Artificial Intelligence Review (2024) 57:139
https://doi.org/10.1007/s10462-024-10767-6

Learning search algorithm: framework and comprehensive performance for solving optimization problems

Chiwen Qu[1] · **Xiaoning Peng**[2] · **Qilan Zeng**[3]

Accepted: 18 April 2024

Abstract
In this study, the Learning Search Algorithm (LSA) is introduced as an innovative optimization algorithm that draws inspiration from swarm intelligence principles and mimics the social learning behavior observed in humans. The LSA algorithm optimizes the search process by integrating historical experience and real-time social information, enabling it to effectively navigate complex problem spaces. By doing so, it enhances its global development capability and provides efficient solutions to challenging optimization tasks. Additionally, the algorithm improves the collective learning capacity by incorporating teaching and active learning behaviors within the population, leading to improved local development capabilities. Furthermore, a dynamic adaptive control factor is utilized to regulate the algorithm's global exploration and local development abilities. The proposed algorithm is rigorously evaluated using 40 benchmark test functions from IEEE CEC 2014 and CEC 2020, and compared against nine established evolutionary algorithms as well as 11 recently improved algorithms. The experimental results demonstrate the superiority of the LSA algorithm, as it achieves the top rank in the Friedman rank-sum test, highlighting its power and competitiveness. Moreover, the LSA algorithm is successfully applied to solve six real-world engineering problems and 15 UCI datasets of feature selection problems, showcasing its significant advantages and potential for practical applications in engineering problems and feature selection problems.

Keywords Learning search algorithm · Swarm intelligence · Optimization · Feature selection

✉ Qilan Zeng
zengqilanzql@163.com

Chiwen Qu
quchiwen@163.com

Xiaoning Peng
pxiaoning@hunnu.edu.cn

[1] Public Health and Management Institute, Youjiang Medical University for Nationalities, Baise 533000, China

[2] Department of Pathology and Pathophysiology, Hunan Normal University School of Medicine, Hunan Normal University, Changsha 410081, China

[3] School of Civil Engineering, Chongqing Jiaotong University, Chongqing 400074, China

Published online: 09 May 2024

1 Introduction

All production or social activities that human beings are engaged in are purposeful. The activity is always under the control of specific values or aesthetic orientations, and it often faces a decision problem of the feasible or even optimal scheme, i.e., an optimization problem (Martello et al. 1984). In recent years, the importance of optimization in engineering design, disease identification, and other issues has been recognized. Specifically, taking the objective function as the predefined measure of decision quality, the best decision or solution to predefined design problems can be obtained by evaluating various methods. Optimization is a problem that people often encounter in the production practice of scientific research and social transformation. It has the characteristics of unknown search space, non-differentiability of the objective function, high-dimensional, and non-convex (Ezugwu 2022). Generally, optimization techniques can be roughly divided into deterministic and non-deterministic ones (Parsopoulos and Vrahatis 2002). Deterministic methods are usually gradient-based and are further divided into linear and nonlinear ones. Although these methods help to solve linear and nonlinear optimization problems, they will fall into local optimization when dealing with non-differentiable optimization problems, so they cannot solve such problems in the minimum time or with accurate complexity. The non-deterministic method uses a random generation strategy to find the near-optimal solution in the problem space, which has the advantage of simple implementation and no gradient-related information (Li et al. 2011; Liu et al. 2013).

With the continuous expansion of engineering application fields and the continuous improvement of the complexity and difficulty of optimization problems, the need for optimization technology is becoming more and more obvious. As a class of non-deterministic methods (random search methods), meta-heuristic algorithms have demonstrated excellent performance when tackling challenges involving multiple-peak, discontinuous, and non-differentiable problems. Therefore, meta-heuristic algorithms have gained significant popularity in efficiently tackling diverse practical optimization problems across numerous fields, such as function optimization (Seo et al. 2006; Pan et al. 2006), pipeline scheduling (Rejowski and Pinto 2003; Li et al. 2021), optimization of neural network parameters (Abdolrasol et al. 2021; Abd Elaziz et al. 2021), key gene identification (Mandal and Mukhopadhyay 2015), image segmentation (Chander et al. 2011; Pham et al. 2018), parameter identification of photovoltaic module models (Ibrahim et al. 2020; Liang et al. 2020), optimization of engineering design, etc. (Kundu and Garg 2022; Onay and Aydemir, S. B. 2022).

In terms of formation principle, meta-heuristic methods can be categorized into four distinct groups, each offering unique approaches to solving optimization problems encountered in various disciplines: the methods based on the evolutionary mechanism, the methods based on physical principles, the methods based on swarm intelligence, and the methods based on human social activities (Sharma et al. 2022; Wilson et al. 2022; Tian et al. 2022; Ewees et al. 2022). Section 2 offers an extensive compilation of the comprehensive literature on the development and application of numerous novel metaheuristics across various domains.

Meanwhile, many scholars have optimized the original basic algorithm to solve the optimization problems of real-world engineering applications more efficiently. For example, due to the large randomness and uncertainty of the randomly generated initial population, many scholars have improved the initial population by using chaos mapping (Tutueva et al. 2020), reverse learning (Ruan et al. 2022), Sobol sequence

(Sun et al. 2021), square neighborhood topology (Rachdi et al. 2020), and other strategies to achieve algorithm optimization and convergence performance improvement. Some scholars have adopted strategies such as sine–cosine optimization (Chen et al. 2020a), Gaussian walk (Khalilpourazari et al. 2021), Levy flight (Kaidi et al. 2022), and quantum behavior (Zhang et al. 2021a) to optimize individual iterative updates. Also, nonlinear inertia weight (Yu et al. 2020), horizontal cross (Chen et al. 2019), spiral update (Lin et al. 2022), and other approaches have been employed to achieve the balance between the global and local development of the algorithm. Moreover, some scholars have combined the advantages of two or more algorithms and proposed an improved hybrid strategy algorithm (Shi et al. 2005; Yaghini et al. 2013; Qiao et al. 2020), such as the exploratory cuckoo search (ECS) algorithm proposed by Abed-Alguni et al. (Abed-Alguni et al. 2021) and the improved SSA algorithm proposed by Dorian (Dorian Sidea 2024).

In metaheuristic algorithms, the exploration phase of the global search space has the ability to escape local optima, while the exploitation phase enhances the algorithm's precise search capability within local regions. Balancing exploration and exploitation is a challenging task in every metaheuristic algorithm, as it affects whether the algorithm can find the global optimum solution. In general, metaheuristic algorithms offer different performances in solving optimization problems due to their different operations and mechanisms. According to the "No Free Lunch" (NFL) theorem (Qiao et al. 2020), all metaheuristic algorithms have the same average performance when solving optimization problems. In other words, there is no optimal algorithm that can solve all optimization problems, which means that the performance of different algorithms varies when providing prior knowledge to solve specific problems with metaheuristic algorithms. Finding the most suitable algorithm for each specific type of optimization problem remains a challenge. Each metaheuristic algorithm has its unique characteristics as they draw inspiration from different natural or biological behaviors. To evaluate performance and find suitable application areas, metaheuristic algorithms require comprehensive testing on various benchmark functions and real-world applications, and continual improvement. These reasons support the innovation and design of metaheuristic algorithms to solve a wide range of optimization problems.

This paper proposes a novel optimization algorithm, called the learning search algorithm (LSA), which is inspired by human learning behaviors and promotes both global exploration and local development phases simultaneously. The LSA algorithm involves dynamic adaptive global exploration to local development phase control parameters, which enhance its global search ability and avoid falling into local optima. In the local development phase, the algorithm exploits the teaching behavior of the model in the current population to actively learn behaviors from role models and improve the learning ability of the entire population. The proposed LSA algorithm is evaluated on 40 benchmark functions from IEEE CEC2014 and CEC2020, 6 real-world engineering optimization problems and 15 feature selection cases in the UCI dataset. Contrasted with nine high-performance algorithms and eleven recently proposed algorithms, the LSA algorithm shows promising results in terms of convergence speed, search accuracy, and scalability. The experiment results suggest that the LSA algorithm outperforms the selected comparison algorithms on most of the selected test problems. The statistical analysis further confirms the proposed algorithm's superiority by conducting the Wilcoxon signed-rank test and the Friedman rank-sum test.

The paper presents several significant contributions:

- In terms of learning mechanism, the LSA algorithm simulates the process of human knowledge acquisition. In this algorithm, a historical experience knowledge base is established, from which humans learn knowledge. Humans also possess the ability of active learning and knowledge acquisition from exemplary radiations. Additionally, the learning outcomes continuously update the historical experience knowledge base.
- Regarding its adaptability, the LSA algorithm exhibits a high level of adaptability. In each iteration process, knowledge and information flow constantly between the historical experience knowledge base and the current individuals. This enables the LSA algorithm to adaptively adjust the search strategy based on the complexity and characteristics of the problem, thereby enhancing the search efficiency and solution quality.
- The concept of idea transmission is another important aspect of the LSA algorithm. It improves the solutions of individuals through the transmission of ideas. The algorithm transfers excellent search ideas to learning individuals based on historical experiences and the most outstanding solutions of the population.
- Furthermore, the LSA algorithm possesses interpretability and ease of implementation. Its ideas are relatively simple and intuitive, making it easy to understand and implement. The role switch between individuals and the process of knowledge transmission can be explained and analyzed, allowing users of the algorithm to better understand the optimization process.

The remaining sections of this paper are organized as follows: In Section 2, an overview of the literature on metaheuristic algorithms is provided. Section 3 provides a detailed introduction to the principle and mathematical model of the LSA algorithm. In Section 4, comprehensive experiments are conducted to demonstrate the superiority of the LSA algorithm over comparative optimization algorithms. Finally, Section 5 presents the conclusions of this paper.

2 Literature review

This section provides an overview of the current advancements in metaheuristics. In recent times, numerous metaheuristic algorithms have been introduced and extensively studied. These algorithms primarily fall into four categories: (1) swarm-based algorithms that emulate swarm intelligence, (2) evolutionary-based algorithms that draw inspiration from natural evolutionary processes, (3) physics or chemistry-based algorithms that are inspired by physical or chemical phenomena, and (4) social or human-based algorithms that are influenced by human or social behaviors. Table 1 presents a compilation of notable and recently developed metaheuristic algorithms.

Natural evolution algorithms are developed from biological phenomena such as natural evolution, and the representative algorithm is the GA algorithm (Mirjalili 2019); Other algorithms include the differential evolution (DE) algorithm (Das and Suganthan 2010), evolution strategy (ES) (Schwefel and Rudolph 1995), memetic algorithm (MA) (Moscato et al. 2004), and genetic programming (GP) (Sette and Boullart 2001). Swarm intelligence optimization algorithms, which simulate the social behavior of animals based on group foraging behaviors, have attracted increasing attention. Notable algorithms in this domain include the particle swarm optimization (PSO) algorithm (Poli et al. 2007), the crow search algorithm (CSA) (Askarzadeh 2016), cuckoo search (CS) algorithm (Yang and Deb 2014), the social spider algorithm (SSA) (James and Li

Table 1 The well-known metaheuristic algorithms proposed in the past decade

Author	Algorithm	Year	Category
Mirjalili (Mirjalili 2015a)	Ant Lion Optimizer(ALO)	2015	Swarm
James et al. (James and Li 2015)	Social Spider Algorithm(SSA)	2015	Swarm
Wang et al. (Wang et al. 2019)	Monarch Butterfly Optimization(MBO)	2015	Swarm
Salimi (Salimi 2015)	Stochastic Fractal Search(SFS)	2015	Physics
Askarzadeh et al. (Askarzadeh 2016)	Crow Search Algorithm(CSA)	2016	Swarm
Yazdani et al. (Yazdani and Jolai 2016)	Lion Optimization Algorithm (LOA)	2016	Swarm
Kaveh et al. (Kaveh and Bakhshpoori 2016)	Water Evaporation Optimization(WEO)	2016	Physics
Mirjalili et al. (Dehghani et al. 2022b)	Whale Optimization Algorithm(WOA)	2016	Swarm
Mirjalili (Chickermane and Gea 1996)	Sine Cosine Algorithm(SCA)	2016	Physics
Mirjalili et al. (Mirjalili et al. 2017)	Salp Swarm Algorithm(SSA)	2017	Swarm
Saremi et al. (Saremi et al. 2017)	Grasshopper optimisation algorithm(GOA)	2017	Swarm
Nematollahi et al. (Nematollahi et al. 2017)	Lightning Attachment Procedure Optimization(LAPO)	2017	Physics
Saremi et al. (Saremi et al. 2017)	Grasshopper Optimization Algorithm(GOA)	2017	Swarm
Kaveh et al. (McFarland et al. 1993)	Thermal Exchange Optimization(TEO)	2017	Physics
Kallioras et al. (Kallioras et al. 2018)	Pity Beetle Algorithm(PBA)	2018	Swarm
Jain et al. (Jain et al. 2019)	Squirrel Search Algorithm(SSA)	2018	Swarm
Dhanya et al. (Dhanya and Arivudainambi 2019)	Dolphin Partner Optimization(DPO)	2019	Swarm
Kılkış et al. (Kılkış and Kılkış 2019)	human urbanization algorithm (HUA)	2019	human
Ahmia et al. (Ahmia and Aider 2019)	monarchy meta-heuristic (MN) optimization algorithm	2019	human
Brammya et al. (Brammya et al. 2019)	deer hunting optimization algorithm (DHOA)	2019	human
Arora and Singh (Arora and Singh 2019)	Butterfly Optimization Algorithm(BOA)	2019	Swarm
Hashim et al. (Hashim et al. 2019)	Henry Gas Solubility Optimization(HGSO)	2019	Physics
Dhiman et al. (Dhiman and Kumar 2019)	Seagull Optimization Algorithm(SOA)	2019	Swarm
Masadeh et al. (Masadeh et al. 2019)	Sea Lion Optimization Algorithm(SLOA)	2019	Swarm
Xue et al. (Xue and Shen 2020)	Sparrow Search Algorithm(SSA)	2020	Swarm
Zervoudakis et al. (Zervoudakis and Tsafarakis 2020)	Mayfly Optimization Algorithm(MOA)	2020	Swarm
Kaveh et al. (Kaveh et al. 2020a)	Black Hole Mechanics Optimization(BHMO)	2020	Physics

Table 1 (continued)

Author	Algorithm	Year	Category
Kaveh et al. (Kaveh et al. 2020b)	Plasma generation optimization(PGO)	2020	Physics
Zitouni et al. (Zitouni et al. 2020)	Solar System Algorithm(SSA)	2020	Physics
Li et al. (Li et al. 2020a)	Atomic Search Algorithm(ASO)	2020	Physics
Askari et al. (Askari et al. 2020)	Heap-based Optimization (HBO)	2020	Human
Hayyolalam et al. (Hayyolalam and Kazem 2020)	Black Widow Optimization Algorithm(BWOA)	2020	Unknown
Nematollahi et al. (Nematollahi et al. 2020)	Golden Ratio Optimization Method (GROM)	2020	Unknown
Li et al. (Li et al. 2020b)	virus propagation optimization (VSO)	2020	Unknown
Alsattar et al. (Alsattar et al. 2020)	Bald Eagle search (BES) Algorithm	2020	Unknown
Braik et al. (Braik et al. 2021)	Capuchin Search Algorithm (CSA)	2021	Swarm
Połap et al. (Połap and Woźniak 2021)	Red Fox Optimization (RFO) Algorithm	2021	Swarm
Peraza-Vázquez et al. (Peraza-Vázquez et al. 2021)	Dingoes Hunting Strategies (DHS)	2021	Swarm
Braik et al. (Braik 2021)	Chameleon Swarm Algorithm (CSA)	2021	Swarm
Feng et al. (Feng et al. 2021)	Cooperative Search Algorithm (CSA)	2021	Human
Abualigah et al. (Abualigah et al. 2021)	Arithmetical Optimization Algorithm (AOA)	2021	Human
Zitouni et al. (Zitouni et al. 2021)	Archerfish Hunting Optimizer(AHO)	2021	Swarm
Meng et al. (Meng et al. 2021)	Carnivorous Plant Algorithm(CPA)	2021	Swarm
Mohammadi-Balani et al. (Mohammadi-Balani et al. 2021)	Golden Eagle Optimizer(GEO)	2021	Swarm
Carreon-Ortiz et al. (Carreon-Ortiz and Valdez 2022)	Mycorrhiza Tree Optimization Algorithm (MTOA)	2022	Swarm
Naruei et al. (Naruei and Keynia 2022)	Wild Horse Optimizer (WHO)	2022	Swarm
Ramshanker et al. (Ramshanker and Chakraborty 2022)	Skill Optimization Algorithm (SOA)	2022	Human
Emami et al. (Emami 2022)	Stock Exchange Trading Optimization (SETO)	2022	Human
Ahmadianfar et al. (Ahmadianfar et al. 2022)	Weighted mean of vectors algorithm(INFO)	2022	Physics
Pan et al. (Pan et al. 2023)	Gannet Optimization Algorithm (GOA)	2022	Swarm
Trojovská et al. (Trojovská et al. 2022)	Zebra Optimization Algorithm (ZOA)	2022	Swarm
Dehghani et al. (Dehghani et al. 2022a)	Tasmanian Devil Optimization (TDO)	2022	Swarm
Dehghani et al. (Dehghani et al. 2023)	Coati Optimization Algorithm (COA)	2022	Swarm

Table 1 (continued)

Author	Algorithm	Year	Category
Wang et al. (Wang et al. 2022a)	Artificial rabbits optimization (RSO)	2022	Swarm
Xue et al. (Xue and Shen 2023)	Dung beetle optimizer (DBO)	2022	Swarm
Ali et al. (Ali et al. 2023)	Peafowl Optimization Algorithm (POA)	2022	Swarm
Zhong et al. (Zhong et al. 2022)	Beluga whale optimization (BWO)	2022	Swarm
Agushaka et al. (Agushaka et al. 2022)	Dwarf Mongoose Optimization (DMO)	2022	Swarm
Seyyedabbasi et al. (Seyyedabbasi and Kiani 2023)	Sand Cat Swarm Optimization (SCSO)	2022	Swarm
Chopra et al. (Chopra and Ansari 2022)	Golden jackal optimization (GJO)	2022	Swarm
Trojovský et al. (Trojovský and Dehghani 2022)	Pelican Optimization Algorithm (POA)	2022	Swarm
Dehghani et al. (Dehghani et al. 2021)	Northern Goshawk Optimization (NGO)	2022	Swarm
Hashim et al. (Hashim and Hussien 2022)	Snake Optimizer (SO)	2022	Swarm
Naruei et al. (Naruei et al. 2022)	Hunter-prey optimization (HPO)	2022	Swarm
Dehghani et al. (Dehghani et al. 2022b)	Driving Training-Based Optimization (DTBO)	2022	Human
Trojovská et al. (Trojovská and Dehghani 2022)	Chef-Based Optimization Algorithm (CBOA)	2022	Human
Ayyarao et al. (Ayyarao et al. 2022)	War Strategy Optimization Algorithm (WSO)	2022	Human
Ayyarao et al. (Ayyarao et al. 2022)	Exponential distribution optimizer(EDO)	2023	Physics
Trojovský et al. (Trojovský and Dehghani 2023)	Subtraction-Average-Based Optimizer(SABO)	2023	Physics
Dehghani et al. (Dehghani and Trojovský 2023)	Osprey Optimization Algorithm (OOA)	2023	Swarm
Abdel-Basset et al. (Abdel-Basset et al. 2023)	Exponential distribution optimizer (EDO)	2023	Swarm

2015), the sparrow search algorithm (SSA) (Xue and Shen 2020), the red fox optimization (RFO) algorithm (Połap and Woźniak 2021), the salp swarm algorithm (SSA) (Mirjalili et al. 2017), dolphin partner optimization (DPO) (Dhanya and Arivudainambi 2019), Lion Optimization Algorithm (LOA) (Yazdani and Jolai 2016), dingoes hunting strategies (DHS) (Peraza-Vázquez et al. 2021), mycorrhiza tree optimization algorithm (MTOA) (Carreon-Ortiz and Valdez 2022), charged system search (CSS) (Kaveh and Talatahari 2010), chameleon swarm algorithm (CSA) (Braik 2021), wild horse optimizer (WHO) (Naruei and Keynia 2022), mayfly optimization algorithm (MOA) (Zervoudakis and Tsafarakis 2020), capuchin search (CSA) (Braik et al. 2021), Zebra Optimization Algorithm (ZOA) (Trojovská et al. 2022), Tasmanian Devil Optimization (TDO) (Dehghani et al. 2022a), Artificial rabbits optimization (RSO) (Wang et al. 2022a), Osprey Optimization Algorithm (OOA) (Dehghani and Trojovský 2023), Exponential distribution optimizer (EDO) (Abdel-Basset et al. 2023), and others. These algorithms have demonstrated promising performance in solving various complex optimization problems. Besides, there is a class of physical search algorithms based on the simulation of physical phenomena, such as the simulated annealing (SA) (Bertsimas and Tsitsiklis 1993), gravitational search algorithm (GSA) (Saremi et al. 2017), curved space optimization (CSO) (Moghaddam and Moghaddam 2012), lighting attachment procedure optimization (LAPO) (Nematollahi et al. 2017), black hole mechanics optimization (BHMO) (Kaveh et al. 2020a), plasma generation optimization (PGO) (Kaveh et al. 2020b), solid system algorithm (SSA) (Zitouni et al. 2020), atomic search algorithm (ASO) (Li et al. 2020a), Heap-based Optimization (HBO) (Askari et al. 2020), Weighted mean of vectors algorithm(INFO) (Ahmadianfar et al. 2022), Exponential distribution optimizer(EDO) (Ayyarao et al. 2022), Subtraction-Average-Based Optimizer(SABO) (Trojovský and Dehghani 2023), etc. Some intelligent algorithms have been designed based on human social activities, such as the teaching–learning-based optimization (TLBO) (Rao et al. 2012), skill optimization algorithm (SOA) (Ramshanker and Chakraborty 2022), cooperative search algorithm (CSA) (Feng et al. 2021), human urbanization algorithm (HUA) (Kılkış and Kılkış 2019), heap-based optimization (HBO) (Askari et al. 2020), stock exchange trading optimization (SETO) (Emami 2022), arithmetical optimization algorithm (AOA) (Abualigah et al. 2021), Driving Training-Based Optimization (DTBO) (Dehghani et al. 2022b), Chef-Based Optimization Algorithm (CBOA) (Trojovská and Dehghani 2022), War Strategy Optimization Algorithm (WSO), and so on. Additionally, in the past three years, many relatively new meta-heuristic algorithms have been proposed, and they are not classified into the mentioned categories. For example, inspired by the management strategy of the constitutional monarchy government, Ahmia et al., proposed the monarchy meta-heuristic (MN) optimization algorithm (Ahmia and Aider 2019). Brammya et al. utilized a simulation of human deer hunting behavior to propose a deer hunting optimization algorithm (DHOA) (Brammya et al. 2019). In a similar vein, Hayyolalam and Kazem devised a black widow optimization (BWO) algorithm (Hayyolalam and Kazem 2020), drawing inspiration from the mating behavior of black widow spiders. Nematollahi et al. introduced the Golden Ratio Optimization Method (GROM) as an optimization approach (Nematollahi et al. 2020). Li et al. developed the virus propagation optimization (VSO) algorithm (Li et al. 2020b), which simulates the propagation process of the virus. Alsattar et al. proposed the bald eagle search (BES) algorithm (Alsattar et al. 2020) based on the hunting process of the bald eagle.

3 Learning search algorithm

This section provides the details and optimization procedure of the proposed learning search algorithm (LSA). The algorithm is inspired by human learning behaviors in the social environment, including the global learning behavior guided by historical experience and other social individuals, and the local learning behavior guided by role models. The analysis of the mathematical model and the realization process and time complexity of LSA is presented below.

3.1 The basics of MHSs and the proposed LSA method

The general framework of meta-heuristic search algorithms (MHSs) typically consists of three essential components: the selection guides mechanism, the search operators design, and the update mechanism design, as demonstrated in Fig. 1 (Kahraman et al. 2023).

The process of selecting candidate solutions from a population to guide the search process is a fundamental aspect of the MHS algorithm. Various methods exist for guiding this selection (Fig. 1), but the dominant approach is currently the survival theorem, which

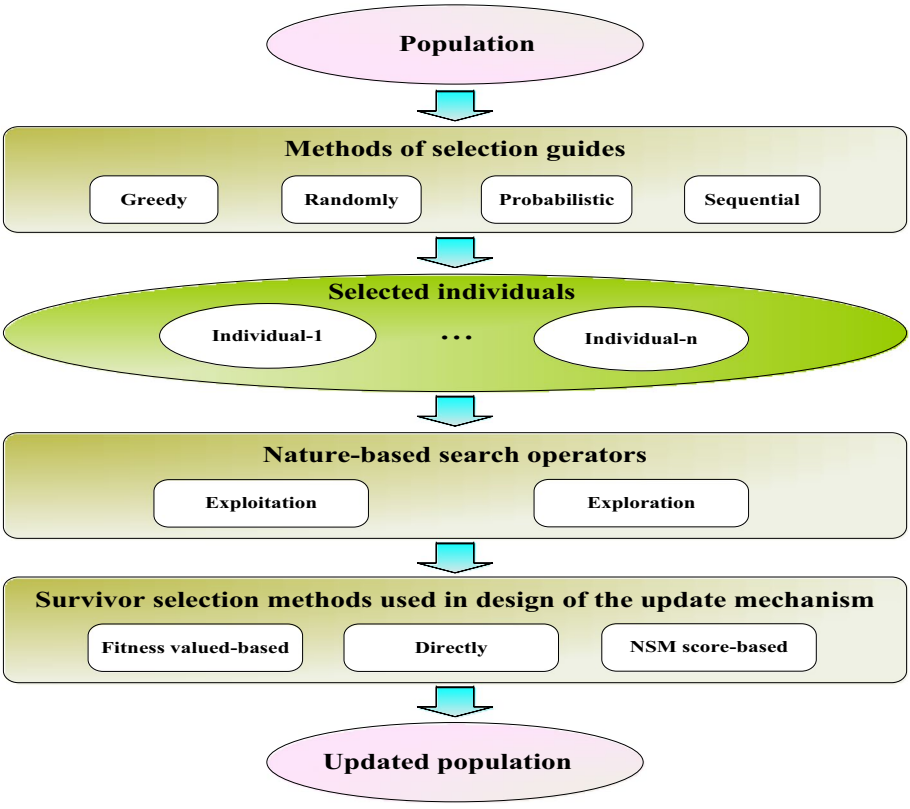

Fig. 1 General steps involved in the MHS process

compares the fitness values of individuals within the population (Forrest 1993; Holland 1992). More recently, the Fitness-Distance Balance (FDB) has emerged as a promising new method for guiding selection in the MHS algorithm (Kahraman et al. 2022; Guvenc et al. 2021; Duman et al. 2023). Selecting excellent individuals as guidance is a critical step in MHS algorithms. Reasonable selection of individuals, balancing diversity and convergence, directly affects the efficiency and quality of search results. The driving force behind human progress is the ability to learn different perspectives and interpretations from different historical periods, cultivating critical thinking and analytical skills. By comparing and evaluating different historical events and interpretations, people can gain a better understanding of the complexity and diversity of history and draw their own conclusions. Additionally, role models serve as symbols of successful experiences, having achieved excellence in a particular field or skill. Humans can use the behavior, thinking, and decision-making of role models to guide the application of knowledge, avoiding some common mistakes and dilemmas. Following the mechanism of MHS algorithms, we can use historical experience and role models as guidance leaders in the search process, emphasizing the balance between diversity and convergence.

The design of search operators is a crucial element of MHS algorithms as it shapes the models simulating the distinct behaviors and survival skills unique to each population. The search operators for various MHS algorithms differ, including genetic-based crossover and mutation operators (Chen et al. 2020a; Mirjalili 2019; Das and Suganthan 2010; Schwefel and Rudolph 1995; Holland 1992), operators based on swarm foraging behavior (Poli et al. 2007; Askarzadeh 2016; Yang and Deb 2014; James and Li 2015; Xue and Shen 2020; Połap and Woźniak 2021), operators based on physical natural phenomena (Bertsimas and Tsitsiklis 1993; Moghaddam and Moghaddam 2012; Li et al. 2020a), and operators based on human social activities (Wilson et al. 2022; Tian et al. 2022; Ewees et al. 2022). A high capacity for summarizing experiences and imitative learning, as well as autonomous learning ability, is why human learning surpasses that of other organisms. The proposed algorithm embodies the subjective autonomy of human learning behavior and the diversity of learning approaches, fully embodying its potential for breakthroughs in MHS.

In MHS algorithms, the majority of update mechanisms employ a greedy approach based on fitness values (Yang and Deb 2014; Carreon-Ortiz and Valdez 2022; Saremi et al. 2017). This approach guarantees a balanced turnover of individuals within the population, ensuring that the introduction of a specific number of new individuals is accompanied by the removal of an equivalent number of existing individuals. An alternative approach for update mechanisms, referred to as the "direct" approach, is depicted in Fig. 1, exemplified by the SCA (Trojovský and Dehghani 2022) and SMA (Li et al. 2020c). In these algorithms, mutated individuals survive at each step of the search process, while previous individuals are eliminated. Furthermore, the NSM score-based approach has also proven to be an efficient method for update mechanisms (Kahraman et al. 2023). Human learning behavior can be characterized as "taking the essence, discarding the dross." Unlike other organisms, humans possess the ability for reflection, critical thinking, and abstract reasoning. They can selectively choose valuable content that aids in personal learning and understanding, assimilating and integrating it into their own knowledge system. Thus, the update mechanism designed for the LSA algorithm adopts a greedy strategy based on fitness values.

3.2 Inspiration

A learning behavior encompasses the acquisition of behaviors by individuals, resulting from a combination of genetic and environmental factors and shaped by life experiences. For human beings, learning is not only an activity of simply adapting to the environment but also has social significance. Therefore, human learning has social characteristics, and this is mainly manifested in its indirect experience and positive initiatives.

Interaction with others allows individuals to acquire knowledge not only from their direct experiences but also from the collective historical experiences of human society. As human culture has evolved, society has accumulated a vast body of knowledge and experience, which has been transmitted through social inheritance. From birth, individuals in human society have the ability to assimilate the wisdom passed down by previous generations through interactions with teachers in educational institutions. Additionally, they also have the opportunity to acquire valuable social experiences through interactions with their contemporaries. This mode of indirect experiential learning is characterized by its rich and diverse content and form, setting it apart from learning processes observed in animals (McFarland et al. 1993; Bennett 2011), as depicted in Fig. 2(a).

Animal learning is primarily an adaptive process driven by environmental factors, making it a passive endeavor. In contrast, human learning encompasses not only a desire to understand the world but also a determination to shape and alter it. Thus, humans engage in active interactions with their surroundings, learning through integration with the individuals they encounter. The purpose of human learning extends beyond merely satisfying physiological needs; it also encompasses the demands of social life. Consequently, humans possess a wide range of learning motivations and objectives. In their pursuit of these objectives, humans actively explore diverse and effective learning

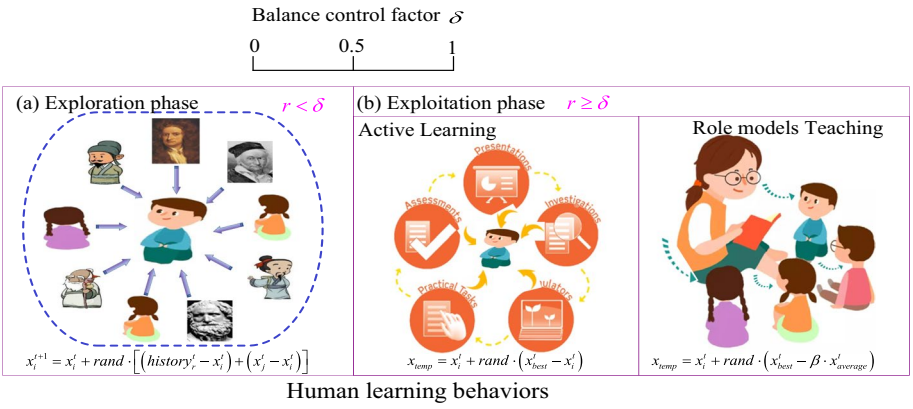

Fig. 2 Depicts human learning behavior through two distinct avenues. Panel (**a**) highlights the role of historical experience and interaction with other individuals in the learning process. Panel (**b**) illustrates how active learning and mentorship from role models also contribute to effective learning outcomes

methods, a capability that surpasses the realm of animal learning (Schoenewolf 1990; Bruner 2009, 1971), exemplified in Fig. 2(b).

