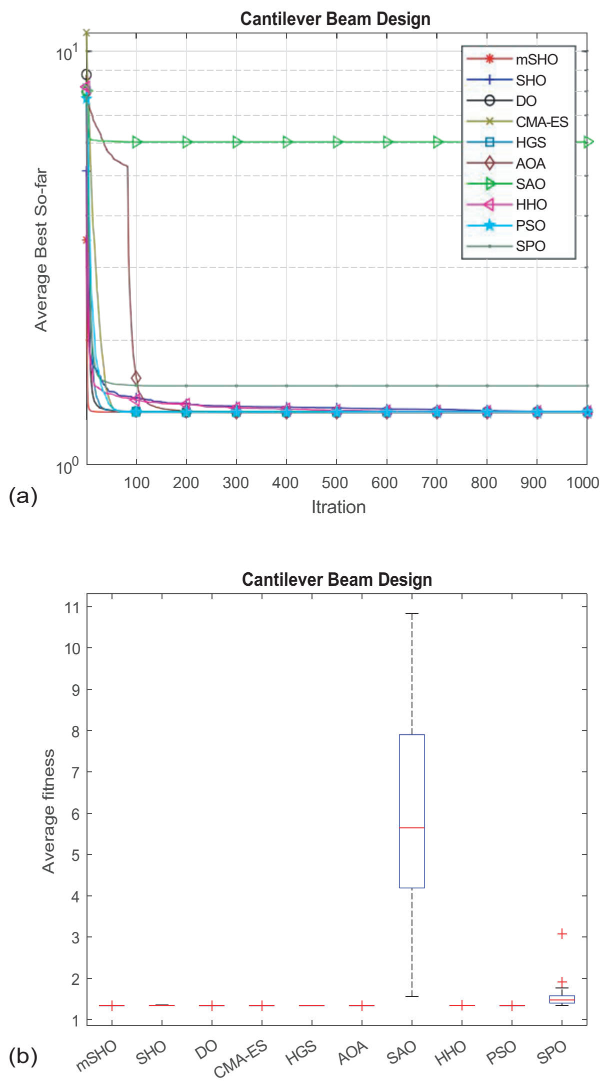


Figure 11: Convergence curve and boxplot for mSHO against other competitors – cantilever beam problem.

Table 22: Results obtained from competitor algorithms for multi-disc clutch brake problem.

Algorithm	x1	x2	x3	x4	x5	cost
mSHO	70	90	1	213.5391	2	0.235 242
SHO	69.99 874	90	1	66.50 295	2	0.235 255
DO	70	90	1	999.9588	2	0.235 242
CMA-ES	69.19 936	90	1	664.1618	2	0.243 435
HGS	70	90	1	1000	2	0.235 242
AOA	70	90	1	858.5333	2	0.235 242
SAO	64.55 532	62.9971	60.45 108	61.2192	71.05524	0.257 118
HHO	70	90	1	946.8728	2	0.235 242
PSO	70	90	1	33.07 846	2	0.235 242
SPO	70	90	1	1000	2	0.235 242

parison with the other algorithms, with a high ability to minimize the weight of the cantilever beam in this problem.

Additionally, Fig. 11presents the convergence curves and boxplot of mSHO and other compared methods for the cantilever beam problem. The figure indicates that the proposed mSHO algorithm exhibits faster convergence than the other algorithms and can usually obtain near-optimal solutions more quickly. Although the other algorithms also demonstrate competitive performance, SAO and SPO exhibit the lowest performance. Furthermore, the boxplot results reveal the stability of the proposed mSHO algorithm, followed by the DO and PSO algorithms. The results of this experiment demonstrate the efficiency and stability of the proposed mSHO algorithm in solving the cantilever beam problem.

