Computers in Biology and Medicine 186 (2025) 109587

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

Success History Adaptive Competitive Swarm Optimizer with Linear Population Reduction: Performance benchmarking and application in eye disease detection

Rui Zhong [a], Zhongmin Wang [b,*], Abdelazim G. Hussien [c,d], Essam H. Houssein [e], Ibrahim Al-Shourbaji [f,g], Mohamed A. Elseify [h], Jun Yu [i]

[a] Information Initiative Center, Hokkaido University, Sapporo, Japan
[b] College of Tropical Crops, Yunnan Agricultural University, Yunnan, China
[c] Department of Computer and Information Science, Linköping University, Linköping, Sweden
[d] Faculty of Science, Fayoum University, Faiyum, Egypt
[e] Faculty of Computers and Information, Minia University, Minia, Egypt
[f] Department of Electrical and Electronics Engineering, Jazan, Jazan University, Jazan, Saudi Arabia
[g] Department of Computer Science, University of Hertfordshire, Hatfield, UK
[h] Department of Electrical Engineering, Faculty of Engineering, Al-Azhar University, Qena, Egypt
[i] Institute of Science and Technology, Niigata University, Niigata, Japan

ARTICLE INFO

Dataset link: https://github.com/RuiZhong961230/L-SHACSO

Keywords:
Metaheuristic algorithms (MAs)
Competitive Swarm Optimizer (CSO)
Success History Adaptation
Linear Population Reduction
Eye disease detection

ABSTRACT

Eye disease detection has achieved significant advancements thanks to artificial intelligence (AI) techniques. However, the construction of high-accuracy predictive models still faces challenges, and one reason is the deficiency of the optimizer. This paper presents an efficient optimizer named Success History Adaptive Competitive Swarm Optimizer with Linear Population Reduction (L-SHACSO). Inspired by the effective success history adaptation scheme and linear population reduction strategy in Differential Evolution (DE), we introduce these techniques into CSO to enable the automatic and intelligent adjustment of hyper-parameters during optimization thereby balancing exploration and exploitation across different phases. To thoroughly investigate the performance of L-SHACSO, we conduct extensive numerical experiments on CEC2017, CEC2020, CEC2022, and eight engineering problems. State-of-the-art optimizers including jSO and L-SHADE-cnEpSin and recently proposed metaheuristic algorithms (MAs) such as RIME and the Parrot Optimizer (PO) are employed as competitors. Experimental results confirm the superiority of L-SHACSO across various optimization tasks. Furthermore, we integrate L-SHACSO into DenseNet and Extreme Learning Machine (ELM) and propose DenseNet-L-SHACSO-ELM for eye disease detection, where the features extracted by the pre-trained DenseNet are fed into L-SHACSO-optimized ELM for classification. Experiments on public datasets confirm the feasibility and effectiveness of our proposed model, which has great potential in real-world scenarios. The source code of this research is available at https://github.com/RuiZhong961230/L-SHACSO.

1. Introduction

In recent years, the application of artificial intelligence (AI) techniques in medical imaging has opened new avenues for enhancing diagnostic precision and operational efficiency in healthcare, particularly in the domain of eye disease detection. Vision-threatening diseases such as cataracts, diabetic retinopathy, and glaucoma continue to rise globally, and this tendency poses a significant public health challenge and impacts the quality of life for millions. Early and precise detection of these diseases is essential for effective intervention and slowing disease progression. However, traditional diagnostic methods are constrained by several limitations, which often rely on subjective interpretation by clinicians, are susceptible to inter-observer variability, and can be resource-intensive and require extensive time and specialized expertise. These limitations have fueled a growing interest in computational approaches, particularly those utilizing machine

* Corresponding author.
E-mail addresses: zhongrui@iic.hokudai.ac.jp (R. Zhong), zhongminwang@ynau.edu.cn (Z. Wang), abdelazim.hussien@liu.se (A.G. Hussien), essam.halim@mu.edu.eg (E.H. Houssein), alshourbajiibrahim@gmail.com (I. Al-Shourbaji), mohamed_1988@azhar.edu.eg (M.A. Elseify), yujun@ie.niigata-u.ac.jp (J. Yu).

https://doi.org/10.1016/j.compbiomed.2024.109587
Received 10 November 2024; Received in revised form 3 December 2024; Accepted 14 December 2024
Available online 2 January 2025

learning and optimization approaches, to support and streamline diagnostic workflows. Pratap and Kokil [1] presented a hybrid deep learning model for identifying various stages of cataract development — categorized as normal, mild, moderate, and severe — by analyzing fundus images. The feature extraction process is conducted using a pre-trained CNN, while the selected features are subsequently input into the Support Vector Machine (SVM) classifier to distinguish between the different stages of cataract progression. This hybrid model leverages the robust feature extraction capabilities of CNN and the high classification accuracy of SVM. Experiments on fundus images achieve 92.91% accuracy, which demonstrates the competitive performance of the proposal in categorizing cataract severity. Sudhan et al. [2] proposed an effective approach for the early detection of glaucoma, where the U-Net is for image segmentation, the DenseNet-201 is employed for feature extraction through transfer learning., and the selected features are fed to Deep CNN (DCNN) for the classification. Experiments in the ORIGA dataset and compared with VGG-19, Inception, ResNet, and DenseNet confirm the superiority of the proposed approach, which achieves 98.82% accuracy in the training dataset and 96.90% in the test dataset. Uyar et al. [3] introduced an efficient approach named ABCEnsemble, which integrates Convolutional Neural Networks (CNNs) into the resilience of ensemble learning to achieve the high-performance classification of eye diseases. Fifteen pre-trained CNN models are employed and fine-tuned in the eye disease dataset, and the best three models are selected for the ensemble using the weighted voting mechanism, where the weights of models are optimized by the Artificial Bee Colony (ABC) algorithm. The effectiveness of ABCEnsemble is validated in the eye disease detection datasets and achieves average performance metrics of 98.84% accuracy, 98.90% precision, 98.84% recall, and a 98.85% F1 score.

Although numerous approaches have achieved substantial success in eye disease detection, constructing predictive models with consistently high accuracy remains challenging One of the key obstacles lies in the limitations of existing optimization algorithms [4–7]. Effective optimizers are essential for fine-tuning model parameters, enhancing convergence speed, and improving overall classification accuracy. However, many conventional optimizers struggle with the complex fitness landscape when training models in medical imaging data. Consequently, deficiencies in optimization can lead to suboptimal model performance and further hamper the ability to detect diseases at early stages reliably.

Fortunately, metaheuristic algorithms (MAs) provide an effective framework for addressing these optimization challenges [8,9]. Drawing inspiration from the behaviors of organisms and natural phenomena, MAs simulate processes such as the preying and mating behaviors of animals [10–12], chemical reactions [13,14], and even mirage formation [15] to refine solutions and approach optima iteratively. Fig. 1 presents a classification of representative MAs. Thanks to the superior characteristics of MAs such as flexibility, adaptability, and applicability, MAs have been one of the most prevalent optimization techniques in both academic research and real-world applications [16,17].

This paper focuses on the Competitive Swarm Optimizer (CSO) [18], an efficient variant of Particle Swarm Optimization (PSO) [19]. Many works attempt to introduce efficient search operators [20,21], integrate learning mechanisms [22,23], and hybridize with other machine learning [24,25] and MA approaches [26,27] to enhance the performance of CSO. However, the importance of hyper-parameter configuration in CSO is commonly neglected. Motivated by the effective and flexible parameter adaptation strategy in Success History Adaptive Differential Evolution with Linear Population Reduction (L-SHADE) [28], we integrate the Success-History-based parameter adaptation and Linear Population Reduction schemes into CSO and proposed an enhanced version called Success History Adaptive CSO with Linear Population Reduction (L-SHACSO). This integration is expected to adjust the hyper-parameters for L-SHACSO automatically and intelligently during optimization. In numerical experiments, we conduct comprehensive comparison experiments to investigate the performance of our proposed L-SHACSO compared with state-of-the-art optimizers such as jSO [29] and L-SHADE-cnEpSin [30] and recently proposed MAs including RIME algorithm [31] and Parrot Optimizer [32] on IEEE-CEC2017, CEC2020, CEC2022, and eight engineering optimization problems. The experimental results and statistical analysis confirm the competitiveness of our proposed L-SHACSO in various optimization challenges. Additionally, we notice the potential of extending L-SHACSO into the eye disease detection domain and integrating it into DenseNet and Extreme Learning Machine (ELM) to form DenseNet-L-SHACSO-ELM for Eye Disease Detection, where the DenseNet is responsible for the feature extraction and L-SHACSO-optimized ELM classifies eye diseases. Experiments on public datasets demonstrate the superiority and flexibility of our proposed model against well-known deep learning methods. Briefly, the main contributions of this paper can be summarized as follows.

- We equip efficient and scalable Success History Adaptation and Linear Population Reduction from L-SHADE to CSO and develop L-SHACSO.
- Comprehensive numerical experiments in CEC2017, CEC2020, CEC2022, and eight engineering problems against state-of-the-art optimizers confirm the competitiveness of the proposed L-SHACSO.
- The sensitivity experiments are conducted to demonstrate the robustness of L-SHACSO.
- We further integrate L-SHACSO into DenseNet and ELM to propose DenseNet-L-SHACSO-ELM for eye disease detection.

The remainder of this paper is organized as follows: Section 2 presents the related works including the original CSO and its literature review. Section 3 details our proposed L-SHACSO. The benchmarking experiments are introduced in Section 4, while the application in eye disease detection is in Section 5. Finally, Section 5 concludes this paper.

2. Preliminaries

2.1. Competitive Swarm Optimizer (CSO)

The framework of traditional PSO is presented in Eq. (1).

$$\begin{aligned} \boldsymbol{v}_i^{t+1} &= \omega \cdot \boldsymbol{v}_i^t + c_1 \cdot \boldsymbol{r}_1 \cdot (\boldsymbol{x}_{pbest}^t - \boldsymbol{x}_i^t) + c_2 \cdot \boldsymbol{r}_2 \cdot (\boldsymbol{x}_{gbest}^t - \boldsymbol{x}_i^t) \\ \boldsymbol{x}_i^{t+1} &= \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1} \end{aligned} \tag{1}$$

Here, $\boldsymbol{v}_i^t$ and $\boldsymbol{x}_i^t$ are D-dimensional vectors to denote the velocity and location of the ith particle in the tth iteration, respectively. ω denotes the inertia weight, c_1 and c_2 represent acceleration coefficients, $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are two random vectors in $[0,1]^D$. $\boldsymbol{x}_{pbest}^t$ is the best solution of $\boldsymbol{x}_i$ found so far representing the cognitive coefficient and $\boldsymbol{x}_{gbest}^t$ is the current best solution of the particle swarm indicating the social coefficient [33]. Based on the conventional framework of PSO, Cheng et al. [18] integrated the competitive mechanism into PSO and developed CSO. Generally, CSO randomly selects two particle individuals $\boldsymbol{x}_{r1}^t$ and $\boldsymbol{x}_{r2}^t$ for competition. The winner which has better fitness is denoted as $\boldsymbol{x}_w^t$, while the loser is denoted as $\boldsymbol{x}_l^t$. Simply, $\boldsymbol{x}_w^t$ directly survives to the next iteration, and $\boldsymbol{x}_l^t$ is updated using Eq. (2).

$$\begin{aligned} \boldsymbol{v}_l^{t+1} &= \boldsymbol{r}_1 \cdot \boldsymbol{v}_l^t + \boldsymbol{r}_2 \cdot (\boldsymbol{x}_w^t - \boldsymbol{x}_l^t) + \varphi \cdot \boldsymbol{r}_3 \cdot (\boldsymbol{x}_{mean}^t - \boldsymbol{x}_l^t) \\ \boldsymbol{x}_i^{t+1} &= \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1} \end{aligned} \tag{2}$$

where φ is a hyper-parameter to control the scaling of $\boldsymbol{r}_3 \cdot (\boldsymbol{x}_{mean}^t - \boldsymbol{x}_l^t)$, and $\boldsymbol{x}_{mean}^t$ is the centroid of the particle swarm. A demonstration is presented in Fig. 2. Here, blue circles denote the particle individuals and the purple star represents $\boldsymbol{x}_{mean}^t$. In Fig. 2(b), two particle individuals in orange are denoted as the randomly selected particle individuals $\boldsymbol{x}_{r1}^t$ and $\boldsymbol{x}_{r2}^t$. Subsequently, the fitness comparison between $\boldsymbol{x}_{r1}^t$ and $\boldsymbol{x}_{r2}^t$

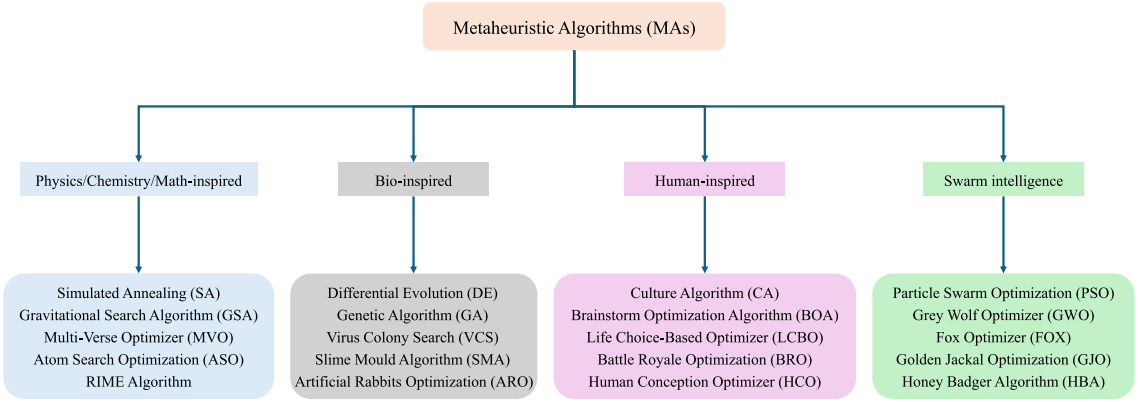

Fig. 1. Classification of representative MAs.