Inspired by the behaviors observed in human life and learning, this paper presents the Learning Search Algorithm (LSA) as a groundbreaking meta-heuristic approach. The LSA's mathematical model is outlined below.

3.3 Mathematical model of the proposed algorithm

Human learning exhibits two distinct modes of behavior, characterized by different approaches to acquiring knowledge. One mode involves the utilization of historical experience and interactions with others, enabling a global search process. In this mode, individuals benefit from the indirect aspect of human learning, where accumulated wisdom and collective experiences guide their learning journey. The other mode involves the active participation of individuals, particularly the role model who represents the current optimal individual. This role model not only imparts knowledge to others but also actively engages in learning, thereby facilitating local search within the learning algorithm. This active aspect of human learning contributes to the refinement and fine-tuning of the individual's knowledge. By incorporating these two modes of learning behavior, the Learning Search Algorithm (LSA) enriches and comprehensively expands the overall knowledge of the population. This algorithm integrates the global exploration facilitated by historical experiences and interactions, along with the localized refinement through active learning from the role model. Such integration leads to a synergistic effect, where the collective wisdom accumulated through historical experiences is combined with the adaptability and learning capabilities of individuals. Furthermore, the LSA incorporates autonomous control factor dynamics, ensuring a seamless wide-ranging exploration to precise refinement. This dynamic adaptation mechanism enables the algorithm to strike a balance between exploration and exploitation, allowing for efficient knowledge acquisition and optimization.

3.3.1 Initialization

In this investigation, we utilized a population-based swarm intelligence optimization technique referred to as the Learning Search Algorithm (LSA). LSA is designed to find optimal solutions by iteratively updating the individual candidate solutions within the population. The population's position is modeled using a matrix, as demonstrated in Formula (1):

$$x = \begin{bmatrix} x_{1,1}, x_{1,2}, \ldots, x_{1,\dim} \\ x_{2,1}, x_{2,2}, \ldots, x_{2,\dim} \\ \ldots, \ldots, \ldots, \ldots \\ x_{n,1}, x_{n,2}, \ldots, x_{n,\dim} \end{bmatrix} \tag{1}$$

where, n represents the number of individuals, dim indicates the dimensionality of the search space, and $x_{i,j}$ represents jth dimension of individual i. It is noteworthy that each position is generated through uniform distribution, as illustrated in Formula (2):

$$x_{i,j} = rand(0,1) \cdot (ub_j - lb_j) + lb_j \tag{2}$$

where, $rand(0,1)$ denotes a random number between 0 and 1, while ub_j and lb_j correspond to the upper and lower bound values, respectively.

Formula (3) provides a means of evaluating the fitness score for each individual in the search population. This score serves as a metric to assess their overall level of fitness within the context of the study.

$$f(x) = \begin{bmatrix} f(x_{1,1}, x_{1,2}, ..., x_{1,\dim}) \\ f(x_{2,1}, x_{2,2}, ..., x_{2,\dim}) \\ ... \\ f(x_{n,1}, x_{n,2}, ..., x_{n,\dim}) \end{bmatrix} \tag{3}$$

In the LSA algorithm, the balance control factor δ realizes the conversion from global exploration to fine-tuning in a dynamic and self-adaptive way, and the calculation method is:

$$\delta = \delta_{init} \cdot |y^t| - (\delta_{init} - \delta_{final}) \cdot \tan(\lambda \cdot \frac{t}{t_{\max}}) \tag{4}$$

$$y^t = 1 - \gamma \cdot (y^{t-1})^2 \tag{5}$$

where, the balance factors δ_{init} and δ_{final} refer to the initial and eventual values, respectively, while $t_{\max}$ signifies the maximum iteration count. Additionally, $y^t \in (-1, 1)$ is a chaotic sequence. The multiplication factors λ and γ are defined.

The selection of multiplication factors λ and γ is discussed herein. To ensure that the balance control factor δ remains within the interval (0, 1), we first analyze the selection of various values for γ (see Figs. 3 and 4) . Figure 4 indicates that the requirement is satisfied when $1.4 < \gamma < 2.2$. Building on this observation, we further refine the selection of γ by testing values of 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, and 2.1 on 30 benchmark functions from CEC 2014. Test results show that the choice of γ minimally affects the

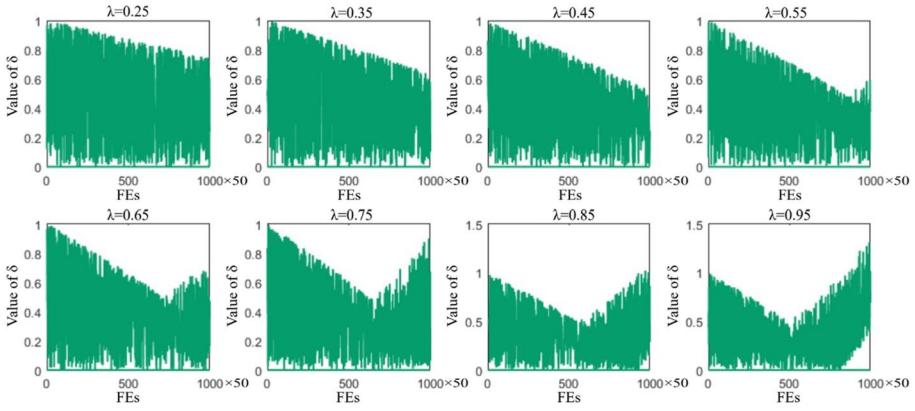

Fig. 3 The range of the balance control factor δ when the multiplication factors λ take different values

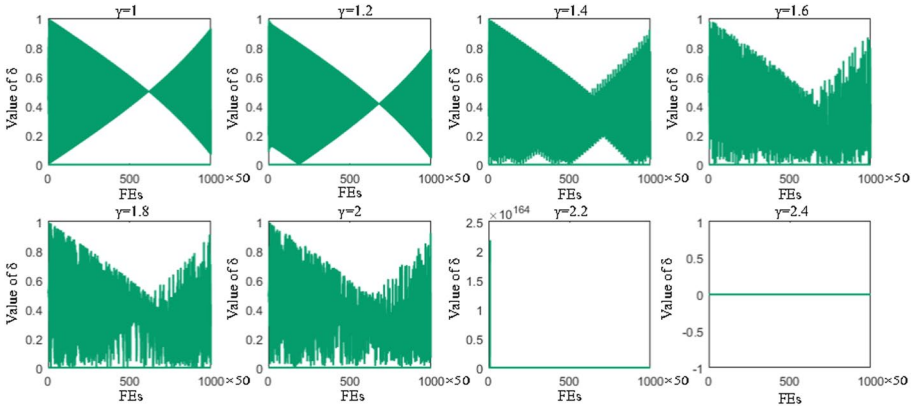

Fig. 4 The range of the balance control factor δ when the multiplication factors γ take different values

LSA algorithm's outcomes; however, slightly superior performance is observed when γ equals 2 (refer to Appendix Table 27). Therefore, for brevity, we set γ to 2 in this paper. Additionally, we explore the impact of λ on the LSA algorithm across various values. To ensure a balance between global exploration and local exploitation capabilities, the LSA algorithm should exhibit strong global exploration abilities in early iterations and potent local exploitation capabilities along with the ability to escape from local optima in later iterations. Consequently, some individuals in later iterations of LSA should conduct global exploration operations to prevent the algorithm from converging to local optima. As depicted in Fig. 3, the value of λ should range between (0.5, 0.85) (when λ is too large, δ exceeds 1). To precisely determine the value of λ, experiments are conducted with λ set to 0.5, 0.6, 0.7, 0.75, and 0.8, respectively. Statistical analysis of the results in Appendix Table 26 reveals that setting λ to 0.75 yields 10 optimal results, making it the most favorable choice among these scenarios. In summary, setting λ to 0.75 and γ to 2 is deemed reasonable. Initially, δ continually decreases, indicating the transition from wide-ranging exploration to focused searching. However, as the process proceeds, some individuals fall into local optimization, resulting in an increase of δ. To mitigate this issue, δ_{init} and δ_{final} are assigned the values 1 and 0, respectively, to enable dynamic balancing between international and regional development in the proposed algorithm.

This approach ensures that the algorithm maintains a balance between wide-ranging exploration and local refinement during the entire iteration process, thus enabling it to achieve both objectives effectively. It increases the likelihood of conducting extensive global search in the initial stages of evolution and progressively shifts focus towards thorough local search in later stages. Consequently, this method optimally balances the algorithm's capacity for broad-based exploration and targeted growth.

3.3.2 Exploration phase

The historical experiences of humanity offer valuable insights for the education of posterity and individual learning journeys. To make the algorithm have a strong global

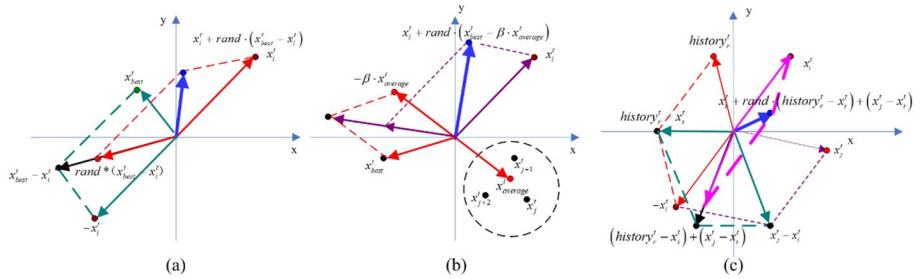

Fig. 5 Different search patterns of the individuals in 2D search space (a) The individual's active learning mode (b) The radiation pattern of role models (c) Global exploration mode

exploitation ability, this paper used the historical experience and other individual information to guide the search progress. The individuals in the population learn from past events and the current inhabitants at a random probability, and the corresponding updated schematic diagram is illustrated in Fig. 5(c). The mathematical model is shown in Formula (6).

$$x_i^{t+1} = x_i^t + rand \cdot \left[\left(history_r^t - x_i^t \right) + \left(x_j^t - x_i^t \right) \right] \tag{6}$$

where, x_i^t represents the ith learning individual, while x_j^t denotes an individual selected at random from the identical population. The new individual is denoted by x_i^{t+1}, and *rand* is randomly generated with support [0,1]. Additionally, $history_r^t$ represents an individual randomly selected from the past experience database. In the initial phase of populating, Formula (2) is utilized to produce a population of equal size, with x being assigned random values. In every iteration, *history* is modified using Eq. (7), and the matrix of *history* suffers random variations according to Formula (8).

$$history = \begin{cases} x, \text{ if a} < \text{b} \\ history, else \end{cases} \tag{7}$$

$$history = permuting(history) \tag{8}$$

where, a and b are randomly generated with support [0,1]. The variable *permuting* represents a random permutation operation.

3.3.3 Exploitation phase

The learning process involves learners acquiring knowledge from historical experiences, while simultaneously enhancing the overall learning ability of the population. This is achieved through active learning from role models, who are identified as optimal individuals exhibiting effective teaching behaviors. Nonetheless, equitable benefits from these role models are not experienced by all individuals within the population due to limited capacity to acquire knowledge. Thus, it is imperative to ascertain the ideal number of beneficiaries derived from the role models to attain optimal enhancement efficacy. Psychological research has indicated that

the average attention span of an individual typically ranges from 7 to 9 (Schoenewolf 1990; Bruner 1971, 2009). Consequently, an excessive number of teachers can hinder the teaching process, leading to challenges in fully utilizing the instructional potential of role models. Considering the aforementioned analysis, it is evident that certain individuals actively learn from role models, while role models specifically instruct select individuals on particular occasions. The learning schematic diagrams for the algorithm are presented in Fig. 5(a) and (b), accompanied by the corresponding mathematical model illustrated in Formula (9).

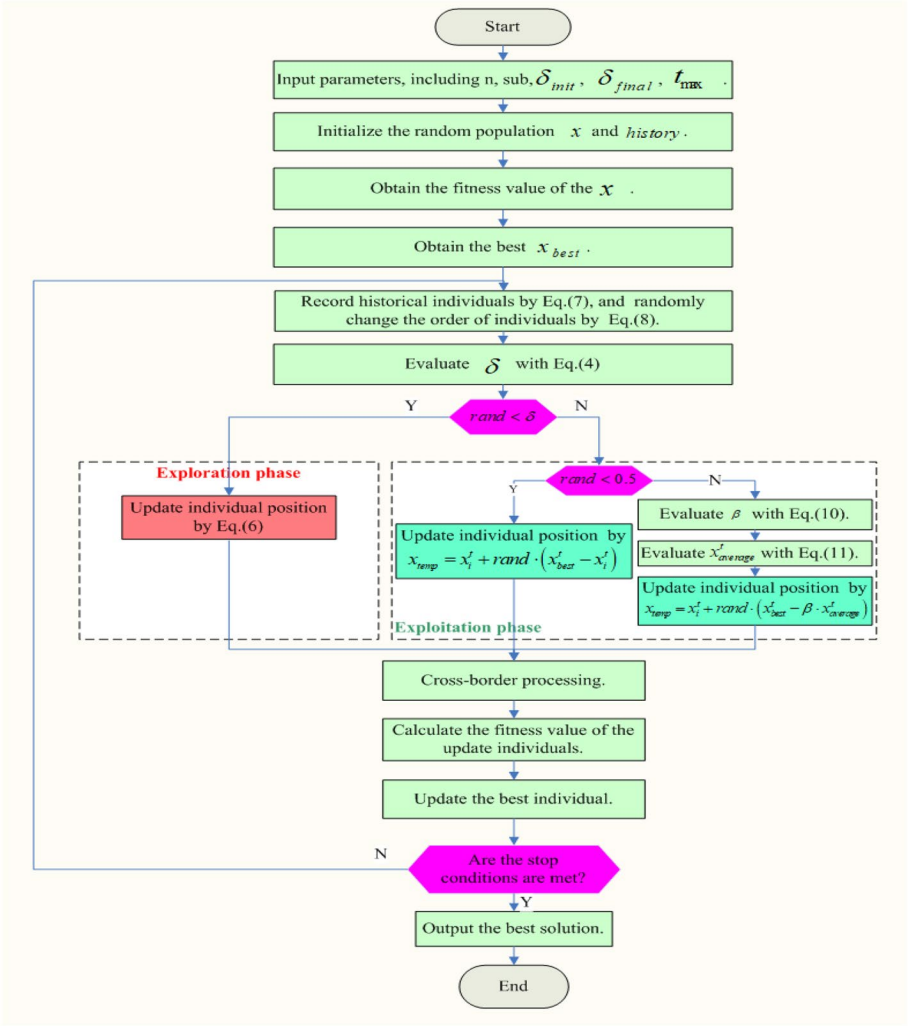

Fig. 6 Flow chart of LSA algorithm

$$x_i^{t+1} = \begin{cases} x_i^t + rand \cdot \left(x_{best}^t - \beta \cdot x_{average}^t\right), r < 0.5 \\ x_i^t + rand \cdot \left(x_{best}^t - x_i^t\right), else \end{cases} \tag{9}$$

where, *rand* and *r* are randomly generated with support [0,1]. x_{best}^t denote the position of the role model. The degree to which the learner acquires knowledge from the role model is modulated by the learning factor denoted as"β", which is computed using Formula (10). Moreover, we also compute the $x_{average}^t$ using Formula (11).

$$\beta = randi([1, sub]) \tag{10}$$

$$x_{average}^t = \frac{1}{sub} \sum_{i=1}^{sub} x_i \tag{11}$$

where, we investigate the relationship between the number of objects taught by role models (*sub*), the algorithm effectiveness (*sub* = 3), and a random integer (*randi*). The experimental measurement process reveals that the algorithm achieves optimal performance when *sub* = 3. Additionally, *randi* is a randomly selected integer within the range of 1 to *sub*, where *sub* represents a specific value.

During this stage, learners enhance their knowledge acquisition through the teaching method employed by role models. This directed learning from role models effectively enhances the overall learning ability of the learners.

3.3.4 Cross boundary processing

During the search process, individuals in the population may exceed the constraint of the problem domain, so it is necessary to handle individuals outside the boundary. Meanwhile, different boundary processing methods have the particular effect on the efficiency. The processing method adopted by this algorithm is shown in Formula (12):

$$x_i = \min(\max(x_i, lb), ub) \tag{12}$$

where, *lb* and *ub* correspond to the upper and lower bound values, respectively.

3.4 The procedure of LSA

Based on the earlier theoretical analysis, the LSA consists of two primary stages: global discovery and regional growth. In the global discovery stage, the algorithm simulates learning behaviors by drawing upon previous learning and unique learning tendencies observed in contemporary society. Conversely, the regional growth stage emulates instruction and acquisition behaviors. In this section, we present a comprehensive overview of the key procedures employed by LSA. Additionally, to assist with implementation, we provide the pseudo-code of the algorithm in Fig. 6 and Algorithm 1.

Algorithm 1 The pseudo code of LSA

Initialization:

Set the parameters: $nPop$, sub, δ_{init}, δ_{final}, $t_{\max}$.

Generate initial population x and $history$, and calculate the fitness value $f(x)$ of x.

Select the optimal individual x_{best} and its corresponding fitness value $fitness(x_{best})$.

while $t < t_{\max}$ do

r1=rand(), r2=rand();

if $r1 < r2$

$history \leftarrow x$

end if

$history = permuting(history)$

$y^t = 1 - 2 \cdot (y^{t-1})^2$

$\delta = \delta_{init} \cdot |\, y^t \,| - (\delta_{init} - \delta_{final}) \cdot \tan(0.75 \cdot \frac{t}{t_{\max}})$

for i = 1 : n

r3=rand();

if $r3 < \delta$

$x_{temp} = x_i^t + rand \cdot \left[\left(history_r^t - x_i^t \right) + \left(x_j^t - x_i^t \right) \right]$.

else

r4=rand();

if $r4 < 0.5$

$x_{temp} = x_i^t + rand \cdot \left(x_{best}^t - x_i^t \right)$

else

$x_{average}^t = \frac{1}{sub} \sum_{i=1}^{sub} x_i$, $\beta = randi([1, sub])$;

$x_{temp} = x_i^t + rand \cdot \left(x_{best}^t - \beta \cdot x_{average}^t \right)$

end if

end if

Cross boundary processing for x_{temp}.

Calculate the fitness value $f(x_{temp})$ of x_{temp}.

if $f(x_{temp}) < f(x_i)$

$x_i = x_{temp}$

$f(x_i) = f(x_{temp})$

end if

if $f(x_{temp}) < f(x_{best})$

$x_{best} = x_{temp}$

$f(x_{best}) = f(x_{temp})$

end if

end for

t=t+1

end while

Output the best solution (x_{best})

end

3.5 Time complexity analysis

The time complexity of the proposed LSA algorithm serves as a crucial performance indicator. The entire LSA process consists of three key steps: the initial setup, evaluation of fitness value, and refinement of the learning search process. The computation complexity of the initialization process is $O(nPop)$. During each iteration, approximately $\delta \cdot nPop$ individuals engage in global development operations, $(1-\delta) \cdot nPop/2$ participants utilize the role model guidance strategy, and $(1-\delta) \cdot nPop/2$ individuals utilize the active learning strategy from role models. As a result, the time complexity of LSA can be estimated as approximately $O(nPop) + O(t_{\max} \cdot \delta \cdot nPop) + O(t_{\max} \cdot (1-\delta) \cdot nPop/2) + O(t_{\max} \cdot (1-\delta) \cdot nPop/2) = O(n + t_{\max} \cdot nPop)$.

A heatmap visually represents data using color gradients to illustrate variations in data magnitude, aiding in the comprehension of correlations and trends. In Fig. 7, darker hues indicate longer algorithm runtimes. Here, we examine the time efficiency of the LSA algorithm in addressing optimization problems from two angles. Firstly, we analyze its overall time complexity, which hinges on factors such as population size, iteration count, and problem intricacy. Figure 7(a) illustrates that, with a set number of iterations, larger populations incur greater time costs (as depicted by colors closer to red), albeit with improved algorithmic precision (as indicated in Appendix Table 28). Conversely, Fig. 7(b) demonstrates that, with a fixed evaluation count, solving more complex problems consumes more time (evident in functions F26 and F27, represented by darker colors), albeit with marginal gains in precision (as shown in Appendix Table 29). Secondly, we scrutinize algorithm runtime and precision through specific execution strategies, comparing outcomes with varied balance control values (denoted by δ) using formula (5). Figure 7 reveals that setting δ to 0 results in maximal execution times (reflected by red hues), while δ set to 1 minimizes execution times (reflected by blue hues). This disparity arises from δ's influence on the algorithm's tendency towards local exploitation (Eqs. (10)-(11)) or global exploration (Eq. (6)), impacting time costs. However, employing a fixed δ value compromises search precision compared to formula (5) (as detailed in Appendix Table 30).

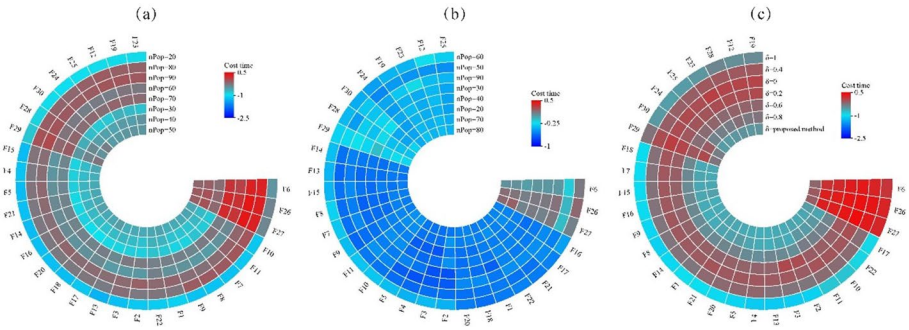

Fig. 7 Time consumption of the LSA algorithm under different parameters. **a** Time spent executing CEC 2014 functions with fixed number of iterations. **b** Time spent executing CEC 2014 functions with fixed number of evaluations. **c** Time spent executing CEC 2014 functions with different values of δ

3.6 Convergence analysis

3.6.1 Markov chain model of the LSA algorithm

Definition 1: The state of a learning agent and state space of a learning agent.

The state of a learning agent is composed of the position x and the global best position $best$, denoted as $I = (x, best)$, where $x \in A, best \in A$ and $f(best) \leq f(x)$, A is a feasible solution within the spatial range. The possible states of all learning agents constitute the state space of a learning agent, denoted as:

$$I = \{I = (x, best) | x, best \in A, f(best) \leq f(x)\} \tag{13}$$

Definition 2: The state of a learning swarm and state space of a learning swarm.

The states of all individual learners within a learning swarm constitute the state of the learning swarm. The state of the m learners in the learning swarm at time t is denoted as $s_t = (I_1^t, I_2^t, ..., I_i^t, ...I_m^t)$, where I_i^t represents the i-th learner in the population at time t, and m is the total number of learners in the population. The collection of all possible states of the learning swarm forms the state space of the learning swarm, denoted as:

$$S = \{s_t = (I_1^t, I_2^t, ..., I_i^t, ...I_m^t) | I_i^t \in I (1 \leq i \leq m, t = 1, 2, ...)\} \tag{14}$$

Definition 3: The state transition of individual learners.

For $\forall I_i = (x_i, best_i)| \in I, \forall I_j = (x_j, best_i)| \in I$, the state I_i of an individual learner transitions to another state I_j in one step, denoted as $T_I(I_i) = I_j$.

Theorem 1: In the LSA algorithm, the probability of the state transition from state I_i to state I_j of an individual learner can be expressed as:

$$p(T_I(I_i) = I_j) = \begin{cases} p_e(T_I(I_i) = I_j), \text{achieved by e}\textit{xploration phase}, \\ p_a(T_I(I_i) = I_j), \text{achieved by active learning}, \\ p_r(T_I(I_i) = I_j), \text{achieved by role models teaching}. \end{cases} \tag{15}$$

Proof:

In the LSA algorithm, the algorithm primarily consists of two phases: the exploration phase and the exploitation phase. The exploitation phase is further comprised of two distinct update modes: "Active learning" and "Role models teaching".

In each position update strategy, the state of an individual learner is formed by the position x and the global best position $best$. Therefore, the corresponding one-step transition probability also varies. Considering that the learner's vector is multidimensional and the population forms a set of points in a multidimensional hyperspace, the process of the learner's position change can be regarded as the transformation between points in this hyperspace.

(i) According to Definition 3 and the geometric interpretation of the LSA algorithm, the one-step transition probability from state I_i to another state I_j in the exploration phase is given by:

$$p(T_e(I_i) = I_j) = p_e(x_i \rightarrow x_j) \cdot p_{best}(best_i \rightarrow best_j) \tag{16}$$

where, the one-step transition probability from the individual's global best solution to another state is given by:

$$p_{best}(best_i \to best_j)=\begin{cases} 1, f(best_i) \le f(best_j) \\ 0, \text{other} \end{cases} \tag{17}$$

The probability of the learning individual transitioning from position x_i to position x_j through the exploration phase search strategy is:

$$p_e(x_i \to x_j) = \begin{cases} \frac{1}{\left|rand \cdot \left[\left(history_r - x_i\right) + \left(x_j - x_i\right)\right]\right|}, x_j \in \left[x_i, x_i + rand \cdot \left(\left(history_r - x_i\right) + \left(x_j - x_i\right)\right)\right], r < \delta \\ 0, \ other \end{cases} \tag{18}$$

(ii) The transition probability from state I_i to another state I_j through the active learning strategy is:

$$p_a(T_I(I_i) = I_j) = p_a(x_i \to x_j) \cdot p_{best}(best_i \to best_j) \tag{19}$$

where, the probability of the learning individual transitioning from position x_i to position x_j in one step is:

$$p_a(x_i \to x_j) = \begin{cases} \frac{1}{\left|x_i + rand \cdot \left(x_{best} - x_i\right)\right|}, x_j \in \left[x_i, x_i + rand \cdot \left(x_{best} - x_i\right)\right], \ r \ge \delta \& r1 < 0.5 \\ 0, \ other \end{cases}. \tag{20}$$

(iii) The transition probability from state I_i to another state I_j through the Role models teaching strategy is:

$$p_r(T_I(I_i) = I_j) = p_r(x_i \to x_j) \cdot p_{best}(best_i \to best_j) \tag{21}$$

where, the probability of the learning individual transitioning from position x_i to position x_j in one step is:

$$p_r(x_i \to x_j) = \begin{cases} \frac{1}{\left|x_i + rand \cdot \left(x_{best} - \beta \cdot x_{average}\right)\right|}, x_j \in \left[x_i, x_i + rand \cdot \left(x_{best} - \beta \cdot x_{average}\right)\right], \ r \ge \delta \& r1 \ge 0.5 \\ 0, \ other \end{cases}. \tag{22}$$

Definition 4: The state transition of learning community.

For $\forall s_i \in S, \forall s_j \in S$, in the iteration of LSA algorithm, the learning community transition from state s_i to another state s_j in a single step, denoted as $T_S(s_i) = s_j$. The transition probability for the learning community state s_i to transition to another state s_j in a single step is:

$$p(T_S(s_i) = s_j) = \prod_{k=1}^{m} p(T_S(s_{i,k}) = s_{j,k}). \tag{23}$$

where, m represents the number of individuals in the population. This equation states that the one-step transition from learning group state s_i to state s_j is the transition of the individual states from all individuals in group space s_i to the corresponding individual states in s_j.

3.6.2 Convergence analysis of LSA algorithm

Definition 5: Markov chain.

In a stochastic process, let process $\{x_n, n \in T\}$ have parameter set T as a discrete time series, denoted as $T = \{0, 1, 2, ...\}$, where the entire set of possible values x_n constitutes a discrete state space $I = \{i_1, i_2, i_3, ...\}$. If for any integer $n \in T$ and any $i_1, i_2, i_3, ..., i_{n+1} \in I$, the following conditional probability $p\{x_{n+1} = i_{n+1} | x_n = i_n\}$ holds, then $\{x_n, n \in \mathrm{T}\}$ is termed as a Markov chain.

Definition 6: Finite Markov chain.

If the state space I is finite, then the Markov chain is referred to as a finite Markov chain.

Definition 7: Homogeneous Markov chain.

$p\{x_{n+1} = i_{n+1} | x_n = i_n\}$ represents the conditional probability that the system, in state i_n at time n, transitions to a new state x_{n+1}. If this probability depends only on the state at time n and not on time n, then the Markov chain is referred to as a homogeneous Markov chain.

Theorem 2: In the LSA algorithm, the state sequence $\{s(n); n \geq 0\}$ of the learning community is a finite homogeneous Markov chain.

Proof:

(i) According to Definition 4, it is known that in the state transition of the learning community, there exists a state $\forall s(n) \in S, \forall s(n+1) \in S$ in the sequence $\{s(n); n \geq 0\}$, and its transition probability $p(T_S(s(n)) = s(n+1))$ is determined by the transition probabilities $p(T_S(I_i(n)) = I_j(n+1))$ of all learning individuals in the community. From Eqs. (15) to (22), it is known that the state transition probability of any individual in the learning community is only related to the control factors δ, the states $(x_i, best)$ at time n, $x_{average}$, a random number r, r_1 between [0,1], and β, but is independent of time n. Based on the above analysis, the state transition probability $p(T_S(s(n)) = s(n+1))$ of the learning community only depends on the individual states at time n, therefore, the sequence $\{s(n); n \geq 0\}$ exhibits the Markov property.

(ii) From Eqs. (16) to (22), it can be seen that $p(T_S(s(n)) = s(n+1))$ is independent of the time n. Combining with (i), it can be inferred that the sequence $\{s(n); n \geq 0\}$ is a homogeneous Markov chain.

(iii) When optimizing any problem in a computer, the variables used to describe the optimization problem are represented with a certain precision, and the search space is finite. In any individual $I_i = (x_i, best)$ of the learners, the dimensions of x_i and $best$ are finite, and x_i and $best$ are also constrained by $[x_{\min}, x_{\max}]$. Therefore, the space of learning individuals I is finite. And the learning community composed of m learning individuals is also finite.

Based on (i), (ii), and (iii), it can be inferred that the state sequence $\{s(n); n \geq 0\}$ of the learning community is a finite homogeneous Markov chain.

Convergence criterion The LSA algorithm belongs to the category of stochastic search algorithms, thus in this paper, the convergence behavior of the LSA algorithm is determined using a convergence criterion based on random algorithms (Solis and Wets 1981).

For the optimization problem $<A, f>$, where A is the feasible solution space and f is the fitness function, if there is a stochastic optimization algorithm D and the result of the k-th iteration is x_k, then the result of the next iteration is $x_k = D(x_k, \zeta)$, where ζ represents

solutions previously encountered during the iterative search process of algorithm D. The lower bound of the search is defined as:

$$\alpha = \inf\{t | v(x \in A | f(x) < t) > 0\} \tag{24}$$

where, $v(x)$ represents the Lebesgue measure on the set x. The region of optimal solutions is defined as:

$$R_{\varepsilon,M} = \begin{cases} x \in A | f(x) < \alpha + \varepsilon\}, \alpha \text{ limit} \\ x \in A | f(x) < -C\}, \alpha = -\infty \end{cases} \tag{25}$$

where, $\varepsilon > 0$ and C are sufficiently large positive numbers. If the stochastic algorithm D finds one point in $R_{\varepsilon,M}$, it can be considered that the algorithm has found an acceptable global optimal or approximately global optimal point.