6.9. Multiple disc clutch brake problem

The multi-plate disc clutch brake is a well-known optimization problem in mechanical engineering, which aims to minimize the total weight of a multiple-disc clutch brake by optimizing five variables: driving force (F), the number of friction surfaces (Z), the thickness of discs (A), outer radius ($r0$), and inner radius ($r1$). These variables are denoted by $x1$, $x2$, $x3$, $x4$, and $x5$. The problem is subject to eight constraints based on the geometry and operating requirements. The mathematical formulation for this engineering optimization problem can be expressed as follows, as stated in Abderazek *et al.*, (2017):

$$\begin{aligned}
&\text{Minimize} \quad f(x) = \pi\left(r_0^2 - r_i^2\right)(Z+1)\rho t \\
&\text{Subject to} \quad g_1(x) = r_0 - r_i - \Delta r \geqslant 0 \\
&\quad g_2(x) = l_{\max} - (Z+1)(t+\delta) \geqslant 0 \\
&\quad g_3(x) = P_{\max} - P_{rz} \geqslant 0 \\
&\quad g_4(x) = P_{\max} v_{vr\max} - P_{rz} v_{sr} \geqslant 0 \\
&\quad g_5(x) = v_{sr\max} - v_{sr} \geqslant 0 \\
&\quad g_6(x) = T_{\max} - T \geqslant 0 \\
&\quad g_7(x) = M_h - sM_s \geqslant 0 \\
&\quad g_8(x) = T \geqslant 0 \\
&\text{Where } M_h = \frac{2}{3}\mu F Z \frac{r_0^3 - r_i^3}{r_0^2 - r_i^2} \\
&P_{rz} = \frac{2}{3}\frac{F}{\pi\left(r_0^2 - r_i^2\right)} \\
&v_{rz} = \frac{2\pi n\left(r_0^3 - r_i^3\right)}{90\left(r_0^2 - r_i^2\right)} \\
&T = \frac{I_z \pi n}{30\left(M_h - M_f\right)} \\
&\Delta r = 20\ \text{mm}, I_z = 55\ \text{kgm}^2, P_{\max} = 1\ \text{MPa} \\
&F_{\max} = 1000\ \text{N}, T_{\max} = 15\ \text{s}, \mu = 0.5 \\
&s = 1.5, M_s = 40\ \text{Nm}, M_f = 3\ \text{Nm}, N = 250\ \text{r/min} \\
&v_{sr\max} = 10\ \text{m/s}, l_{\max} = 30\ \text{mm} \\
&60\ \text{mm} \le r_i \le 80\ \text{mm}, 90\ \text{mm} \le r_0 \le, 110\ \text{mm}, \\
&1.5\ \text{mm} \le t \le 3\ \text{mm}, 600\ \text{N} \le F \le 1000\ \text{N}, 2 \le Z \le 9.
\end{aligned} \tag{30}$$

The multi-plate disc clutch brake problem was solved by applying the mSHO algorithm and other competitive algorithms, as presented in Table 22. The statistical analysis of the results is shown in Tables 23 and 24, which indicates that the mSHO algorithm, along with the AOA algorithm, achieved the minimum objective function value of 0.235 242. These results demonstrate that the mSHO algorithm performs better than other algorithms for minimizing the weight of the clutch brake in this engineering problem.

Figure 12 shows the convergence curves and boxplot for mSHO and all other compared methods, revealing that the proposed mSHO algorithm converged faster than the other algorithms and was able to obtain near-optimal solutions more quickly. While the other algorithms also showed competitive performance, the SAO and SPO algorithms exhibited the lowest performance. Additionally, the results of the boxplot demonstrated the stability of the proposed mSHO algorithm, followed by the DO and PSO algorithms. Overall, these findings indicate the efficiency and stability of the mSHO algorithm in handling the multi-plate disc clutch brake problem.