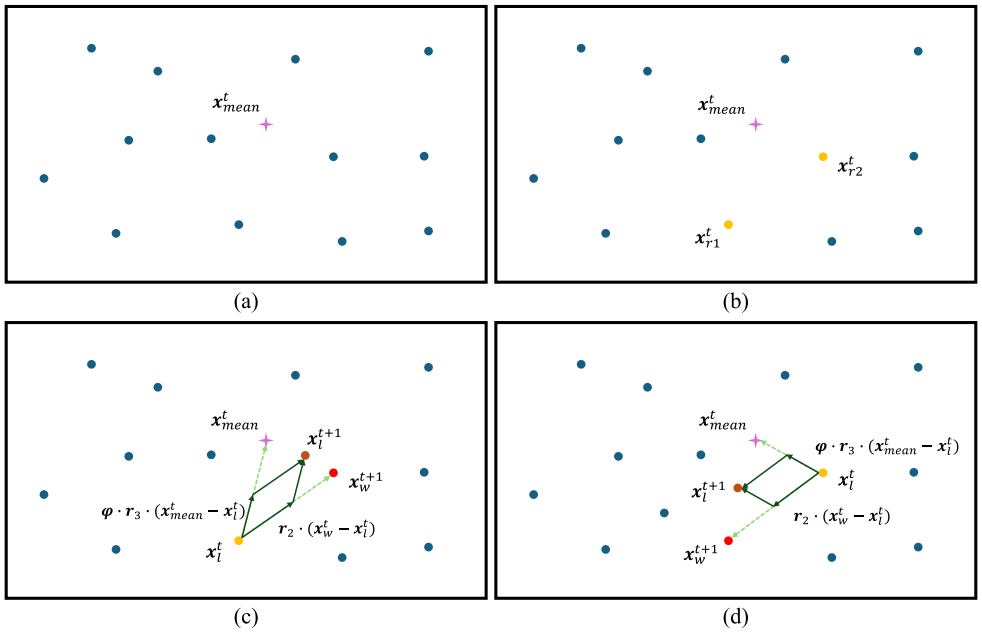

Fig. 2. A demonstration of competitive mechanism in CSO when $v^t_l = \{0\}^D$. (a). The distribution of particle swarm. (b). Two randomly selected individuals x^t_{r1} and x^t_{r2}. (c). The update for x^t_w and x^t_l when x^t_{r2} is the winner. (d). The update for x^t_w and x^t_l when x^t_{r1} is the winner.

are conducted. Fig. 2(c) demonstrates that x^t_{r1} is worse than x^t_{r2} and denoted as x^t_l, while x^t_{r2} is denoted as x^t_w. Simply, x^t_w survives to the next iteration and denoted as x^{t+1}_w, while x^t_l is updated based on two differential vector: $r_2 \cdot (x^t_w - x^t_l)$ and $\varphi \cdot r_3 \cdot (x^t_{mean} - x^t_l)$ when v^t_l is assumed as a zero vector. Fig. 2(d) demonstrates a similar instance when x^t_{r1} is better than x^t_{r2}. In each iteration, only half of the particle swarm is updated. This unique competitive mechanism as well as the balanced exploration–exploitation search operator enables CSO to achieve more effective performance in optimization. In summary, the pseudocode of CSO for minimization problems is presented in Algorithm 1.

2.2. Literature review of CSO

As an efficient approach to numerical optimization, CSO has attracted significant attention from researchers and scholars: Mohapatra et al. [34] introduced a tri-competitive criterion into CSO and presented a novel variant called Modified CSO (MCSO). Specifically, three particle individuals are involved in competition at a time. The particle individual with the best fitness is declared the winner, while the remaining particle individuals are identified as the losers. Among the losers, the particle individual being closer to the winner is determined as the loser1, and the other is the loser2. In the update phase, the winner progresses directly to the next iteration, while loser1 and loser2 are updated using a similar search operator in the original CSO. In each iteration, two-thirds of the particle swarm is updated, which accelerates optimization convergence while preserving swarm diversity. Huang et al. [35] enhanced the optimization performance of CSO by incorporating a novel three-phase cooperative evolutionary strategy and proposed an advanced version termed Three-Phase CSO (TPCSO). In TPCSO, the particle swarm is divided into two equal-sized sub-swarms based on the concept of fitness competition. The winners are transferred to the next iteration while the losers are updated using the three-phase cooperative evolutionary mechanism. TPCSO specifies that in the early stage of optimization, the exploration behavior of particle individuals is encouraged. In the middle stage, both convergence and diversity are emphasized. In the late stage, the focus shifts primarily to optimization convergence. Depending on the requirement of convergence speed and swarm diversity in different optimization stages, the losers adaptively select the appropriate search operators for evolution. Pan et al. [36] integrated CSO with a multi-strategy surrogate-assisted technique and extended the proposed Surrogate-Assisted CSO (SACSO) to address expensive optimization problems. SACSO incorporates three distinct search principles — global search, local search, and opposition-based search — used for the particle individual update. A dynamic

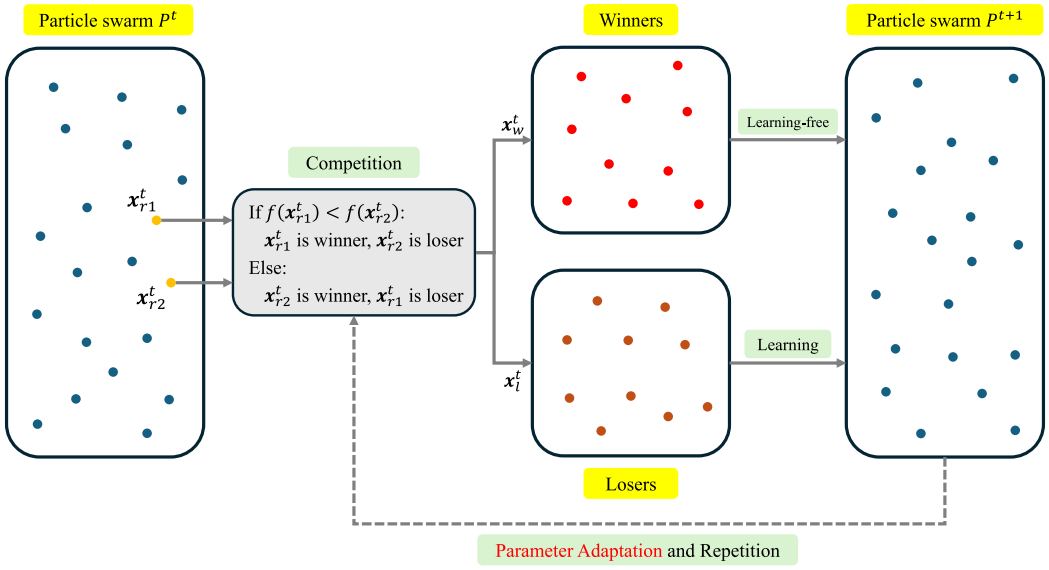

Fig. 3. A demonstration of L-SHACSO.

Algorithm 1: CSO

Input: Population size: N, Dimension: D, Maximum iteration: T
Output: Optimum: x_{best}^t

Function CSO(N, D, T_{max})**:**
  Initialize the particle swarm X
  $t = 0$
  $x_{best}^t \leftarrow$ **best**(X)
  while $t < T$ **do**
    $P \leftarrow$ **copy**(X)
    while P *is not empty* **do**
      Select two random particle individuals x_{r1}^t and x_{r2}^t
      if $f(x_{r1}^t)$ *is better than* $f(x_{r2}^t)$ **then**
        Assign x_{r1}^t as x_w^t and x_{r2}^t as x_l^t
      end
      else
        Assign x_{r1}^t as x_l^t and x_{r2}^t as x_w^t
      end
      Remove x_{r1}^t and x_{r2}^t from P
      Survive x_w^t to X^{t+1}
      Update x_l^t using Eq. (2) and survive to X^{t+1}
    end
    $x_{best}^t \leftarrow$ **best**(X)
    $t \leftarrow t + 1$
  end
  return x_{best}^t

adaptation scheme is introduced to balance exploration and exploitation intelligently. Furthermore, the ensemble of the global Generalized Surrogate Model (GSM) and the Elite Surrogate Model (ESM), combined with local and opposition-based search, strengthens the optimization performance of SACSO in expensive optimization scenarios. In the meantime, some researchers have noticed the potential of CSO and further developed CSO to tackle multi-objective optimization challenges. Huang et al. [37] presented an Adaptive Multi-Objective CSO (AMOCSO), where the domination relationship is considered in the modified competitive mechanism. When the domination relationship between two selected particle individuals exists, the non-dominated particle individual is assigned as the winner and the dominated particle individual is assigned as the loser. If the domination relationship does not exist, the particle individual with a smaller numerical value is identified as the winner and the particle individual with a larger numerical value is identified as the loser. An improved learning mechanism is developed in AMOCSO to update the losers, which is expected to enhance the optimization quality and balance convergence and diversity. Additionally, an external archive is maintained during the optimization to prevent population degeneration. The optimization performance in DTLZ, WFG, and CEC2009 benchmarks confirmed the competitiveness of AMOCSO with other state-of-the-art optimizers. Ge et al. [38] presented an Adaptive Competitive Multi-Objective Swarm Algorithm based on Inverse Modeling (AIMOCSO), where an adaptive parameter model is adopted to accelerate the convergence when the direction of optimization does trap into local optima. A random parameter model is also employed to prevent stagnation when the particle swarm falls into local optima. Additionally, inverse modeling is utilized to update the winner particles to enhance exploitation capabilities and mitigate slow evolution in the early stages. An adjacent individual competition method is also proposed to improve the distribution of solutions. More detailed reviews for CSO can refer to [39,40].

3. Our proposal: L-SHACSO

This section details our proposed L-SHACSO for optimization problems. We first present a demonstration in Fig. 3. The primary architecture of L-SHACSO inherits from the original CSO and the main modification is in the parameter adaptation scheme. In the following context, we will introduce two advanced approaches integrated into L-SHACSO: the success history adaptation and the linear population reduction strategies.

3.1. Success history adaptation

The success history adaptation strategy described in [41] emphasizes that a successful parameter adaptation in DE seeks to closely approximate problem-specific ideal hyperparameters. Building on this concept, we focus on the crucial hyperparameter φ in CSO and introduce the success history adaptation methodology. We hypothesize that specific φ values, derived from successful evolution — where the offspring particle individuals achieve better fitness than their parents — can guide the evolution of other particle individuals towards the global optimum. Based on these descriptions, the successful parameter

$\mu_{\varphi,1}$	$\mu_{\varphi,2}$	$\mu_{\varphi,3}$	$\mu_{\varphi,4}$	...	$\mu_{\varphi,k}$

Fig. 4. Historical memory of μ_{φ}.

adaptation in L-SHACSO is started by initializing the historical memory of μ_{φ}, as presented in Fig. 4. The initial value of $\mu_{\varphi,i}$ $(i = \{1, 2, \dots, k\})$ is fixed at 0.3, and the sensitivity experiments are conducted in Section 4. k is the length of the historical memory and set to 5 as suggested in [42,43]. In each iteration, L-SHACSO randomly selects a $\mu_{\varphi,i}$ from the historical memory, and the hyperparameter φ for the jth particle individual is sampled using Eq. (3).

$$\varphi_j \sim \textbf{truncate}(\textbf{randn}(\mu_{\varphi,i}, 0.1), \varphi_{lb}, \varphi_{ub}) \tag{3}$$

Here, φ_j is sampled from a normal distribution where the mean is $\mu^j_{\varphi,i}$ and the standard deviation is 0.1. Given the randomness in the normal distribution and the specific sampled number may be smaller than 0 or much larger, the truncation function is employed to ensure φ_j is within $[\varphi_{lb}, \varphi_{ub}]$, where φ_{lb}=0.001 and φ_{ub}=0.5 to ensure the global convergence and balance the exploration and exploitation as suggested in [18]. If the generated offspring particle individual has a better fitness than its parent, this evolution is successful, and φ_j is stored into S_{φ}. At the end of this iteration, Eq. (4) is activated to update $\mu_{\varphi,i}$.

$$\mu_{\varphi,i} = (1-c)\cdot\mu_{\varphi,i} + c\cdot \text{meanL}(S_{\varphi})$$
$$\text{meanL}(S_{\varphi}) = \frac{\sum_{\varphi\in S_{\varphi}} \varphi^2}{\sum_{\varphi\in S_{\varphi}} \varphi} \tag{4}$$

where c is the coefficient to balance the proportion between the historical information and the knowledge from the recent evolution, as suggested within [0.05, 0.2] in [44]. We fixed it at 0.1 in our numerical experiments. meanL$(\cdot)$ denotes the Lehmer mean function. The integration of the success history adaptation scheme aims to enhance the overall optimization capacity by systematically transferring the knowledge from successful adaptations to the broader particle swarm.