Condition H1: If $f(D(x_k, \zeta) \leq f(x))$, then $\zeta \in A$ implies $f(D(x_k, \zeta) \leq f(\zeta))$.

Condition H2: For $\forall B \in A$, s.t. $v(B) > 0$, there exists $\prod_{k=0}^{\infty}(1 - u_k(B)) = 0$, where $u_k(B)$ is the probability measure of the *k-th* iteration search solution of algorithm D on set B.

Theorem 3: Let f be measurable, and let A be a measurable subset of R^n. Suppose algorithm D satisfies conditions H1 and H2, and $\{x_k\}_{k=0}^{\infty}$ is the sequence generated by algorithm D. Then, there exists a probability measure $\lim_{k\to\infty} P(x_k \in R_{\varepsilon,M}) = 1$,where $R_{\varepsilon,M}$ is the optimal region, i.e., algorithm D globally converges.

LSA Algorithm Convergence Theorem 4: The LSA algorithm satisfies condition H1.

Proof: In the LSA algorithm, individual current optimal positions are updated at each iteration, denoted as follows.

$$x_i(t) = \begin{cases} x_i(t-1), f(x_i(t) \geq f(x_i(t-1)) \\ x_i(t), f(x_i(t) < f(x_i(t-1)) \end{cases} \tag{26}$$

Therefore, the LSA algorithm preserves the best position of the population at each iteration, satisfying condition H1.

Definition 8: Set of optimal states of learners, denoted as G.

Let g^* be the optimal solution of the optimization problem $<A, f>$, and let $G = \{s = (x) | f(x) = f(g^*), s \in S\}$ denote the set of optimal states of learners. If $G = S$, then each solution in the feasible solution space is not only feasible but also optimal. In this case, optimization is meaningless, and the following discussions are based on $G \subset S$.

Definition 9: Absorbing state Markov chain.

Based on the population sequence from the LSA algorithm, a Markov chain $\{s(t), t \geq 0\}$ and the set of optimal states $G = S$ are defined. If $\{s(t), t \geq 0\}$ satisfies the conditional probability $p\{x_{k+1} \notin G | x_k \in G\} = 0$, then this Markov chain is called an absorbing state Markov chain.

Theorem 5: The Markov chain generated by the LSA algorithm is an absorbing state Markov chain.

Proof:

In the optimization of LSA, the update of individuals adopts the mechanism of preserving the best individuals. That is, only when the fitness value of the current best individual is better than the fitness value of the original individual, will the original best individual be replaced, as shown in Eq. (26). This ensures that in each iteration of the algorithm's evolution process, the newly generated individuals are not inferior to those generated before.

Therefore, the conditional probability $p\{x_{k+1} \notin G | x_k \in G\} = 0$ is satisfied, that is, the population sequence $\{s(t), t \geq 0\}$ of the LSA algorithm forms an absorbing state Markov chain.

Definition 10: Let set D be a non-empty subset of the state space S. If $\forall i \in D, \forall j \notin D$, there exists $p(i \notin D \lfloor j \in D) = 0$, then D is called a closed set.

Definition 11: Let the set of optimal learning individual states be denoted as M, the set of optimal learning group states be denoted as G, and the global optimal solution of the optimization problem be denoted as $best^*$, then

$$M = \{I^* = (x, best), n \geq 1\} = \{I^* = (x, best^*), n \geq 1\} \tag{27}$$

where, $I_{\mathrm{t}}^* \in S$ represents the optimal learning individual state.

$$G = \{\mathrm{S}_n^*, \mathrm{t} \geq 1 | \mathrm{S}_n^* = (\mathrm{I}_1^*, \mathrm{I}_2^*, ..., \mathrm{I}_i^*, ...\mathrm{I}_m^*) | \exists \mathrm{I}_i^* \in M\} \tag{28}$$

where $\mathrm{S}_n^* = (\mathrm{I}_1^*, \mathrm{I}_2^*, ..., \mathrm{I}_i^*, ...\mathrm{I}_m^*)$ represents the optimal state of the population.

Theorem 6: The set of optimal states for individual learning, denoted as M, is a closed set.

Proof:

Let the learning individual state $I_i^n = (x_i^n, best^*)$ be the optimal state. According to the execution strategy of the LSA algorithm, it is evident that the next moment's state $I_i^{n+1} = (x_i^{n+1}, best^*)$ is also the optimal state. This can be concluded based on formulas (16)-(22).

$$p(I_i^n \to I_i^{n+1}) = p(x_i^n \to x_i^{n+1}) \cdot p(best^* \to best^*) = 1 \tag{29}$$

In other words, $\forall I_i^n \in M, I_j^{n+1} \notin M$, $p(I_i^n \to I_j^{n+1}) = 0$. Therefore, the set M of optimal states for the individual learner is a closed set.

Theorem 7: In the LSA algorithm, the set of optimal states for the learning population,G, is a closed set.

Proof:

$\forall s_i \in G, \forall s_j \notin G, s_j \in S$, for any step size $l, l \geq 1$, according to the Chapman-Kolmogorov equation, we can obtain:

$$\begin{aligned} p(i \notin G \lfloor j \in G) = p_{i,j}^l = \sum_{s_{r1} \in S} ... \sum_{s_{rl-1} \in S} & p(T_S(s_i) = s_{r1}) \cdot p(T_S(s_{r1}) = s_{r2}) \cdot ... \cdot p(T_S(s_{ru-1}) \\ & = s_{ru}) \cdot ... \cdot p(T_S(s_{rl-1}) = s_j) \end{aligned} \tag{30}$$

where,$s_{ru-1} \in G, s_{ru} \notin G, 1 \leq u \leq l$, it can be inferred from Definition 4:

$$p(T_S(s_{ru-1}) = s_{ru}) = \prod_{k=1}^{m} p(T_S(I_k^{ru-1}) = I_k^{ru-1}) \tag{31}$$

Due to $\exists I_k^{ru-1} \in M, I_k^{ru} \notin M$, then $f(I_k^{ru}) > f(I_k^{ru-1}) = f(I^*)$. According to Eq. (17), $p_*(best_k^{ru-1} \to best_k^{ru}) = 0$, so $p(T_S(s_{ru-1}) = s_{ru}) = 0$, which means $p(i \notin G \lfloor j \in G) = 0$. Therefore, G is a closed set in the S space.

Definition 12: For any $n \geq 1$, when $l \geq n + 1$, if $best^l = best^n$ always holds true, $s(n) \in S$ is referred to as an absorbing state. A set H constructed solely from a single absorbing state is called a closed set.

Theorem 8: Let {s(n),n ≥ 1} be a Markov chain representing the state sequence of a learning population, and $\Lambda=G\cup H$ be a closed set. When $\sum_{n=1}^{\infty} p_S\{s(n)\}<\infty$, then $p(\lim_{n\to\infty} s(n)\in\Lambda)=1$.

Proof:

Based on the fact that all G, H are closed sets, it follows that $\Lambda=G\cup H$ is also a closed set. Assuming that the learning population is in set Λ at time n and in state $S(n+1)$ at time $n+1$ while being in state $S(n)$, we can conclude $p_S\{s(n+1)\notin\Lambda|s(n)\in\Lambda\}=0$. Therefore,

$$\begin{aligned} p_S\{s(n+1)\notin\Lambda\} &= p_S\{s(n)\notin\Lambda\}\cdot p_S\{s(n+1)\notin\Lambda|s(n)\notin\Lambda\} \\ &\quad + p_S\{s(n)\notin\Lambda\}\cdot p_S\{s(n+1)\notin\Lambda|s(n)\notin\Lambda\} \\ &= p_S\{s(n)\notin\Lambda\}\cdot p_S\{s(n+1)\notin\Lambda|s(n)\notin\Lambda\} \\ &= p_S\{s(n-1)\notin\Lambda\}\cdot p_S\{s(n)\notin\Lambda|s(n-1)\notin\Lambda\} \\ &\quad \cdot p_S\{s(n+1)\notin\Lambda|s(n)\notin\Lambda\} \\ &= p_S\{s(l)\notin\Lambda\}\cdot p_S\{s(l+1)\notin\Lambda|s(l)\notin\Lambda\}\cdot\ldots \\ &\quad \cdot p_S\{s(n+1)\notin\Lambda|s(n)\notin\Lambda\} \\ &= p_S\{s(l)\notin\Lambda\}\cdot\prod_{l=1}^{n} p_S\{s(l+1)\notin\Lambda|s(l)\notin\Lambda\} \end{aligned} \tag{32}$$

If the learning population is in state $S(n)=(I_1^n,I_2^n,\ldots,I_m^n)\notin\Lambda$, then $\forall i\in[1,m], I_i^n\notin G, and\ I_i^n\notin H$ holds.

$$\begin{aligned} p_S\{s(l+1)\notin\Lambda|s(l)\notin\Lambda\} &= \prod_{i=1}^{m} p\{I_i^{l+1}\notin\Lambda|I_i^l\notin\Lambda\} \\ &= \prod_{i=1}^{m}(1-p\{x_i^{l+1}|x_i^l\}\cdot p\{best_i^{l+1}|best_i^l\}) \end{aligned} \tag{33}$$

Then

$$\begin{aligned} p_S\{\,s(n)\notin\Lambda\} &= p_S\{\,s(l)\notin\Lambda\}\cdot\prod_{l=1}^{n-1} p_S\{\,s(l+1)\notin\Lambda|s(l)\notin\Lambda\} \\ &= p_S\{\,s(l)\notin\Lambda\}\cdot\prod_{l=1}^{n-1}\prod_{i=1}^{m}(1-p\{x_i^{l+1}|x_i^l\}\cdot p\{best_i^{l+1}|best_i^l\}) \end{aligned} \tag{34}$$

By summing up n, we can obtain

$$\begin{aligned} \sum_{n=1}^{\infty} p_S\{\,s(n)\notin\Lambda\} &= p_S\{\,s(l)\notin\Lambda\}\cdot \\ \sum_{n=1}^{\infty}\prod_{l=1}^{n-1}\prod_{i=1}^{m}&(1-p\{x_i^{l+1}|x_i^l\}\cdot p\{best_i^{l+1}|best_i^l\}) \end{aligned} \tag{35}$$

Due to

$$\sum_{n=1}^{\infty} p_S\{s(n)\}<\infty \tag{36}$$

So

$$\sum_{n=1}^{\infty} p_S\{\mathrm{s}(n) \notin \Lambda\} < \infty \tag{37}$$

Then

$$\lim_{n\to\infty}(1 - p\{\ x_i^n | x_i^{n-1}\}\ \cdot p\{\ best_i^n | best_i^{n-1}\}\ ) = 0 \tag{38}$$

So, when $n \to \infty$ and $\sum_{n=1}^{\infty} p_S\{\ \mathrm{s}(n)\} < \infty$, $p(\lim_{n\to\infty} \mathrm{s}(n) \in \Lambda) = 1 - p(\lim_{n\to\infty} \mathrm{s}(n) \notin \Lambda) = 1$.

Theorem 9: The LSA algorithm converges to the global optimum.

Proof: According to Theorem 4, it is known that the LSA algorithm satisfies condition H1. By Theorem 8, it is known that the probability of the LSA algorithm continuously searching for the global optimum for an infinite number of times is zero, thus there exists an $\prod_{k=0}^{\infty}(1 - u_k(B)) = 0$ satisfying condition H2. According to Theorem 3, it can be concluded that the LSA algorithm is a globally convergent algorithm.

3.7 The difference between TLBO and LSA

Teaching–learning-based optimization (TLBO) and Learning Search Algorithm (LSA) share common features as population-based algorithms inspired by human learning behavior. However, they diverge significantly in several aspects. Firstly, TLBO draws inspiration from the classroom teaching model, where knowledge transfer occurs bidirectionally: students learn from teachers, and teachers impart knowledge to students through direct instruction. Conversely, LSA predominantly acquires knowledge through individual learning from historical experiences (including both past and contemporary sources) and individuals emulating role models around them, with these role models disseminating knowledge to those receptive to specific learning aspects. Hence, the learning mechanisms of the two algorithms differ. Secondly, TLBO adopts a uniform knowledge dissemination approach throughout the entire population, overlooking individual physiological traits, which can hinder knowledge acquisition. In contrast, LSA, during its developmental phase, fully integrates learners' unique attributes into the acquisition process, leveraging their ability to absorb knowledge from role models and learn from exceptional individuals. Thirdly, TLBO lacks distinction between global exploration and local exploitation phases, with all individuals following uniform learning and teaching strategies. In contrast, LSA embodies both phases and achieves a harmonious balance between global exploration and local exploitation through adaptive adjustment of balancing factors. Lastly, owing to its diverse learning strategies, LSA surpasses TLBO in search results for optimization problems like Unimodal Functions, Multimodal Functions, Hybrid Functions, and Composition Functions. This comprehensive superiority stems from LSA's multifaceted learning approaches.

4 Experiment and simulation

To estimate the adequacy of the proposed approach, we conducted a comparative analysis against other state-of-the-art algorithms. Our implementation of the LSA algorithm was conducted utilizing MATLAB R2016a. The experimental setup comprises a personal computer equipped with an AMD Ryzen 74700G with Radeon Graphics and 12 GB main memory. The study employed a diverse set of test problems, including 30 IEEE CEC2014 and 10 IEEE CEC2020 benchmark functions, 6 challenging real-world engineering design optimization problems, and 15 feature selection datasets from UCI. Table 2 presents a collection of 40 minimization functions called CEC2014 and CEC2020, which is a powerful set of real-parameter optimization benchmarks. These functions effectively mimic real-world optimization problems. To assess the performance of the LSA, we selected 9 prominent swarm intelligence optimization algorithms and 11 powerful recently developed algorithms as a comparison. The population size (*nPop*) was set at 50, and the number of evaluations (FEs) was set at 1000**nPop*. The search dimension was fixed at 10, whereas the parameter values for the comparison algorithms are presented in Table 3.

In order to ensure the reliability of our results, we executed each test 20 times independently and highlighted the best-performing outcomes in the data statistics tables. Furthermore, to investigate the statistical significance of our results, we employed the Wilcoxon test at a significance level of 0.05. The symbols "/=/-" were adopted to indicate whether the LSA algorithm is superior, equal to, or inferior to the comparison algorithms in terms of performance.

4.1 Results of IEEE CEC 2014 and CEC 2020

This section presents a comprehensive analysis of the proposed LSA algorithm as well as various state-of-the-art original and enhanced algorithms with improved search performance. The IEEE CEC 2014 and CEC 2020 benchmark functions have been chosen as the evaluation benchmarks for this study.

4.1.1 The search process experiment of LSA

In this subsection, the proposed LSA algorithm is utilized to solve a range of benchmark functions, and a detailed analysis of the optimization process is conducted.

Figure 8(a) illustrates the three-dimensional mathematical model of the benchmark function. Notably, the mathematical model of F2 exhibits relative simplicity, while the remaining four functions possess a higher level of complexity. Figure 8(b) demonstrates the search trajectory of the proposed algorithm from a top-down perspective. The larger blue dots represent the best search positions during the search process, while the other colored dots denote the positions of the search individuals throughout iterations. Additionally, Fig. 8(c) depicts the progression of the average fitness value for the entire population. It is evident from Fig. 8(b) that for both simple and complex functions, numerous historical search trajectories of individuals are concentrated in proximity to the global optimum, effectively ensuring the thoroughness of local search. Moreover, several discrete colored points are scattered in other regions, demonstrating the capability of the algorithm to perform global exploration and avoid being trapped in local optima. The convergence of the average fitness curve in all mathematical models is evident in Fig. 8(c), underscoring the

Table 2 The benchmark suite of CEC2014 and CEC2020 specifically designed for optimizing real-valued parameters in single-objective tasks

Functions	Function type	Characteristics	Performance testing
F1-F3(CEC2014), F31(CEC2020)	Unimodal Functions	There is only one global optimal solution, and the curve shape of the function exhibits a characteristic of a single peak	Evaluating the performance of algorithms on simple global optimization problems
F4-F16(CEC2014), F32-F34(CEC2020)	Multimodal Functions	There are multiple global optimal solutions, and the curve shape of the function exhibits a characteristic of multiple peaks	Assessing the ability of algorithms in multi-solution search and global optimization problems
F17-F22(CEC2014), F35-F37(CEC2020)	Hybrid functions	It simulates optimization problems involving mixed parameter types by combining continuous and discrete variables	Evaluating the performance of algorithms on optimization problems involving both continuous and discrete variables
F23-F30(CEC2014), F38-F40(CEC2020)	Composition Functions	It is a complex function constructed by combining multiple basic functions	They simulate complex optimization problems in the real world and are used to evaluate the performance of algorithms on large-scale global optimization problems

Table 3 The parameter configurations of the LSA and rival algorithms

	Algorithm	Values of the parameter
Original algorithms	GWO	Based on the references (Mohammadi-Balani et al. 2021; Ahmadianfar et al. 2022);
	HHO	Decreaing energy of rabbit: Emax = 2,Emin = 0;
	HPO	Constriction Coefeicient: B = 0.1;
	MFO	Based on the reference (Mirjalili 2015b);
	SSA	Based on the reference (Abualigah et al. 2020);
	BWOA	Based on the reference (Hayyolalam and Kazem 2020);
	SOA	Based on the reference (Ramshanker and Chakraborty 2022);
	TSA	xmin = 1; xmax = 4;
	TLBO	Based on the reference (Rao et al. 2011);
Improved algorithms	GNHGWO	Based on (Akbari et al. 2021);
	GWOCS	Based on (Wang et al. 2022b);
	HPSOBOA	p = 0.6; a = 0.1; c = 0.01; wMax = 0.9; wMin = 0.2; Vmax = 1;
	NCHHO	a1 = 4; teta = 0.7;
	PSOBOA	p = 0.6; a = 0.1; c = 0.01; w = 0.7; Vmax = 1;
	HFPSO	c1 = 1.49445, c2 = 1.49445, vmax_coef = 0.1;
	FDB-AGDE	Based on the reference (Long et al. 2020);
	dFDB-MRFO	Based on the reference (Mirjalili et al. 2014);
	AFDB-SFS	Maximum_Diffusion = 1, Walk = 1;
	FDB- TLABC	CR = 0.5;
	TSALSHADE	p_best_rate = 0.11, arc_rate = 1, memory_size = 5, min_pop_size = 4, subpoppoulation size is 1/6, tangent flight modification rate is 0.01
	LSA	$sub = 3, \delta_{init} = 1, \delta_{final} = 0$, the initial chaotic sequence $y^0 = 0.75$

robust search capability of the LSA algorithm. In Fig. 8(e), it's evident that the LSA algorithm achieves a balanced approach between Exploration and Exploitation over time (the computational methods for Exploration and Exploitation are based on literature (Cheng et al. 2014; Hussain et al. 2019)). As iterations progress, Exploitation steadily approaches 100 while Exploration declines towards 0. Analysis of functions F2, F8, and F15 reveals a rapid decrease in population diversity, showcasing the algorithm's robust exploitation abilities. Conversely, for hybrid and composite functions like F25 and F27, population diversity fluctuates notably (the computational methods for calculating population diversity are based on literature (Cheng et al. 2014; Hussain et al. 2019).), consistently remaining at a higher level, demonstrating the algorithm's strong global exploration capabilities (as shown in Fig. 8(d)).

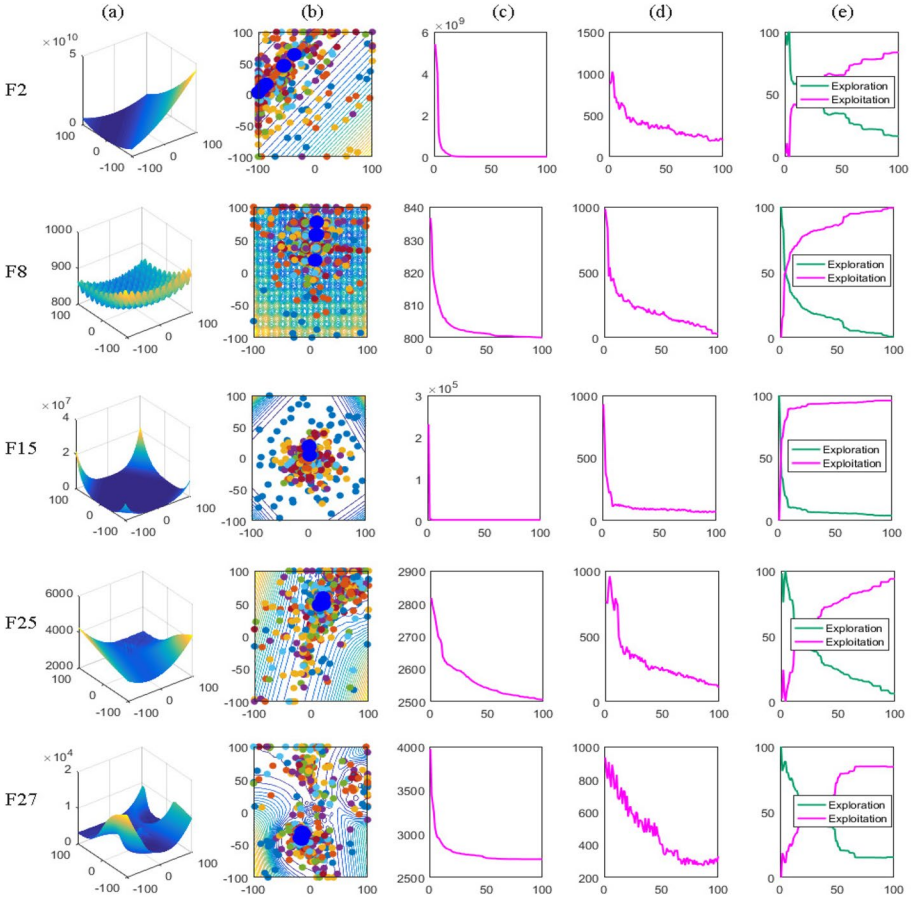

Fig. 8 Visualization of the algorithm's search process. **a** 3-dimensional mathematical models of the function. **b** The search trajectory during the optimization process is displayed from a top view. **c** Changes in the average fitness value of the population. **d** Population diversity. **e** Exploration and exploitation capabilities of the algorithm

Table 4 The Avage and Std values of the LSA algorithm in comparison to other benchmark algorithms for both the CEC 2014 and 2020 competitions

	F1		F2		F3	
	Avage	Std	Avage	Std	Avage	Std
GWO	7.3990E+06	4.55690E+06	9.1321E+04	3.59190E+05	6829.1724	3.56353E+03
HHO	5.9485E+06	4.28566E+06	3.0612E+05	9.99881E+04	4121.9835	1.54798E+03
HPO	1.1264E+05	7.09479E+04	6457.0263	3.95105E+03	1184.7682	5.10976E+02
SSA	9.2149E+05	9.50630E+05	3591.5371	3.20325E+03	8359.8877	5.70965E+03
BWOA	4.6031E+07	3.55861E+07	3.46019E+09	2.60632E+09	15,063.7426	6.68398E+03
MFO	1.9464E+06	2.17012E+06	9.0759E+03	4.25634E+03	21,664.9321	1.41015E+04
SOA	2.3202E+07	2.12349E+07	1.5816E+09	1.34092E+09	10,483.1522	3.32589E+03
TSA	2.2291E+07	4.43164E+07	2.2233E+09	2.26854E+09	8771.7316	4.48969E+03
TLBO	9.3430E+04	1.05505E+05	669.0509	4.24107E+02	6610.1866	2.62832E+03
LSA	**116.6784**	**3.94191E+00**	**200.0000**	**2.70800E-05**	**300.0000**	**6.96451E-07**
	F4		F5		F6	
	Avage	Std	Avage	Std	Avage	Std
GWO	436.2236	2.06953E+01	520.4559	7.60797E-02	602.2885	1.15348E+00
HHO	429.3707	1.93111E+01	520.1261	1.07791E-01	607.2109	1.40264E+00
HPO	**415.8509**	**1.63915E+01**	520.0152	3.21176E-02	604.5557	1.36612E+00
SSA	430.3959	1.59243E+01	520.0352	7.54122E-02	603.5568	1.56615E+00
BWOA	1142.3322	6.17904E+02	520.5403	1.71310E-01	609.5351	1.37675E+00
MFO	451.2517	3.74510E+01	520.1037	1.68917E-01	603.6946	1.60060E+00
SOA	697.0973	2.29154E+02	**520.0000**	**4.32590E-05**	607.1850	1.01637E+00
TSA	519.0999	1.93319E+02	520.4281	1.04081E-01	606.2421	1.99843E+00
TLBO	427.8243	1.55541E+01	520.4179	1.20977E-01	**602.0702**	1.25953E+00
LSA	417.8237	1.79205E+01	520.2482	8.66241E-02	602.7809	**8.04295E-01**
	F7		F8		F9	
	Avage	Std	Avage	Std	Avage	Std
GWO	701.0452	7.08238E-01	813.4327	6.43647E+00	916.5673	7.23977E+00
HHO	700.9995	3.98162E-01	822.4818	1.05324E+01	937.3829	1.09189E+01

Table 4 (continued)

HPO	**700.1675**	**9.27887E-02**	**805.3728**	**3.99808E + 00**	926.9161	1.10740E + 01
SSA	700.2252	1.59946E-01	823.2980	1.02919E + 01	928.0578	1.12084E + 01
BWOA	788.6180	6.49690E + 01	863.0067	1.90000E + 01	957.6514	1.46143E + 01
MFO	707.9165	1.23292E + 01	818.0696	8.71082E + 00	926.5253	9.80669E + 00
SOA	735.2068	2.03182E + 01	828.8105	8.16202E + 00	928.4226	6.04461E + 00
TSA	756.9418	3.36636E + 01	841.6423	1.57187E + 01	946.8281	1.74275E + 01
TLBO	700.3627	1.06491E-01	808.4321	4.44300E + 00	915.5713	9.00259E + 00
LSA	700.2244	1.10526E-01	814.1284	6.28916E + 00	**914.1284**	**4.95928E + 00**
	F10		F11		F12	
	Avage	Std	Avage	Std	Avage	Std
GWO	1346.2413	2.08608E + 02	1641.4664	**1.56318E + 02**	1200.7001	6.27785E-01
HHO	1475.0159	2.27734E + 02	2173.5971	2.64857E + 02	1200.7217	3.33102E-01
HPO	1304.7791	1.73130E + 02	1855.1108	2.74637E + 02	1200.2962	2.10140E-01
SSA	1602.7777	1.84629E + 02	2103.2159	3.02030E + 02	1200.2795	1.86608E-01
BWOA	2439.2799	3.82383E + 02	2735.6573	2.46593E + 02	1201.2318	5.84348E-01
MFO	1638.9129	3.69535E + 02	1956.0341	4.45354E + 02	1200.3704	2.50162E-01
SOA	1352.3077	1.87587E + 02	1704.3085	1.81635E + 02	**1200.2028**	**8.92927E-02**
TSA	1951.7134	3.60254E + 02	2269.0931	3.72123E + 02	1201.1155	2.94628E-01
TLBO	1650.1523	2.92586E + 02	1944.8514	5.40353E + 02	1201.5641	3.70105E-01
LSA	**1200.9479**	**1.51257E + 02**	**1564.4724**	2.59065E + 02	1200.5530	2.70114E-01
	F13		F14		F15	
	Avage	Std	Avage	Std	Avage	Std
GWO	1300.2249	**4.29903E-02**	**1400.2242**	1.12212E-01	1501.9157	1.06723E + 00
HHO	1300.4944	1.75577E-01	1400.2459	1.45172E-01	1507.1216	3.75392E + 00
HPO	1300.5010	2.09226E-01	1400.4023	2.48553E-01	1502.3169	1.03619E + 00
SSA	1300.3607	1.73029E-01	1400.4413	2.53068E-01	1501.3407	5.77058E-01
BWOA	1302.3458	1.00251E + 00	1416.8026	9.80996E + 00	2718.1110	2.79063E + 03

Table 4 (continued)

MFO	1300.3669	1.57935E-01	1401.3131	1.97052E+00	1502.1048	1.31971E+00
SOA	1300.8804	8.06660E-01	1406.4269	4.22619E+00	1570.2442	1.15764E+02
TSA	1300.9033	7.51587E-01	1410.9123	1.05740E+01	1813.6231	7.51868E+02
TLBO	1300.4347	7.34791E-02	1400.3967	**5.07606E-02**	1501.6956	**5.33348E-01**
LSA	**1300.1910**	7.02282E-02	1400.2764	6.75515E-02	**1501.2107**	6.36206E-01
	F16		F17		F18	
	Avage	Std	Avage	Std	Avage	Std
GWO	1602.7551	4.19258E-01	24,820.4140	7.74793E+04	11,015.4546	9.32691E+03
HHO	1603.1799	3.73264E-01	45,608.9498	4.52630E+04	12,826.8976	8.46428E+03
HPO	1603.2811	4.62445E-01	4672.3637	2.23567E+03	11,130.5817	7.50249E+03
SSA	1602.9995	3.72352E-01	6627.8919	4.11049E+03	11,627.9544	9.99290E+03
BWOA	1603.7311	3.38014E-01	9876.8465	1.45924E+04	3740.5347	2.42401E+03
MFO	1603.3254	5.38780E-01	70,867.0609	8.95067E+04	11,878.3394	1.51538E+04
SOA	1602.8837	2.73145E-01	263,958.1772	1.80454E+05	10,633.9149	3.02889E+03
TSA	1603.3487	4.78868E-01	510,461.6264	3.36017E+05	12,939.4862	1.09698E+04
TLBO	1602.8972	**2.65046E-01**	4270.2034	2.30274E+03	5122.7304	4.83494E+03
LSA	**1602.2662**	3.05751E-01	**1839.0022**	**1.10524E+02**	**1808.3092**	**7.39303E+00**
	F19		F20		F21	
	Avage	Std	Avage	Std	Avage	Std
GWO	1902.7209	8.66568E-01	8868.2193	5.78006E+03	8987.1258	4.75179E+03
HHO	1904.7027	1.75686E+00	6552.4805	3.46760E+03	9319.7142	8.26677E+03
HPO	1902.7435	9.58475E-01	3830.2130	2.85467E+03	6964.5421	3.88615E+03
SSA	1903.8431	1.45828E+00	6159.4326	5.11421E+03	9529.0228	7.39480E+03
BWOA	1924.7208	2.89854E+01	22,605.9797	6.96757E+04	184,590.4869	7.86305E+05
MFO	1903.0187	1.04551E+00	11,016.1281	1.14348E+04	12,669.3902	1.16516E+04
SOA	1911.0729	8.01496E+00	7317.0094	1.88980E+03	28,434.7016	4.53276E+04
TSA	1912.7613	1.83284E+01	9871.1467	5.24048E+03	49,462.7224	7.89425E+04

Table 4 (continued)