7. Discussion

The aforementioned results show that the proposed mSHO has advanced results compared with the other metaheuristic algorithms, including SHO, DO, CMA-ES, HGS, AOA, SAO, HHO, PSO, and SPO. In addition, as optimization issues get more challenging, mSHO's effectiveness remains unchanged, demonstrating its stability and aptitude for addressing challenging search domains. This demonstrates that it is a powerful tool for addressing challenging optimization problems. The results can be summarized as follows:

(i) CEC2020 test function
 (a) mSHO demonstrates highly competitive fitness values, ranking first for all functions except F6 and F10.
 (b) The proposed mSHO algorithm achieves an overall ranking of 1.
 (c) According to the Friedman test, the proposed mSHO exhibits the lowest value of 1.3.

(ii) Engineering problems
 (a) Pressure vessel design problem: The optimal function value is 0.012 665, attained using the mSHO algorithm.
 (b) Speed reducer design problem: The optimal function value is 2993.634, achieved using the mSHO, HGS, AOA, and PSO algorithms.
 (c) Tension/compression spring problem: The mSHO algorithm achieves the optimal function value of 0.01 266.
 (d) Welded beam design problem: The optimal function value is 1.724 967, obtained using the mSHO algorithm.
 (e) Three-bar truss engineering design problem: The mSHO algorithm attains the optimal function value of 263.8915.

Table 23: Best solution obtained from the comparative algorithms for solving multi-disc clutch brake problem.

Mea.	mSHO	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Min	0.235 242	0.235 255	0.235 242	0.243 435	0.235 242	0.235 242	0.257 118	0.235 242	0.235 242	0.235 242
Max	0.235 242	0.237 529	0.235 243	0.24 762	0.250 273	0.235 242	0.568 008	0.235 243	0.235 243	0.264 648
Mean	0.235 242	0.235 562	0.235 243	0.247 341	0.235 743	0.235 242	0.410 957	0.235 242	0.235 242	0.239 091
Std	1.41E−16	0.000 445	6.97E−08	0.001 062	0.002 744	1.41E−16	0.081 127	5.97E−08	2.39E−08	0.008 781
Rank	2	6	5	9	7	1	10	4	3	8

Table 24: Wilcoxon's signed rank test.

mSHO versus	SHO	DO	CMA-ES	HGS	AOA	SAO	HHO	PSO	SPO
Pressure vessel design problem	3.02E−11	1.41E−09	1.72E−12	0.001 936	1.07E−07	3.02E−11	3.02E−11	3.02E−11	3.16E−10
Speed reducer problem	3.02E−11	6.72E−10	1.72E−12	1.14E−11	0.005 757	3.02E−11	3.02E−11	1.14E−11	3.02E−11
Tension/compression spring problem	9.92E−11	1.07E−09	1.72E−12	5.49E−11	1.16E−07	3.02E−11	6.07E−11	3.01E−07	3.19E−09
Welded beam design problem	3.02E−11	7.04E−07	2.36E−12	1.21E−10	0.035 137	3.02E−11	3.02E−11	0.016 955	8.87E−10
Three-bar truss engineering design problem	3.02E−11	0.012 732	1.72E−12	3.01E−11	4.12E−11	3.02E−11	3.02E−11	1.18E−08	5.46E−09
Industrial refrigeration system problem	3.02E−11	0.000 225	1.72E−12	9.2E−05	0.149 449	3.02E−11	3.02E−11	0.000 691	3.02E−11
Multi-product batch plant problem	3.34E−11	0.994 102	3.69E−11	0.002 499	1.73E−06	3.02E−11	3.34E−11	0.001 518	4.08E−11
Cantilever beam design	3.02E−11	2.15E−10	1.72E−12	3.02E−11	4.62E−10	3.02E−11	3.02E−11	8.1E−10	3.02E−11
Multi-disc clutch brake problem	3.02E−11	5.09E−08	2.36E−12	4.56E−11	1.21E−12	3.02E−11	2.93E−09	1.21E−12	0.063 525