3.2. Linear population reduction

Population reduction techniques have been practically proven effective in balancing exploration and exploitation during different optimization phases and have been widely applied in MAs [45–47]. This technique is consistent with the concept that, in the early stage of optimization, a large population can ensure uniformly distributed solutions across the search space as much as possible. This preference enhances the effectiveness of explorative search due to the broader coverage of the population. In the middle stage, a moderate-sized population achieves a balance between exploration and exploitation and contributes to avoiding premature convergence and stagnation during the optimization process. In the late stage, a small-sized population focuses on the exploitative search. MAs can benefit from the convergence acceleration towards the optimal solution. By incorporating this simple yet effective population reduction technique, MAs can achieve improved optimization accuracy and faster convergence. Therefore, we integrate the linear population reduction proposed in [28] into L-SHACSO, and the population size of the particle swarm in the tth iteration can be determined using Eq. (5).

$$N^t = \textbf{round}\left(\frac{N_{min} - N_{max}}{FE_{max}}\cdot FE + N_{max}\right) \tag{5}$$

where **round**$(\cdot)$ converts the input value into an integer number, N_{max} and N_{min} denote the maximum and minimum population sizes. FE_{max} and FE denote the maximum and current fitness evaluations. In this study, we set N_{max} and N_{min} to 400 and 4, respectively.

3.3. Architecture of L-SHACSO

This section presents the main architecture of L-SHACSO. The flowchart of L-SHACSO is demonstrated in Fig. 5 and the pseudocode of L-SHACSO is presented in Algorithm 2.

Algorithm 2: L-SHACSO

Input: Max. and Min. population size: N_{max} and N_{min}, Dimension: D, Max. fitness evaluation: FE_{max}, History memory: $[\mu_{\varphi,1}, ..., \mu_{\varphi,k}]$
Output: Optimum: $\boldsymbol{x}^t_{best}$

Function L-SHACSO(N, D, T_{max}):
  Initialize the particle swarm X
  $FE = 0$, $t = 0$
  $\boldsymbol{x}^t_{best} \leftarrow$ **best**(X)
  while $FE < FE_{max}$ **do**
    $P \leftarrow$ **copy**(X)
    Initialize success history S_{φ}
    Sample a specific $\mu_{\varphi,i}$ from the history memory
    while *P is not empty* **do**
      Select two random particle individuals $\boldsymbol{x}^t_{r1}$ and $\boldsymbol{x}^t_{r2}$
      if $f(\boldsymbol{x}^t_{r1})$ *is better than* $f(\boldsymbol{x}^t_{r2})$ **then**
        Assign $\boldsymbol{x}^t_{r1}$ as $\boldsymbol{x}^t_{w}$ and $\boldsymbol{x}^t_{r2}$ as $\boldsymbol{x}^t_{l}$
      end
      else
        Assign $\boldsymbol{x}^t_{r1}$ as $\boldsymbol{x}^t_{l}$ and $\boldsymbol{x}^t_{r2}$ as $\boldsymbol{x}^t_{w}$
      end
      Survive $\boldsymbol{x}^t_{w}$ to X^{t+1}
      Sample φ_j using Eq. (3)
      Update $\boldsymbol{x}^t_{l}$ using Eq. (2) and survive to X^{t+1}
      $FE \leftarrow FE + 1$
      if $f(\boldsymbol{x}^{t+1}_{l})$ *is better than* $f(\boldsymbol{x}^t_{l})$ **then**
        Save φ_j to S_{φ}
      end
      Remove $\boldsymbol{x}^t_{r1}$ and $\boldsymbol{x}^t_{r2}$ from P
    end
    $t \leftarrow t + 1$
    $\boldsymbol{x}^t_{best} \leftarrow$ **best**(X)
    Resize the particle swarm X using Eq. (5)
    Update $\mu_{\varphi,i}$ in historical memory using Eq. (4)
  end
  return $\boldsymbol{x}^t_{best}$

3.4. Computational complexity of L-SHACSO

This section analyzes the computational complexity of L-SHACSO. Initially, we define the population size as N, dimension size as D, and maximum iteration as T. The crucial components in L-SHACSO are analyzed as follows:

- Particle swarm initialization: $O(N \times D)$.
- Update $\boldsymbol{x}^t_l$ using Eq. (2): $O(T \times N \times D)$.
- Resize particle swarm: $O(T)$.
- Update $\mu_{\varphi,i}$ in historical memory: The best is $O(1)$ when there is no success history. The worst is $O(T \times N)$ when all particle individuals evolve successfully.

In summary, the complete computational complexity of L-SHACSO is formulated in Eq. (6).

$$\begin{aligned} &O(N\times D) + O(T\times N\times D) + O(T) + \{O(1), O(T\times N)\} \\ :=&O(T\times N\times D) \end{aligned} \tag{6}$$

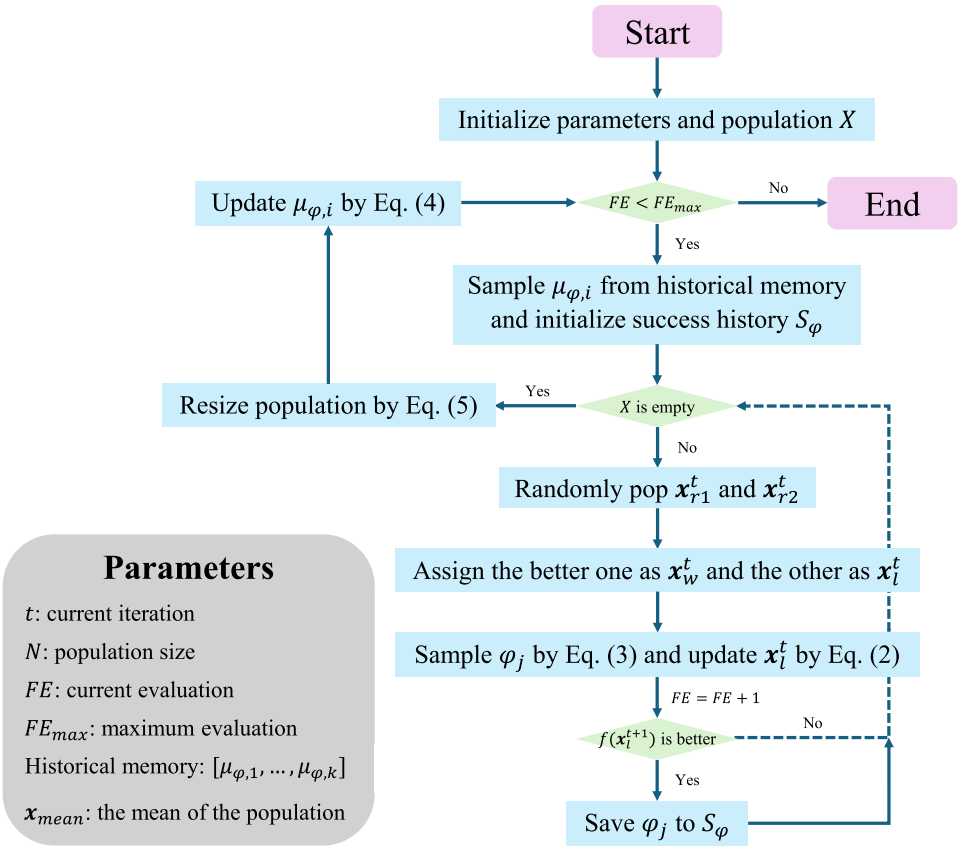

Fig. 5. The flowchart of L-SHACSO.

Table 1
Computational complexity of representative MAs.

MAs	Complexity
GA [48]	$O(T \times N \times D)$
DE [49]	$O(T \times N \times D)$
PSO [19]	$O(T \times N \times D)$
CSO [18]	$O(T \times N \times D)$
Gray Wolf Optimizer (GWO) [50]	$O(T \times N \times D)$

We also present the computational complexity of representative MAs in Table 1. The computational complexity of L-SHACSO is theoretically comparable to that of other well-known MAs. This analysis underscores the flexibility of L-SHACSO in tackling various optimization challenges.

4. Numerical experiments

This section presents the numerical experiments to investigate the performance of our proposed L-SHACSO comprehensively. Section 4.1 details the experimental settings including the experimental environments, benchmark functions, and competitor algorithms. Section 4.2 summarizes the experimental results across various benchmarks.

4.1. Experimental settings

4.1.1. Experimental environments

The numerical experiments in this research are implemented on the following system:

- Device: Lenovo Legion R9000p
- OS: Windows 11 Home 64 bit
- CPU: Core i7-14700HX Processor
- GPU: NVIDIA GeForce RTX 4060 Laptop GPU 8 GB GDDR6
- RAM: 16 GB DDR5-5600MHz (SODIMM) - (2 × 8 GB)
- Language: Python 3.11

The detailed information on experimental environments ensures the reproducibility of this research.

4.1.2. Benchmark functions

We conduct numerical experiments on the following benchmarks:

- IEEE-CEC2017: This benchmark contains 29 functions ranging from unimodal, multimodal, hybrid, and composite. Details of this benchmark are summarized in Appendix A Table A.16. It is important to note that f_2 is disabled in CEC2017, with f_{30} serving as a replacement.
- IEEE-CEC2020: This benchmark contains 10 functions ranging from unimodal, multimodal, hybrid, and composite. Details of this benchmark are summarized in Appendix A Table A.17.
- IEEE-CEC2022: This benchmark contains 12 functions ranging from unimodal, basic, hybrid, and composite. Details of this benchmark are summarized in Appendix A Table A.18.
- Engineering problems: We investigate the performance of L-SHACSO in Cantilever Beam Design Problem (CBDP), Corrugated Bulkhead Design Problem (CBHDP), Gear Train Design Problem (GTDP), Piston Lever Design Problem (PLDP), Speed Reducer Design Problem (SRDP), Three Bar Truss Design Problem (TBTDP), Tubular Column Design Problem (TCDP), and Welded Beam Design Problem (WBDP). These problems are detailed in Appendix B.

The CEC benchmarks are provided by Opfunu library [51], and engineering problems are available in Enoppy library [52].

Table 2
Parameter settings of algorithms.

MAs	Parameters	Value
CMA-ES	population size N	100
	σ	1.3
JADE	population size N	100
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
L-SHADE	N_{max} and N_{min}	18 · D and 4
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
CSO	population size N	200
	ϕ	0.15
MPEDE	N_{max} and N_{min}	250 and 20
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
	λ_1, λ_2, and λ_3	0.2, 0.2, and 0.2
jSO	N_{max} and N_{min}	12 ×D and 4
	p_{max} and p_{min}	0.25 and 0.125
	$\boldsymbol{M}_F$	0.3
L-SHADE-cnEpSin	N_{max} and N_{min}	18 ×D and 4
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
	memory size H	5
	ps and pc	0.5 and 0.4
INFO	population size N	100
SCSO	population size N	100
	sensitivity range	[2, 0]
RIME	population size N	100
	parameter w	5
PO	population size N	100
L-SHACSO	N_{max} and N_{min}	400 and 4
	μ_ϕ	0.3
	memory size k	5

4.1.3. Competitor algorithms and parameters

This section introduces the competitor algorithms and parameter settings. To comprehensively and fairly investigate the performance of L-SHACSO, we adopt two categories of MAs for comparison, which are listed as follows:

- State-of-the-art optimizers: Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [53], JADE [44], L-SHADE [28], CSO [18], Multi-Population Ensemble DE (MPEDE) [54], jSO [29], and L-SHADE-cnEpSin [30].
- Recently proposed high-performance MAs: Weighted Mean of Vectors (INFO) [55], Sand Cat Swarm Optimization (SCSO) [56], RIME algorithm[1] [31], and Parrot Optimizer (PO)[2] [32].

The advanced DE variants are obtained from the PyADE library,[3] and MAs are partially provided by the Mealpy library [57]. The detailed parameter settings of competitor algorithms are listed in Table 2.