TLBO	1902.0031	**5.25756E-01**	3209.4489	1.99267E+03	2588.4073	2.51284E+02
LSA	**1901.6906**	5.94024E-01	**2002.0537**	**1.48489E+00**	**2117.5061**	**1.55947E+01**
	F22		F23		F24	
	Avage	Std	Avage	Std	Avage	Std
GWO	2285.5655	6.23276E+01	2633.5264	3.47638E+00	2543.6391	3.14967E+01
HHO	2327.9463	7.31593E+01	**2500.0000**	**0.00000E+00**	2586.5613	2.44666E+01
HPO	2305.2269	7.89118E+01	**2500.0000**	**0.00000E+00**	2566.5938	3.16486E+01
SSA	2275.4347	6.56510E+01	2629.4575	1.01843E-04	**2534.8457**	2.26120E+01
BWOA	2437.2275	9.74387E+01	**2500.0000**	**0.00000E+00**	2595.1449	**1.23340E+01**
MFO	2330.6703	8.75456E+01	2637.1887	1.08958E+01	2551.3704	1.88547E+01
SOA	2311.3580	5.60100E+01	**2500.0000**	**0.00000E+00**	2585.3719	2.27450E+01
TSA	2360.2906	8.85419E+01	2636.0565	3.26819E+01	2572.9484	2.44993E+01
TLBO	2228.5902	**4.25537E+00**	2629.4575	1.57529E-12	2550.4046	4.62088E+01
LSA	**2222.8540**	5.32006E+00	2629.4575	3.71300E-13	2538.1719	2.47893E+01
	F25		F26		F27	
	Avage	Std	Avage	Std	Avage	Std
GWO	2692.9333	1.67354E+01	2700.1609	4.99284E-02	3021.9060	1.12136E+02
HHO	2699.2862	3.19227E+00	2700.3597	1.66692E-01	2900.0000	**0.00000E+00**
HPO	2695.0774	1.23218E+01	2700.4038	1.55911E-01	2881.2304	5.77732E+01
SSA	2683.9658	2.67555E+01	2700.2317	1.00065E-01	3004.7134	1.78426E+02
BWOA	2697.1775	8.76470E+00	2701.4975	1.48063E+00	2887.4249	3.95780E+01
MFO	2697.9483	1.19654E+01	2700.3454	1.27208E-01	3029.8593	1.73959E+02
SOA	2699.0402	**2.38754E+00**	2701.1593	1.73793E+00	**2865.6306**	7.10854E+01
TSA	2701.7789	2.94306E+00	2706.2469	2.20796E+01	3155.6174	1.24477E+02
TLBO	**2662.7561**	2.49444E+01	2700.1811	**3.94067E-02**	2997.6947	1.65760E+02
LSA	2662.9213	2.43880E+01	**2700.1530**	9.95153E-02	3017.0294	1.65329E+02

Table 4 (continued)

	F28		F29		F30	
	Avage	Std	Avage	Std	Avage	Std
GWO	3284.9543	9.71994E+01	210,968.0651	6.37969E+05	4334.8720	8.91234E+02
HHO	**3000.0000**	**0.00000E+00**	187,104.5980	8.18936E+05	5039.5694	7.60581E+02
HPO	3009.0348	4.04049E+01	360,192.9237	7.33729E+05	4085.9483	5.43899E+02
SSA	3201.6662	5.65332E+01	5068.3765	2.23908E+03	4289.5109	6.64679E+02
BWOA	**3000.0000**	**0.00000E+00**	**3278.9238**	**3.30452E+02**	3848.8791	1.12257E+03
MFO	3188.0670	1.30118E+01	4123.7981	6.34265E+02	3683.8613	**9.37370E+01**
SOA	**3000.0000**	**0.00000E+00**	3596.2931	3.42329E+02	5658.2707	1.65245E+03
TSA	3648.2761	2.93305E+02	1,925,676.8894	3.04470E+06	5170.5328	5.62931E+02
TLBO	3219.3493	4.70776E+01	43,922.6854	8.99989E+04	3838.0880	3.74922E+02
LSA	3178.8117	3.48971E+00	13,255.8509	3.20348E+04	**3596.8012**	1.25551E+02
	F31		F32		F33	
	Avage	Std	Avage	Std	Avage	Std
GWO	2.03907E+08	4.82615E+08	1668.7929	2.91711E+02	730.2925	9.50160E+00
HHO	444,014.2003	3.28970E+05	1983.9660	2.02991E+02	793.5116	1.81555E+01
HPO	4258.0138	3.15042E+03	1940.6188	3.59744E+02	747.3638	1.44439E+01
SSA	3014.9256	3.42207E+03	1938.0480	3.14606E+02	738.5734	1.57097E+01
BWOA	4.32667E+09	2.70767E+09	2614.0298	3.99210E+02	807.6041	2.97580E+01
MFO	1.88029E+08	4.50827E+08	1990.6956	3.19331E+02	734.7929	1.08735E+01
SOA	5.46057E+09	2.44291E+09	2000.6583	**1.91209E+02**	755.3606	1.64074E+01
TSA	2.18337E+09	2.42227E+09	2179.2532	3.35972E+02	802.0716	3.09218E+01
TLBO	1405.4536	6.17055E+02	1902.5692	4.17357E+02	**724.8578**	**6.44039E+00**
LSA	**101.3899**	**3.94191E+00**	**1485.2930**	2.54472E+02	729.6887	8.15806E+00
	F34		F35		F36	
	Avage	Std	Avage	Std	Avage	Std
GWO	1902.5353	8.23058E-01	25,556.1718	8.46286E+04	1607.6003	1.48008E+01

Table 4 (continued)

HHO	1909.2929	4.45981E+00	29,305.5409	3.42607E+04	1611.8300	9.54484E+00
HPO	1903.4585	1.46518E+00	6014.0886	4.41564E+03	1606.4848	1.39540E+01
SSA	1901.5311	7.12197E-01	6519.5331	4.68533E+03	1601.0683	5.38584E-01
BWOA	63,053.0122	1.38377E+05	134,941.0442	4.02536E+05	1640.5201	6.91740E+01
MFO	1903.7229	5.69356E+00	166,026.8452	2.79813E+05	1604.5447	6.50658E+00
SOA	37,769.9149	5.01122E+04	361,505.9992	1.98667E+05	1607.9505	7.48245E+00
TSA	42,562.1626	7.50114E+04	459,202.0444	4.10504E+05	1619.7106	1.89657E+01
TLBO	1902.6576	1.43506E+00	4032.8841	1.12822E+03	1601.3840	2.93073E-01
LSA	**1900.9662**	**4.47520E-01**	**1795.5017**	**5.62097E+01**	**1600.6600**	**1.53184E-01**
	F37		F38		F39	
	Avage	Std	Avage	Std	Avage	Std
GWO	10,308.0604	4.40677E+03	2310.0420	3.41865E+01	2746.9784	**1.06160E+01**
HHO	11,470.7522	9.86467E+03	2314.2470	4.86108E+00	2781.6500	1.03406E+02
HPO	5550.9542	3.71685E+03	2367.3745	2.73715E+02	2771.4691	1.36124E+01
SSA	6608.0698	5.03890E+03	2341.4958	2.13088E+02	2725.0554	7.74500E+01
BWOA	8157.9587	1.00508E+04	2817.5149	3.88472E+02	2793.3408	6.85923E+01
MFO	20,895.3920	5.77797E+04	2309.6302	1.45067E+01	2754.4516	4.29342E+01
SOA	209,779.2113	4.87385E+05	2601.9910	1.63004E+02	2754.5643	1.18158E+02
TSA	842,967.0293	3.63649E+06	2553.3653	3.10692E+02	2815.4810	2.37148E+01
TLBO	2413.9364	1.30395E+02	**2303.3124**	**1.14951E+00**	2689.8806	1.05603E+02
LSA	**2144.9284**	**5.47718E+01**	2303.8299	1.99960E+00	**2600.8587**	1.19417E+02
	F40					
	Avage	Std	Number of champions	Number of runners-up	ARV	Rank
GWO	2932.0090	**1.54126E+01**	1	6	4.75	5
HHO	2935.8414	2.19992E+01	0	1	6.225	7
HPO	2925.4023	7.88282E+01	4	4	4.225	3
SSA	**2919.8040**	2.43778E+01	2	5	4.35	4

Table 4 (continued)

BWOA	3228.9628	2.09030E+02	1	2	8.35	9
MFO	2938.0174	2.94748E+01	0	1	6.075	6
SOA	3161.2961	1.67114E+02	4	2	6.6	8
TSA	3120.7528	3.02077E+02	0	0	8.9	10
TLBO	2926.3744	2.51975E+01	4	13	3.425	2
LSA	2928.7641	2.26328E+01	**24**	**6**	**2.1**	**1**

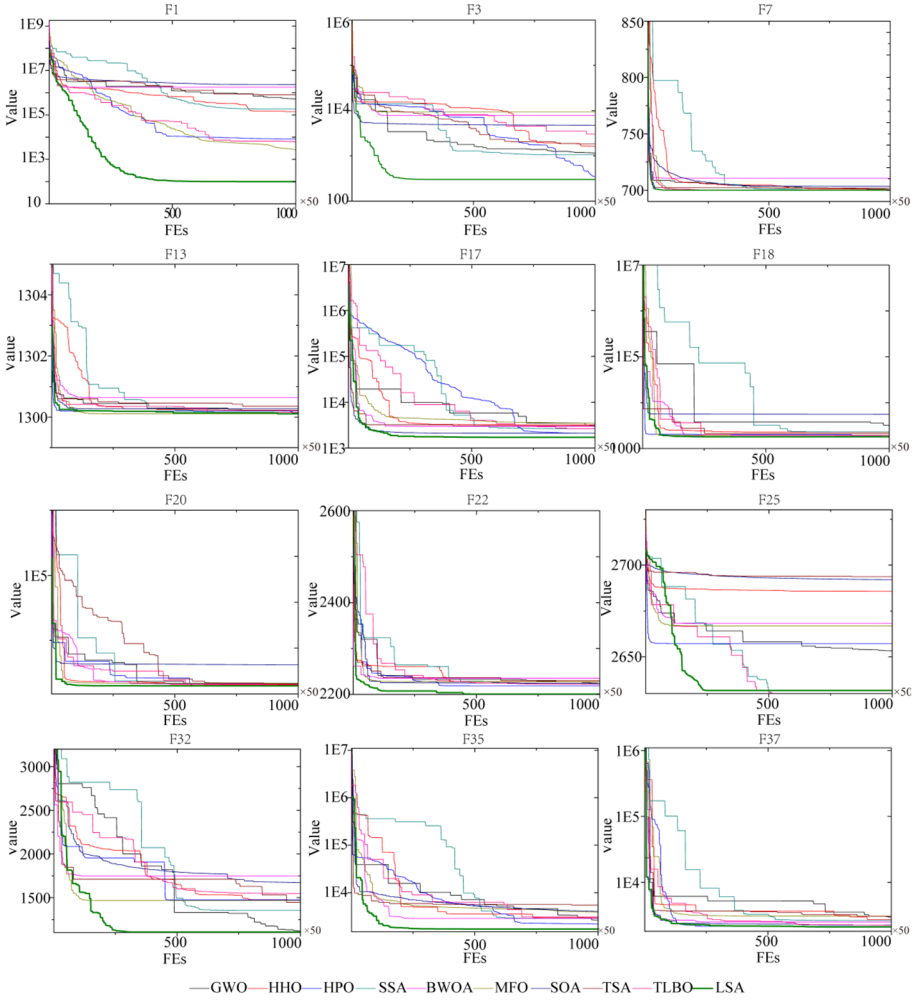

Fig. 9 The convergence curves of LSA and other original algorithms on 12 benchmark functions

4.1.2 Comparison with well-known original algorithms

This subsection tests the performance of the target algorithm LSA and other well-known original algorithms, including GWO (Mirjalili et al. 2014; Long et al. 2020), HHO (Chen et al. 2020b), HPO (Naruei et al. 2022), MFO (Mirjalili 2015b), SSA (Mirjalili et al. 2017; Abualigah et al. 2020), BWOA (Hayyolalam and Kazem 2020), SOA (Ramshanker and Chakraborty 2022), TLBO (Rao et al. 2011), and TSA (Kaur et al. 2020). In the CEC 2014 benchmark functions, F1-F3 are single-mode functions, and they are used to test the development ability of the algorithm; F4-F15 are multi-mode functions, and they have multiple local optimal values. If the algorithm's exploration ability is not sufficient, it will converge prematurely and quickly, or even fall into a local optimum, resulting in low convergence accuracy. F16-F21 are mixing functions, and F22-F30 are composition functions. These two types of functions are more challenging than the previous two types of functions, so

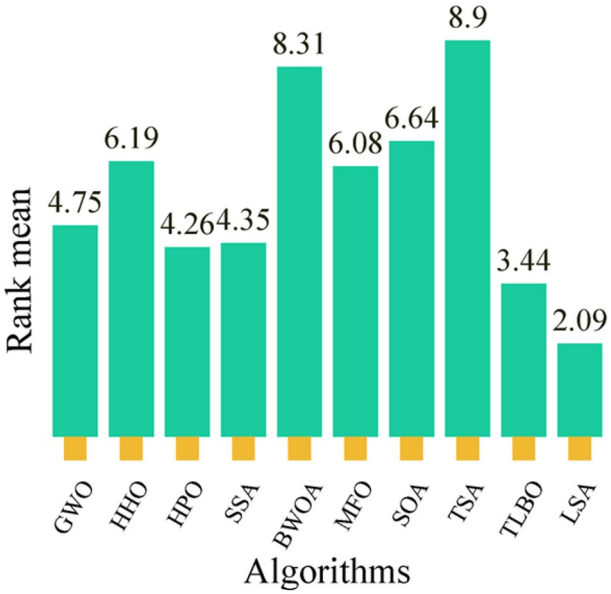

Fig. 10 The results of the Friedman test for the LSA and other original algorithms

they can better reflect the search performance of the algorithm. F31-F40 are the CEC 2020 benchmark functions. Table 4 presents the average and variance results of each algorithm for solving the CEC 2014 and CEC 2020 benchmark functions, and the best results are highlighted in bold. Figure 9 shows the convergence effect of the LSA and the comparison algorithms in solving 12 benchmark functions. Figure 10 illustrates the Friedman rank sum sorting results of each algorithm. Table 5 and Fig. 11 show the results of the Wilcoxon rank-sum experiment with a 5% significance.individual's active learning mode

It can be seen from Table 4 that LSA ranks first on 24 functions and ranks second on six functions, with an overall rank of #1 (average rank AVG of 2.1). The results of the p values in Table 5 indicate that compared with the algorithms GWO, HHO, HPO, SSA, BWOA, MFO, SOA, TSA, and TLBO, the LSA obtained 27, 37, 35, 33, 38, 36, 36, 40, and 24 victories, showing a significant difference. Although the LSA algorithm obtained 3, 5, and 7 optimal results when solving Unimodal Functions (F1-F3), Multimodal Functions (F4-F16), and Hybrid functions (F17-F22) problems respectively, its overall optimal result rate is only $(3+5+7)/22=68.18\%$. Moreover, when solving Composition Functions (F23-F30) problems, LSA only achieved 2 optimal results, whereas SOA obtained 3 optimal results. This indicates that SOA has certain advantages in solving Composition Functions problems. This also confirms the "No Free Lunch" (NFL) theorem, demonstrating that there is no one-size-fits-all solution for all optimization problems. However, compared to other algorithms, the LSA algorithm obtains the most optimal results in solving these two types of problems. The bubble chart in Fig. 11 vividly demonstrates the statistical superiority of the proposed algorithm over these well-known original algorithms in terms of search results.

According to the results of the Friedman rank sum test shown in Fig. 10, the LSA algorithm achieved the highest rank among all algorithms, with a rank mean of 2.09. Overall, these experimental results indicate that the LSA performs superiorly on the CEC 2014 and CEC 2020 functions and is stronger than other comparison algorithms.

Table 5 The *p*-value and significance of the Wilcoxon signed-rank test

Fun	GWO	HHO	HPO	SSA	BWOA	MFO	SOA	TSA	TLBO
F1	6.6e-8(+)	6.6e-8(+)	6.6e-8(+)	6.6e-8(+)	6.6e-8(+)	6.5e-8(+)	6.6e-8(+)	6.6e-8(+)	6.6e-8(+)
F2	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	5.2e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F3	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.6e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F4	3.8e-6(+)	2.2e-4(+)	7.4e-2(=)	1.7e-5(+)	5.3e-8(+)	8.1e-4(+)	5.3e-8(+)	3.8e-6(+)	5.4e-1(=)
F5	1.2e-6(+)	4.6e-4(+)	6.7e-8(-)	1.6e-6(+)	6.7e-8(+)	6.5e-5(+)	6.7e-8(+)	2.6e-5(+)	6.6e-5(+)
F6	1.5e-1(=)	6.7e-8(+)	1.3e-4(+)	4.1e-2(+)	6.7e-8(+)	9.7e-3(+)	6.7e-8(+)	2.2e-7(+)	4.6e-4(+)
F7	6.8e-7(+)	6.7e-8(+)	4.1e-2(+)	8.0e-1(+)	6.7e-8(+)	8.6e-2(=)	6.7e-8(+)	6.7e-8(+)	1.5e-1(=)
F8	6.0e-1(=)	5.1e-3(+)	3.3e-5(+)	3.0e-3(+)	6.7e-8(+)	1.6e-1(=)	2.0e-6(+)	2.2e-7(+)	2.9e-2(-)
F9	2.2e-1(=)	2.9e-7(+)	6.5e-5(+)	3.2e-5(+)	6.6e-8(+)	2.5e-5(+)	1.2e-6(+)	2.2e-7(+)	5.1e-3(+)
F10	1.3e-2(+)	5.2e-5(+)	1.2e-1(=)	3.9e-7(+)	6.7e-8(+)	2.4e-4(+)	4.7e-2(+)	1.6e-7(+)	6.8e-2(-)
F11	3.6e-1(=)	2.9e-7(+)	5.1e-3(+)	7.5e-6(+)	6.7e-8(+)	1.8e-3(+)	8.6e-2(=)	1.2e-6(+)	1.5e-2(+)
F12	8.8e-1(=)	1.2e-1(=)	1.2e-3(+)	1.0e-3(+)	6.7e-8(+)	9.7e-3(+)	3.5e-6(-)	4.5e-6(+)	1.2e-7(+)
F13	3.6e-2(+)	3.9e-7(+)	1.6e-6(+)	2.0e-4(+)	6.7e-8(+)	1.0e-4(+)	1.2e-7(+)	6.7e-8(+)	1.2e-1(=)
F14	6.0e-3(-)	7.6e-2(+)	7.6e-2(=)	2.0e-1(=)	6.7e-8(+)	2.4e-4(+)	2.2e-7(+)	9.7e-6(+)	3.6e-1(=)
F15	5.3e-2(=)	9.0e-8(+)	3.3e-5(+)	2.0e-1(=)	6.7e-8(+)	7.1e-3(+)	9.0e-8(+)	9.0e-8(+)	6.0e-2(-)
F16	5.6e-4(+)	1.6e-6(+)	1.2e-6(+)	2.7e-6(+)	2.9e-1(=)	2.0e-6(+)	2.7e-6(+)	9.0e-7(+)	1.0e-4(+)
F17	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.6e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F18	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.6e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F19	2.6e-5(+)	3.9e-7(+)	3.3e-5(+)	9.0e-7(+)	6.7e-8(+)	5.8e-6(+)	6.7e-8(+)	1.6e-7(+)	1.5e-1(=)
F20	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.6e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F21	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)	6.6e-8(+)	6.7e-8(+)	6.7e-8(+)	6.7e-8(+)
F22	2.2e-7(+)	1.6e-7(+)	4.6e-4(+)	1.2e-6(+)	7.1e-3(+)	1.2e-7(+)	2.2e-7(+)	1.2e-7(+)	6.0e-3(+)
F23	8.0e-9(+)	4.7e-10(+)	4.7e-10(+)	8.0e-9(+)	8.0e-9(+)	2.1e-3(+)	4.7e-10(-)	2.1e-7(+)	3.4e-1(=)
F24	7.6e-1(=)	5.1e-5(+)	7.1e-3(+)	4.6e-1(=)	1.1e-1(=)	1.8e-3(+)	2.4e-5(+)	1.2e-5(+)	1.5e-3(+)
F25	4.1e-5(+)	3.0e-8(+)	4.5e-6(+)	7.1e-3(+)	6.7e-8(+)	2.0e-6(+)	2.9e-7(+)	3.9e-7(+)	1.8e-1(=)
F26	8.4e-1(=)	3.3e-5(+)	4.5e-6(+)	2.7e-2(+)	6.7e-8(+)	4.1e-5(+)	8.3e-4(+)	6.7e-8(+)	2.2e-1(=)
F27	9.6e-2(=)	5.5e-4(+)	6.9e-4(+)	8.3e-3(+)	5.2e-5(+)	3.1e-2(+)	8.3e-4(-)	9.7e-6(+)	2.0e-1(=)
F28	8.3e-4(+)	7.9e-9(-)	2.7e-7(+)	7.1e-1(=)	3.7e-4(+)	5.6e-4(+)	7.9e-9(+)	6.7e-8(+)	9.7e-6(+)
F29	9.7e-6(+)	3.6e-3(+)	5.8e-6(+)	1.6e-5(+)	6.7e-8(+)	1.6e-5(+)	1.3e-4(+)	9.0e-7(+)	9.7e-6(+)
F30	3.9e-7(+)	6.7e-8(+)	3.6e-3(+)	1.2e-6(+)	2.0e-6(+)	7.1e-3(+)	6.7e-8(+)	6.7e-8(+)	2.0e-1(=)
F31	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	9.2e-8(+)	6.8e-8(+)	9.1e-8(+)	6.8e-8(+)	6.8e-8(+)	1.2e-6(+)
F32	2.7e-2(+)	7.6e-6(+)	2.9e-5(+)	8.3e-5(+)	1.7e-7(+)	1.3e-5(+)	4.5e-6(+)	2.4e-6(+)	2.6e-2(+)
F33	9.2e-1(=)	6.8e-8(+)	1.8e-4(+)	3.4e-2(+)	6.8e-8(+)	1.3e-1(=)	3.5e-6(+)	6.8e-8(+)	1.7e-1(=)
F34	5.9e-6(+)	6.8e-8(+)	3.4e-7(+)	2.6e-2(+)	6.8e-8(+)	7.6e-4(+)	6.8e-8(+)	6.8e-8(+)	3.0e-4(+)
F35	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)
F36	3.9e-1(=)	6.8e-8(+)	1.4e-5(+)	1.2e-3(+)	4.5e-7(+)	6.8e-8(+)	5.9e-6(+)	2.1e-6(+)	6.8e-8(+)
F37	6.8e-8(+)	6.8e-8(+)	1.7e-7(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	7.9e-8(+)	6.8e-8(+)	1.1e-7(+)
F38	1.1e-2(+)	7.9e-8(+)	7.2e-2(=)	9.0e-1(=)	6.8e-8(+)	5.4e-1(=)	6.8e-8(+)	1.6e-5(+)	4.4e-2(+)
F39	1.8e-4(+)	7.5e-7(+)	8.5e-8(+)	3.6e-5(+)	5.7e-7(+)	2.1e-7(+)	2.7e-4(+)	5.4e-8(+)	2.0e-2(+)
F40	4.2e-1(=)	1.2e-2(+)	7.2e-4(+)	4.7e-1(=)	6.8e-8(+)	3.2e-2(+)	4.5e-7(+)	1.0e-4(+)	9.8e-1(=)
+/=/-	27/12/1	37/2/1	35/4/1	33/7/0	38/2/0	36/4/0	36/1/3	40/0/0	24/14/2

Figure 9 illustrates the convergence curves of benchmark functions for GWO, HHO, HPO, SSA, BWOA, MFO, SOA, TSA, TLBO, and the proposed LSA algorithm. Based on the convergence curves of functions F1, F3, F7, F13, F17, F18, F20, F22, F25,

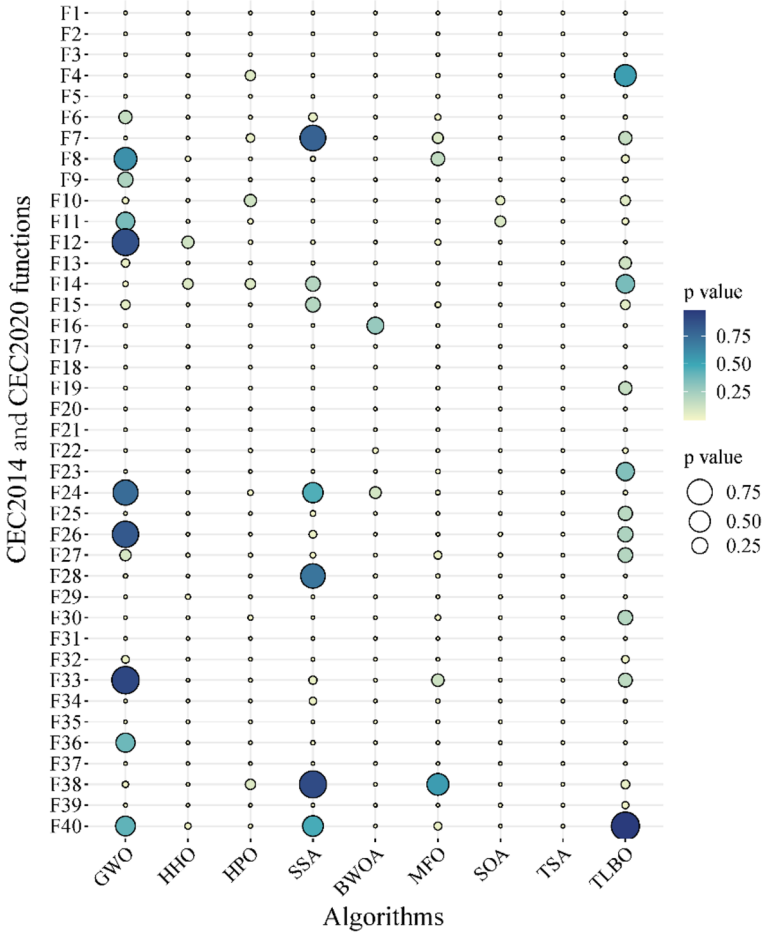

Fig. 11 The graphical representation of the Wilcoxon signed rank test results of the LSA algorithm compared to the original algorithms

F32, F35, and F37, the LSA algorithm achieves the highest fitness values and the fastest convergence speed among these unimodal functions, multimodal functions, hybrid functions, and composition functions. In contrast, other algorithms fail to obtain global solutions due to being trapped in local optima. Therefore, the experimental results demonstrate that LSA effectively utilizes its exploitation capability for unimodal functions and exploration capability for multimodal functions. The incorporation of exploration and exploitation stages ensures the global convergence of the LSA algorithm.

The stability of an algorithm is also an important indicator of whether it is good or bad. To examine the reliability, we selected F1, F3, F10, F18, F21, F26, F21, F30, and F34 as the test subjects. From the box diagram in Fig. 12, it can be seen that the box diagram of LSA algorithm is the most flat. This indicates that LSA algorithm has good stability.

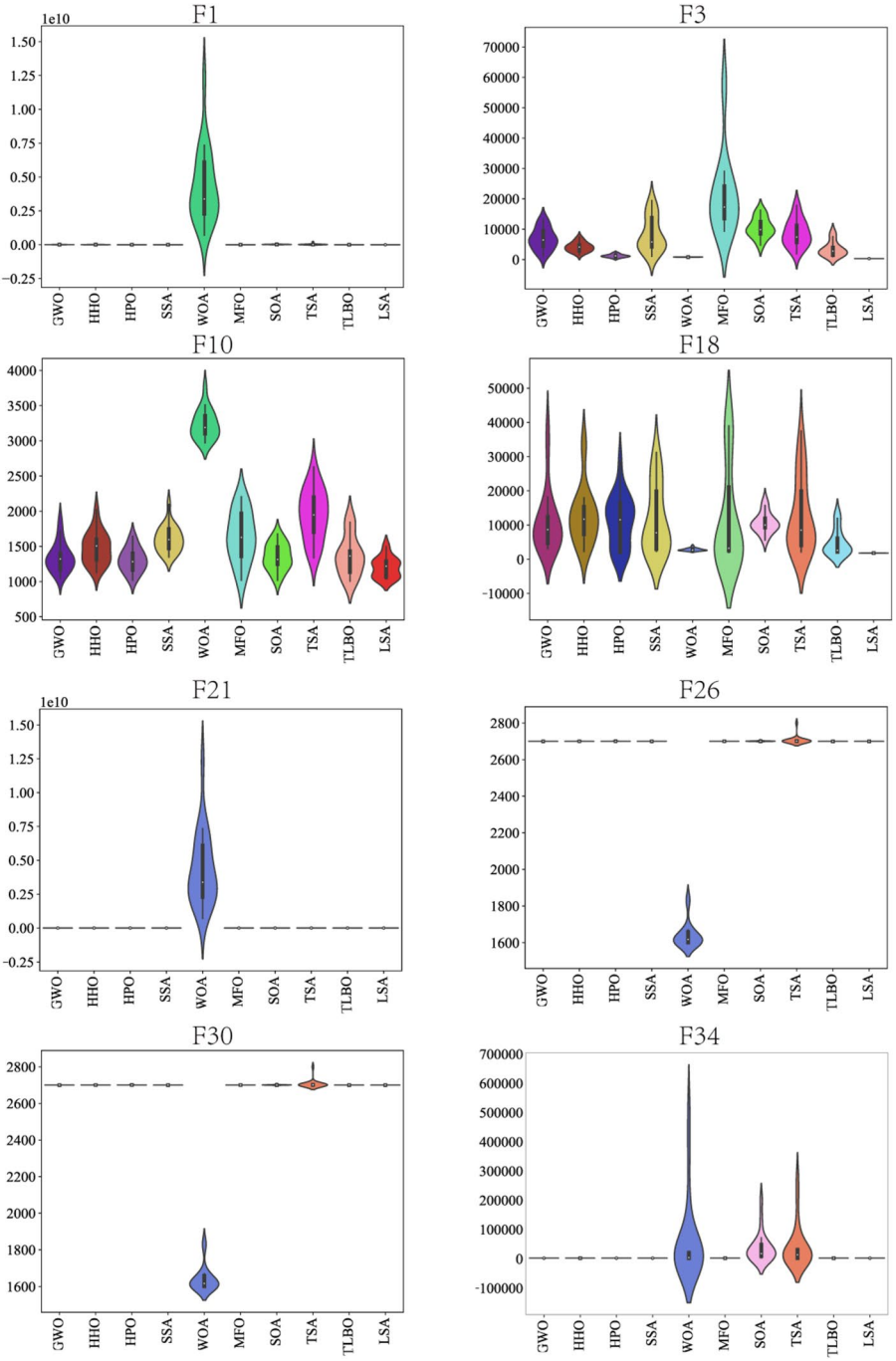

Fig. 12 The box plot of the results by well-known original algorithms

Table 6 The Average and Std of the LSA and other advanced improved algorithms for CEC 2020

	F31		F32		F33	
	Avage	Std	Avage	Std	Avage	Std
GNHGWO	2.29749E+04	8.90630E+03	1509.3832	4.71149E+02	732.8926	5.76382E+00
GWOCS	4.41822E+07	1.33006E+08	1644.1078	2.19760E+02	735.7466	1.20817E+01
HPSOBOA	2.89045E+10	4.46195E+08	3046.7200	1.75496E+02	871.3040	4.02032E+00
NCHHO	4.43941E+09	2.15579E+09	2496.4916	1.42381E+02	800.8964	1.89861E+01
PSOBOA	9.80322E+09	3.52858E+09	2975.6998	1.96950E+02	834.0457	1.00161E+01
HFPSO	2.29067E+03	2.99775E+03	1569.7595	2.05936E+02	722.0455	5.05387E+00
FDB-AGDE	**100**	**1.75087E-12**	1379.5603	**8.61286E+01**	**719.4734**	**2.23856E+00**
dFDB-MRFO	1832.636718	1.32543E+03	1765.1352	2.35801E+02	739.9340	1.03716E+01
AFDB-SFS	174.297294	2.41013E+02	**1378.1035**	1.74399E+02	721.5749	4.09895E+00
FDB-TLABC	2036.630182	7.27001E+02	2091.4635	2.95764E+02	727.7579	6.60168E+00
TSALSHADE	7813.065933	4.94819E+03	1530.9438	1.62318E+02	732.3411	8.31024E+00
LSA	101.3899171	3.94191E+00	1485.2930	2.54472E+02	729.6887	8.15806E+00
	F34		F35		F36	
	Avage	Std	Avage	Std	Avage	Std
GNHGWO	1902.4678	4.75003E-01	6809.3620	5.22632E+03	1601.0570	1.96680E-01
GWOCS	1906.8593	1.83302E+01	43,881.9263	1.13483E+05	1620.8909	1.36073E+01
HPSOBOA	477,424.5836	5.58245E+05	719,221.5580	1.02389E+04	1731.1794	3.90045E+01
NCHHO	7796.5739	6.21368E+03	98,606.1207	7.91887E+04	1616.7368	1.25908E+01
PSOBOA	547,051.2652	2.53396E+05	589,052.8448	9.49337E+04	1631.2794	1.37839E+01
HFPSO	**1900.9075**	4.73336E-01	5481.3176	3.41465E+03	1605.8144	8.15710E+00
FDB-AGDE	1903.1957	2.26815E-01	1706.6560	5.38846E+00	1601.3187	**7.01129E-04**
dFDB-MRFO	1903.4178	7.89383E-01	3918.7789	1.84470E+03	1600.9950	1.94709E-01
AFDB-SFS	1903.0536	4.01990E-01	2170.5805	7.23065E+02	1600.9237	1.56906E-01
FDB-TLABC	1901.7999	4.98936E-01	2503.1503	2.16456E+02	1600.7051	1.83889E-01
TSALSHADE	1903.2750	1.43789E+00	1,498,978.3995	2.80362E+06	1601.2228	3.47466E-01
LSA	1900.9662	4.47520E-01	1795.5017	5.62097E+01	**1600.6600**	1.53184E-01
	F37		F38		F39	
	Avage	Std	Avage	Std	Avage	Std

Table 6 (continued)

GNHGWO	9904.0481	4.41090E+03	2303.3639	2.08884E+00	2752.6495	6.42091E+00
GWOCS	9800.0872	4.02991E+03	2306.9350	7.25168E+00	2610.2406	1.16546E+02
HPSOBOA	3.25662E+06	8.96197E+05	3255.2398	3.20567E+00	3172.2764	7.30846E+01
NCHHO	1.01195E+06	1.34855E+06	2810.4899	3.48426E+02	2880.2015	6.39614E+01
PSOBOA	1.36267E+05	3.05329E+05	3315.6645	3.50204E+02	3002.4244	7.46094E+01
HFPSO	2748.4970	7.04504E+02	2393.3900	2.89001E+02	2731.4673	8.24944E+01
FDB-AGDE	2100.6788	4.59590E-01	2309.3270	3.55972E+00	2719.0915	5.63984E+01
dFDB-MRFO	2666.9684	3.10524E+02	2300.9809	5.86035E-01	2686.9302	1.08507E+02
AFDB-SFS	2226.8862	3.55730E+01	**2292.6142**	2.67242E+01	2693.4764	9.93525E+01
FDB-TLABC	2216.7655	2.69779E+01	2301.2093	**4.26428E-01**	2704.2605	7.25950E+01
TSALSHADE	457,998.8214	7.99551E+05	2319.2812	5.09644E+01	2618.8838	9.67995E+01
LSA	2144.9284	5.47718E+01	2303.8299	1.99960E+00	**2600.8587**	1.19417E+02
	F40					
	Avage	Std	Number of champions	Number of runners-up	ARV	Rank
GNHGWO	2946.7370	**6.52872E-01**	0	0	6.4	7
GWOCS	2930.2175	1.73780E+01	0	1	7.2	9
HPSOBOA	4727.0939	2.38437E+01	0	0	11.7	12
NCHHO	3167.4394	1.15613E+02	0	0	9.9	10
PSOBOA	3971.7804	2.24326E+02	0	0	10.9	11
HFPSO	2922.1098	2.45300E+01	1	1	5.5	6
FDB-AGDE	2937.2386	1.91791E+01	4	1	4.1	3
dFDB-MRFO	2923.7268	2.32046E+01	0	1	5.2	5
AFDB-SFS	2932.1738	2.33114E+01	2	1	3.4	2
FDB-TLABC	2926.2011	2.24320E+01	0	1	4.4	4
TSALSHADE	**2880.4910**	1.00513E+02	1	0	6.5	8
LSA	2928.7641	2.26328E+01	2	4	2.8	1

Bold entries indicates optimal result

4.1.3 Comparison of LSA with recently proposed advanced improved algorithms

In this subsection, LSA is compared with 6 high-performance improved algorithms proposed in recent years, including GNHGWO (Akbari et al. 2021), GWOCS (Wang et al. 2022b), HPSOBOA (Zhang et al. 2020), NCHHO (Dehkordi et al. 2021), PSOBOA (Zhang et al. 2020), HFPSO (Aydilek 2018), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC (Duman et al. 2022), TSALSHADE (Abdesslem layeb 2023), and ISSA (Dorian Sidea 2024).