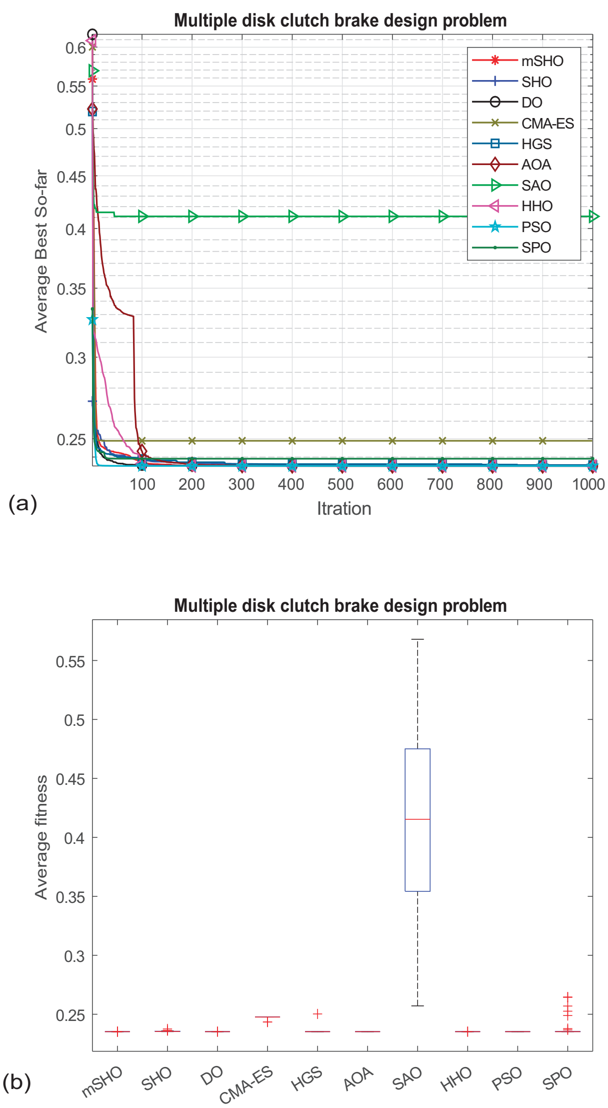


Figure 12: Convergence curve and boxplot for mSHO against other competitors – multi-disc clutch brake problem.

(f) Industrial refrigeration system problem: The mSHO algorithm reaches the optimal function value of 0.032 255.
(g) Multi-product batch plant problem: The mSHO algorithm yields the optimal function value of 58 507.14.
(h) Cantilever beam problem: The optimal function value is 1.339 956, achieved using the mSHO algorithm.
(i) Multiple disc clutch brake problem: The mSHO algorithm, in conjunction with the AOA algorithm, achieves the minimum objective function value of 0.23 524.

This study is limited to the selected problems, which can be extended to experiment mSHO with machine-learning-related problems, such as feature engineering and selection, hyperparameter tuning, ensemble learning, neural architecture search, model selection, and model compression. In addition, the study is limited to single-objective optimization, which can be extended to solve multi-objective optimization problems to represent trade-offs between conflicting objectives.

8. Conclusions and Future Research

SHO is a notable metaheuristic algorithm designed to emulate the nuanced behaviors of sea horses, encompassing their feeding patterns, male reproductive strategies, and intricate movement dynamics. This study introduces an evolved version of the SHO algorithm, referred to as mSHO, which uses a set of distinct mechanisms aimed at boosting its local search capabilities by substituting the original approach with an innovative local search strategy executed through three strategic phases: a neighborhood-based local search, a global non-neighbor-based search, and a circumferential exploration strategy, which helps mSHO to attain heightened performance in exploring the search spaces. The proficiency of the mSHO algorithm is rigorously examined through evaluations encompassing CEC2020 benchmark functions and a spectrum of nine intricate engineering problems. A meticulous comparative analysis with nine robust metaheuristic algorithms is conducted to corroborate and validate the achieved outcomes. Statistical techniques, including Wilcoxon's rank-sum and Friedman's tests, are judiciously employed to unveil substantial disparities among the examined algorithms. The empirical findings unequivocally affirm mSHO's supremacy, consistently demonstrating its superior performance across a diverse range of benchmark functions. Moreover, the effectiveness of mSHO persists unaffected as the difficulty of optimization problems increases, confirming its robustness and skill in handling complex search areas. This validates its strength as a highly valuable instrument for tackling intricate optimization challenges. Looking ahead, the prospective applications of mSHO appear promising, spanning domains such as feature selection, cloud job scheduling, multi-level threshold image segmentation, and hyperparameter optimization for different machine learning models.