The maximum fitness evaluations (FEs) are fixed at 1000 $\times D$ for CEC2017, CEC2020, and CEC2022 and 10,000 for the engineering problems. To alleviate the effect of randomness, each optimizer in a single function is repeated 30 times. Additionally, we integrate optimizers with a static penalty function [58] as defined in Eq. (7) to handle constrained engineering problems.

$$F(\boldsymbol{x}_i) = f(\boldsymbol{x}_i) + w \cdot \sum_{i=1}^{m}(\max(0, g_i(\boldsymbol{x}_i))) \tag{7}$$

$F(\cdot)$ is the fitness function, $f(\cdot)$ is the objective function, and $g_i(\cdot)$ is the constraint function. w is a penalty constant set to $10e7$ by default.

[1] https://aliasgharheidari.com/RIME.html
[2] https://aliasgharheidari.com/PO.html
[3] https://github.com/xKuZz/pyade

4.2. Experimental results and statistical analysis

This section presents a summary of the experimental results and statistical analysis in CEC2017, CEC2020, CEC2022, and eight engineering problems. To assess the statistical significance between L-SHACSO and other competitors, the Mann–Whitney U test is applied to measure the p-value for every pair of optimizers. The Holm multiple comparison test is then employed to adjust the p-values obtained from the Mann–Whitney U test. We use +, ≈, and − to indicate whether L-SHACSO is significantly better, without significant difference, and significantly worse than the specific competitor. The average ranks of the optimizers are provided, and the best fitness values are highlighted in bold. Furthermore, we quantitatively investigate the optimization performance of L-SHACSO by examining the proportion between exploration and exploitation during optimization, as defined in Eq. (8).

$$\begin{aligned} Div^t &= \frac{1}{D}\sum_{d=1}^{D}\frac{1}{N}\sum_{i=1}^{N}|\boldsymbol{x}^t_{mean,d} - \boldsymbol{x}^t_{i,d}| \\ Exploration &= \frac{Div^t}{Div_{max}} \\ Exploitation &= \frac{|Div^t - Div_{max}|}{Div_{max}} \end{aligned} \tag{8}$$

where Div^t measures the distribution of the particle swarm relative to the centroid.

4.2.1. Results on CEC2017

The summary of the average rank and statistical results in CEC2017 is presented in Table 3, and the detailed results can be found in Appendix C. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in CEC2017 (i.e., f_1: unimodal function; f_4: multimodal function; f_{15}: hybrid function; f_{25} and f_{28}: composite functions) are presented in Figs. 6 and 7.

Based on the experimental results and statistical analysis, our proposed L-SHACSO demonstrates remarkable competitiveness compared to other algorithms, both in terms of statistical summaries and average ranks. These findings confirm the effectiveness and efficiency of integrating the success history adaptation and linear population reduction strategies into CSO. As a flexible technique, we believe that the collaboration between these two techniques has a promising potential to extend any MAs for performance advancement. Additionally, the excellent robustness and scalability of L-SHACSO are evident in the boxplots, while the well-balanced exploration and exploitation are clearly depicted in the variation curves shown in Figs. 6 and 7. These results prove that L-SHACSO has the potential to address unknown optimization tasks.

Although the exceptional optimization performance of L-SHACSO is observable, some significant inferiority is non-negligible. Specifically, in the 30-D benchmark, L-SHACSO performs significantly worse than CMA-ES four times, JADE twice, L-SHADE once, MPEDE once, jSO once, and L-SHADE-cnEpSin twice. In the 50-D benchmark, L-SHACSO performs significantly worse than CMA-ES four times, JADE once, L-SHADE three times, MPEDE three times, jSO once, and L-SHADE-cnEpSin twice. We attempt to introduce the No Free Lunch (NFL) Theorem [59] to explain this phenomenon. NFL states that no single optimization algorithm is universally superior for all possible problems. In other words, an algorithm that performs well on one class of problems may perform poorly on another. This theorem underscores the importance of tailoring algorithms to specific problem characteristics.

4.2.2. Results on CEC2020

The summary of the average rank and statistical results in CEC2020 is presented in Table 4, and the detailed results can be found in Appendix D. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in CEC2020 (i.e., f_1: unimodal function; f_3: multimodal function; f_6:

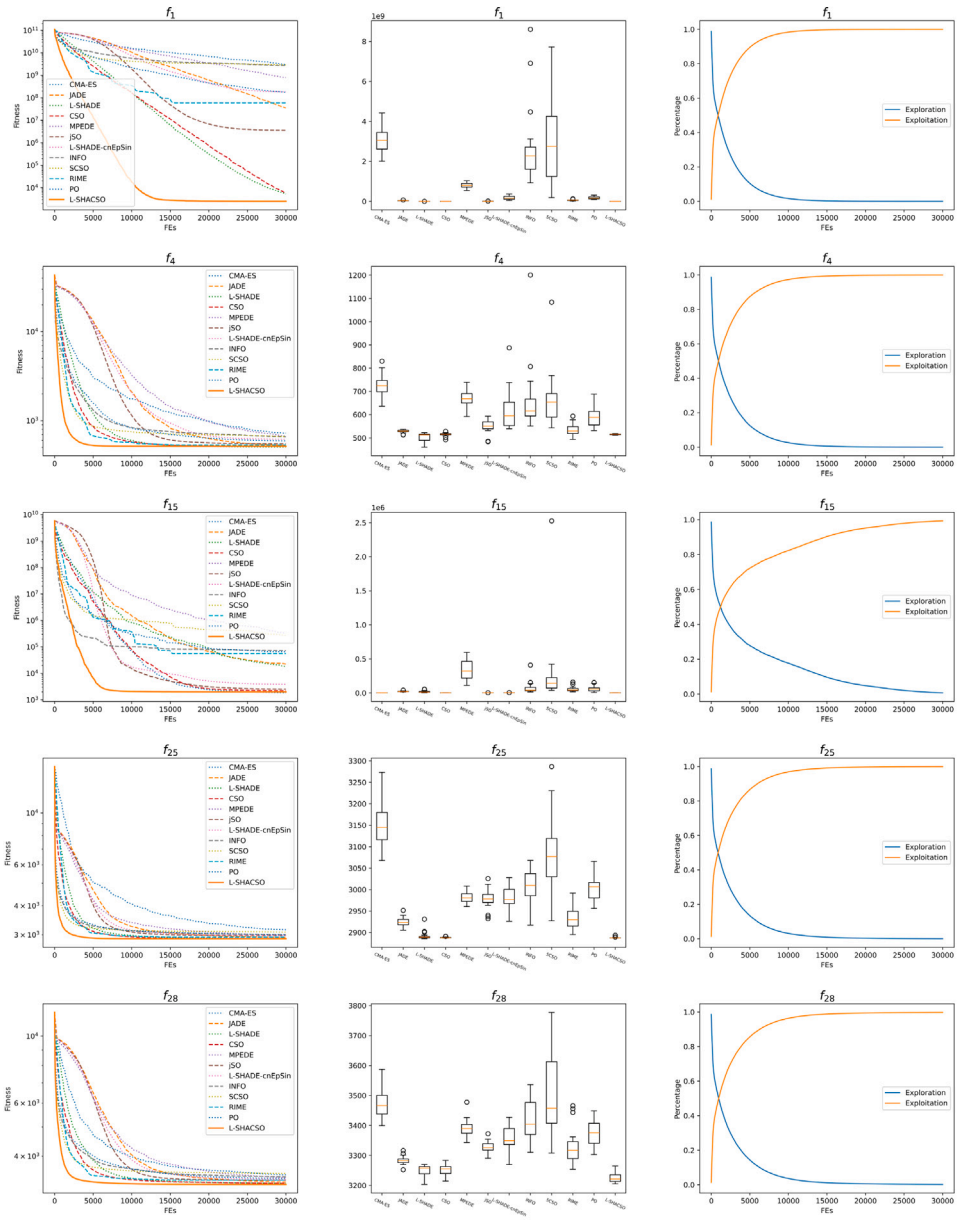

Fig. 6. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 30-D CEC2017.

Table 3
The average rank and statistical summary in CEC2017.

Dim.		CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
30	+ /≈/−:	24/1/4	25/2/2	26/2/1	22/7/0	27/1/1	26/2/1	25/2/2	29/0/0	29/0/0	28/1/0	28/1/0	–
	Avg. ranks:	7.7	6.6	4.3	3.2	7.8	6.7	7.6	8.8	8.6	5.4	9.1	**1.6**
50	+ /≈/−:	24/1/4	26/2/1	22/4/3	12/17/0	25/1/3	26/0/3	26/1/2	29/0/0	29/0/0	28/1/0	29/0/0	–
	Avg. ranks:	8.0	7.2	3.9	2.9	7.1	5.9	7.2	9.2	9.4	5.4	9.7	**1.8**

Table 4
The average rank and statistical summary in CEC2020.

Dim.		CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
30	+ /≈/−:	9/1/0	10/0/0	5/5/0	8/2/0	10/0/0	8/1/1	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	–
	Avg. ranks:	8.6	5.8	2.8	3.1	8.8	4.7	7.1	9.8	10.5	6.7	8.7	**1.4**
50	+ /≈/−:	10/0/0	10/0/0	4/5/1	2/8/0	10/0/0	8/2/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	–
	Avg. ranks:	9.2	6.6	2.6	2.5	6.6	4.6	7.7	10.2	10.8	6.3	9.2	**1.7**

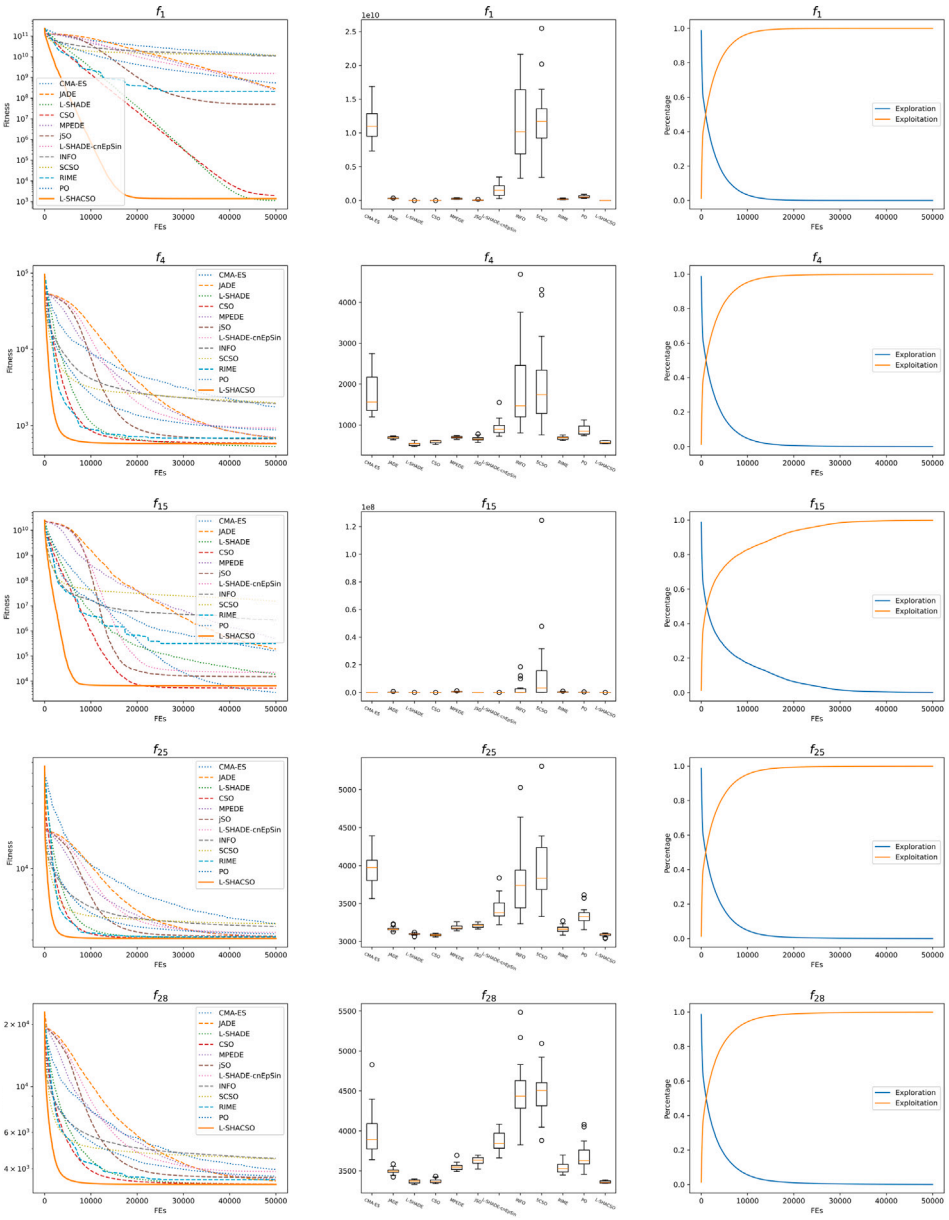

Fig. 7. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 50-D CEC2017.

hybrid function; f_9 and f_{10}: composite functions) are presented in Figs. 8 and 9.