Table 6 shows the experimental results of these 12 algorithms on CEC 2020 functions. Table 7 presents the p-value results of the Wilcoxon test. It can be seen from Table 6 that the LSA obtained an AVG value of 2.8, ranking first overall. Meanwhile, the p-values in Table 7 and Fig. 16 indicate that compared to the algorithms GNHGWO, GWOCS, HPSOBOA, NCHHO, HFPSO, PSOBOA, FDB-AGDE, dFDB-MRFO, AFDB-SFS, FDB-TLABC, and TSALSHADE, the LSA obtained 7, 6, 10,10,10,7,3,6,4,6, and 6 victories, respectively, showing a significant difference. The Friedman rank sum value of the LSA in Fig. 13 is 2.8, which is the smallest among all algorithms. The sorting results indicate that the LSA ranks first among all algorithms. These experimental results show that the LSA achieves superior performance on the CEC 2020 function and is stronger than other algorithms. The excellent performance of the LSA makes it a new optimizer that can be applied to solve complex and realistic optimization problems.

Appendix Table 25 presents the results of LSA and ISSA algorithms in solving CEC 2014 problem. From appendix Table 25, it can be observed that the LSA algorithm achieved victories in 28 test functions. Whether in single-modal, multi-modal, hybrid, or composite functions, the LSA algorithm outperformed the ISSA algorithm comprehensively.

The excellent convergence performance of the LSA algorithm is demonstrated in Fig. 14, compared with GNHGWO, GWOCS, HPSOBOA, NCHHO, HFPSO, PSOBOA, FDB-AGDE, dFDB-MRFO, AFDB-SFS, FDB-TLABC, and TSALSHADE. The main reason for this achievement is the combined effect of the LSA algorithm's local exploitation strategies (Active learning strategy and Role models teaching strategy) and global exploration strategy.

In summary, LSA shows strong competitiveness compared with the top-performing meta-heuristic algorithms put forth in recent years. This indicates that our proposed LSA obtains good results in dealing with numerical optimization problems.

To test the stability of the LSA algorithm, we selected F31, F34, F35, F36, F37, and F38 as the test subjects. According to Fig. 15, compared with the distributions of optimal solutions, the box diagram of LSA algorithm is the most flat, indicating its good stability (Fig. 16).

4.1.4 Consumption time cost analysis

Tables 8 and 9 present the time consumption (unit: second) of each algorithm when solving the CEC2014 and CEC2020 functions. To more intuitively show the results, the proportion of time consumption is shown in Figs. 17 and 18.

It can be seen from Table 8 that compared with the original benchmark algorithm, the average time consumption of the LSA when solving the CEC 2014 and CEC 2020 problems is 0.305 s, ranking in the middle of the nine algorithms. Figure 17 illustrates the

percentage of the cost time by each algorithm in executing different test functions, providing a more intuitive reflection of the performance of the proposed algorithm in terms of execution time. Meanwhile, Table 9 and Fig. 18 demonstrate that compared with the newly proposed improved algorithm, the average time consumption of the LSA in solving the CEC 2014 and CEC 2020 problems ranks 6th. The relatively high time expense of the target algorithm is mainly due to the computation cost of determining the number of beneficiaries in the exemplar instructing process, which is calculated using Formula (11).

4.1.5 Parameter sensitivity analysis

Different parameters may have different influences on the performance of the algorithm. To explore the influence of the parameter *sub* (the number of subjects taught by role models) on the performance of the LSA, this paper selected different values of *sub,* i.e., 2, 3, 4, 5, 6, 7, and 8, to conduct experiments, and the values of other parameters remained the same. In addition, we discussed the impact of different values of parameter y^0 and parameter $\delta_{init} > 1$ on the algorithm. The test case was obtained from CEC 2014 and CEC 2020. Each test case was independently run 20 times, and the final results are presented in Table 10, where the optimal results are highlighted in bold.

It can be seen from Table 10 that when *sub* equals 3, the number of average optimal values obtained by the algorithm is the largest, indicating that the number of role models in teaching must be controlled within a certain range so that the teaching object effect can reach the best level. This is consistent with the analysis results in Sect. 3.3.3.

From the statistical results in Appendix Table 23, it can be observed that the best results were achieved when $y^0 = 0.75$, with a total of 21 instances. On the other hand, the least favorable results were obtained when $y^0 = 0.5$. Therefore, the value of y^0 does have some impact on the target algorithm, although the fluctuation in the results of the problem-solving process is not significant.

According to the statistical results from Appendix Table 24, it can be observed that when $\delta_{init} = 1$, the proposed algorithm achieves the best results with a count of 15, outperforming the cases where $\delta_{init} > 1$. The analysis suggests that this improvement can be attributed to the excessive global exploration carried out by the algorithm during the iterative process when $\delta_{init} > 1$, which consequently undermines the algorithm's capability for local exploitation.

4.2 Results of real-world constrained optimization problems

The design of many real-world engineering structures is usually limited by various conditions. When solving such problems, engineers need to deal with additional constraints. To test the LSA algorithm in solving engineering real optimization problems, this paper selected 6 real-world engineering design problems, such as speed reducer design et al.

4.2.1 Speed Reducer Design (SRD)

The goal of SRD is to design a reducer with a minimum weight. SRD contains seven design variables and eleven inequality constraints. A detailed description of this problem can be found in the reference (Dhiman and Kumar 2017). The mathematical model of the problem is as follows:

Table 7 The *p*-value and significance of the Wilcoxon signed rank test between the LSA and other recently improved algorithms

Fun	GNH-GWO	GWOCS	HPSO-BOA	NCHHO	PSOBOA	HFPSO	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB-TLABC	TSAL SHADE
F31	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.7e-8(+)	5.9e-8(-)	6.8e-8(+)	6.6e-5(+)	7.9e-8(+)	6.8e-8(+)
F32	3.2e-1(=)	1.8e-2(+)	6.8e-8(+)	7.9e-8(+)	6.8e-8(+)	3.6e-2(+)	9.1e-2(=)	1.1e-3(+)	3.4e-1(=)	1.8e-6(+)	4.9e-1(=)
F33	7.6e-2(=)	1.5e-1(=)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.0e-3(-)	8.6e-6(-)	1.3e-1(=)	7.6e-4(-)	7.8e-1(=)	6.8e-1(=)
F34	6.8e-8(+)	6.6e-5(+)	5.7e-8(+)	6.8e-8(+)	6.8e-8(+)	8.0e-1(=)	1.6e-1(=)	9.6e-2(=)	6.9e-1(=)	3.4e-4(+)	1.2e-7(+)
F35	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	6.7e-8(+)	9.2e-8(-)	6.8e-8(+)	3.4e-2(+)	6.8e-8(+)	6.8e-8(+)
F36	6.8e-8(+)	3.9e-7(+)	6.8e-8(+)	4.5e-7(+)	6.8e-8(+)	1.0e-3(+)	6.8e-8(+)	2.7e-4(+)	2.4e-1(=)	1.6e-1(=)	1.1e-2(+)
F37	6.8e-8(+)	6.8e-8(+)	6.6e-8(+)	6.8e-8(+)	6.8e-8(+)	1.0e-4(+)	8.3e-5(+)	2.2e-7(+)	6.0e-1(=)	4.6e-4(+)	6.8e-8(+)
F38	3.5e-1(=)	3.8e-1(=)	6.8e-8(+)	6.8e-8(+)	6.8e-8(+)	3.1e-2(+)	6.8e-8(-)	3.1e-6(-)	7.6e-6(-)	1.3e-5(-)	1.1e-1(=)
F39	3.7e-7(+)	1.6e-1(=)	4.8e-8(+)	5.4e-8(+)	5.4e-8(+)	1.4e-5(+)	6.6e-2(=)	2.4e-2(+)	7.9e-3(+)	1.4e-2(+)	5.5e-2(+)
F40	8.3e-5(+)	7.1e-1(=)	6.8e-8(+)	1.1e-7(+)	6.8e-8(+)	6.7e-1(=)	7.8e-4(+)	2.9e-1(=)	1.9e-2(+)	3.9e-1(=)	3.6e-1(=)
+/=/-	7/3/0	6/4/0	10/0/0	10/0/0	10/0/0	7/2/1	3/3/4	6/3/1	4/4/2	6/3/1	6/4/0

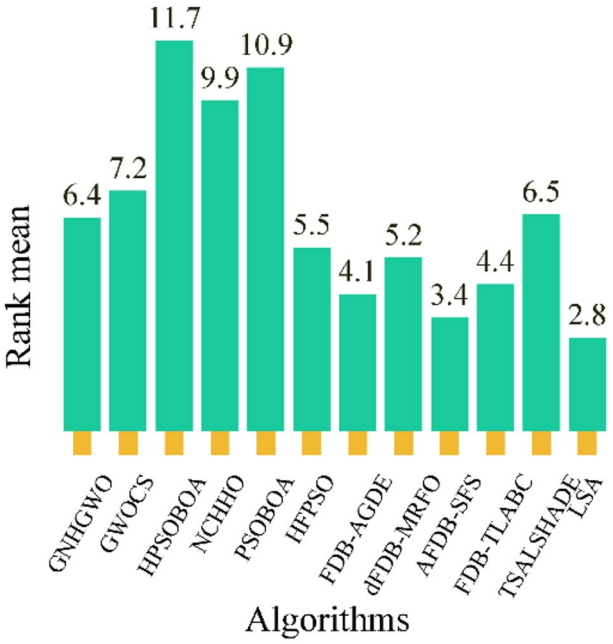

Fig. 13 Friedman test was performed to compare the results of the LSA with other enhanced algorithms

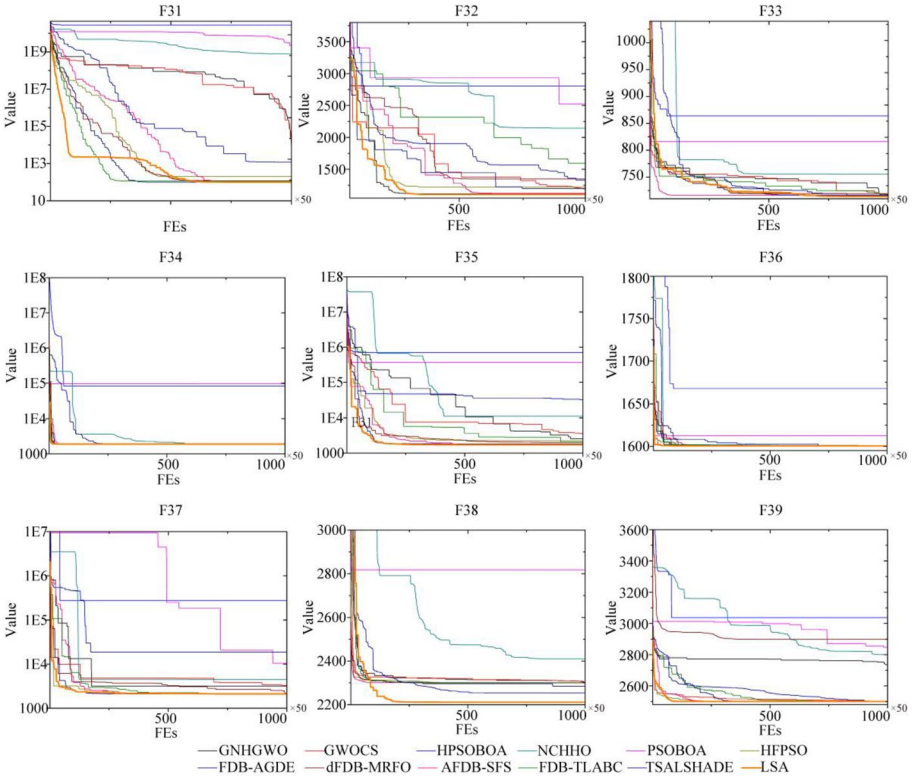

Fig. 14 LSA and other improved algorithms' convergence curves on 9 CEC 2020 benchmark functions

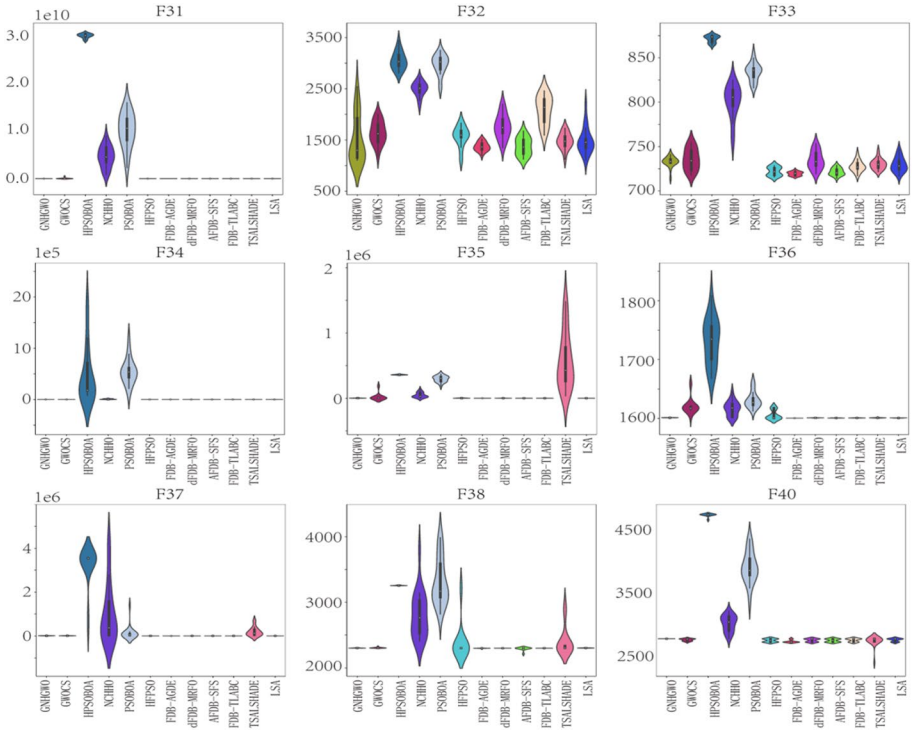

Fig. 15 The box plot of the results by recently proposed advanced improved algorithms for CEC 2020

$$
\begin{aligned}
&\text{Consider}[x_1, x_2, x_3, x_4, x_5, x_6, x_7]\\
&\min f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934)\\
&\text{-}1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)\\
&\qquad g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0,\\
&\qquad g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0,\\
&\qquad g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0,\\
&\qquad g_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0,\\
&\qquad g_5(x) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times10^6}}{110x_6^3} - 1 \le 0,\\
&\text{s.t.}\\
&\qquad g_6(x) = \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.6\times10^6}}{85x_7^3} - 1 \le 0,\\
&\qquad g_7(x) = \frac{x_2x_3}{40} - 1 \le 0,\\
&\qquad g_8(x) = 5x_2 - x_1 - 1 \le 0,\\
&\qquad g_9(x) = \frac{x_1}{12x_2} - 1 \le 0,\\
&\qquad g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0,\\
&\qquad g_{11}(x) = \frac{1.5x_7 + 1.9}{x_5} - 1 \le 0
\end{aligned}
\tag{39}
$$

where $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5 \le x_7 \le 5.5$.

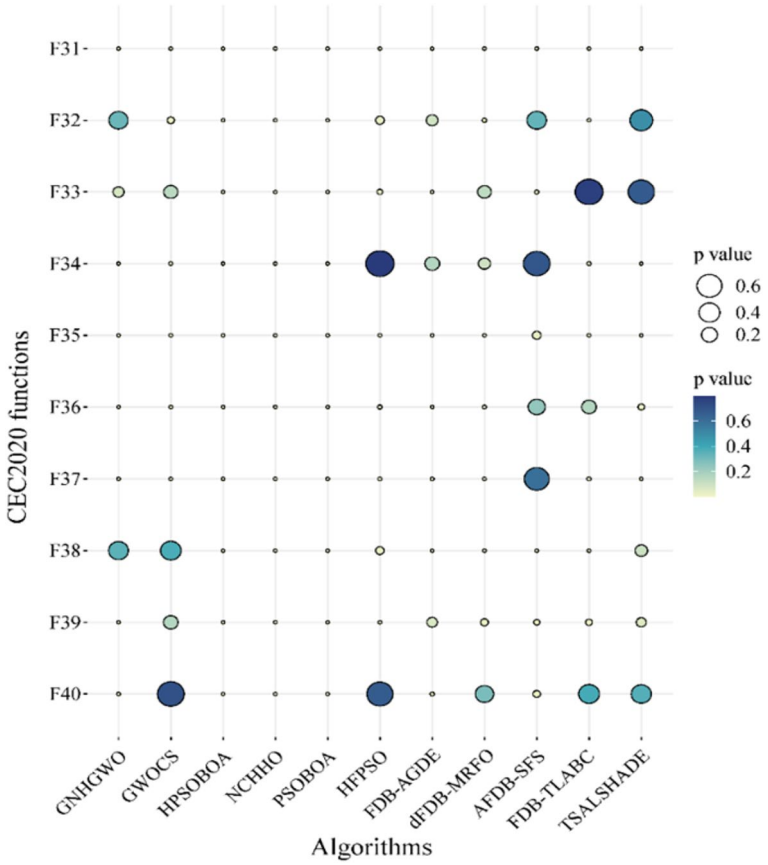

Fig. 16 The graphical representation of the Wilcoxon signed rank test results of the LSA algorithm compared to the other improved algorithms

When testing LSA to solve this problem, this paper selected some well-known meta-heuristic algorithms proposed in recent years including STOA (Dhiman and Kaur 2019), TQA (Chen et al. 2022), HS (Dhiman and Kumar 2017), ESMA (Örnek et al. 2022), GSA (Karami et al. 2021), EJAYA (Zhang et al. 2021b), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC (Duman et al. 2022),and TSALSHADE (Abdesslem layeb 2023) as comparison algorithms. Table 11 shows the statistical results of the LSA and comparison algorithms for solving this problem. It can be seen from Table 11 that the LSA obtained the result of 2986.064(consistent with the results of FDB-AGDE, dFDB-MRFO, AFDB-SFS, and FDB-TLABC), which is the best among all compared algorithms.

Table 8 The time consumption of the original algorithms for CEC 2014 and CEC 2020 functions

Fun	GWO	HHO	HPO	SSA	BWOA	MFO	SOA	TSA	TLBO	LSA
F1	0.151	0.345	4.005	0.403	0.133	0.216	0.424	0.141	0.231	0.255
F2	0.133	0.316	3.770	0.195	0.114	0.186	0.380	0.134	0.219	0.233
F3	0.139	0.313	4.027	0.422	0.116	0.175	0.371	0.134	0.216	0.236
F4	0.134	0.300	4.042	0.411	0.116	0.172	0.373	0.134	0.225	0.223
F5	0.145	0.376	4.023	0.416	0.117	0.188	0.441	0.134	0.209	0.231
F6	0.575	1.372	6.451	0.610	0.528	0.708	1.344	0.577	0.491	0.659
F7	0.145	0.331	4.279	0.142	0.124	0.188	0.343	0.137	0.213	0.238
F8	0.145	0.331	3.905	0.141	0.117	0.181	0.344	0.131	0.209	0.233
F9	0.156	0.471	4.223	0.144	0.125	0.181	0.387	0.141	0.206	0.219
F10	0.177	0.577	4.362	0.155	0.135	0.203	0.497	0.154	0.222	0.244
F11	0.158	0.518	3.998	0.161	0.144	0.213	0.420	0.148	0.219	0.253
F12	0.221	0.737	4.558	0.218	0.230	0.308	0.579	0.220	0.266	0.313
F13	0.135	0.410	4.099	0.137	0.134	0.169	0.271	0.130	0.203	0.222
F14	0.138	0.420	3.927	0.141	0.127	0.183	0.280	0.134	0.206	0.239
F15	0.141	0.480	4.363	0.138	0.126	0.178	0.280	0.129	0.209	0.241
F16	0.145	0.465	4.395	0.141	0.128	0.189	0.282	0.127	0.203	0.269
F17	0.146	0.467	4.202	0.145	0.130	0.198	0.292	0.137	0.206	0.289
F18	0.145	0.465	4.222	0.140	0.135	0.188	0.290	0.141	0.206	0.242
F19	0.234	0.745	4.897	0.226	0.210	0.288	0.441	0.220	0.269	0.328
F20	0.145	0.503	4.454	0.145	0.131	0.197	0.282	0.139	0.206	0.236
F21	0.150	0.554	4.129	0.145	0.137	0.189	0.292	0.141	0.213	0.241
F22	0.156	0.545	4.261	0.155	0.148	0.203	0.299	0.144	0.213	0.242
F23	0.245	0.752	4.447	0.218	0.206	0.272	0.407	0.208	0.263	0.313
F24	0.262	0.610	4.478	0.202	0.179	0.245	0.374	0.190	0.244	0.286
F25	0.278	0.726	4.370	0.209	0.195	0.266	0.423	0.190	0.250	0.300
F26	0.871	2.088	6.634	0.699	0.791	0.783	1.345	0.653	0.556	0.789
F27	0.848	1.970	6.339	0.681	0.808	0.858	1.265	0.648	0.553	0.753
F28	0.304	0.851	4.396	0.294	0.285	0.345	0.438	0.231	0.281	0.334
F29	0.352	0.845	4.790	0.388	0.351	0.377	0.588	0.277	0.297	0.531
F30	0.286	0.510	4.380	0.230	0.245	0.295	0.430	0.196	0.259	0.459
F31	0.145	0.316	4.098	0.163	0.132	0.202	0.163	0.138	0.406	0.245
F32	0.155	0.361	4.006	0.162	0.142	0.213	0.162	0.147	0.341	0.255
F33	0.161	0.364	4.178	0.168	0.133	0.206	0.157	0.146	0.303	0.248
F34	0.153	0.336	4.066	0.149	0.129	0.203	0.160	0.141	0.319	0.241
F35	0.148	0.332	4.552	0.147	0.146	0.204	0.166	0.215	0.338	0.238
F36	0.154	0.327	4.393	0.153	0.138	0.213	0.144	0.233	0.303	0.235
F37	0.154	0.337	4.370	0.159	0.153	0.209	0.162	0.248	0.331	0.240
F38	0.189	0.577	4.367	0.200	0.170	0.249	0.194	0.271	0.369	0.276
F39	0.204	0.674	4.484	0.207	0.183	0.266	0.210	0.295	0.381	0.288
F40	0.188	0.583	4.744	0.191	0.169	0.249	0.208	0.371	0.356	0.279
Avg	0.223	0.590	4.442	0.239	0.199	0.264	0.398	0.211	0.280	0.305
Rank	3	9	10	4	1	5	8	2	6	7

Table 9 LSA and the improved algorithms' time consumption for CEC 2020 functions

fun	GNH GWO	GWOCS	HPSO BOA	NCHHO	PSO BOA	HFPSO	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB-TLABC	TSALS HADE	LSA
F1	0.654	0.781	0.116	0.350	0.127	0.213	1.116	0.205	0.288	0.480	0.103	0.245
F2	0.763	0.759	0.120	0.397	0.128	0.116	1.128	0.201	0.296	0.456	0.095	0.255
F3	0.712	0.756	0.113	0.393	0.131	0.114	1.130	0.188	0.318	0.443	0.098	0.248
F4	0.687	0.791	0.120	0.375	0.130	0.130	1.122	0.188	0.292	0.439	0.109	0.241
F5	0.652	0.817	0.124	0.358	0.144	0.109	1.123	0.195	0.305	0.441	0.102	0.238
F6	0.644	0.870	0.127	0.348	0.129	0.106	1.116	0.190	0.285	0.433	0.103	0.235
F7	0.660	0.900	0.121	0.364	0.121	0.122	1.125	0.185	0.286	0.439	0.095	0.240
F8	0.754	1.002	0.153	0.470	0.168	0.147	1.166	0.222	0.321	0.479	0.127	0.276
F9	0.711	1.023	0.155	0.513	0.300	0.169	1.217	0.238	0.338	0.484	0.122	0.288
F10	0.692	0.801	0.141	0.508	0.278	0.155	1.154	0.225	0.322	0.466	0.125	0.279
Avg	0.693	0.850	0.129	0.408	0.166	0.138	1.140	0.203	0.305	0.456	0.108	0.254
Rank	10	11	2	8	4	3	12	5	7	9	1	6

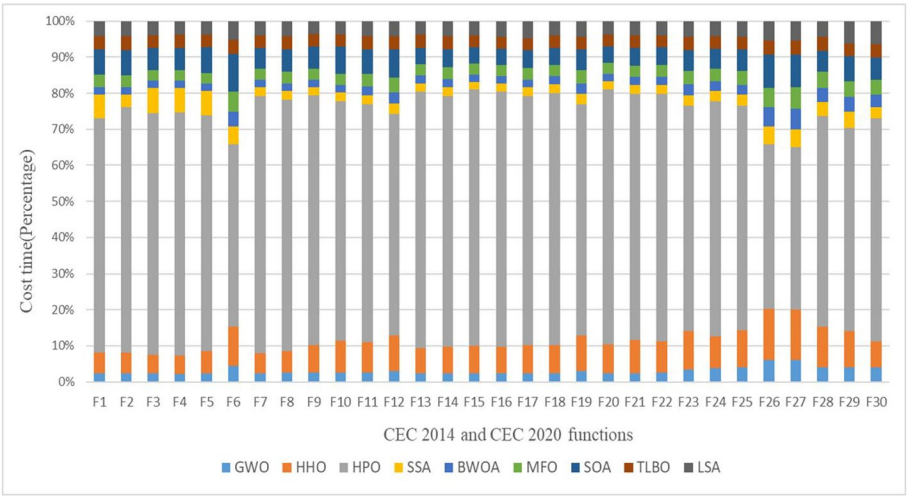

Fig. 17 The proportion of cost time for LSA compared to the original algorithms

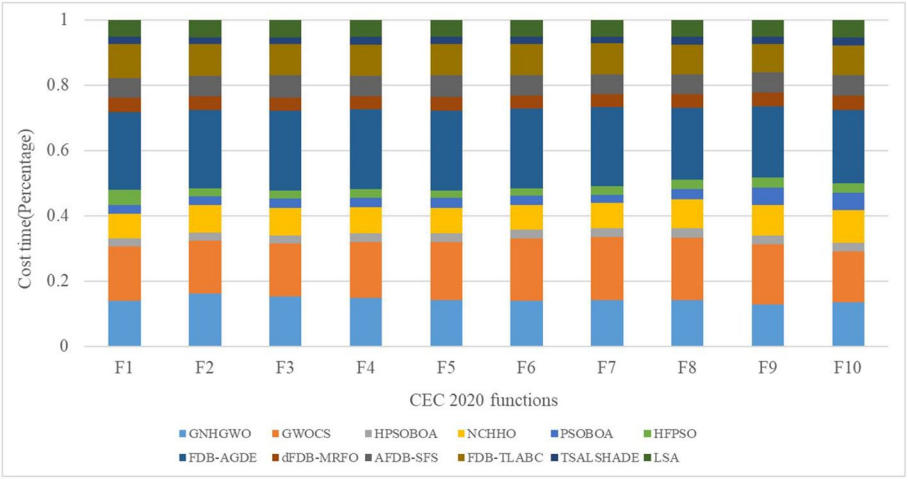

Fig. 18 The proportion of cost time for LSA compared to the improved algorithms

4.2.2 The Tension/Compression Spring Design (TCSD)

In the design of engineering problems, in addition to considering the optimal objective function of the mathematical model of the designed product, designers also need to consider the corresponding constraints. TCSD is a classic engineering design problem, and its goal is to minimize the weight of the designed product. In this problem, there are three variables and four inequality constraints (Faramarzi et al. 2020), and the mathematical model is as follows:

Table 10 The results when the parameter sub takes different values

Fun	$sub=2$	$sub=3$	$sub=4$	$sub=5$	$sub=6$	$sub=7$	$sub=8$
F1	609.9626577	**116.6784178**	608.321445	131.6911778	385.617495	117.8124101	583.0712154
F2	**200**	**200**	**200**	**200**	**200**	**200**	**200**
F3	**300**	**300**	**300**	**300**	**300**	**300**	**300**
F4	424.7797331	417.8236886	**413.9595865**	414.3462733	418.2572802	411.3011639	418.2572189
F5	520.209536	520.2482056	520.2300414	520.2390617	520.2043731	**520.1820433**	520.1953175
F6	603.0717908	602.7808713	**602.4843001**	602.6232063	603.1365271	603.7197096	603.3281788
F7	700.1902034	700.2244117	700.2354042	700.2119559	700.1715279	**700.1601685**	700.2143042
F8	813.5314291	814.1283979	814.7253784	812.6359699	814.0288995	813.4319327	**811.9394931**
F9	916.3173079	**914.128404**	914.8248662	916.0188136	918.207725	920.2971305	920.0981239
F10	1205.711159	**1200.947857**	1251.577491	1310.424259	1296.566687	1276.515618	1351.723663
F11	1628.827741	**1564.472366**	1705.076561	1688.600349	1665.299006	1662.486605	1649.825325
F12	1200.855214	**1200.552966**	1200.649862	1200.662828	1200.679571	1200.577318	1200.786121
F13	**1300.176382**	1300.191003	1300.245354	1300.240482	1300.219048	1300.237982	1300.255564
F14	1400.282115	**1400.276366**	1400.334527	1400.335936	1400.313986	1400.319259	1400.280061
F15	**1501.13756**	1501.210708	1501.191756	1501.157614	1501.250565	1501.165412	1501.275308
F16	1602.337659	1602.266232	1601.954572	1602.252587	1602.130453	**1601.952192**	1602.279844
F17	1855.556203	1839.002225	1868.615055	1772.066196	1830.37407	1811.764686	**1803.201491**
F18	1809.808767	1808.309217	1808.9655	**1806.136571**	1810.905358	1807.574215	1806.228484
F19	1902.215136	**1901.690645**	1902.346684	1902.145933	1901.885896	1901.813335	1901.797411
F20	2003.604335	**2002.053681**	2003.258793	2003.809543	2004.071637	2003.81932	2005.081206
F21	2124.908538	2117.506148	2114.182004	**2108.364146**	2129.222914	2132.510882	2122.747525
F22	2222.669351	2222.853952	2222.922669	2224.67913	2222.444711	2219.635019	**2215.399956**
F23	**2629.457475**	**2629.457475**	**2629.457475**	**2629.457475**	**2629.457475**	**2629.457475**	**2629.457475**
F24	2523.918425	2538.171865	**2523.488357**	2526.894086	2528.841622	2526.028004	2533.111258
F25	2662.520648	2662.921326	2651.682872	2662.5512	2663.658993	**2656.052593**	2666.729285
F26	**2700.128906**	2700.153013	2700.134672	2700.141367	2700.129121	2700.149678	2700.128579
F27	2817.909147	3017.029424	2853.61205	**2705.430799**	2903.276787	2814.246832	2820.860768
F28	3210.704945	3178.811713	**3177.857597**	3198.976216	3190.698723	3178.880343	3217.133256
F29	3124.993883	13,255.85092	3110.464066	3126.223415	3122.03686	**3107.76455**	3126.949094
F30	3525.691305	3596.801155	3584.915137	3582.474921	3571.198181	3678.346548	**3488.935807**
F31	102.9920416	**101.3899171**	101.4225509	107.3581254	104.7162445	104.392711	102.0000919
F32	1503.299207	**1485.292961**	1578.50125	1583.870475	1657.355378	1575.706236	1619.170036
F33	727.0853667	729.6887389	**726.4573311**	730.8014118	728.1224648	726.9000811	728.1392809
F34	1901.06296	**1900.966212**	1901.018139	1901.157811	1901.267822	1901.387834	1901.098329
F35	1845.338036	1795.501677	1796.956647	1791.903712	1812.279532	1808.905185	**1783.121345**
F36	1601.083725	**1600.659988**	1600.933328	1601.155258	1600.699197	1601.025983	1600.860962
F37	2133.616615	2144.9284	2156.038302	**2120.192204**	2116.620856	2136.509958	2137.57255
F38	**2289.899124**	2303.829855	2298.361709	2299.178388	2303.132056	2295.350297	2298.687978
F39	2708.724163	**2600.858706**	2596.971466	2612.8588	2587.882027	2659.207818	2659.122948
F40	2931.259278	2928.764136	2924.794399	2926.430333	2926.729055	**2920.221649**	2919.497356
Winner	7	16	8	7	3	9	8

$$\begin{aligned}
&\text{Consider}[x_1, x_2, x_3]\\
&\text{Minimize } f(x) = (x_3 + 2)x_2x_1^2\\
&\text{s.t.} \begin{cases} g_1(x) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \le 0,\\ g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} \le 0,\\ g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0,\\ g_4(x) = \frac{x_1+x_2}{1.5} - 1 \le 0. \end{cases}
\end{aligned} \tag{40}$$

where $0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15$.

Various intelligent algorithms have been used to solve this engineering design problem, such as EO (Faramarzi et al. 2020), RL-BA (Meng et al. 2019), DDAO (Ghafil and Jármai 2020), SDO (Zhao et al. 2019), AFA (Dhrubajyoti et al. 2021), mGWO (Shubham and Kusum 2020), PFA (Yapici and Cetinkaya 2019), GCHHO (Song et al. 2021), VAGWO (Farshad et al. 2022), ExPSO (Khelil et al. 2022), TEO (Kaveh and Dadras 2017), QS (Zhang et al. 2018), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC (Duman et al. 2022),and TSALSHADE (Abdesslem layeb 2023). Table 12 presents the solution results of the LSA and the above comparison algorithms to this problem, and the best optimization results are marked in bold. It can be seen from Table 12 that in terms of Best, Mean, Worst, and Std, the LSA obtains the best solution result. Table 13 indicates that the best result of the LSA for solving this engineering problem is 0.009872, and the values of x_1, x_2 and x_3 are 0.05, 0.374433, and 8.546567, respectively.

4.2.3 Pressure Vessel Design (PVD)

PVD is another classic constrained optimization problem with four optimization variables and four constraints. Its goal is to minimize the total cost of materials in forming and welding cylindrical vessels. The mathematical model is as follows:

$$\begin{aligned}
&\text{Consider}[x_1, x_2, x_3, x_4]\\
&\text{Minimize } f(x) = 0.6224x_1x_3x_4 + 1.7881x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3\\
&\text{s.t.} \begin{cases} g_1(x) = -x_1 + 0.0193x_3 \le 0,\\ g_2(x) = -x_2 + 0.00954x_3 \le 0,\\ g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0,\\ g_4(x) = x_4 - 240 \le 0 \end{cases}
\end{aligned} \tag{41}$$

where $0 \le x_1, x_2 \le 100, 10 \le x_3, x_4 \le 200$.

To explore the performance of the LSA in solving PVD, this paper selected some high-performance improved meta-heuristic algorithms recently proposed as comparison algorithms, including BIANCA (Montemurro et al. 2013), G-QPSO (Santos Coelho 2010), HAIS-GA (Coello and Cortés 2004), CB-ABC (Brajevic 2015), NHAIS-GA(Bernardino et al. 2008), DEC-PSO (Chun et al. 2013), T-Cell (Aragón et al. 2010), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC(Duman et al. 2022),and TSALSHADE (Abdesslem layeb 2023). It can be seen from Tables 14 and 15 that the LSA, FDB-AGDE, dFDB-MRFO, AFDB-SFS, and FDB- TLABC achieve the best optimization result, and the objective function value is 5885.333, which is obviously better than those of other comparative algorithms.

Table 11 The statistical results of the SRD problem

	LSA	STOA	TQA	HS	ESMA	GSA	EJAYA	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB-TLABC	TSALS HADE
Best	**2986.064**	3011.9	2994.6	3029.00	2993.76	3301.6	2994.47	2986.064	2986.064	2986.064	2986.064	2986.064
x_1	3.4764	3.5029	3.5	3.5201	3.50000	3.53420	3.5	3.476	3.476	3.476	3.4764	3.4764
x_2	0.7	0.70	0.7	0.700	0.70000	0.701	0.7	0.7	0.7	0.7	0.7	0.7
x_3	17	17.0	17.0	17.000	17.00000	17.817	17	17	17	17	17	17
x_4	7.3	7.5156	7.31	8.370	7.30000	7.39780	7.3	7.3	7.3	7.3	7.3	7.3
x_5	7.698	8.0238	7.72	7.800	7.8	8.2455	7.71532	7.6977	7.6977	7.6977	7.6977	7.6977
x_6	3.3486	3.3735	3.35	3.367	3.3434	3.4921	3.35026	3.3486	3.3486	3.3486	3.3486	3.3486
x_7	5.2766	5.2892	5.286655	5.2887	5.2854	5.4292	5.28665	5.2766	5.2766	5.2766	5.2766	5.2766
Mean	2986.064	3028.1	3002.9	-	-	-	2994.47	2986.06	2986.064	2986.064	2986.064	2986.065
Worst	2986.064	-	-	-	-	-	2994.47	2986.06	2986.064	2986.064	2986.064	2986.083
Std	3.4e-13	9.5	5.6	-	-	-	7.2e-6	0	4.6e-13	0	3.216E-13	0.004

Bold entries indicates optimal result

Table 12 The statistical results of each algorithm

Algorithms	Best	Mean	Worst	Std
LSA	**0.009872**	**0.009872**	**0.009872**	**8.18E-19**
EO	0.012666	0.013017	0.013997	3.91E − 04
RL-BA	0.0126764	0.012745	0.0129281	7.19E − 04
DDAO	0.0129065	0.0151829	0.0173199	1.26E − 03
SDO	0.0126663	0.0126724	0.0126828	6.1899E − 06
AFA	0.0126653049	0.0126770446	0.0000128058	0.012711688
mGWO	0.012668	-	-	-
PFA	0.01266528	-	-	-
GCHHO	0.012665264	-	-	-
VAGWO	0.0127	-	-	-
ExPSO	0.012665	0.012857	0.015234	0.000523
TEO	0.012665	0.012685	0.012715	4.4079E − 06
QS	0.012665	0.012666	0.012669	-
FDB-AGDE	0.009872455	0.009872455	0.009872455	1.50231E-18
dFDB-MRFO	0.009872455	0.009872455	0.009872455	1.22663E-18
AFDB-SFS	0.009872	0.009872	0.009872	1.5E-18
FDB-TLABC	0.009872455	0.009872455	0.009872455	1.73472E-18
TSALS-HADE	0.009913	0.011094	0.015782	0.001369

4.2.4 Three-bar Truss Design (TTD)

The TTD problem is another classic minimization problem in engineering design, and its structure can be found in the reference (Ghasemi et al. 2022). It has 3 variants and 4 inequality constraints, and its goal is to minimize the weight of the three trusses.

$$\begin{aligned} &\text{Consider}[x_1, x_2] \\ &\min f(x) = (2\sqrt{2}x_1 + x_2) \times l \\ &\text{s.t.} \begin{cases} g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \le 0, \\ g_2(x) = \frac{x_2}{\sqrt{2x_1^2 + 2x_1x_2}} P - \sigma \le 0, \\ g_3(x) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \le 0, \\ g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \end{cases} \end{aligned} \tag{42}$$

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

Table 16 shows the statistical results of each algorithm to solve the TTD problem, where the best results are highlighted in bold. According to the statistical results, LSA, FDB-AGDE, dFDB-MRFO, AFDB-SFS and FDB-TLABC obtained the best optimal value (263.8523) among all the 15 comparison algorithms.

Table 13 The optimal solution results of the LSA algorithm

	x_1	x_2	x_3	Best
LSA	0.05	0.374433	8.546567	0.009872

4.2.5 Cantilever Beam Design (CBD)

The CBD is a civil engineering structural design problem consisting of 5 hollow elements, each of which has an equal thickness. Its objective function is to minimize the weight of the cantilever beam. The mathematical model of this problem is represented below:

$$\begin{aligned} &\text{Consider } [x_1, x_2, x_3, x_4, x_5] \\ &\min f(x) = 0.624 \sum_{i}^{5} x_i \\ &\text{s.t. } g(x) = \sum_{i=1}^{5} \frac{b_i}{x_i^3} - 1 \le 0 \end{aligned} \tag{43}$$

where $b = (b_1, b_2, b_3, b_4, b_5) = (67, 37, 19, 7, 1) 0.01 \le x_i \le 100, i = 1, ..., 5.$

To test the ability of the proposed LSA to solve this problem, this paper selected 5 algorithms, STOA (Dhiman and Kaur 2019), TQA (Chen et al. 2022), GCA_I (Kumar et al. 2020), GCA_II (Kumar et al. 2020), SMA (Li et al. 2020c), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC (Duman et al. 2022),and TSALSHADE (Abdesslem layeb 2023) as comparison algorithms. Table 17 presents the statistical results of each algorithm's performance in addressing this problem, with the best results highlighted in bold. From the statistical outcomes depicted in Table 17, it is observed that LSA, FDB-AGDE, dFDB-MRFO, AFDB-SFS, and FDB-TLABC exhibit the most favorable optimization outcomes. In other words, these algorithms achieved the best optimal values among all the evaluated approaches.

4.2.6 Car Side Impact Design (CSID)

The goal of CSID is to minimize weight, which is related to 11 variables and 10 inequality constraints. Its detailed description can be found in the reference (Huang et al. 2015). Meanwhile, EJAYA (Zhang et al. 2021b), TLCS (Huang et al. 2015), AOSMA (Naik et al. 2021), WOAGWO (Mohammed and Rashid 2020), PGJAYA (Yu et al. 2019), ERao-1 (Jian and Zhu 2021), CLJAYA (Zhang and Jin 2022), FDB-AGDE (Guvenc et al. 2021), dFDB-MRFO (Kahraman et al. 2022), AFDB-SFS (Duman et al. 2023), FDB-TLABC (Duman et al. 2022),and TSALSHADE (Abdesslem layeb 2023) were selected to test their performance in solving this problem. The mathematical model of this problem is as follows.

Table 14 The statistical results of solving PVD by each algorithm

	LSA	BIANCA	G-QPSO	HAIS-GA	CB-ABC	NHAIS-GA	DEC-PSO	T-Cell	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB- TLABC	TSALS HADE
Best	5885.3	6059.9	6059.7	6832.6	6059.7	6061.1	6059.7	6390.6	5885.3	5885.3	5885.3	5885.3	5922.8
Mean	5885.3	6182.0	6059.7	7187.3	6126.6	6743.1	6060.3	6737.1	5885.3	5885.3	5885.3	5885.3	6369.0
Worst	5885.3	6447.3	7544.5	8012.6	-	7368.1	6090.5	7694.1	5885.3	5885.3	5885.3	5885.3	7319.0
Std	0	122.3	448.5	276.0	114.0	458.0	4.4	357.0	0	2.98E-04	0	0	3.52E+02

Table 15 The LSA algorithm to solve the optimal solution results of PVD

	x_1	x_2	x_3	x_4	Best
LSA	0.778168641	0.384649163	40.31961872	200	5885.333

(44)

where $0.5 \le x_1, x_2, x_3, x_4, x_5, x_6, x_7 \le 1.5, 0.192 \le x_8, x_9 \le 0.345, -30 \le x_{10}, x_{10} \le 30$.

$$
\begin{aligned}
&\text{Consider}[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}].\\
&\text{Objective}: \min f(x) = 1.98 + 4.90x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 2.73x_7\\
&\quad g_1(x) = 1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10} \le 1,\\
&\quad g_2(x) = 0.261 - 0.0159x_1x_2 - 0.188x_1x_8 - 0.019x_2x_7 + 0.0144x_3x_5 + 0.0008757x_5x_{10}\\
&\quad +0.08045x_6x_9 + 0.00139x_8x_{11} + 0.00001575x_{10}x_{11} \le 0.32,\\
&\quad g_3(x) = 0.214 + 0.00817x_5 - 0.131x_1x_8 - 0.0704x_1x_9 + 0.03099x_2x_6\\
&\quad -0.018x_2x_7 + 0.0208x_3x_8 + 0.121x_3x_9 - 0.00364x_5x_6\\
&\quad +0.0007715x_5x_{10} - 0.0005354x_6x_{10} + 0.00121x_8x_{11} \le 0.32,\\
&\text{s.t. } g_4(x) = 0.74 - 0.61x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2 \le 0.32,\\
&\quad g_5(x) = 28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.7x_7x_8 + 0.32x_9x_{10} \le 32,\\
&\quad g_6(x) = 33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22x_8x_9 \le 32,\\
&\quad g_7(x) = 46.36 - 9.9x_2 - 12.9x_1x_8 - 5.057x_1x_2 + 0.1107x_3x_{10} \le 32,\\
&\quad g_8(x) = 4.72 - 0.5x_4 - 0.19x_2x_3 - 0.0122x_4x_{10} + 0.009325x_6x_{10} + 0.000191x_{11}^2 \le 4,\\
&\quad g_9(x) = 10.58 - 0.674x_1x_2 - 1.95x_2x_8 + 0.02054x_3x_{10} - 0.0198x_4x_{10} + 0.028x_6x_{10} \le 9.9,\\
&\quad g_{10}(x) = 16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} - 0.000786x_{11}^2 \le 15.7
\end{aligned}
$$

The statistical results of LSA on this problem are presented in Table 18. It can be seen from this table that the optimal solution of LSA is 22.842, which is consistent with the same results by FDB-AGDE, dFDB-MRFO, AFDB-SFS, and FDB-TLABC algorithms. The values corresponding to each variable are 0.5, 1.116, 0.5, 1.302, 0.5, 1.5, 0.5, 0.964338, 1.000, -19.577, and 3.73E-07. These results indicate that the LSA algorithm has strong competitiveness.

4.3 Application to real-world optimization of feature selection

In this subsection, the proposed LSA is applied to the feature selection problem to verify the performance of the algorithm in solving optimization problems in this domain. Feature selection refers to selecting a feature subset with good distinguishing characteristics from a feature set according to a target. Therefore, feature selection requires a specific feature evaluation function. The K-nearest neighbor (KNN) algorithm is a classification technique based on supervised machine learning. Because of its characteristics of easy implementation and fast operation, it is often selected as a wrapper-based feature selection method. The fitness function used in feature selection problems has two main goals: a small number of features and the minimum classification error. The most ideal solution to the feature selection problem is to achieve the minimum error by selecting the fewest features. In this paper, the following objective function is adopted to calculate the objective function:

Table 16 The statistical results of the TTD

	Best	x_1	x_2	Mean	Worst	Std
LSA	**263.8523**	0.788415	0.408114	263.8523	263.8523	2.68E-14
FA1 → 3 (Ghasemi et al. 2022)	263.8958	0.7886699	0.4082631	263.895850	263.8959	5.23E − 6
CS (Gandomi et al. 2013)	263.9716	-	-	264.0669	-	9.0E − 5
WSO (Ayyarao et al. 2022)	263.8958	-	-	-	-	
CDESSA (Zhang et al. 2022)	263.895843	-	-	-	-	-
MDM-GWO (Shitu and Jagdish 2022)	263.89581	-	-	-	-	-
STOA (Dhiman and Kaur 2019)	263.89704	0.788115	0.40984	267.70719	-	7.6972
TQA (Dhiman and Kaur 2019)	263.89584	0.788688	0.40821	263.89751	-	2.6324E − 3
COLSHADE (Gurrola-Ramos et al. 2020)	263.8958	-	-	263.8958	263.8958	0
IUDE (Trivedi et al. 2018)	2.64E + 2	-	-	2.64E + 2	2.64E + 02	0
eMAgES (Hellwig and Beyer 2018)	2.64E + 2	-	-	2.65E + 2	2.74E + 02	2.88
SASS (Kumar and Zelinka 2020)	263.896	-	-	263.896	263.896	5.8E-14
sCMAGES (Kumar et al. 2020)	263.896	-	-	263.896	263.896	2.17E-12
FDB-AGDE (Guvenc et al. 2021)	**263.8523**	0.788415	0.408114	263.8523	263.8523	0
dFDB-MRFO (Kahraman et al. 2022)	**263.8523**	0.788415	0.408114	263.8523464	263.8523464	0
AFDB-SFS (Duman et al. 2023)	**263.8523**	0.788415	0.408114	263.8523	263.8523	2.84E-14
FDB-TLABC (Duman et al. 2022)	**263.8523**	0.788415	0.408114	263.8523	263.8523	4.92E-14
TSALSHADE (Abdesslem layeb 2023)	263.8545	0.786731	0.41293	264.263	267.3196	0.764696

Table 17 Statistical results of CBD problem

	LSA	STOA	TQA	GCA_I	GCA_II	SMA	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB- TLABC	TSALS HADE
Best	**1.33996**	1.340096	1.339957	1.3400	1.3400	**1.33996**	**1.33996**	**1.33996**	1.33996	1.33996	1.34535
x_1	6.016011	6.012432	6.013593	6.0100	6.0100	6.0177	6.01601	**6.01601**	6.016011	6.016011	6.30664
$_2$	5.30917	5.330040	5.307230	5.3000	5.3000	5.3108	6.01601	**6.01601**	6.01601	6.01601	6.30664
x_3	4.494326	4.458157	4.498556	4.4900	4.4900	4.4937	5.30917	**5.30917**	5.30917	5.30917	5.06950
x_4	3.501472	3.541110	3.502111	3.4900	3.4900	3.5011	4.49433	**4.49433**	4.49433	4.49433	4.30018
x_5	2.152664	2.134160	2.152181	2.1500	2.1500	2.1501	3.50147	**3.50147**	3.50147	3.50147	3.65317
Mean	1.339956	1.340710	1.339975	-	-	-	2.15266	**2.15266**	2.15266	2.15266	2.23056
Worst	1.339956	-	-	-	-	-	1.33996	**1.33996**	1.33996	1.33996	1.37581
Std	0	7.5025e-4	1.6258e − 5	-	-	-	0	**2.96e-13**	2.22E-16	0	0.030

$$objective\ \text{fitness} = \text{Argmin}(\alpha \cdot (1 - acc) + \beta \cdot \frac{N_i}{N}) \tag{45}$$

where,*acc* represents the accuracy of KNN classification, N represents the total number of features, and N_i represents the number of features selected by the i-th candidate solution. $\alpha \in [0, 1]$ denotes the weight, $\beta = 1 - \alpha$. According to the literature (Xu et al. 2023), the values of α and β are set to 0.99 and 0.01, respectively.

In the above tests, it is shown that the LSA is suitable for solving continuous optimization problems. To this end, it is necessary to convert the value of the search individual in the algorithm into a discrete value of 0 or 1. Denoting the j-th dimension value of the ith individual is $x_{i,j}$, the conversion is shown below:

$$s = \frac{1}{1 + \exp(-10 \cdot (x_{i,j} - 0.5))} \tag{46}$$

$$x_{i,j} = \begin{cases} 1, s \geq rand \\ 0, else \end{cases} \tag{47}$$

where, $rand \in (0, 1)$ is a random number.

To verify the effect of the LSA in feature selection, this paper experimented with the LSA on 15 UCI public datasets. These 15 test cases have different sample numbers (from 72 to 846), and different feature numbers (from 8 to 30). Table 19 presents the basic information of these 15 datasets. The original dataset can be downloaded from the UCI machine learning website http://archive.ics.uci.edu/ml/index.php.

Five algorithms were selected for comparison, including BGWO (Emary et al. 2016), CCSA (Zamani et al. 2019), SCA (Mirjalili 2016), SSA (Xue and Shen 2020), and WOA (Hayyolalam and Kazem 2020). The population size was set to 30, each test case was run 20 times independently, and the average value was taken as the statistical value. All other parameters were kept the same as their corresponding references. Tables 20, 21, and 22 present the feature selection results of each algorithm on the UCI dataset.

Table 20 shows the average fitness value of each algorithm, and the optimal result is shown in bold. It can be seen from Table 20 that except for the four data sets of Breast, Fertility, WDBC, and Vehicle, LSA obtained the best average fitness value among all the other 11 feature selection algorithms. On these 4 datasets, the performance of the LSA is second only to BGWO. Although the feature selection capabilities of CCSA, SCA, SSA, and WOA are very powerful, their results are still significantly worse than those of the LSA. The rankings of these six algorithms in terms of fitness value are LSA, BGWO, WOA, SCA, CCSA, and SSA

Table 21 shows the classification error rate of the LSA and other comparison algorithms on each dataset. It can be seen from this table that the classification error rate of the LSA on Ceramic and Audit-2 is 0 datasets. In the 15 test datasets, LSA ranked first on 12 and ranked second on the other 3 datasets. This proves the absolute superiority of the LSA algorithm. The features of all algorithms on the BreastTissue and Vehicle datasets are difficult to distinguish because the error rate of all algorithms exceeds 25%.

Table 22 presents the average number of features selected by each algorithm. It was found that no algorithm obtained an absolute advantage in the number of features. The main reason is that the weight of the selected feature number in the fitness function is relatively small. As a result, although some algorithms select a few features, their classification accuracy is low. The above analysis indicates that the LSA has strong competitiveness in feature selection.

Table 18 Statistical results of CSID problem

	LSA	EJAYA	TLCS	AOSMA	WOA GWO	PGJ AYA	ERao-1	CLJ AYA	FDB-AGDE	dFDB-MRFO	AFDB-SFS	FDB-TLABC	TSALS HADE
Best	**22.842**	22.843	22.843	22.843	22.85	22.844	22.843	22.843	**22.842**	**22.842**	**22.842**	**22.842**	23.084
x_1	0.5	0.5	0.5	0.5	0.5	0.5	0.500009	0.5	0.5	0.500001	0.5	0.5	0.5
x_2	1.116	1.116	1.116	1.114	1.117	1.120	1.116	1.116	1.116	1.116	1.116	1.116	1.140
x_3	0.5	0.5	0.5	0.5	0.5	0.50005	0.5	0.5	0.5	0.500001	0.5	0.5	0.5
x_4	1.302	1.302	1.302	1.306	1.303	1.297	1.303	1.302	1.302	1.302	1.302	1.302	1.323
x_5	0.5	0.500	0.5	0.5	0.501	0.5	0.5	0.5	0.5	0.500004	0.5	0.5	0.5
x_6	1.5	1.500	1.5	1.5	1.500	1.49998	1.5	1.49999	1.5	1.5	1.5	1.5	1.5
x_7	0.5	0.500	0.5	0.5	0.500	0.5 0.3450	0.50002	0.5	0.5	0.500005	0.5	0.5	0.5
x_8	0.964	0.345	0.345	0.345	0.345	0.2338	0.344998	0.34999	0.876	0.537	0.577	0.622	0.814
x_9	1.000	0.327	0.192	0.193	0.333	-18.997	0.216895	0.1925	0.945	0.669	0.281	0.155	0.381
x_{10}	−19.577	−19.571	−19.572	− 19.914	− 19.491	0.083	−19.582	−19.566	−19.577	−19.577	−19.590	−19.577	−15.409
x_{11}	3.73E-7	0.008	0.0157	0.299	0.992	22.844	−0.00679	−0.0079	0.000	0.052	0.000	0.000	−0.114
Mean	22.842	22.944	-	-	-	-	-	-	22.842	22.993	22.842	22.842	23.770
Worst	22.842	23.262	-	-	-	-	-	-	22.842	23.223	22.842	22.842	24.748
Std	3.55E-15	1.71E-01	-	-	-	-	-	-	1.78E-15	2.07E-01	1.29E-05	2.51E-15	4.45E-01

Bold entries indicates optimal result

Table 19 The details of 15 UCI datasets

Items	Name	No. of the features	No. of the samples
FS01	Algerian	13	244
FS02	Audit-1	26	776
FS03	BreastTissue	9	106
FS04	Ceramic	17	88
FS05	Breast	9	116
FS06	Fertility	9	100
FS07	HCV	12	616
FS08	Heartstatlog	13	270
FS09	Lymphography	18	148
FS10	Cervical	19	72
FS11	Audit-2	17	776
FS12	WBC	9	683
FS13	WDBC	30	569
FS14	Wine	13	178
FS15	Vehicle	17	846

Figure 19 shows the convergence curves of each algorithm when solving the feature selection problem. It can be seen from Fig. 19 that the LSA has high convergence accuracy and speed when solving such problems.

5 Conclusion

This paper introduces a novel learning search algorithm (LSA) designed to efficiently and accurately address optimization problems. In the global expansion stage, the algorithm leverages historical knowledge and up-to-date community information to guide the search direction, thereby enhancing its global search capability. In the local development phase, the algorithm employs the teaching behavior and direction of the role model within the population to enhance the learning capability of the entire population. By dynamically adapting the control factor, the algorithm strikes a balance between exploration and exploitation, thereby avoiding local optima and improving convergence speed. Experimental results vividly demonstrate the LSA's search process for the optimal solution. Initially, 40 CEC 2014 and CEC 2020 benchmark functions are subjected to comparative testing using well-known original algorithms and recently proposed high-performing improved algorithms. Statistical analysis and the Wilcoxon signed rank test substantiate the LSA's commendable performance and robust competitiveness vis-à-vis other meta-heuristic algorithms. Furthermore, six subsequent engineering design experiments underscore the LSA's efficacy in solving real-world engineering applications with constraints. Finally, the LSA is used to solve the feature selection problem, and the experimental results on 15 UCI datasets further verify that the proposed algorithm performs significantly better than other methods in terms of classification accuracy and fitness value.

In this study, despite utilizing the LSA algorithm for solving continuous single-objective optimization problems, real-world constrained optimization problems, and real-world optimization of feature selection, limited research has been conducted on solving multi-objective problems. Many practical decision-making problems involve multiple criteria. For example, resource scheduling problems in cloud computing encompass objectives such as minimizing completion time and cost, and maximizing profit. Therefore, in the near future, we intend to further develop and enhance the LSA algorithm to tackle multi-objective optimization problems. Additionally, we aim to incorporate discretization methods into the LSA algorithm to enable it to handle discrete optimization problems, such as resource scheduling problems.