Acknowledgments

This work was supported by the Ministerio de Ciencia e Innovación de España under project number PID2020-115570GB-C22 MCIN/AEI/10.13039/501100011033 and by the Cátedra de Empresa Tecnología para las Personas (UGR-Fujitsu).

Conflict of interest statement

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

Abderazek, H., Ferhat, D., & Ivana, A. (2017). Adaptive mixed differential evolution algorithm for bi-objective tooth profile spur gear optimization. *The International Journal of Advanced Manufacturing Technology*, **90**(5), 2063–2073.

Ahmadianfar, I., Heidari, A. A., Gandomi, A. H., Chu, X., & Chen, H. (2021). Run beyond the metaphor: An efficient optimization algorithm based on Runge Kutta method. *Expert Systems with Applications*, **181**, 115079.

Ahmadianfar, I., Heidari, A. A., Noshadian, S., Chen, H., & Gandomi, A. H. (2022). Info: An efficient optimization algorithm based on weighted mean of vectors. *Expert Systems with Applications*, **195**, 116516.

Ahmed, R., Rangaiah, G. P., Mahadzir, S., Mirjalili, S., Hassan, M. H., & Kamel, S. (2023). Memory, evolutionary operator, and local search based improved grey wolf optimizer with linear population size reduction technique. *Knowledge-Based Systems*, **264**, 110297.

Alahmer, H., Alahmer, A., Alamayreh, M. I., Alrbai, M., Al-Rbaihat, R., Al-Manea, A., & Alkhazaleh, R. (2023). Optimal water addition in emulsion diesel fuel using machine learning and sea-horse optimizer to minimize exhaust pollutants from diesel engine. *Atmosphere*, **14**(3), 449.

Alweshah, M., Alkhalaileh, S., Al-Betar, M. A., & Bakar, A. A. (2022). Coronavirus herd immunity optimizer with greedy crossover for feature selection in medical diagnosis. *Knowledge-Based Systems*, **235**, 107629.

Aribowo, W. (2023). A novel improved sea-horse optimizer for tuning parameter power system stabilizer. *Journal of Robotics and Control (JRC)*, **4**(1), 12–22.

Awadallah, M. A., Hammouri, A. I., Al-Betar, M. A., Braik, M. S., & Abd Elaziz, M. (2022). Binary horse herd optimization algorithm with crossover operators for feature selection. *Computers in Biology and Medicine*, **141**, 105152.

Bhadoria, A., & Kamboj, V. K. (2019). Optimal generation scheduling and dispatch of thermal generating units considering impact of wind penetration using HGWO-RES algorithm. *Applied Intelligence*, **49**(4), 1517–1547.

Chakraborty, S., Saha, A. K., Nama, S., & Debnath, S. (2021a). COVID-19 X-ray image segmentation by modified whale optimization algorithm with population reduction. *Computers in Biology and Medicine*, **139**, 104984.

Chakraborty, S., Saha, A. K., Sharma, S., Chakraborty, R., & Debnath, S. (2023). A hybrid whale optimization algorithm for global optimization. *Journal of Ambient Intelligence and Humanized Computing*, **14**(1), 431–467.