The results obtained from the CEC2020 benchmark underscore the exceptional optimization performance, scalability, and robustness of the proposed L-SHACSO. L-SHACSO effectively combines success history adaptation with linear population reduction, which allows it to dynamically adjust its parameters as the optimization process progresses and enables L-SHACSO to maintain a balance between exploration and exploitation.

However, a notable limitation emerges when considering the scalability of L-SHACSO in higher-dimensional search space. As the problem dimension increases from 30 to 50, the average rank of L-SHACSO shows a declining trend, and the performance gap between CSO and L-SHACSO narrows. This pattern is observed in both CEC2017 and CEC2020 benchmarks, which suggests that the dominant performance of L-SHACSO becomes less pronounced in higher-dimensional search spaces. We infer that the integration of this combination within CSO may be less effective in handling the increased complexity and dimensionality of the problem landscape, potentially limiting the ability to efficiently navigate high-dimensional solution spaces.

Despite this, the overall results remain encouraging. Although the performance gap narrows, there is no significant performance degradation between CSO and L-SHACSO. In fact, L-SHACSO consistently outperforms CSO in terms of average rank, even in higher-dimensional scenarios. These observations highlight the robustness and effectiveness of the proposed L-SHACSO.

4.2.3. Results on CEC2022

The summary of the average rank and statistical results in CEC2022 is presented in Table 5, and the detailed results can be found in Appendix E. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in CEC2022 (i.e., f_1: unimodal function; f_3: multimodal function; f_6: hybrid function; f_9 and f_{10}: composite functions) are presented in Figs. 10 and 11.

The competitiveness of L-SHACSO is evidenced by its remarkable performance on the CEC2022 benchmark. A significant factor of this success can be attributed to the inherently high performance of the original CSO. Furthermore, the integration of two efficient strategies —

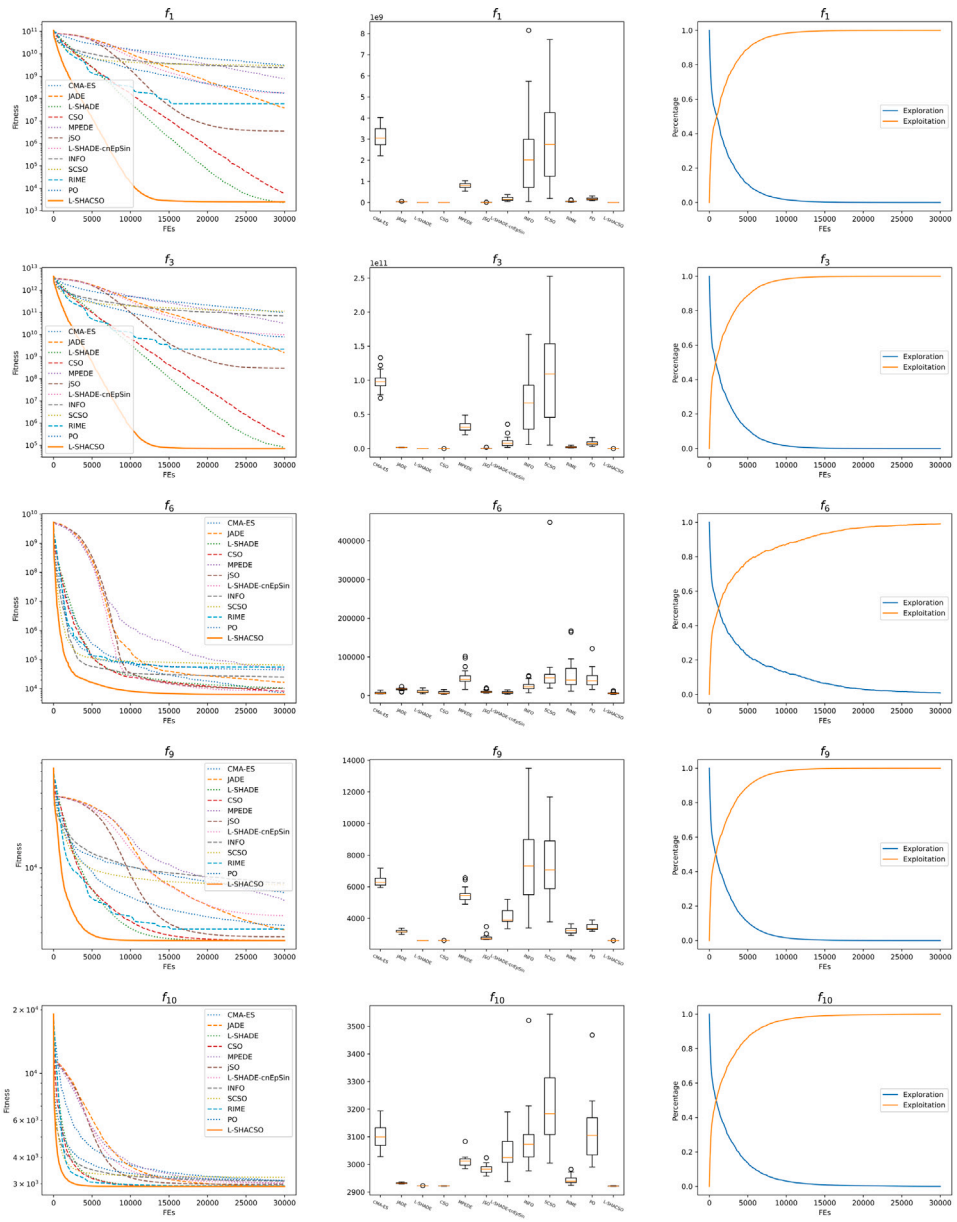

Fig. 8. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 30-D CEC2020.

Table 5
The average rank and statistical summary in CEC2022.

Dim.		CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
10	+ /≈/−:	10/0/2	10/0/2	10/2/0	11/1/0	12/0/0	10/2/0	10/1/1	12/0/0	11/1/0	10/2/0	12/0/0	–
	Avg. ranks:	7.0	4.2	3.9	5.0	10.3	8.0	7.3	8.3	8.4	5.5	8.2	**1.7**
20	+ /≈/−:	11/0/1	11/0/1	9/2/1	9/3/0	12/0/0	9/1/2	11/0/1	12/0/0	12/0/0	12/0/0	12/0/0	–
	Avg. ranks:	8.0	5.5	4.2	3.7	9.6	5.4	7.9	8.7	8.8	6.2	8.2	**1.6**

success history adaptation and linear population reduction — provides a comprehensive enhancement to L-SHACSO. The success history adaptation mechanism allows L-SHACSO to dynamically fine-tune its search parameters based on the success history of previous iterations, and the linear population reduction scheme gradually reduces the population size as the optimization process advances to balance the exploration and exploitation. The integration of these two strategies cooperatively contributes to the capacity of L-SHACSO and enables it to overcome local optima traps and efficiently explore the global solution space.

4.2.4. Results on engineering problems

CBDP: The experimental results and statistical analysis in CBDP are summarized in Table 6, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 12.

In CBDP, the top three optimizers are L-SHACSO, SCSO, and CSO, with our proposed L-SHACSO significantly outperforming both of them and other competitor algorithms. These results underscore the strong competitiveness of L-SHACSO in CBDP. Interestingly, SCSO, which performed poorly in the CEC benchmarks, proves to be highly competitive in CBDP. This observation aligns with the NFL Theorem, which suggests

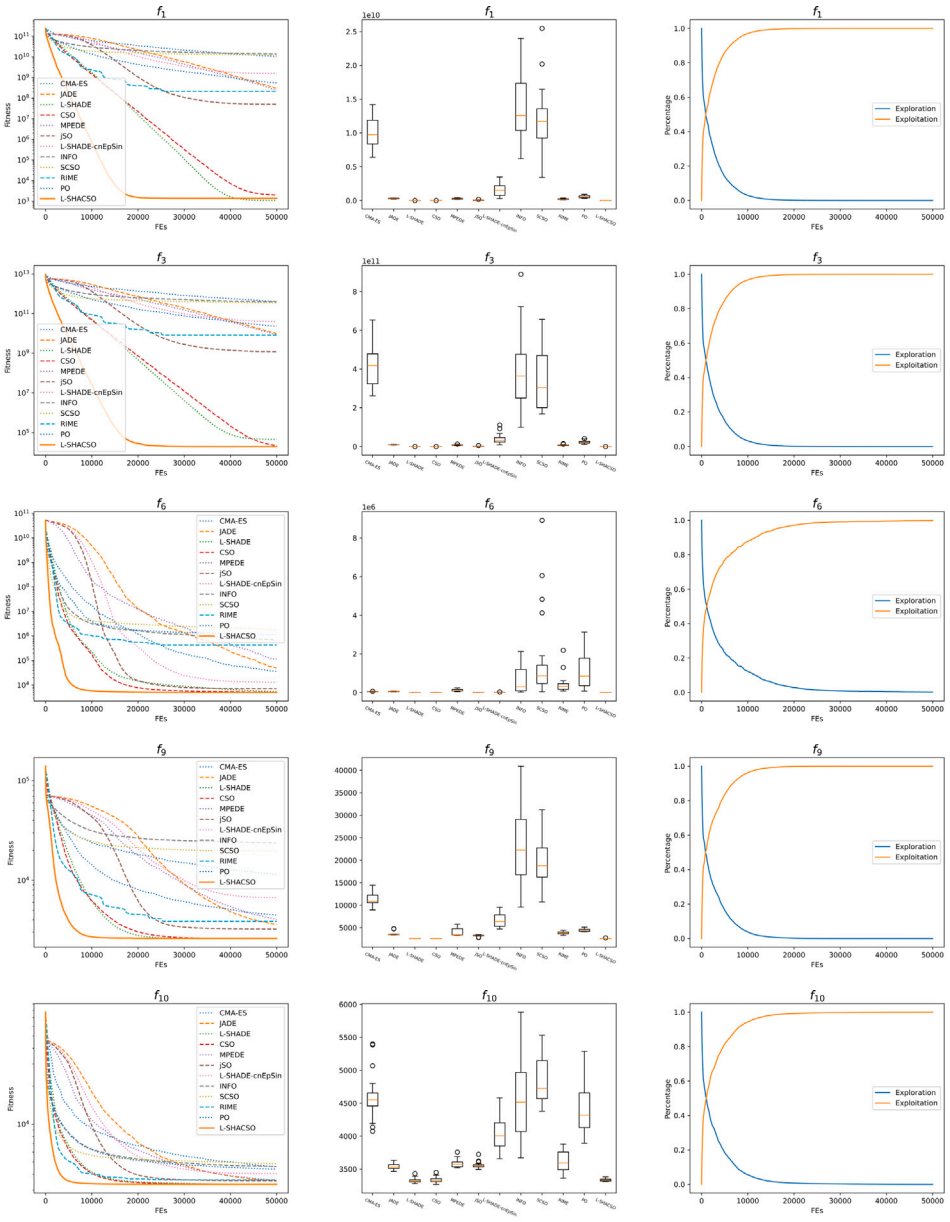

Fig. 9. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 50-D CEC2020.

Table 6
Experimental results and statistical analysis in CBDP.