Table 20 The average fitness value of the algorithms on 15 UCI datasets

Items		BGWO	CCSA	SCA	SSA	WOA	LSA
Algerian	mean	1.69%	2.86%	3.20%	3.99%	1.90%	**1.68%**
	std	3.41E-03	9.84E-03	7.30E-03	5.93E-03	3.78E-03	**2.40E-03**
Audit-1	mean	0.19%	1.20%	1.34%	1.59%	0.11%	**0.10%**
	std	4.52E-04	2.03E-03	2.39E-03	1.77E-03	7.71E-04	**6.67E-04**
BreastTissue	mean	30.57%	30.25%	31.51%	34.50%	30.47%	**30.16%**
	std	1.09E-02	9.30E-03	1.14E-02	7.58E-03	7.70E-03	**5.83E-03**
Ceramic	mean	0.19%	0.57%	0.54%	0.95%	0.07%	**0.06%**
	std	3.52E-03	3.48E-03	1.64E-03	7.68E-04	1.98E-04	**1.14E-19**
Breast	mean	**13.55%**	16.62%	14.65%	21.10%	14.56%	13.61%
	std	**7.76E-03**	3.38E-02	7.24E-03	9.41E-03	1.52E-02	7.82E-03
Fertility	mean	**8.70%**	9.60%	9.76%	11.06%	9.36%	8.79%
	std	**7.21E-03**	1.10E-02	8.17E-03	6.72E-03	8.91E-03	5.91E-03
HCV	mean	5.06%	5.75%	5.64%	6.28%	5.38%	**5.02%**
	std	8.43E-04	3.93E-03	3.22E-03	1.30E-03	3.07E-03	**1.35E-03**
Heartstatlog	mean	13.99%	19.46%	16.16%	22.09%	14.78%	**13.88%**
	std	5.17E-03	2.02E-02	1.63E-02	3.53E-02	1.29E-02	**5.69E-03**
Lymphography	mean	11.50%	15.00%	13.79%	18.15%	13.67%	**10.93%**
	std	1.36E-02	9.87E-03	1.42E-02	9.52E-03	2.15E-02	**8.56E-03**
Cervical	mean	4.86%	5.68%	5.50%	6.86%	5.02%	**4.35%**
	std	**5.99E-03**	8.46E-03	7.55E-03	7.35E-03	9.05E-03	7.44E-03
Audit-2	mean	**0.06%**	0.80%	0.84%	1.10%	0.06%	**0.06%**
	std	**1.14E-19**	3.52E-03	4.52E-03	5.41E-03	1.14E-19	**1.14E-19**
WBC	mean	2.70%	2.91%	2.92%	2.96%	2.78%	**2.64%**
	std	1.22E-03	9.00E-04	8.51E-04	6.85E-04	1.33E-03	**1.05E-03**
WDBC	mean	**4.50%**	5.38%	5.48%	5.68%	4.66%	4.54%
	std	**1.13E-03**	4.64E-03	4.51E-03	2.77E-03	1.96E-03	8.12E-04
Wine	mean	3.82%	5.78%	4.22%	6.75%	4.50%	**3.61%**
	std	4.57E-03	8.46E-03	4.30E-04	8.75E-03	5.38E-03	**4.42E-03**
Vehicle	mean	**25.61%**	28.57%	27.60%	30.22%	26.79%	25.93%
	std	**2.96E-03**	1.04E-02	7.23E-03	9.43E-03	5.52E-03	4.92E-03
Rank		**2**	5	4	6	3	1

Table 21 Average classification accuracy of the algorithms on 15 UCI datasets

Items		BGWO	CCSA	SCA	SSA	WOA	LSA
Algerian	mean	1.49%	2.53%	2.67%	3.68%	1.74%	**1.46%**
	std	4.71E-03	9.58E-03	5.97E-03	5.93E-03	4.94E-03	2.85E-03
Audit-1	mean	0.12%	0.81%	0.85%	1.14%	0.07%	**0.04%**
	std	4.07E-04	1.48E-03	1.41E-03	1.91E-03	6.82E-04	6.23E-04
BreastTissue	mean	30.52%	30.06%	31.13%	34.14%	30.34%	**30.05%**
	std	1.09E-02	9.70E-03	1.21E-02	7.00E-03	8.19E-03	5.95E-03
Ceramic	mean	0.12%	0.03%	0.03%	0.06%	**0**	**0**
	std	3.59E-03	2.07E-04	9.75E-05	4.57E-05	1.17E-05	0
Breast	mean	**13.09%**	16.20%	14.06%	20.53%	14.12%	13.17%
	std	7.58E-03	3.53E-02	6.86E-03	9.15E-03	1.57E-02	7.92E-03
Fertility	mean	**8.35%**	9.17%	9.27%	10.75%	9.05%	8.45%
	std	8.13E-03	1.19E-02	7.92E-03	8.06E-03	1.04E-02	6.88E-03
HCV	mean	4.60%	5.20%	4.85%	5.48%	4.79%	**4.45%**
	std	1.05E-03	4.42E-03	2.64E-03	2.14E-03	3.02E-03	1.46E-03
Heartstatlog	mean	13.67%	19.19%	15.65%	21.89%	14.48%	**13.56%**
	std	5.44E-03	2.07E-02	1.62E-02	3.64E-02	1.34E-02	6.08E-03
Lymphography	mean	11.17%	14.63%	13.22%	17.48%	13.34%	**10.50%**
	std	1.40E-02	1.02E-02	1.40E-02	8.37E-03	2.21E-02	8.58E-03
Cervical	mean	4.60%	5.30%	4.90%	6.42%	4.74%	**4.05%**
	std	6.67E-03	8.77E-03	7.33E-03	7.18E-03	9.68E-03	7.86E-03
Audit-2	mean	0	0.40%	0.39%	0.73%	0	**0**
	std	0	3.14E-03	3.34E-03	5.08E-03	0	0
WBC	mean	2.19%	2.20%	1.98%	2.02%	2.21%	**2.12%**
	std	1.87E-03	1.35E-03	8.96E-04	1.53E-03	1.75E-03	1.32E-03
WDBC	mean	4.42%	5.01%	4.82%	5.27%	4.47%	**4.38%**
	std	1.73E-03	4.25E-03	4.15E-03	2.92E-03	2.61E-03	1.71E-03
Wine	mean	3.46%	5.38%	3.44%	6.33%	4.08%	**3.18%**
	std	4.89E-03	8.48E-03	3.34E-05	8.70E-03	6.30E-03	3.97E-03
Vehicle	mean	**25.36%**	28.33%	27.07%	30.05%	26.47%	25.69%
	std	3.14E-03	1.05E-02	7.04E-03	9.33E-03	6.49E-03	5.25E-03
Rank		2	5	4	6	3	1

Bold entries indicates optimal result

In our future work, we can employ adaptive mechanisms to adjust the parameters and operations of algorithms, enabling them to automatically adapt and improve their performance to different problems. Additionally, we can combine or cooperate metaheuristic algorithms with other optimization algorithms, machine learning methods, etc., to enhance the performance and adaptability of the algorithms. Moreover, LSA can be expanded to solve different optimization problems in various domains, such as neural networks, gene feature selection, shop floor scheduling, big data applications, and more.

Table 22 The average feature number of the algorithms on 15 UCI datasets

Items		BGWO	CCSA	SCA	SSA	WOA	LSA
Algerian	mean	2.7	4.6	7.2	4.5	2.4	3.3
	std	1.64E+00	1.58E+00	2.30E+00	1.58E+00	1.71E+00	1.16E+00
Audit-1	mean	1.8	10.1	13.2	12.1	1.2	1.6
	std	6.32E-01	1.91E+00	3.71E+00	2.56E+00	4.22E-01	5.16E-01
BreastTissue	mean	3.2	4.4	6.3	6.3	3.9	3.7
	std	9.19E-01	1.07E+00	8.23E-01	1.89E+00	1.29E+00	6.75E-01
Ceramic	mean	1.3	9.1	8.7	15.2	1.1	1
	std	6.75E-01	5.57E+00	2.63E+00	1.23E+00	3.16E-01	0.00E+00
Breast	mean	5.3	5.3	6.6	7	5.2	5.1
	std	4.83E-01	1.49E+00	5.16E-01	1.70E+00	9.19E-01	3.16E-01
Fertility	mean	3.9	4.7	5.2	3.7	3.6	3.8
	std	8.76E-01	1.06E+00	9.19E-01	1.42E+00	1.51E+00	9.19E-01
HCV	mean	6	7.3	10.1	10.3	7.7	7.3
	std	6.67E-01	9.49E-01	9.94E-01	1.57E+00	1.06E+00	8.23E-01
Heartstatlog	mean	5.9	6	8.7	5.5	5.7	6
	std	1.20E+00	1.25E+00	6.75E-01	1.58E+00	1.25E+00	1.33E+00
Lymphography	mean	7.8	9.4	12.6	15.2	8.4	9.5
	std	1.48E+00	1.35E+00	1.58E+00	3.55E+00	1.58E+00	1.43E+00
Cervical	mean	5.8	8.2	12.4	9.6	6.2	6.6
	std	1.75E+00	1.14E+00	1.90E+00	1.96E+00	1.87E+00	8.43E-01
Audit-2	mean	1	6.9	7.7	6.4	1	1
	std	0.00E+00	1.37E+00	2.26E+00	1.65E+00	0.00E+00	0.00E+00
WBC	mean	4.8	6.6	8.6	8.6	5.3	4.9
	std	1.03E+00	1.71E+00	9.66E-01	1.26E+00	1.95E+00	7.38E-01
WDBC	mean	3.8	12.7	21.3	13.8	7.1	6
	std	2.10E+00	1.77E+00	4.35E+00	1.75E+00	5.15E+00	3.59E+00
Wine	mean	5.1	6	10.6	6.3	5.9	5.9
	std	7.38E-01	1.25E+00	5.16E-01	2.06E+00	1.79E+00	9.94E-01
Vehicle	mean	9	9.4	14.4	8.4	10.6	9
	std	8.16E-01	1.84E+00	1.17E+00	2.76E+00	2.37E+00	1.41E+00

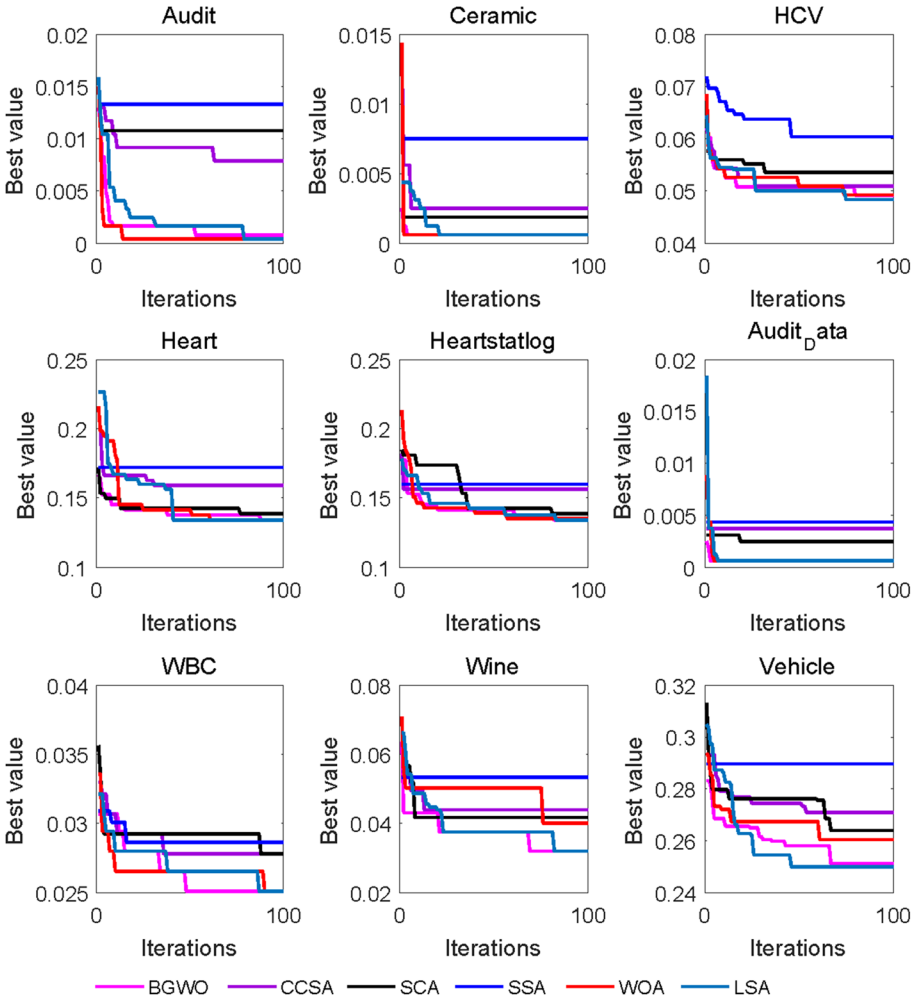

Fig. 19 The convergence curves of all algorithms for the feature section on 9 UCI datasets

Appendix

Table 23 The impact of parametery^{0}on the results

Fun	$y^0 = 0.25$	$y^0 = 0.5$	$y^0 = 0.75$	$y^0 = 0.9$
F1	114.585	119.95	**116.6784**	120.984
F2	200.4871	229.9816	**200.0000**	200.6111
F3	300.0094	301.3022	**300.0000**	300.0017
F4	417.937	418.0485	**417.8237**	428.2595
F5	**520.2452**	520.309	520.2482	520.2693
F6	603.4826	603.9908	**602.7809**	602.8813
F7	**700.162**	700.3432	700.2244	700.2948
F8	815.5213	817.5112	**814.1284**	817.7102
F9	917.4118	917.1133	914.1284	**913.5314**
F10	1251.554	1290.472	**1200.9479**	1270.914
F11	1771.266	1685.999	**1564.4724**	1719.725
F12	1200.731	1200.915	**1200.5530**	1200.624
F13	1300.244	**1300.115**	1300.1910	1300.214
F14	1400.297	1400.311	**1400.2764**	1400.312
F15	1501.452	1502.424	**1501.2107**	1501.417
F16	1602.491	**1602.261**	1602.2662	1602.287
F17	1893.462	1906.35	**1839.0022**	1907.909
F18	1818.709	1832.98	**1808.3092**	1824.406
F19	1902.704	1902.703	**1901.6906**	1902.565
F20	2021.548	2016.049	**2002.0537**	2009.782
F21	2152.634	2183.795	**2117.5061**	2138.145
F22	**2220.749**	2233.7	2222.8540	2224.576
F23	**2629.457**	2629.457	2629.4575	2629.457
F24	**2530.389**	2552.695	2538.1719	2534.363
F25	2669.59	2668.453	**2662.9213**	2670.571
F26	2700.178	2700.239	2700.1530	**2700.151**
F27	**2935.943**	2987.152	3017.0294	2995.386
F28	3202.26	3216.833	**3178.8117**	3204.359
F29	**3148.37**	4520.16	13,255.8509	3294.059
F30	3654.251	3809.743	**3596.8012**	3631.247
F31	**100.5524**	288.9802	101.3899	101.5678
F32	1598.588	1591.394	**1485.2930**	1606.152
F33	726.8894	732.7538	729.6887	**726.4377**
F34	1901.258	1901.517	**1900.9662**	1901.225
F35	**1787.05**	1798.053	1795.5017	1822.845
F36	**1600.238**	1600.49	1600.6600	1600.662
F37	**2121.5**	2154.921	2144.9284	2122.386
F38	2303.595	2304.972	2303.8299	**2299.844**
F39	2633.207	**2591.83**	2600.8587	2612.066
F40	**2914.985**	2924.608	2928.7641	2926.414
Winner	12	3	21	4

Bold entries indicates optimal result

Table 24 The Impact of parameterδ_{int}on the result

Fun	$\delta_{\text{int}} = 1$	$\delta_{\text{int}} = 1.5$	$\delta_{\text{int}} = 2$	$\delta_{\text{int}} = 2.5$
F1	**116.6784**	791.146	127.705	297.151
F2	**200.0000**	200.1356	200.8179	200.1755
F3	**300.0000**	300.0092	300.0243	300.0049
F4	**417.8237**	428.2578	417.8237	421.7353
F5	520.2482	**520.2365**	520.2586	520.239
F6	602.7809	**602.5988**	602.9983	602.673
F7	700.2244	**700.0854**	700.1518	700.123
F8	**814.1284**	818.9042	816.7153	816.0188
F9	**914.1284**	916.2178	915.9193	916.3173
F10	**1200.9479**	1280.227	1256.636	1300.639
F11	**1564.4724**	1635.436	1654.232	1815.683
F12	**1200.5530**	1200.939	1200.77	1200.852
F13	**1300.1910**	1300.249	1300.255	1300.213
F14	1400.2764	1400.29	1400.386	**1400.242**
F15	1501.2107	1501.258	**1501.178**	1501.204
F16	1602.2662	1602.319	1602.378	**1602.226**
F17	1839.0022	**1806.637**	1971.399	1855.309
F18	**1808.3092**	1817.286	1818.005	1812.247
F19	1901.6906	1902.03	1901.619	**1901.45**
F20	**2002.0537**	2008.921	2006.869	2007.434
F21	2117.5061	2140.667	2120.362	**2117.022**
F22	2222.8540	2222.039	**2218.991**	2219.463
F23	2629.4575	**2629.457**	**2629.457**	**2629.457**
F24	2538.1719	2531.707	**2529.317**	2531.985
F25	2662.9213	2682.573	**2660.942**	2666.1
F26	2700.1530	**2700.121**	2700.154	2700.179
F27	3017.0294	2931.191	3016.677	**2852.374**
F28	**3178.8117**	3189.78	3205.766	3180.484
F29	13,255.8509	3229.163	3239.356	**3135.162**
F30	3596.8012	**3583.588**	3603.031	3601.192
F31	101.3899	101.4211	100.0279	**100.0116**
F32	**1485.2930**	1588.763	1556.156	1494.462
F33	729.6887	724.6936	721.8807	**721.132**
F34	1900.9662	**1900.828**	1900.912	1900.853
F35	1795.5017	1820.51	1800.555	**1786.028**
F36	1600.6600	**1600.17**	1600.629	1600.617
F37	2144.9284	**2108.14**	2123.441	2112.596
F38	2303.8299	**2302.555**	2297.725	2298.434
F39	**2600.8587**	2675.827	2654.631	2644.652
F40	2928.7641	2931.209	2935.241	**2914.703**
Winner	15	11	5	11

Bold entries indicates optimal result

Table 25 Results of LSA and ISSA algorithms in solving CEC 2014 problem

Fun	ISSA	LSA
F1	2.2557e6	**116.6784**
F2	1.3746e7	**200.0000**
F3	1479.475	**300.0000**
F4	424.9021	**417.8237**
F5	520.4126	**520.2482**
F6	605.2097	**602.7809**
F7	701.3192	**700.2244**
F8	832.7926	**814.1284**
F9	924.5887	**914.1284**
F10	1672.985	**1200.9479**
F11	2088.802	**1564.4724**
F12	1201.067	**1200.5530**
F13	1300.271	**1300.1910**
F14	**1400.243**	1400.2764
F15	1503.347	**1501.2107**
F16	1602.944	**1602.2662**
F17	7225.925	**1839.0022**
F18	13,912.32	**1808.3092**
F19	1903.505	**1901.6906**
F20	4721.7	**2002.0537**
F21	7463.577	**2117.5061**
F22	2263.028	**2222.8540**
F23	2629.721	**2629.4575**
F24	2548.125	**2538.1719**
F25	2668.062	**2662.9213**
F26	2700.247	**2700.1530**
F27	3061.494	**3017.0294**
F28	3389.011	**3178.8117**
F29	**4046.771**	13,255.8509
F30	4979.605	**3596.8012**
Winner	2	28

Bold entries indicates optimal result

Table 26 Search results obtained by algorithm LSA with different values of multiplication factors λ

Fun	λ = 0.5	λ = 0.6	λ = 0.7	λ = 0.75	λ = 0.8
F1	389.2927	1351.276	554.1418	**116.6784**	520.6036
F2	200.0012	200.0001	200.0006	**200**	200.0013
F3	**300**	**300**	**300**	**300**	300.0001
F4	418.2572	421.3516	421.7945	**417.8237**	418.2572
F5	**520.208**	520.2278	520.2702	520.2482	518.5535
F6	603.4072	**602.7543**	603.2434	602.7809	602.9616
F7	**700.1664**	700.2778	700.2111	700.2244	700.2321
F8	**812.835**	816.4168	813.5314	814.1284	815.6208
F9	917.8097	917.8097	916.8148	**914.1284**	915.1234
F10	1277.732	1280.893	**1188.601**	1200.9479	1319.877
F11	1650.111	1610.87	1712.185	**1564.4724**	1790.733
F12	1200.672	1200.679	1200.662	**1200.553**	1200.755
F13	1300.257	**1300.18**	1300.212	1300.191	1300.278
F14	1400.382	1400.336	**1400.275**	1400.2764	1400.308
F15	1501.319	1501.474	1501.359	1501.2107	**1501.149**
F16	1602.201	1602.127	1602.111	1602.2662	1602.651
F17	1838.028	1867.354	**1834.001**	1839.0022	1788.407
F18	1810.405	1808.802	1813.644	**1808.3092**	1811.829
F19	**1901.681**	1902.174	1902.085	1901.6906	1902.077
F20	2003.407	2003.769	2004.865	**2002.0537**	2003.702
F21	**2110.026**	2113.388	2149.881	2117.5061	2146.973
F22	2224.655	2226.586	**2222.822**	2222.854	2224.404
F23	**2629.457**	**2629.457**	**2629.457**	**2629.4575**	**2629.457**
F24	2527.859	2525.993	2526.544	2538.1719	**2525.779**
F25	2679.929	2676.853	**2656.514**	2662.9213	2661.241
F26	2700.169	2700.159	**2700.133**	2700.153	2700.173
F27	**2821.804**	2928.728	2970.83	3017.0294	2895.594
F28	3248.059	**3178.386**	3187.39	3178.8117	3201.156
F29	3129.188	3126.948	3172.588	13,255.8509	3136.061
F30	3612.884	3599.84	3577.167	3596.8012	**3564.047**
Number of champions	8	5	8	10	4

Bold entries indicates optimal result

Table 27 Search results obtained by algorithm LSA with different values of multiplication factors γ

Fun	γ = 1.5	γ = 1.6	γ = 1.7	γ = 1.8	γ = 1.9	γ = 2	γ = 2.1
F1	5832.924	2417.435	4644.85	1051.279	1984.819	116.6784	**112.9029**
F2	200.1858	200.0128	200.0065	200.0013	200.006	**200**	261.9749
F3	300.0002	300.0001	300.0001	**300**	**300**	**300**	300.2432
F4	427.8746	**417.4406**	408.3118	417.5387	411.7614	417.8237	414.7921
F5	520.2512	**518.4806**	520.2565	520.0827	520.2105	520.2482	520.2415
F6	603.2477	603.6562	603.1728	603.1822	603.2571	602.7809	**601.2539**
F7	700.258	700.1526	**700.025**	700.2055	700.2005	700.2244	700.0645
F8	816.0188	815.3223	812.5365	815.2229	815.7203	814.1284	**810.9248**
F9	917.7102	917.0138	919.3022	915.9193	917.3123	914.1284	**911.9095**
F10	1265.834	1246.19	1226.324	1255.956	1331.176	**1200.948**	1271.055
F11	1585.293	1710.105	1730.506	1714.479	1670.867	**1564.472**	1751.723
F12	1200.734	1200.707	1200.601	1200.66	1200.594	**1200.553**	1200.791
F13	1300.215	1300.231	1300.252	1300.239	1300.189	1300.191	**1300.188**
F14	1400.278	1400.254	1400.269	1400.295	**1400.228**	1400.276	1400.33
F15	**1501.005**	1501.547	1501.38	1501.24	1501.422	1501.211	1501.074
F16	1602.302	1602.256	1602.432	1602.32	**1602.191**	1602.266	1602.368
F17	1836.441	1858.814	**1793.049**	1849.001	1841.15	1839.002	1813.121
F18	1810.518	1816.244	1810.608	1813.768	1812.191	1808.309	**1806.637**
F19	1901.921	1902.046	1902.216	1901.864	1901.842	1901.691	**1900.994**
F20	2005.563	2006.161	2005.169	2003.693	2004.762	**2002.054**	2002.452
F21	2149.737	2125.418	2160.184	2115.564	2126.626	2117.506	**2109.612**
F22	2222.52	2221.15	**2219.844**	2222.06	2223.285	2222.854	2226.179
F23	**2629.457**	**2629.457**	**2629.457**	**2629.457**	**2629.457**	2629.458	2629.459
F24	**2524.249**	2532.657	2528.388	2525.16	2527.53	2538.172	2524.29
F25	**2654.697**	2659.588	2679.282	2692.717	2668.654	2662.921	2660.918
F26	2700.176	2700.157	2700.148	2700.155	**2700.131**	2700.153	2700.16
F27	2925.864	2900.529	**2897.543**	2967.184	2935.63	3017.029	3010.969
F28	3178.111	3199.563	3189.493	**3177.755**	3190.326	3178.812	3200.985
F29	3132.611	3136.36	3164.515	**3118.285**	3271.837	13,255.85	3119.387
F30	3608.091	3712.528	3614.472	**3532.666**	3556.58	3596.801	3598.349
Number of champions	4	3	5	5	5	6	8

Bold entries indicates optimal result

Table 28 Search results with fixed number of iterations

	nPop = 20	nPop = 30	nPop = 40	nPop = 50	nPop = 60	nPop = 70	nPop = 80	nPop = 90
F1	3797.91	444.967	345.3231	116.6784	125.473	151.8832	126.7637	**115.4954**
F2	2822.545	233.118	201.9384	**200**	200.0451	200.0018	200.001	200.0001
F3	340.6389	305.1787	300.0626	**300**	300.0042	300.0001	300	300
F4	417.9964	422.0836	418.7905	417.8237	417.3912	420.8682	**417.3901**	427.8242
F5	520.3459	520.3352	520.2663	520.2482	520.2631	**519.6942**	520.2708	520.2081
F6	605.417	603.3589	603.8683	602.7809	603.539	603.07	**602.6236**	603.354
F7	700.4501	700.289	700.2344	700.2244	700.1939	**700.1409**	700.1791	700.1981
F8	825.1724	821.1926	818.4387	**814.1284**	816.4168	819.7002	815.1234	815.7203
F9	926.0678	917.5112	924.9734	**914.1284**	920.3966	914.9244	916.8148	916.7153
F10	1466.059	1320.501	1387.136	**1200.948**	1265.836	1277.511	1216.037	1225.086
F11	1941.725	1738.592	1717.158	**1564.472**	1684.141	1801.496	1774.288	1605.927
F12	1201.159	1200.805	1200.63	1200.553	1200.667	1200.847	1200.774	**1200.496**
F13	1300.407	1300.356	1300.28	**1300.191**	1300.228	1300.23	1300.253	1300.234
F14	1400.299	1400.384	1400.316	1400.276	1400.294	1400.275	1400.265	**1400.237**
F15	1503.779	1502.862	1501.364	1501.211	1501.583	1501.656	1501.573	**1500.948**
F16	1602.698	1602.447	1602.494	1602.266	1602.268	1602.367	**1602.198**	1602.282
F17	2249.726	2041.029	1874.486	1839.002	1841.242	1850.969	1824.514	**1813.747**
F18	1845.567	1847.628	1822.815	1808.309	1819.426	1815.561	1816.504	**1805.552**
F19	1904.197	1902.632	1902.678	1901.691	1901.693	1902.449	**1901.63**	1901.759
F20	2058.07	2020.344	2010.684	**2002.054**	2006.184	2004.892	2007.202	2007.143
F21	2211.304	2163.826	2157.31	2117.506	**2112.8**	2140.21	2129.978	2133.894
F22	2244.867	2227.23	2233.11	**2222.854**	2224.439	2226.168	2226.095	2223.659
F23	2629.458	2629.457	2629.457	2629.458	**2629.457**	**2629.457**	**2629.457**	**2629.457**
F24	2550.759	2552.91	2537.899	2538.172	2532.535	**2527.619**	2527.957	2535.498
F25	2679.428	2684.352	2677.57	2662.921	2674.209	2675.827	**2662.6**	2663.46
F26	2700.249	2700.225	2700.244	2700.153	2700.186	2700.163	2700.169	**2700.103**
F27	2996.736	2943.105	3023.28	3017.029	2975.272	2936.147	2934.469	**2855.932**
F28	3244.612	3201.239	3193.525	**3178.812**	3191.545	3182.843	3214.867	3201.026
F29	412,854.5	3407.271	3148.586	13,255.85	3141.757	3121.399	3126.969	**3118.959**
F30	4169.874	3959.778	3618.634	3596.801	3611.175	**3524.844**	3601.457	3604.931
Winners	0	0	0	10	2	5	6	10

Bold entries indicates optimal result

Table 29 Search results with fixed number of evaluations

Fun	nPop = 20	nPop = 30	nPop = 40	nPop = 50	nPop = 60	nPop = 70	nPop = 80	nPop = 90
F1	4283.161	565.9753	150.971	116.6784	258.9861	125.1742	118.3251	**116.5818**
F2	200.6976	200.9973	200.2697	**200**	200.4441	200.6402	204.3914	203.3108
F3	300.8961	300.016	300.0469	**300**	300.0204	300.0173	300.0814	300.0443
F4	424.9098	420.9051	424.4858	417.8237	417.4002	420.918	414.4264	**410.4952**
F5	520.2372	520.2734	**520.186**	520.2482	520.2959	520.2557	520.2549	520.2613
F6	605.2757	604.1976	604.0057	**602.7809**	604.1918	603.372	603.5676	603.1121
F7	700.6025	700.6395	700.262	700.2244	700.1887	700.1692	**700.1611**	700.1776
F8	823.481	823.879	820.7946	814.1284	820.7946	816.7153	815.3223	**813.7304**
F9	923.282	920.2971	919.2027	914.1284	916.5163	920.6951	917.2128	**913.1334**
F10	1582.267	1433.113	1356.143	1200.948	1196.711	1244.736	**1184.043**	1260.187
F11	1844.374	1753.634	1899.132	**1564.472**	1646.956	1679.11	1712.04	1712.921
F12	1200.783	1200.75	1200.84	**1200.553**	1200.695	1200.878	1201.011	1201.137
F13	1300.291	1300.341	1300.33	**1300.191**	1300.255	1300.291	1300.246	1300.242
F14	1400.414	1400.518	1400.395	1400.276	1400.355	**1400.264**	1400.308	1400.266
F15	1502.76	1503.492	1501.799	1501.211	1501.467	1501.669	**1501.21**	1501.382
F16	1602.433	1602.536	1602.432	1602.266	1602.22	1602.639	1602.472	**1602.013**
F17	2153.641	1952.321	1901.494	1839.002	1815.351	**1791.693**	1839.327	1866.854
F18	1834.794	1830.361	1824.924	**1808.309**	1823.552	1825.354	1810.834	1810.818
F19	1902.667	**1901.63**	1902.77	1901.691	1902.13	1902.056	1901.884	1901.731
F20	2032.775	2028.723	2016.601	**2002.054**	2010.298	2008.055	2006.567	2005.591
F21	2218.033	2189.543	2156.74	2117.506	2125.934	2135.775	**2116.474**	2147.361
F22	2240.321	2232.654	2224.209	2222.854	2225.862	2226.197	2225.023	**2221.161**
F23	2629.457	2629.457	2629.457	2629.458	**2629.457**	**2629.457**	**2629.457**	**2629.457**
F24	2556.539	2554.222	2534.898	2538.172	2544.651	**2527.94**	2556.267	2530.144
F25	2690.97	2686.706	2671.949	**2662.921**	2671.948	2665.437	2665.308	2680.513
F26	2700.256	2700.269	**2700.149**	2700.153	2700.172	2700.158	2700.172	2700.161
F27	2943.831	2981.291	2975.66	3017.029	2941.171	2856.468	2902.391	**2783.943**
F28	3204.621	3227.672	3208.537	3178.812	3204.033	3204.432	3194.06	**3178.225**
F29	3165.568	5806.237	3165.784	13,255.85	**3129.914**	3166.625	3141.616	3142.912
F30	3771.269	3899.005	**3596.717**	3596.801	3618.722	3613.521	3606.269	3596.835
Winners	0	1	3	9	2	4	5	9

Bold entries indicates optimal result

Table 30 Search results with different values of δ

Fun	δ = 0	δ = 0.2	δ = 0.4	δ = 0.6	δ = 0.8	δ = 1	δ = Eq. (4)
F1	317,819.86	2430.592	826.07691	100.0448	**100.023**	141.993	116.6784
F2	732,655.78	200.0009	200.00001	**200**	200	554.582	**200**
F3	504.68492	300.0001	300	**300**	300	300.002	**300**
F4	452.09056	411.5226	410.86763	424.7797	**410.434**	421.307	417.8237
F5	520.34575	520.2965	**519.4691**	520.1842	520.201	520.211	520.2482
F6	604.46775	604.0055	604.27494	603.8807	602.848	602.825	**602.7809**
F7	700.78768	700.4033	700.18136	700.204	700.194	**700.169**	700.2244
F8	815.22334	816.7153	816.4168	818.8047	813.133	**807.739**	814.1284
F9	922.98362	922.287	915.81983	918.6057	914.128	**905.381**	914.1284
F10	1455.3281	1389.785	1356.4294	1261.933	1215.92	1374.33	**1200.948**
F11	1786.6301	1822.949	1689.0956	1761.276	1568.97	1904.15	**1564.472**
F12	**1200.3741**	1200.686	1200.6787	1200.709	1200.69	1200.84	1200.553
F13	1300.3293	1300.292	1300.3327	1300.229	1300.2	1300.26	**1300.191**
F14	1400.344	1400.358	1400.2996	1400.283	1400.3	1400.44	**1400.276**
F15	1503.2749	1502.086	1501.2956	1501.335	**1500.95**	1501.09	1501.211
F16	1602.4396	1602.584	1602.2072	**1602.168**	1602.98	1602.35	1602.266
F17	4847.8004	1976.426	1832.1438	1875.533	1812.47	**1797.51**	1839.002
F18	5555.0635	1815.195	1807.389	1812.052	**1804.76**	1809.51	1808.309
F19	1903.2821	1902.642	1901.8899	1902.146	1902.01	1901.94	**1901.691**
F20	2081.0569	2008.833	2007.6522	2005.49	2003.02	2004.05	**2002.054**
F21	2405.8557	2172.905	2152.2877	2143.051	**2112.38**	2121.92	2117.506
F22	2264.4418	2234.151	2225.7286	2222.621	**2219.04**	2222.1	2222.854
F23	2630.6873	2629.457	**2629.4575**	**2629.457**	2629.46	2629.46	**2629.458**
F24	2536.1801	2528.348	2538.9154	2527.384	2525.43	**2515.61**	2538.172
F25	2686.3032	2666.987	2667.6324	2681.748	2673.34	**2651.6**	2662.921
F26	2700.202	**2700.146**	2700.1983	2700.155	2700.18	2700.18	2700.153
F27	2905.5407	2976.658	2862.6442	2934.809	**2858.73**	2915.64	3017.029
F28	3310.3997	3204.126	3238.5707	3187.722	3179.95	3179.94	**3178.812**
F29	3481.0725	3137.565	**3111.7781**	3120.363	3114.52	3117.13	13,255.85
F30	4732.4171	3885.27	3763.0558	3551.25	3546.31	**3502.98**	3596.801
Winners	1	1	3	4	7	7	10

Bold entries indicates optimal result

Acknowledgements This study was jointly supported by the National Natural Science Foundation of China (Grant Number 62341210), Science and Technology Development Plan for Baise City (Grant Number 20233654).