Chakraborty, S., Saha, A. K., Sharma, S., Mirjalili, S., & Chakraborty, R. (2021b). A novel enhanced whale optimization algorithm for global optimization. *Computers & Industrial Engineering*, **153**, 107086.

Chen, H., Li, W., & Yang, X. (2020). A whale optimization algorithm with chaos mechanism based on quasi-opposition for global optimization problems. *Expert Systems with Applications*, **158**, 113612.

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

Ewees, A. A., Mostafa, R. R., Ghoniem, R. M., & Gaheen, M. A. (2022). Improved seagull optimization algorithm using Lévy flight and mutation operator for feature selection. *Neural Computing and Applications*, **34**(10), 7437–7472.

Fan, Q., Chen, Z., Zhang, W., & Fang, X. (2020). ESSAWOA: Enhanced whale optimization algorithm integrated with salp swarm algorithm for global optimization. *Engineering with Computers*, **38**, 1–18.

Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, **9**(2), 159–195.

Hashim, F. A., & Hussien, A. G. (2022). Snake optimizer: A novel meta-heuristic optimization algorithm. *Knowledge-Based Systems*, **242**, 108320.

Hashim, F. A., Mostafa, R. R., Hussien, A. G., Mirjalili, S., & Sallam, K. M. (2023). Fick's law algorithm: A physical law-based algorithm for numerical optimization. *Knowledge-Based Systems*, **260**, 110146.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, **97**, 849–872.

Hongwei, L., Jianyong, L., Liang, C., Jingbo, B., Yangyang, S., & Kai, L. (2019). Chaos-enhanced moth-flame optimization algorithm for global optimization. *Journal of Systems Engineering and Electronics*, **30**(6), 1144–1159.

Houssein, E. H., Hashim, F. A., Ferahtia, S., & Rezk, H. (2022a). Battery parameter identification strategy based on modified COOT optimization algorithm. *Journal of Energy Storage*, **46**, 103848.

Houssein, E. H., Rezk, H., Fathy, A., Mahdy, M. A., & Nassef, A. M. (2022b). A modified adaptive guided differential evolution algorithm applied to engineering applications. *Engineering Applications of Artificial Intelligence*, **113**, 104920.

Hussien, A. G., Hashim, F. A., Qaddoura, R., Abualigah, L., & Pop, A. (2022). An enhanced evaporation rate water-cycle algorithm for global optimization. *Processes*, **10**(11), 2254.

Kamel, S., Houssein, E. H., Hassan, M. H., Shouran, M., & Hashim, F. A. (2022). An efficient electric charged particles optimization algorithm for numerical optimization and optimal estimation of photovoltaic models. *Mathematics*, **10**(6), 913.

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, **80**, 8091–8126.

Kaveh, A., Talatahari, S., & Khodadadi, N. (2020). Stochastic paint optimizer: Theory and application in civil engineering. *Engineering with Computers*, **38**, 1–32.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*(Vol. **4**, pp. 1942–1948). IEEE.

Khasanov, M., Kamel, S., Houssein, E. H., Rahmann, C., & Hashim, F. A. (2023). Optimal allocation strategy of photovoltaic-and wind turbine-based distributed generation units in radial distribution networks considering uncertainty. *Neural Computing and Applications*, **35**(3), 2883–2908.

Khurma, R. A., Aljarah, I., Castillo, P. A., & Sabri, K. E. (2022). An enhanced opposition-based evolutionary feature selection approach. In *Proceedings of the Applications of Evolutionary Computation: 25th European Conference, EvoApplications 2022, Held as Part of EvoStar 2022*(pp. 3–14). Springer.

Khurma, R. A., Aljarah, I., & Sharieh, A. (2020a). An efficient moth flame optimization algorithm using chaotic maps for feature selection in the medical applications. In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*(pp. 175–182).