MAs	Mean	std	Best	Worst
CMA-ES	1.535982e+00 +	8.822704e−02	1.430083e+00	1.749333e+00
JADE	1.354398e+00 +	5.448735e−03	1.347093e+00	1.373137e+00
L-SHADE	1.343406e+00 +	1.534474e−03	1.340507e+00	1.346403e+00
CSO	1.340319e+00 +	1.012553e−04	1.340076e+00	1.340444e+00
MPEDE	1.346106e+00 +	2.728504e−03	1.341716e+00	1.351344e+00
jSO	2.047484e+00 +	2.321759e−01	1.636356e+00	2.525738e+00
L-SHADE-cnEpSin	3.763554e+00 +	5.085213e−01	3.015637e+00	4.615789e+00
INFO	1.344483e+00 +	3.400181e−03	1.340523e+00	1.350431e+00
SCSO	1.340280e+00 +	1.874746e−04	1.339993e+00	1.340642e+00
RIME	1.417897e+00 +	5.814267e−02	1.354553e+00	1.566944e+00
PO	1.461427e+00 +	8.327753e−02	1.346014e+00	1.682413e+00
L-SHACSO	**1.340067e+00**	6.665529e−05	1.339967e+00	1.340222e+00

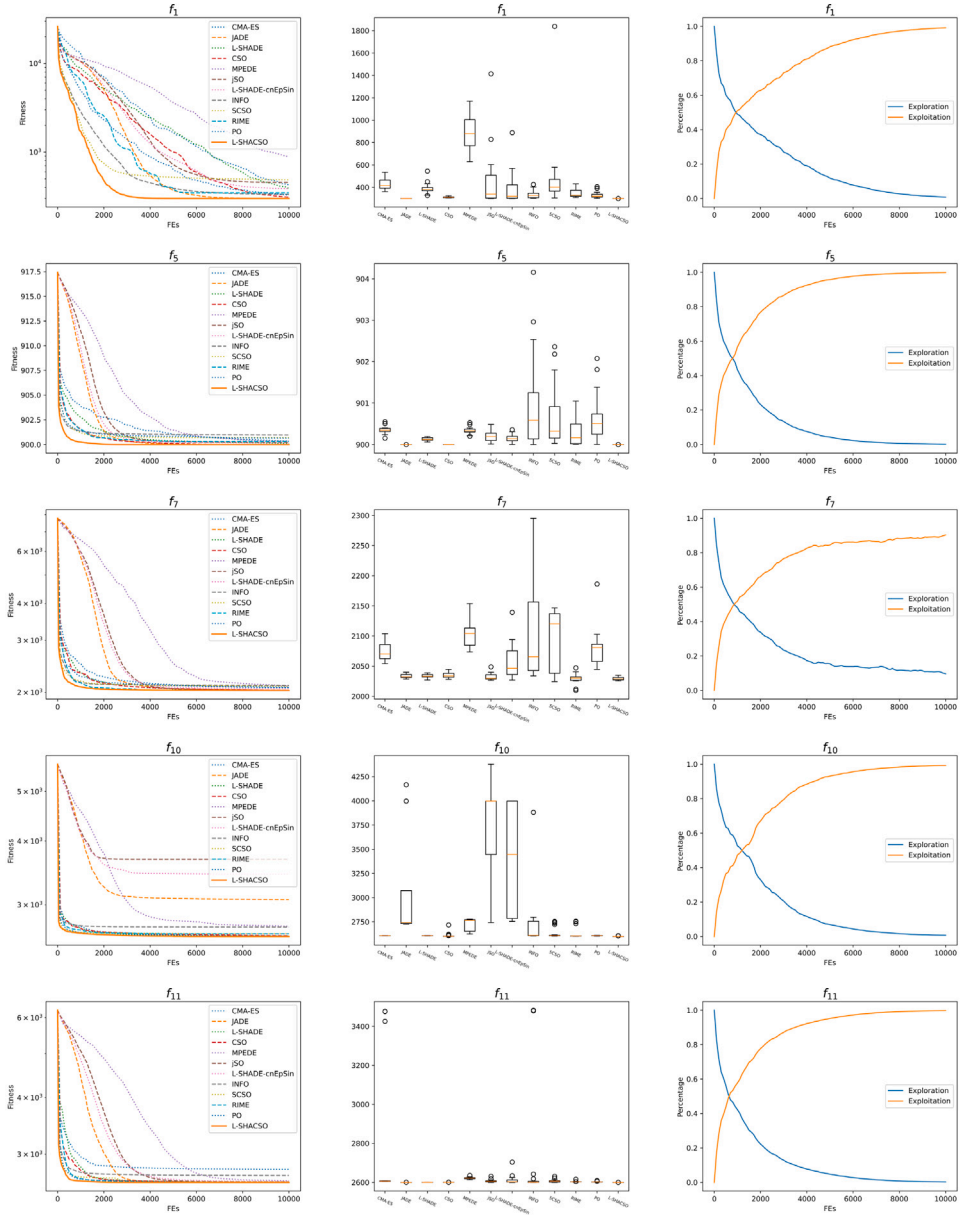

Fig. 10. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 10-D CEC2022.

that the performance of an optimization algorithm can vary significantly across different problem domains. This phenomenon reinforces the idea that no single method is universally superior.

CBHDP: The experimental results and statistical analysis in CBHDP are summarized in Table 7, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 13.

In CBHDP, the top three optimizers are L-SHACSO, CSO, and L-SHADE, with our proposed L-SHACSO significantly outperforming both of them and other competitor algorithms. Furthermore, the best optimum found by L-SHACSO across 30 trial runs surpasses that of the other algorithms. These findings further underscore the superiority of L-SHACSO in CBHDP.

GTDP: The experimental results and statistical analysis in GTDP are summarized in Table 8, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 14.

In GTDP, the top three optimizers are L-SHACSO, INFO, and SCSO. In this instance, the performance of our proposed L-SHACSO is comparable to that of INFO and significantly better than the other competing algorithms, which demonstrates the effectiveness of integrating success history adaptation and linear population reduction schemes into CSO. Similarly, INFO and SCSO outperform CSO, L-SHADE, L-SHADE-cnEpSin, and other MAs that excel in CEC benchmarks. These observations further support the practical implications of the NFL Theorem.

PLDP: The experimental results and statistical analysis in PLDP are summarized in Table 9, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 15.

In PLDP, the top three optimizers are jSO, L-SHADE, and CSO. In this instance, our proposed L-SHACSO ranks as the fourth-best optimizer, performing significantly worse than jSO. However, the performance of L-SHACSO is comparable to CMA-ES, JADE, L-SHADE, and CSO, and significantly better than MPEDE, L-SHADE-cnEpSin, INFO, SCSO, RIME, and PO. These results confirm that L-SHACSO is a competitive and acceptable optimizer in PLDP.

SRDP: The experimental results and statistical analysis in SRDP are summarized in Table 10, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 16.

In SRDP, the top three optimizers are L-SHACSO, MPEDE, and L-SHADE. In this instance, the performance of our proposed L-SHACSO

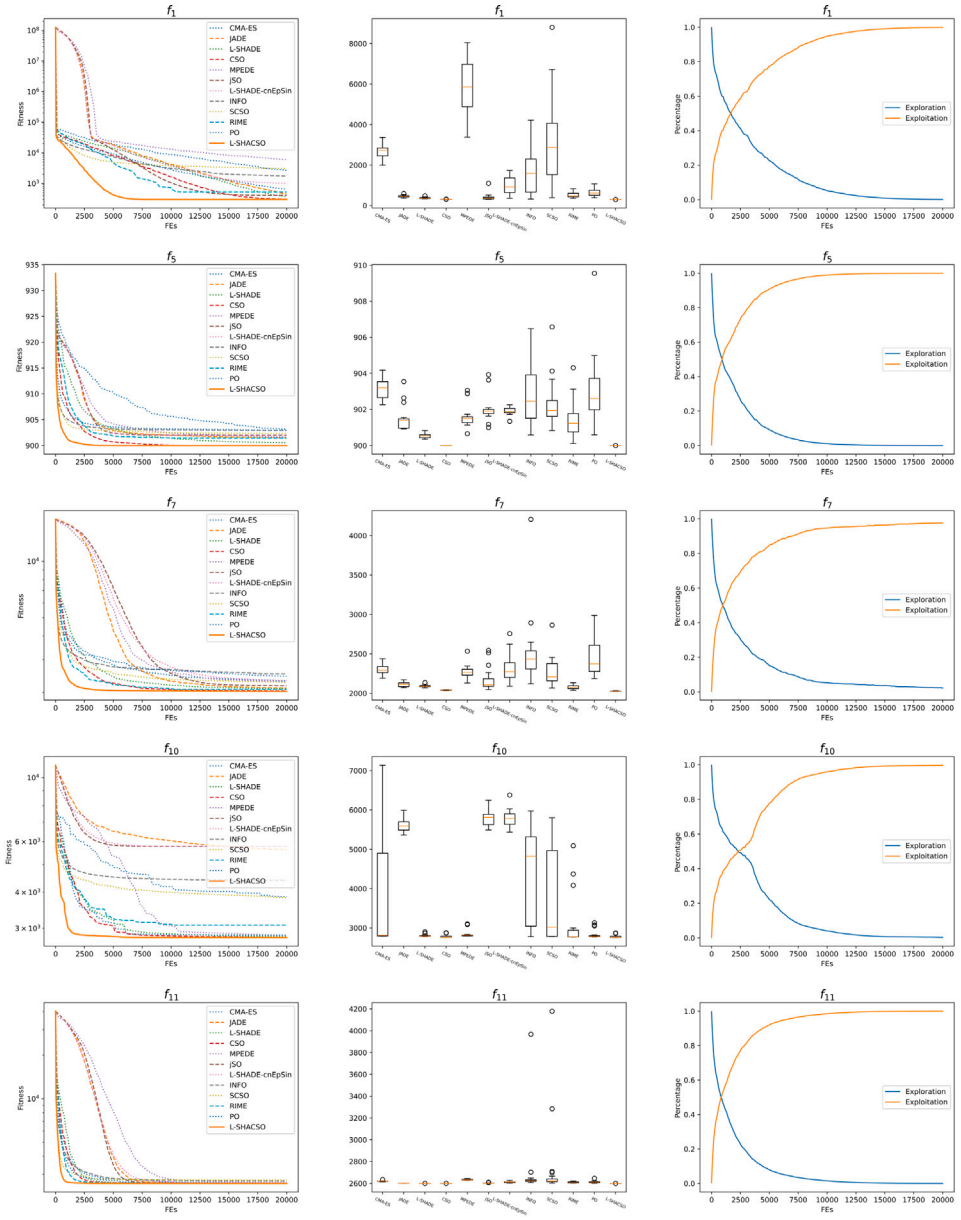

Fig. 11. The convergence curves, boxplots, and proportion between exploration and exploitation of representative functions in 20-D CEC2022.

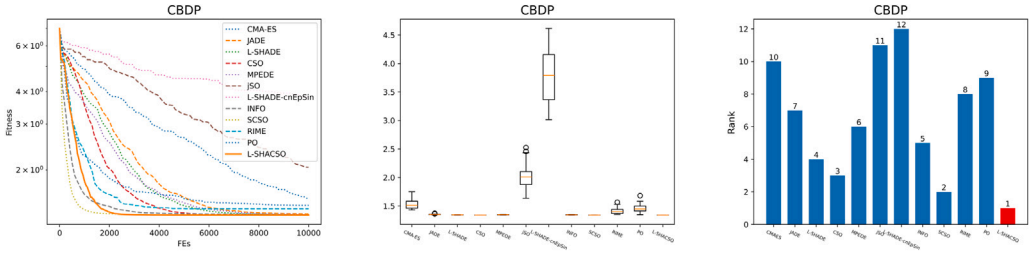

Fig. 12. Convergence curves, boxplots, and ranks of optimizers in CBDP.

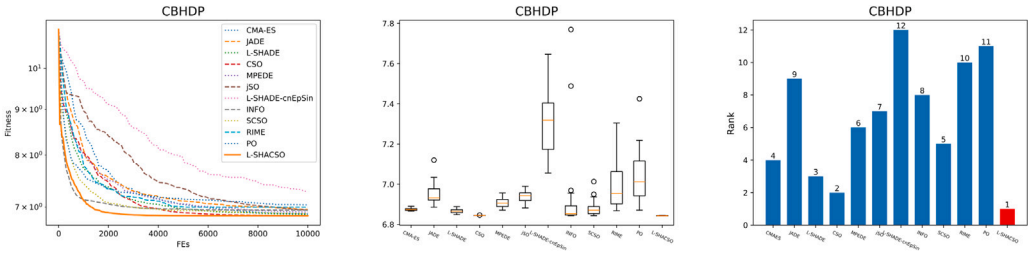

Fig. 13. Convergence curves, boxplots, and ranks of optimizers in CBHDP.

Table 7
Experimental results and statistical analysis in CBHDP.

MAs	Mean	std	Best	Worst
CMA-ES	6.875724e+00 +	7.170983e−03	6.865683e+00	6.890311e+00
JADE	6.957924e+00 +	5.658671e−02	6.885943e+00	7.120213e+00
L-SHADE	6.866804e+00 +	1.098877e−02	6.849223e+00	6.889056e+00
CSO	6.845188e+00 +	5.950060e−04	6.844287e+00	6.846736e+00
MPEDE	6.907986e+00 +	2.443504e−02	6.871067e+00	6.956464e+00
jSO	6.940912e+00 +	2.852034e−02	6.882437e+00	6.989862e+00
L-SHADE-cnEpSin	7.289160e+00 +	1.452637e−01	7.055867e+00	7.646100e+00
INFO	6.944796e+00 +	2.351929e−01	6.844141e+00	7.769633e+00
SCSO	6.883124e+00 +	4.161706e−02	6.844012e+00	7.013499e+00
RIME	6.998168e+00 +	1.296440e−01	6.868836e+00	7.303676e+00
PO	7.039513e+00 +	1.365860e−01	6.870918e+00	7.424646e+00
L-SHACSO	**6.843600e+00**	6.685836e−04	6.842965e+00	6.845047e+00

Table 8
Experimental results and statistical analysis in GTDP.