Author contributions Chiwen Qu: Writing - original draft, Visualization, Formal analysis, Validation. Xiaoning Peng: Supervision, Project administration. Qilan Zeng: Project Administration, Writing - Review & Editing, Supervision.

Data availability Data is provided within the manuscript.

Declarations

Competing interests The authors declare no competing interests.

References

Abd Elaziz M, Dahou A, Abualigah L, Yu L, Alshinwan M, Khasawneh AM, Lu S (2021) Advanced metaheuristic optimization techniques in applications of deep neural networks: a review. Neural Comput Appl 33(21):14079–14099

Abdel-Basset M, El-Shahat D, Jameel M et al (2023) Exponential distribution optimizer (EDO): a novel math-inspired algorithm for global optimization and engineering problems. Artif Intell Rev:1–72

Abdesslem layeb (2023) TSALSHADE: improved LSHADE algorithm with tangent search. MATLAB central file exchange. https://www.mathworks.com/maTLABCentral/fileexchange/123400-tsalshade-improved-lshade-algorithm-with-tangent-search

Abdolrasol MGM, Hussain SMS, Ustun TS, Sarker MR, Hannan MA, Mohamed R, Ali JA, Mekhilef S, Milad A (2021) Artificial Neural networks based optimization techniques: a review. Electronics 10(21):2689

Abed-Alguni BH, Alawad NA, Barhoush M et al (2021) Exploratory cuckoo search for solving single-objective optimization problems[J]. Soft Comput 25(15):10167–10180

Abualigah L, Shehab M, Alshinwan M et al (2020) Salp swarm algorithm: a comprehensive survey. Neural Comput Appl 32(15):11195–11215

Abualigah L, Diabat A, Mirjalili S et al (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609

Agushaka JO, Ezugwu AE, Abualigah L (2022) Dwarf mongoose optimization algorithm. Comput Methods Appl Mech Eng 391:114570

Ahmadianfar I, Heidari AA, Noshadian S et al (2022) INFO: an efficient optimization algorithm based on weighted mean of vectors. Expert Syst Appl 195:116516

Ahmia I, Aider M (2019) A novel metaheuristic optimization algorithm: the monarchy metaheuristic. Turk J Electr Eng Comput Sci 27(1):362–376

Akbari E, Rahimnejad A, Gadsden SA (2021) A greedy non-hierarchical grey wolf optimizer for real-world optimization. Electron Lett 57(13):499–501

Ali MH, El-Rifaie AM, Youssef AAF et al (2023) Techno-economic strategy for the load dispatch and power flow in power grids using peafowl optimization algorithm. Energies 16(2):846

Alsattar HA, Zaidan AA, Zaidan BB (2020) Novel meta-heuristic bald eagle search optimisation algorithm. Artif Intell Rev 53(3):2237–2264

Aragón VS, Esquivel SC, Coello CAC (2010) A modified version of a T-cell algorithm for constrained optimization problems. Internat J Numer Methods Engrg 84(3):351–378
Arora S, Singh S (2019) Butterfly optimization algorithm: a novel approach for global optimization. Soft Comput 23:715–734
Askari Q, Saeed M, Younas I (2020) Heap-based optimizer inspired by corporate rank hierarchy for global optimization. Expert Syst Appl 161:113702
Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12
Aydilek IB (2018) A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Appl Soft Comput 66:232–249
Ayyarao TSLV, Ramakrishna NSS, Elavarasan RM et al (2022) War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization. IEEE Access 10:25073–25105
Bennett S (2011) Learning behaviors and learning spaces. portal: libraries and the academy. 11(3):765–789
Bernardino HS, Barbosa HJC, Lemonge ACC, Fonseca LG (2008) A new hybrid AIS-GA for constrained optimization problems in mechanical engineering. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence). pp 1455–1462
Bertsimas D, Tsitsiklis J (1993) Simulated annealing. Stat Sci 8(1):10–15
Braik MS (2021) Chameleon swarm algorithm: a bio-inspired optimizer for solving engineering design problems. Expert Syst Appl 174:114685
Braik M, Sheta A, Al-Hiary H (2021) A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm. Neural Comput Appl 33(7):2515–2547
Brajevic I (2015) Crossover-based artificial bee colony algorithm for constrained optimization problems. Neural Comput Appl 26(7):1587–1601
Brammya G, Praveena S, Ninu Preetha NS et al (2019) Deer hunting optimization algorithm: a new nature-inspired meta-heuristic paradigm. Comput J 2019:bxy133
Bruner JS (1971) "The process of education" revisited. Phi Delta Kappan 53(1):18–21
Bruner JS (2009) The process of education. Harvard University Press
Carreon-Ortiz H, Valdez F (2022) A new mycorrhized tree optimization nature-inspired algorithm. Soft Comput 26:4797–4817
Chander A, Chatterjee A, Siarry P (2011) A new social and momentum component adaptive PSO algorithm for image segmentation. Expert Syst Appl 38(5):4998–5004
Chen J, Xu H, Wu J et al (2019) Deer crossing road detection with roadside LiDAR sensor. IEEE Access 7:65944–65954
Chen H, Heidari AA, Zhao X et al (2020a) Advanced orthogonal learning-driven multi-swarm sine cosine optimization: framework and case studies. Expert Syst Appl 144:113113
Chen H, Jiao S, Wang M, Heidari AA, Zhao X (2020b) Parameters identification of photovoltaic cells and modules using diversification-enriched Harris hawks optimization with chaotic drifts. J Clean Prod 244:118778
Chen P, Zhou S, Zhang Q et al (2022) A meta-inspired termite queen algorithm for global optimization and engineering design problems. Eng Appl Artif Intell 111:104805
Cheng S, Shi Y, Qin Q et al (2014) Population diversity maintenance in brain storm optimization algorithm[J]. J Artif Intell Soft Comput Res 4(2):83–97
Chickermane H, Gea HC (1996) Structural optimization using a new local approximation method. Internat J Numer Methods Engrg 39(5):829–846
Chopra N, Ansari MM (2022) Golden jackal optimization: a novel nature-inspired optimizer for engineering applications. Expert Syst Appl 198:116924
Chun S, Kim YT, Kim TH (2013) A diversity-enhanced constrained particle swarm optimizer for mixed integer-discrete-continuous engineering design problems. Adv Mech Eng 5:130750
Coello CAC, Cortés NC (2004) Hybridizing a genetic algorithm with an artificial immune system for global optimization. Eng Optim 36(5):607–634
Das S, Suganthan PN (2010) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 15(1):4–31
Dehghani M, Trojovský P (2023) Osprey optimization algorithm: a new bio-inspired metaheuristic algorithm for solving engineering optimization problems. Front Mech Eng 8:1126450
Dehghani M, Hubálovský Š, Trojovský P (2021) Northern goshawk optimization: a new swarm-based algorithm for solving optimization problems. Ieee Access 9:162059–162080
Dehghani M, Hubálovský Š, Trojovský P (2022a) Tasmanian devil optimization: a new bio-inspired optimization algorithm for solving optimization algorithm. IEEE Access 10:19599–19620
Dehghani M, Montazeri Z, Trojovská E et al (2023) Coati optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems. Knowl-Based Syst 259:110011

Dehghani M, Trojovská E, Trojovský P (2022b) Driving training-based optimization: a new human-based metaheuristic algorithm for solving optimization problems. 4. https://doi.org/10.21203/rs.3.rs-1506972/v1
Dehkordi A, Sadiq A, Mirjalili S et al (2021) Nonlinear-based chaotic harris hawks optimizer: algorithm and internet of vehicles application. Appl Soft Comput 109:107574
Dhanya D, Arivudainambi D (2019) Dolphin partner optimization based secure and qualified virtual machine for resource allocation with streamline security analysis. Peer- Peer Netw Appl 12(5):1194–1213
Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174
Dhiman G, Kumar V (2017) Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 114:48–70
Dhiman G, Kumar V (2019) Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. Knowl-Based Syst 165:169–196
Dhrubajyoti G, Ananda Ra DR, Shibendu Shekhar R (2021) A partition cumunification based genetic-firefly algorithm for single objective optimization. Sādhanā 46(3):1–31
Dorian Sidea (2024) Improved salp swarm algorithm. MATLAB Central File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/155984-improved-salp-swarm-algorithm
Dos Santos Coelho L (2010) Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. Expert Syst Appl 37(2):1676–1683
Duman S, Kahraman HT, Sonmez Y, Guvenc U, Kati M, Aras S (2022) A powerful meta-heuristic search algorithm for solving global optimization and real-world solar photovoltaic parameter estimation problems. Eng Appl Artif Intell 111:104763
Duman S, Kahraman HT, Kati M (2023) Economical operation of modern power grids incorporating uncertainties of renewable energy sources and load demand using the adaptive fitness-distance balance-based stochastic fractal search algorithm. Eng Appl Artif Intell 117:105501
Emami H (2022) Stock exchange trading optimization algorithm: a human-inspired method for global optimization. J Supercomput 78(2):2125–2174
Emary E, Zawbaa HM, Hassanien AE (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371–381
Ewees AA, Al-qaness MAA, Abualigah L (2022) HBO-LSTM: optimized long short term memory with heap-based optimizer for wind power forecasting. Energy Convers Manage 268:116022
Ezugwu AE (2022) Advanced discrete firefly algorithm with adaptive mutation-based neighborhood search for scheduling unrelated parallel machines with sequence-dependent setup times. Int J Intell Syst 37(8):4612–4653
Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190
Farshad R, Hamid RS, Mohamed AE, Shaker H, Ali E, Mohammed A, Tamer A (2022) An enhanced grey wolf optimizer with a velocity-aided global search mechanism. Mathematics 10(3):351
Feng Z, Niu W, Liu S (2021) Cooperation search algorithm: a novel metaheuristic evolutionary intelligence algorithm for numerical optimization and engineering optimization problems. Appl Soft Comput 98:106734
Forrest S (1993) Genetic algorithms: principles of natural selection applied to computation. Science 261(5123):872–878
Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29(1):17–35
Ghafil HN, Jármai K (2020) Dynamic differential annealed optimization: new metaheuristic optimization algorithm for engineering applications. Appl Soft Comput 93:106392
Ghasemi M, Kadkhoda Mohammadi S, Zare M et al (2022) A new firefly algorithm with improved global exploration and convergence with application to engineering optimization. Decis Anal J 5:100125
Gurrola-Ramos J, Hernàndez-Aguirre A, Dalmau-Cedeño O (2020) COLSHADE for real-world single-objective constrained optimization problems. 2020 IEEE Congress on Evolutionary Computation (CEC) 2020, 1–8
Guvenc U, Duman S, Kahraman HT, Aras S, Katı M (2021) Fitness-distance balance based adaptive guided differential evolution algorithm for security-constrained optimal power flow problem incorporating renewable energy sources. Appl Soft Comput 108:107421
Hashim FA, Hussien AG (2022) Snake optimizer: a novel meta-heuristic optimization algorithm. Knowl-Based Syst 242:108320

Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S (2019) Henry gas solubility optimization: a novel physics-based algorithm. Futur Gener Comput Syst 101:646–667
Hayyolalam V, Kazem AAP (2020) Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Eng Appl Artif Intell 87:103249
Hellwig M, Beyer H (2018) A matrix adaptation evolution strategy for constrained real-parameter optimization. In: 2018 IEEE Congress on Evolutionary Computation (CEC), 2018: 1–8
Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, pp 211
Huang J, Gao L, Li X (2015) An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes. Appl Soft Comput 36:349–356
Hussain K, Salleh M, Cheng S et al (2019) On the exploration and exploitation in popular swarm-based metaheuristic algorithms. Neural Comput Appl 31:7665–7683
Ibrahim I, Hossain M, Duck B, Nadarajah M (2020) An improved wind driven optimization algorithm for parameters identification of a triple-diode photovoltaic cell model. Energy Convers Manag 213:112872
Jain M, Singh V, Rani A (2019) A novel nature-inspired algorithm for optimization: squirrel search algorithm. Swarm Evol Comput 44:148–175
James JQ, Li VOK (2015) A social spider algorithm for global optimization. Appl Soft Comput 30:614–627
Jian X, Zhu Y (2021) Parameters identification of photovoltaic models using modified Rao-1 optimization algorithm. Optik 231:166439
Kahraman H, Bakir H, Duman S, Katı M, Aras S, Guvenc U (2022) Dynamic FDB selection method and its application: modeling and optimizing of directional overcurrent relays coordination. Appl Intell 52:4873–4908
Kahraman HT, Katı M, Aras S, Taşci D (2023) Development of the Natural Survivor Method (NSM) for designing an updating mechanism in metaheuristic search algorithms. Eng Appl Artif Intell 122:106121
Kaidi W, Khishe M, Mohammadi M (2022) Dynamic levy flight chimp optimization. Knowl-Based Syst 235:107625
Kallioras NA, Lagaros ND, Avtzis DN (2018) Pity beetle algorithm – a new metaheuristic inspired by the behavior of bark beetles. Adv Eng Softw 121:147–166
Karami H, Anaraki MV, Farzin S, Mirjalili S (2021) Flow direction algorithm (fda): a novel optimization approach for solving optimization problems. Comput Ind Eng 156:107224
Kaur S, Awasthi LK, Sangal AL et al (2020) Tunicate Swarm Algorithm: a new bio-inspired based metaheuristic paradigm for global optimization. Eng Appl Artif Intell 90:103541
Kaveh A, Bakhshpoori T (2016) Water evaporation optimization: a novel physically inspired optimization algorithm. Comput Struct 167:69–85
Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Softw 110:69–84
Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3):267–289
Kaveh A, Seddighian MR, Ghanadpour E (2020a) Black hole mechanics optimization: a novel meta-heuristic algorithm. Asian Journal of Civil Engineering 21(7):1129–1149
Kaveh A, Akbari H, Hosseini SM (2020b) Plasma generation optimization: a new physically-based metaheuristic algorithm for solving constrained optimization problems. Eng Comput 38(4):1554–1606
Khalilpourazari S, Doulabi HH, Çiftçioğlu AÖ et al (2021) Gradient-based grey wolf optimizer with Gaussian walk: application in modelling and prediction of the COVID-19 pandemic. Expert Syst Appl 177:114920
Khelil K, Nicolas Z, Naoufel C, Samir BB (2022) Exponential particle swarm optimization for global optimization. IEEE Access 10:78320–78344
Kılkış Ş, Kılkış B (2019) An urbanization algorithm for districts with minimized emissions based on urban planning and embodied energy towards net-zero exergy targets. Energy 179:392–406
Kumar DS, Zelinka I (2020) A self-adaptive spherical search algorithm for real-world constrained optimization problems. Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020:13–14
Kumar A, Das S, Zelinka I (2020) A modified covariance matrix adaptation evolution strategy for real-world constrained optimization problems. Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020:11–12

Kundu T, Garg H (2022) A hybrid TLNNABC algorithm for reliability optimization and engineering design problems. Eng Comp 38(6):5251–5295
Li C, Yang S, Nguyen TT (2011) A self-learning particle swarm optimizer for global optimization problems. IEEE Trans Syst Man Cybernetics B Cybern 42(3):627–646
Li L, Chang YB, Tseng ML et al (2020a) Wind power prediction using a novel model on wavelet decomposition-support vector machines-improved atomic search algorithm. J Clean Prod 270:121817
Li S, Chen H, Wang M et al (2020c) Slime mould algorithm: a new method for stochastic optimization. Futur Gener Comput Syst 111:300–323
Li Z, Liang Y, Liao Q, Zhang H (2021) A review of multiproduct pipeline scheduling: from bibliometric analysis to research framework and future research directions. J Pipeline Sci Eng 1(4):395–406
Li Z, Tam V, Yeung LK (2020b) A study on parameter sensitivity analysis of the virus spread optimization. 2020 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 1535–1542
Liang J, Qiao K, Yu K, Ge S, Qu B, Xu R, Li K (2020) Parameters estimation of solar photovoltaic models via a self-adaptive ensemble-based differential evolution. Sol Energy 207:336–346
Lin X, Yu X, Li W (2022) A heuristic whale optimization algorithm with niching strategy for global multi-dimensional engineering optimization. Comput Ind Eng 171:108361
Liu H, Gu F, Zhang Q (2013) Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. IEEE Trans Evol Comput 18(3):450–455
Long W, Wu T, Tang M, Xu M, Cai S (2020) Grey wolf optimizer algorithm based on lens imaging learning strategy. Acta Automat Sin 46(10):2148–2164
Mandal M, Mukhopadhyay A (2015) A novel PSO-based graph-theoretic approach for identifying most relevant and non-redundant gene markers from gene expression data. Int J Parallel Emergent Distrib Syst 30(3):175–192
Martello S, Pulleyblank WR, Toth, de Werra D (1984) Balanced optimization problems. Oper Res Lett 3(5):275–278
Masadeh R, Mahafzah BA, Sharieh A (2019) Sea lion optimization algorithm. Int J Adv Comput Sci Appl 10(5):388–395
McFarland D, Bösser T, Bosser T (1993) Intelligent behavior in animals and robots. Mit Press
Meng XB, Li HX, Gao XZ (2019) An adaptive reinforcement learning-based bat algorithm for structural design problems. Int J Bio-Inspired Comput 14(2):114–124
Meng OK, Pauline O, Kiong SC (2021) A carnivorous plant algorithm for solving global optimization problems. Appl Soft Comput J 98:106833
Mirjalili S (2015a) The ant lion optimizer. Adv Eng Softw 83:80–98
Mirjalili S (2015b) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249
Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
Mirjalili S, Gandomi AH, Mirjalili SZ, Shahrzad S, Hossam F, Seyed M (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
Mirjalili S (2019) Genetic algorithm. Evolutionary algorithms and neural networks. Springer, Cham, pp 43–55
Moghaddam FF, Moghaddam RF (2012) Cheriet M. Curved space optimization: a random search based on general relativity theory. arXiv preprint arXiv:1208.2214
Mohammadi-Balani A, Nayeri MD, Azar A, Taghizadeh-Yazdi M (2021) Golden eagle optimizer: a nature-inspired metaheuristic algorithm. Comput Ind Eng 152:107050
Mohammed H, Rashid T (2020) A novel hybrid GWO with WOA for global numerical optimization and solving pressure vessel design. Neural Comput Appl 32:14701–14718
Montemurro M, Vincenti A, Vannucci P (2013) The automatic dynamic penalization method (ADP) for handling constraints with genetic algorithms. Comput Methods Appl Mech Engrg 256:70–87
Moscato P, Cotta C, Mendes A (2004) Memetic algorithms. New optimization techniques in engineering. Springer, Berlin, Heidelberg, pp 53–85
Naik MK, Panda R, Abraham A (2021) Adaptive opposition slime mould algorithm. Soft Comput 25:14297–14313
Naruei I, Keynia F (2022) Wild horse optimizer: a new meta-heuristic algorithm for solving engineering optimization problems. Eng with Comput 38(4):3025–3056
Naruei I, Keynia F, Sabbagh Molahosseini A (2022) Hunter–prey optimization: algorithm and applications. Soft Comput 26(3):1279–1314
Nematollahi AF, Rahiminejad A, Vahidi B (2017) A novel physical based meta-heuristic optimization method known as lightning attachment procedure optimization. Appl Soft Comput 59:596–621

Nematollahi AF, Rahiminejad A, Vahidi B (2020) A novel meta-heuristic optimization method based on golden ratio in nature. Soft Comput 24(2):1117–1151
Onay FK, Aydemır SB (2022) Chaotic hunger games search optimization algorithm for global optimization and engineering problems. Math Comput Simul 192:514–536
Örnek BN, Aydemir SB, Düzenli T et al (2022) A novel version of slime mould algorithm for global optimization and real world engineering problems: enhanced slime mould algorithm. Math Comput Simul 198:253–288
Pan H, Wang L, Liu B (2006) Particle swarm optimization for function optimization in noisy environment. Appl Math Comput 181(2):908–919
Pan JS, Sun B, Chu SC et al (2023) A parallel compact gannet optimization algorithm for solving engineering optimization problems. Mathematics 11(2):439
Parsopoulos E, Vrahatis MN (2002) Recent approaches to global optimization problems through particle swarm optimization. Nat Comput 1(2):235–306
Peraza-Vázquez H, Peña-Delgado AF, Echavarría-Castillo G et al (2021) A bio-inspired method for engineering design optimization inspired by dingoes hunting strategies. Math Probl Eng 2021:9107547
Pham TX, Siarry P, Oulhadj H (2018) Integrating fuzzy entropy clustering with an improved PSO for MRI brain image segmentation. Appl Soft Comput 65:230–242
Połap D, Woźniak M (2021) Red fox optimization algorithm. Expert Syst Appl 166:114107
Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization. Swarm Intell 1(1):33–57
Qiao W, Lu H, Zhou G et al (2020) A hybrid algorithm for carbon dioxide emissions forecasting based on improved lion swarm optimizer. J Clean Prod 244:118612
Rachdi E, El Merabet Y, Akhtar Z et al (2020) Directional neighborhood topologies based multi-scale quinary pattern for texture classification. IEEE Access 8:212233–212246
Ramshanker A, Chakraborty S (2022) Maiden application of skill optimization algorithm on cascaded multi-level neuro-fuzzy based power system stabilizers for damping oscillations. Int J Renew Energy Res (IJRER) 12(4):2152–2167
Rao R, Savsani V, Vakharia D (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315
Rao RV, Savsani VJ, Balic J (2012) Teaching-learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems. Eng Optim 44(12):1447–1462
Rejowski J, Pinto JM (2003) Scheduling of a multiproduct pipeline system. Comput Chem Eng 27(8):1229–1246
Ruan Y, Li KY, Zheng R et al (2022) Cholinergic neurons in the pedunculopontine nucleus guide reversal learning by signaling the changing reward contingency. Cell Rep 38(9):110437
Salimi H (2015) Stochastic fractal search. a powerful metaheuristic algorithm. Knowl-Based Syst 75:1–18
Saremi S, Mirjalili S, Lewis A (2017) Grasshopper optimisation algorithm: theory and application. Adv Eng Softw 105:30–47
Schoenewolf G (1990) Emotional contagion: behavioral induction in individuals and groups. Mod Psychoanal 15(1):49–61
Schwefel HP, Rudolph G (1995) Contemporary evolution strategies. European conference on artificial life. Springer, Berlin, Heidelberg, pp 891–907
Seo JH, Im CH, Heo CG (2006) Multimodal function optimization based on particle swarm optimization. IEEE Trans Magn 42(4):1095–1098
Sette S, Boullart L (2001) Genetic programming: principles and applications. Eng Appl Artif Intell 14(6):727–736
Seyyedabbasi A, Kiani F (2023) Sand Cat swarm optimization: a nature-inspired algorithm to solve global optimization problems. Eng Comput 39(4):2627–2651
Sharma A, Shoval S, Sharma A, Jitendra KP (2022) Path planning for multiple targets interception by the swarm of UAVs based on swarm intelligence algorithms: a review. IETE Tech Rev 39(3):675–697
Shi XH, Liang YC, Lee HP et al (2005) An improved GA and a novel PSO-GA-based hybrid algorithm. Inf Process Lett 93(5):255–261
Shitu S, Jagdish CB (2022) Mutation-driven grey wolf optimizer with modified search mechanism. Expert Syst Appl 194:116450
Shubham G, Kusum D (2020) A memory-based grey wolf optimizer for global optimization tasks. Appl Soft Comput 93:106367
Solis F, Wets J (1981) Minimization by random search techniques. Math Oper Res 6(1):19–30
Song S, Wang P, Heidari AA, Wang M, Zhao X, Chen H, He W, Xu S (2021) Dimension decided Harris Hawks optimization with Gaussian mutation: balance analysis and diversity patterns. Knowl-Based Syst 215:106425

Sun X, Croke B, Roberts S et al (2021) Comparing methods of randomizing Sobol′ sequences for improving uncertainty of metrics in variance-based global sensitivity estimation. Reliab Eng Syst Saf 210:107499
Tian G, Fathollahi-Fard AM, Ren Y, Li Z, Jiang X (2022) Multi-objective scheduling of priority-based rescue vehicles to extinguish forest fires using a multi-objective discrete gravitational search algorithm. Inf Sci 608:578–596
Trivedi A, Srinivasan D, Biswas N (2018) An improved unified differential evolution algorithm for constrained optimization problems. 2018 IEEE Congress on Evolutionary Computation (CEC), 2018:1–10
Trojovská E, Dehghani M (2022) A new human-based metahurestic optimization method based on mimicking cooking training. Sci Rep 12(1):14861
Trojovská E, Dehghani M, Trojovský P (2022) Zebra optimization algorithm: a new bio-inspired optimization algorithm for solving optimization algorithm. IEEE Access 10:49445–49473
Trojovský P, Dehghani M (2022) Pelican optimization algorithm: a novel nature-inspired algorithm for engineering applications. Sensors 22(3):855
Trojovský P, Dehghani M (2023) Subtraction-average-based optimizer: a new swarm-inspired metaheuristic algorithm for solving optimization problems. Biomimetics 8(2):149
Tutueva AV, Nepomuceno EG, Karimov AI et al (2020) Adaptive chaotic maps and their application to pseudo-random numbers generation. Chaos Solitons Fractals 133:109615
Wang GG, Deb S, Cui Z (2019) Monarch butterfly optimization. Neural Comput Appl 31:1995–2014
Wang L, Cao Q, Zhang Z et al (2022a) Artificial rabbits optimization: a new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. Eng Appl Artif Intell 114:105082
Wang Z, Pan J, Huang K et al (2022b) Hybrid gray wolf optimization and cuckoo search algorithm based on the taguchi theory. Advances in intelligent information hiding and multimedia signal processing. Springer, Singapore, pp 219–228
Wilson AJ, Pallavi DR, Ramachandran M (2022) A review on memetic algorithms and its developments. Electrical Automation Eng 1(1):7–12
Xu Z, Heidari AA, Kuang F et al (2023) Enhanced Gaussian bare-bones grasshopper optimization: mitigating the performance concerns for feature selection. Expert Syst Appl 212:118642
Xue J, Shen B (2020) A novel swarm intelligence optimization approach: sparrow search algorithm. Syst Sci Control Eng 8(1):22–34
Xue J, Shen B (2023) Dung beetle optimizer: a new meta-heuristic algorithm for global optimization. J Supercomput 79(7):7305–7336
Yaghini M, Khoshraftar MM, Fallahi M (2013) A hybrid algorithm for artificial neural network training. Eng Appl Artif Intell 26(1):293–301
Yang XS, Deb S (2014) Cuckoo search: recent advances and applications. Neural Comput Appl 24(1):169–174
Yapici H, Cetinkaya N (2019) A new meta-heuristic optimizer: pathfinder algorithm. Appl Soft Comput 78:545–568
Yazdani M, Jolai F (2016) Lion Optimization Algorithm (LOA): a nature-inspired metaheuristic algorithm. J Comput Des Eng 3(1):24–36
Yu K, Qu B, Yue C, Ge S, Chen X, Liang J (2019) A performance-guided JAYA algorithm for parameters identification of photovoltaic cell and module. Apply Energy 237:241–257
Yu H, Gao Y, Wang L et al (2020) A hybrid particle swarm optimization algorithm enhanced with nonlinear inertial weight and Gaussian mutation for job shop scheduling problems. Mathematics 8(8):1355
Zamani H, Nadimi-Shahraki MH, Gandomi AH (2019) CCSA: conscious neighborhood-based crow search algorithm for solving global optimization problems. Appl Soft Comput 85:105583
Zervoudakis K, Tsafarakis S (2020) A mayfly optimization algorithm. Comput Ind Eng 145:106559
Zhang Y, Jin Z (2022) Comprehensive learning Jaya algorithm for engineering design optimization problems. J Intell Manuf 33(5):1229–1253
Zhang J, Xiao M, Gao L, Pan Q (2018) Queuing search algorithm: a novel metaheuristic algorithm for solving engineering optimization problems. Appl Math Model 63:464–490
Zhang M, Long D, Qin T et al (2020) A chaotic hybrid butterfly optimization algorithm with particle swarm optimization for high-dimensional optimization problems. Symmetry 12(11):1800
Zhang Q, Li H, Liu Y et al (2021a) A new quantum particle swarm optimization algorithm for controller placement problem in software-defined networking. Comput Electr Eng 95:107456
Zhang Y, Chi A, Mirjalili S (2021b) Enhanced Jaya algorithm: a simple but efficient optimization method for constrained engineering design problems. Knowl-Based Syst 233:107555
Zhang H, Liu T, Ye X, Heidari AA, Liang G, Chen H, Pan Z (2022) Differential evolution-assisted salp swarm algorithm with chaotic structure for real-world problems. Eng Comput:1–35

Zhao W, Wang L, Zhang Z (2019) Supply-demand-based optimization: a novel economics-inspired algorithm for global optimization. IEEE Access 7:73182–73206
Zhong C, Li G, Meng Z (2022) Beluga whale optimization: a novel nature-inspired metaheuristic algorithm. Knowl-Based Syst 251:109215
Zitouni F, Harous S, Maamri R (2020) The solar system algorithm: a novel metaheuristic method for global optimization. IEEE Access 9:4542–4565
Zitouni F, Harous S, Belkeram A, Hammou LEB (2021) The Archerfish hunting optimizer: a novel metaheuristic algorithm for global optimization. Arab J Sci Eng. https://doi.org/10.1007/s13369-021-06208-z