Khurma, R. A., Aljarah, I., & Sharieh, A. (2020b). Rank based moth flame optimisation for feature selection in the medical application. In *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*(pp. 1–8). IEEE.

Khurma, R. A., Aljarah, I., & Sharieh, A. (2021). A simultaneous moth flame optimizer feature selection approach based on levy flight and selection operators for medical diagnosis. *Arabian Journal for Science and Engineering*, **46**, 8415–8440.

Khurma, R. A., Aljarah, I., Sharieh, A., & Mirjalili, S. (2020c). EvoloPy-FS: An open-source nature-inspired optimization framework in python for feature selection. In *Evolutionary machine learning techniques: Algorithms and applications*(pp. 131–173). Springer.

Kumar, A., Wu, G., Ali, M. Z., Mallipeddi, R., Suganthan, P. N., & Das, S. (2020). A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm and Evolutionary Computation*, **56**, 100693.

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*, **111**, 300–323.

Marechal, F., & Kalitventzeff, B. (2001). A tool for optimal synthesis of industrial refrigeration systems. In *Computer aided chemical engineering*(Vol. **9**, pp. 457–462). Elsevier.

Matoušová, I., Trojovskỳ, P., Dehghani, M., Trojovská, E., & Kostra, J. (2023). Mother optimization algorithm: A new human-based metaheuristic approach for solving engineering optimization. *Scientific Reports*, **13**(1), 10312.

Morales-Castañeda, B., Zaldivar, D., Cuevas, E., Fausto, F., & Rodríguez, A. (2020). A better balance in metaheuristic algorithms: Does it exist?. *Swarm and Evolutionary Computation*, **54**, 100671.

Mostafa, R. R., Ewees, A. A., Ghoniem, R. M., Abualigah, L., & Hashim, F. A. (2022). Boosting chameleon swarm algorithm with consumption AEO operator for global optimization and feature selection. *Knowledge-Based Systems*, **246**, 108743.

Mostafa, R. R., Gaheen, M. A., Abd ElAziz, M., Al-Betar, M. A., & Ewees, A. A. (2023). An improved gorilla troops optimizer for global optimization problems and feature selection. *Knowledge-Based Systems*, **269**, 110462.

Mousavirad, S. J., & Ebrahimpour-Komleh, H. (2017). Human mental search: A new population-based metaheuristic optimization algorithm. *Applied Intelligence*, **47**, 850–887.

Nadimi-Shahraki, M. H., Taghian, S., & Mirjalili, S. (2021). An improved grey wolf optimizer for solving engineering problems. *Expert Systems with Applications*, **166**, 113917.

Osaba, E., Villar-Rodriguez, E., Del Ser, J., Nebro, A. J., Molina, D., LaTorre, A., Suganthan, P. N., Coello, C. A. C., & Herrera, F. (2021). A tutorial on the design, experimentation and application of metaheuristic algorithms to real-world optimization problems. *Swarm and Evolutionary Computation*, **64**, 100888.

Pierre, D. A. (1986). *Optimization theory with applications*. Courier Corporation.

Piri, J., & Mohapatra, P. (2021). An analytical study of modified multiobjective Harris hawk optimizer towards medical data feature selection. *Computers in Biology and Medicine*, **135**, 104558.

Qaddoura, R., Aljarah, I., Faris, H., & Mirjalili, S. (2021). A grey wolfbased clustering algorithm for medical diagnosis problems. In *Evolutionary data clustering: Algorithms and applications*(pp. 73–87). Springer.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2011). Teaching–learningbased optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, **43**(3), 303–315.

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

Saha, A. K. (2022). Multi-population-based adaptive sine cosine algorithm with modified mutualism strategy for global optimization. *Knowledge-Based Systems*, **251**, 109326.

Salawudeen, A. T., Mu'azu, M. B., Yusuf, A., & Adedokun, A. E. (2021). A novel smell agent optimization (SAO): An extensive CEC study and engineering application. *Knowledge-Based Systems*, **232**, 107486.