MAs	Mean	std	Best	Worst
CMA-ES	4.009765e−11 +	5.196439e−11	7.378192e−14	2.008061e−10
JADE	7.687673e−11 +	1.252313e−10	2.343566e−13	4.507041e−10
L-SHADE	1.800312e−11 +	2.507299e−11	2.420462e−15	8.271061e−11
CSO	6.946254e−12 +	1.593815e−11	3.257459e−16	6.605303e−11
MPEDE	2.369712e−10 +	5.783699e−10	5.807461e−13	2.413778e−09
jSO	1.506650e−10 +	3.449084e−10	1.872801e−14	1.567181e−09
L-SHADE-cnEpSin	5.188376e−10 +	8.421392e−10	1.402512e−11	3.347669e−09
INFO	2.870679e−14 ≈	3.349224e−14	5.046282e−16	1.414286e−13
SCSO	6.862586e−13 +	1.233325e−12	1.490850e−15	4.258717e−12
RIME	7.003150e−10 +	1.178444e−09	2.403351e−13	4.074187e−09
PO	2.484175e−10 +	8.475851e−10	1.430783e−15	3.901934e−09
L-SHACSO	**2.700234e−14**	3.055144e−14	6.929571e−20	8.398523e−14

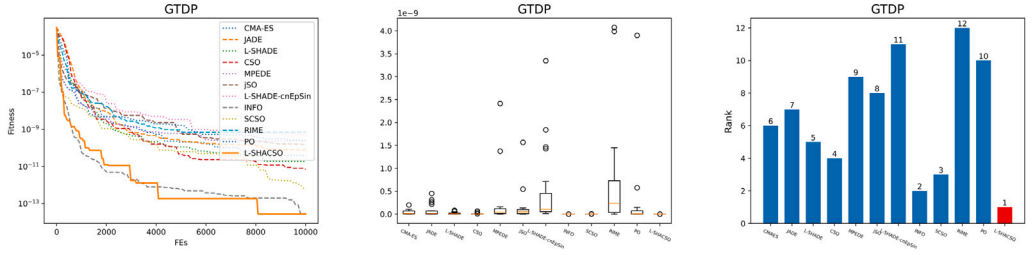

Fig. 14. Convergence curves, boxplots, and ranks of optimizers in GTDP.

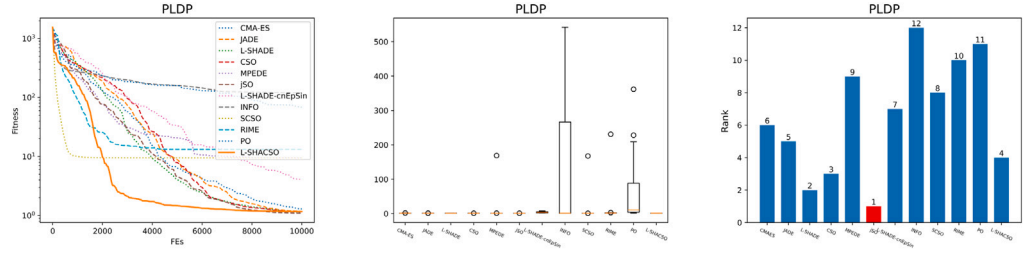

Fig. 15. Convergence curves, boxplots, and ranks of optimizers in PLDP.

Table 9
Experimental results and statistical analysis in PLDP.

MAs	Mean	std	Best	Worst
CMA-ES	1.294350e+00 ≈	2.052036e−01	1.107425e+00	1.889472e+00
JADE	1.170518e+00 ≈	8.107073e−02	1.100639e+00	1.393350e+00
L-SHADE	1.134440e+00 ≈	4.837909e−02	1.075458e+00	1.221853e+00
CSO	1.147615e+00 ≈	6.015120e−02	1.076325e+00	1.330939e+00
MPEDE	9.477042e+00 +	3.655977e+01	1.058682e+00	1.688368e+02
jSO	**1.092877e+00** −	2.310255e−02	1.067815e+00	1.175191e+00
L-SHADE-cnEpSin	4.106175e+00 +	1.795011e+00	1.857727e+00	7.726751e+00
INFO	1.204096e+02 +	1.587483e+02	1.057504e+00	5.419237e+02
SCSO	9.384546e+00 +	3.628401e+01	1.057590e+00	1.675429e+02
RIME	1.314956e+01 +	4.998653e+01	1.211308e+00	2.310266e+02
PO	6.868348e+01 +	1.020787e+02	1.410823e+00	3.619390e+02
L-SHACSO	1.166797e+00	4.186567e−02	1.074387e+00	1.216342e+00

Table 10
Experimental results and statistical analysis in SRDP.

MAs	Mean	std	Best	Worst
CMA-ES	2.990130e+03 +	9.435769e−01	2.988100e+03	2.991947e+03
JADE	2.988371e+03 +	4.403611e−01	2.987632e+03	2.988968e+03
L-SHADE	2.987949e+03 +	3.488800e−01	2.987495e+03	2.988955e+03
CSO	2.988593e+03 +	3.651721e−01	2.987999e+03	2.989297e+03
MPEDE	2.987099e+03 ≈	8.277077e−02	2.986966e+03	2.987247e+03
jSO	2.991224e+03 +	1.135956e+00	2.988902e+03	2.994047e+03
L-SHADE-cnEpSin	3.003367e+03 +	4.712294e+00	2.996049e+03	3.015320e+03
INFO	2.999197e+03 +	1.927673e+01	2.988394e+03	3.081033e+03
SCSO	3.002677e+03 +	4.572125e+00	2.993261e+03	3.010789e+03
RIME	3.016512e+03 +	1.560335e+01	2.993585e+03	3.063717e+03
PO	3.095396e+03 +	4.290130e+01	3.035086e+03	3.215812e+03
L-SHACSO	2.987072e+03	1.372724e−01	2.986894e+03	2.987450e+03

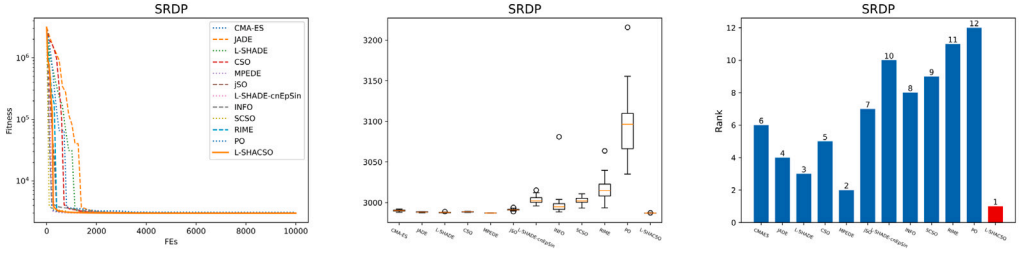

Fig. 16. Convergence curves, boxplots, and ranks of optimizers in SRDP.

Table 11
Experimental results and statistical analysis in TBTDP.

MAs	Mean	std	Best	Worst
CMA-ES	**2.638958e+02** −	2.618599e−06	2.638958e+02	2.638959e+02
JADE	2.638984e+02 +	2.283112e−03	2.638962e+02	2.639053e+02
L-SHADE	2.638961e+02 +	2.180641e−04	2.638959e+02	2.638967e+02
CSO	2.638977e+02 +	2.414009e−03	2.638963e+02	2.639060e+02
MPEDE	2.638984e+02 +	1.499701e−03	2.638964e+02	2.639024e+02
jSO	2.638959e+02 ≈	1.505237e−05	2.638958e+02	2.638959e+02
L-SHADE-cnEpSin	2.638970e+02 +	5.197752e−04	2.638959e+02	2.638985e+02
INFO	2.643019e+02 +	5.781759e−01	2.638966e+02	2.658899e+02
SCSO	2.639025e+02 +	7.016504e−03	2.638960e+02	2.639234e+02
RIME	2.643520e+02 +	4.807496e−01	2.639189e+02	2.657381e+02
PO	2.639592e+02 +	8.193222e−02	2.638968e+02	2.642509e+02
L-SHACSO	2.638959e+02	4.655094e−05	2.638959e+02	2.638960e+02

is comparable to MPEDE and significantly better than the other competing algorithms. The notable convergence speed observed in the convergence curve and the robustness shown in the boxplot further confirm the practical efficiency of L-SHACSO in SRDP.

TBTDP: The experimental results and statistical analysis in TBTDP are summarized in Table 11, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 17.

In TBTDP, the top three optimizers are CMA-ES, jSO, and L-SHACSO. In this instance, CMA-ES significantly outperforms our proposed L-SHACSO, while the performance of jSO is comparable to L-SHACSO. However, L-SHACSO is significantly better than the remaining competing algorithms, which demonstrates its competitiveness in TBTDP.

TCDP: The experimental results and statistical analysis in TCDP are summarized in Table 12, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 18.

In TCDP, the top three optimizers are CSO, CMA-ES, and L-SHACSO. While CSO and CMA-ES perform numerically better than L-SHACSO, statistical analysis confirms that the differences are not significant. Additionally, L-SHACSO shows no significant difference compared to the fourth-best optimizer, jSO, but is significantly better than the rest of the

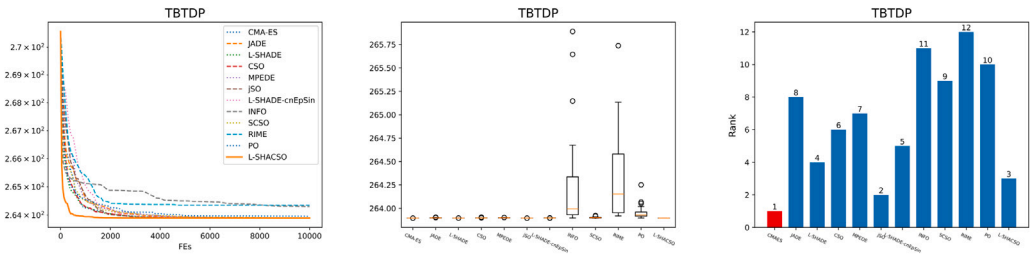

Fig. 17. Convergence curves, boxplots, and ranks of optimizers in TBTDP.

Table 12
Experimental results and statistical analysis in TCDP.

MAs	Mean	std	Best	Worst
CMA-ES	3.014980e+01 ≈	4.252909e−05	3.014974e+01	3.014992e+01
JADE	3.015848e+01 +	4.300876e−03	3.015281e+01	3.016584e+01
L-SHADE	3.015073e+01 +	5.361841e−04	3.014987e+01	3.015258e+01
CSO	**3.014979e+01** ≈	3.099130e−05	3.014974e+01	3.014984e+01
MPEDE	3.015997e+01 +	4.216113e−03	3.015316e+01	3.016959e+01
jSO	3.014986e+01 ≈	9.185516e−05	3.014977e+01	3.015013e+01
L-SHADE-cnEpSin	3.015463e+01 +	3.148996e−03	3.015056e+01	3.016239e+01
INFO	3.015144e+01 +	2.419055e−03	3.014997e+01	3.016114e+01
SCSO	3.015172e+01 +	1.489919e−03	3.015004e+01	3.015534e+01
RIME	3.054853e+01 +	2.384785e−01	3.020629e+01	3.093800e+01
PO	3.016984e+01 +	2.823850e−02	3.015234e+01	3.027707e+01
L-SHACSO	3.014985e+01	9.994708e−05	3.014974e+01	3.015012e+01

Table 13
Experimental results and statistical analysis in WBDP.

MAs	Mean	std	Best	Worst
CMA-ES	1.713375e+00 +	6.941640e−03	1.698954e+00	1.729968e+00
JADE	2.033112e+00 +	1.421205e−01	1.786404e+00	2.325266e+00
L-SHADE	1.863820e+00 +	1.114383e−01	1.706492e+00	2.056812e+00
CSO	1.743892e+00 +	6.216808e−02	1.688057e+00	1.892704e+00
MPEDE	1.758716e+00 +	2.850954e−02	1.714099e+00	1.826376e+00
jSO	1.755918e+00 +	3.272496e−02	1.708903e+00	1.820989e+00
L-SHADE-cnEpSin	1.988417e+00 +	1.066476e−01	1.841539e+00	2.224430e+00
INFO	1.993645e+00 +	4.203366e−01	1.700798e+00	3.381378e+00
SCSO	1.697768e+00 ≈	2.044130e−02	1.684011e+00	1.775577e+00
RIME	2.084290e+00 +	1.798984e−01	1.726304e+00	2.391074e+00
PO	1.933671e+00 +	1.760364e−01	1.708329e+00	2.287301e+00
L-SHACSO	**1.695231e+00**	1.589750e−02	1.682949e+00	1.728256e+00

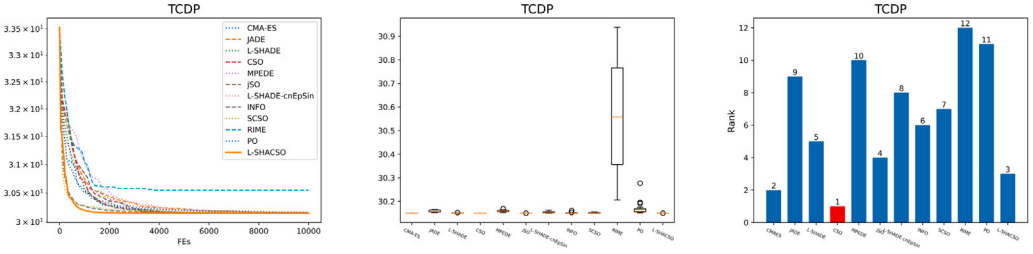

Fig. 18. Convergence curves, boxplots, and ranks of optimizers in TCDP.

optimizers. This confirms the scalability and robustness of L-SHACSO in TCDP.