Sharma, S., Chakraborty, S., Saha, A. K., Nama, S., & Sahoo, S. K. (2022). MLBOA: A modified butterfly optimization algorithm with Lagrange interpolation for global optimization. *Journal of Bionic Engineering*, **19**(4), 1161–1176.

Sheikhi Azqandi, M., Delavar, M., & Arjmand, M. (2020). An enhanced time evolutionary optimization for solving engineering design problems. *Engineering with Computers*, **36**(2), 763–781.

Smith, J. M. (1978). Optimization theory in evolution. *Annual Review of Ecology and Systematics*, **9**(1), 31–56.

Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. *Neurocomputing*, **532**, 183–214.

Sun, S., Cao, Z., Zhu, H., & Zhao, J. (2019). A survey of optimization methods from a machine learning perspective. *IEEE Transactions on Cybernetics*, **50**(8), 3668–3681.

Thawkar, S., Sharma, S., Khanna, M., & Singh, L. K. (2021). Breast cancer prediction using a hybrid method based on butterfly optimization algorithm and ant lion optimizer. *Computers in Biology and Medicine*, **139**, 104968.

Tolba, M. A., Houssein, E. H., Eisa, A. A., & Hashim, F. A. (2022). Optimizing the distributed generators integration in electrical distribution networks: Efficient modified forensic-based investigation. *Neural Computing and Applications*, **35**, 1–36.

Tu, J., Chen, H., Wang, M., & Gandomi, A. H. (2021). The colony predation algorithm. *Journal of Bionic Engineering*, **18**, 674–710.

Wang, Z., Luo, Q., & Zhou, Y. (2021). Hybrid metaheuristic algorithm using butterfly and flower pollination base on mutualism mechanism for global optimization problems. *Engineering with Computers*, **37**, 3665–3698.

Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Breakthroughs in statistics*(pp. 196–202). Springer.

Xing, J., Zhao, H., Chen, H., Deng, R., & Xiao, L. (2023). Boosting whale optimizer with quasi-oppositional learning and Gaussian barebone for feature selection and COVID-19 image segmentation. *Journal of Bionic Engineering*, **20**(2), 797–818.

Yang, Y., Chen, H., Heidari, A. A., & Gandomi, A. H. (2021). Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Systems with Applications*, **177**, 114864.

Yıldız, B. S., Mehta, P., Panagant, N., Mirjalili, S., & Yildiz, A. R. (2022). A novel chaotic Runge Kutta optimization algorithm for solving constrained engineering problems. *Journal of Computational Design and Engineering*, **9**(6), 2452–2465.

Zamani, H., Nadimi-Shahraki, M. H., & Gandomi, A. H. (2019). CCSA: Conscious neighborhood-based crow search algorithm for solving global optimization problems. *Applied Soft Computing*, **85**, 105583.

Zhang, Y., Chi, A., & Mirjalili, S. (2021). Enhanced JAYA algorithm: A simple but efficient optimization method for constrained engineering design problems. *Knowledge-Based Systems*, **233**, 107555.

Zhang, X., Xu, Y., Yu, C., Heidari, A. A., Li, S., Chen, H., & Li, C. (2020). Gaussian mutational chaotic fruit fly-built optimization and feature selection. *Expert Systems with Applications*, **141**, 112976.

Zhao, S., Zhang, T., Ma, S., & Chen, M. (2022a). Dandelion optimizer: A nature-inspired metaheuristic algorithm for engineering applications. *Engineering Applications of Artificial Intelligence*, **114**, 105075.

Zhao, S., Zhang, T., Ma, S., & Wang, M. (2022b). Sea-horse optimizer: A novel nature-inspired meta-heuristic for global optimization problems. *Applied Intelligence*, **53**, 1–28.

Received: April 22, 2023. **Revised:** November 13, 2023. **Accepted:** November 13, 2023