WBDP: The experimental results and statistical analysis in WBDP are summarized in Table 13, and convergence curves, boxplots, and ranks of optimizers are presented in Fig. 19.

In WBDP, the top three optimizers are L-SHACSO, SCSO, and CMA-ES. In this instance, the performance of L-SHACSO is statistically comparable only to SCSO, while it is significantly better than the remaining competitor algorithms. Overall, L-SHACSO demonstrates competitive performance across all engineering problems, and we believe it can be effectively extended to other optimization challenges in real-world scenarios.

4.2.5. Results on sensitivity experiments

As we mentioned in Section 3.1, the initial value of μ_φ may significantly affect the performance of L-SHACSO in various benchmarks. Therefore, we fix the $\mu_\varphi = \{0.1, 0.15, 0.2, 0.25, 0.3\}$ and conduct the sensitivity experiments in the CEC2017, CEC2020, and CEC2022 benchmarks. Table 14 summarizes the average rank and statistical analysis, while the detailed experimental results are presented in Appendix F.

Comprehensive sensitivity experiments show that L-SHACSO is not highly sensitive to the initial setting of the hyper-parameter μ_φ, which demonstrates the excellent robustness of L-SHACSO in parameter settings. However, $\mu_\varphi = 0.3$ performs numerically better than other settings when evaluated by average rank. Therefore, we have selected $\mu_\varphi = 0.3$ as the base setting for our comparison experiments.

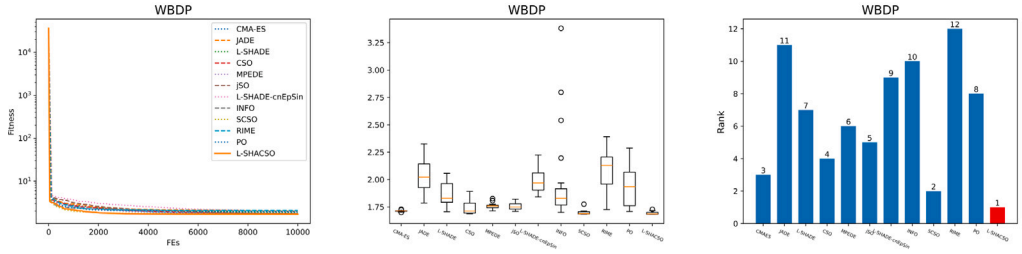

Fig. 19. Convergence curves, boxplots, and ranks of optimizers in WBDP.

Table 14
The average rank and statistical summary of sensitivity experiments.

Bench.	Dim.		μ_φ=0.1	μ_φ=0.15	μ_φ=0.2	μ_φ=0.25	μ_φ=0.3
CEC2017	30	+ /≈/−:	5/24/0	3/25/1	6/23/0	2/27/0	–
		Avg. ranks:	2.7	2.9	3.5	3.3	**2.5**
	50	+ /≈/−:	10/17/2	11/17/1	10/18/1	9/20/0	–
		Avg. ranks:	3.3	3.8	2.7	3.3	**1.9**
CEC2020	30	+ /≈/−:	1/8/1	1/8/1	1/8/1	2/8/0	–
		Avg. ranks:	3.2	3.0	3.0	3.0	**2.8**
	50	+ /≈/−:	6/4/0	3/6/1	4/5/1	3/7/0	–
		Avg. ranks:	3.4	2.7	3.1	3.4	**2.4**
CEC2022	10	+ /≈/−:	0/12/0	2/10/0	0/12/0	1/11/0	–
		Avg. ranks:	2.9	3.8	2.9	3.1	**2.3**
	20	+ /≈/−:	2/10/0	2/10/0	2/10/0	1/11/0	–
		Avg. ranks:	**2.4**	3.3	3.0	3.3	3.0

5. Application in eye disease detection

Early detection is crucial for managing progressive eye diseases that can lead to permanent vision loss if untreated. Deep learning models can identify early signs of disease even in asymptomatic stages, which provides a significant window for intervention. Therefore, we present a hybrid model named DenseNet-L-SHACSO-ELM for eye disease detection. Section 5.1 details the architecture of DenseNet-L-SHACSO-ELM, Section 5.2 describes the utilized eye disease dataset, and Section 5.3 presents the experimental results and analysis.

5.1. Architecture of DenseNet-L-SHACSO-ELM

This section presents the architecture of the proposed DenseNet-L-SHACSO-ELM model, designed to enhance accuracy and efficiency in eye disease detection. As presented in Fig. 20, the dataset is divided into three subsets: training, test, and validation datasets, allocated in proportions of 60%, 20%, and 20%, respectively. Initially, the training data is input into DenseNet-169, a densely connected convolutional neural network (CNN) famous for its ability to capture both low-level textures and high-level representations, which are essential for distinguishing subtle variations in retinal images.

After DenseNet-169 extracts these high-dimensional features, data are subsequently fed into an ELM for classification. ELM, known for its rapid training speed and strong generalization ability, serves as the classifier that identifies eye disease categories based on the features extracted by DenseNet-169. However, to achieve optimal performance, the weights and biases of ELM require fine-tuning. To address this, L-SHACSO is employed to optimize the weights and biases. L-SHACSO combines the benefits of CSO and efficient parameter adaptation schemes to achieve precise optimization of ELM parameters.

5.2. Eye disease dataset

The eye disease detection dataset utilized in this research downloaded from Kaggle[4] consists of retinal images categorized into four primary classes: Cataract with 1038 images, Diabetic Retinopathy with 1098 images, Glaucoma with 1007 images, and Normal with 1074 images, where an overview is presented in Fig. 21. Each category represents a different condition affecting eye health, with specific visual characteristics in retinal images that are used to identify and classify these diseases.

- **Cataract**: In retinal images, cataracts may manifest as opacities or hazy regions in the eye lens area.
- **Diabetic Retinopathy**: A common complication of diabetes, diabetic retinopathy results from damage to the blood vessels in the retina, leading to abnormalities such as microaneurysms, hemorrhages, and exudates.
- **Glaucoma**: This disease is associated with increased intraocular pressure, which damages the optic nerve, leading to progressive vision loss. In retinal images, glaucoma can often be identified by a change in the optic disc's size and shape, including increased cupping of the optic nerve.
- **Normal**: This category includes healthy retinal images with no signs of the above conditions. These images serve as a baseline, enabling the model to differentiate normal eye structures from those affected by disease.

Each image in the dataset has been labeled based on expert diagnosis, ensuring high-quality, accurate data for training, validation, and testing of the DenseNet-L-SHACSO-ELM model.

5.3. Experimental results and analysis

This section presents the experimental results and analysis of the proposed DenseNet-L-SHACSO-ELM in eye disease detection tasks. To quantitatively analyze the performance, four performance indicators

[4] https://www.kaggle.com/datasets/gunavenkatdoddi/eye-diseases-classification

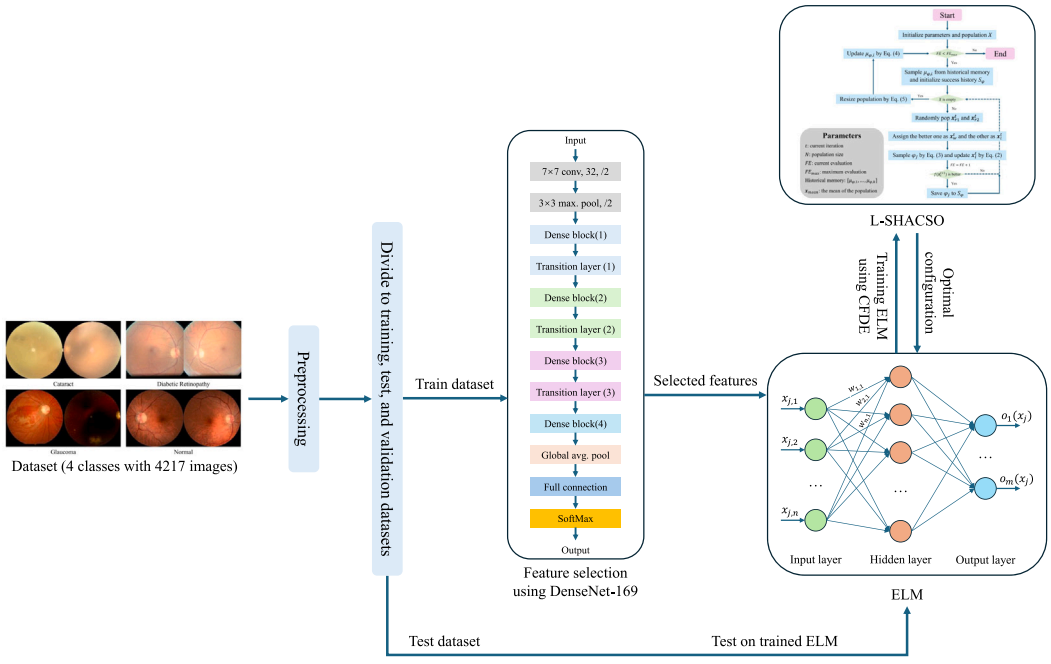

Fig. 20. The architecture of DenseNet-L-SHACSO-ELM.

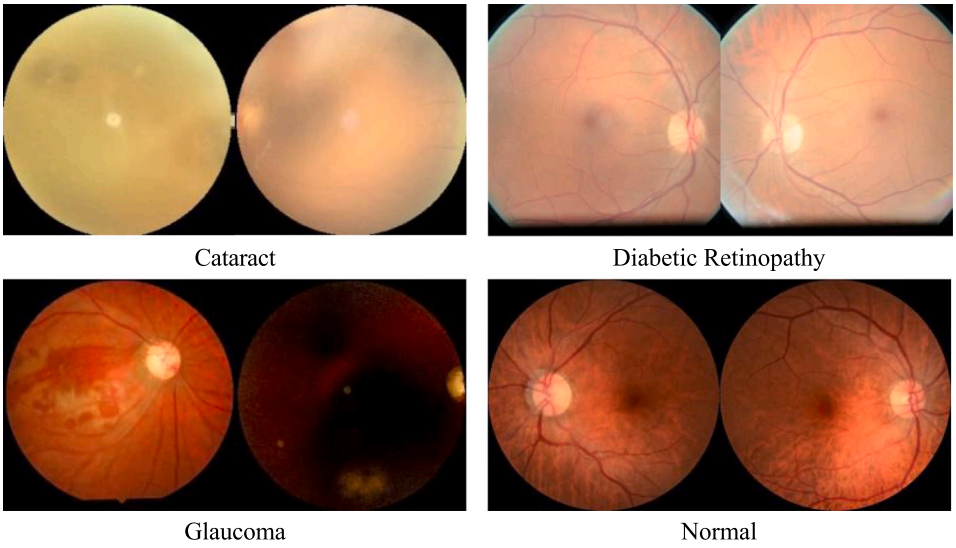

Fig. 21. An overview of the eye disease detection dataset.

are employed: accuracy, precision, recall, and F1-score, which are defined in Eq. (9).

$$\begin{aligned}\text{Accuracy} &= \frac{TP+TN}{TP+TN+FP+FN}\\ \text{Precision} &= \frac{TP}{TP+FP}\\ \text{Recall} &= \frac{TP}{TP+FN}\\ \text{F1-score} &= \frac{2*precision*recall}{precision+recall}\end{aligned} \tag{9}$$

where TP (true positive) denotes the number of instances that are correctly predicted as the positive class. TN (true negative), FP (false positive), and FN (false negative) have similar definitions.

To further demonstrate the competitiveness of DenseNet-L-SHACSO-ELM, five well-known deep learning models are employed: AlexNet [60], MobileNetV2 [61], DenseNet-169 [62] ResNet-18 [63], VGG-16 [64], and Inception [65], which are provided by the PyTorch library [66]. The batch size, learning rate, and training epoch of deep learning models are 30, 0.001, and 30, respectively, while the maximum and minimum population size and the maximum iteration of L-SHACSO are 40, 4, and 200, respectively. Each training process is repeated 10 times to alleviate the impact of randomness. Table 15 summarizes the experimental results while the confusion matrix of the ResNet-CFRIME-ELM model is presented in Fig. 22.

The experimental results indicate that our proposed DenseNet-L-SHACSO-ELM model significantly outperforms other competitor deep learning models in eye disease detection, which achieves superior performance across accuracy, precision, recall, and F1-score. This enhanced performance highlights the ability of the proposed model to reliably identify disease patterns within complex eye disease image datasets. Furthermore, the effectiveness of L-SHACSO is particularly remarkable in this challenge, as it plays a crucial role in optimizing ELM parameters. L-SHACSO enhances the detection capacity by fine-tuning

Table 15
Comparison results against deep learning models in eye disease detection.

Models	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
AlexNet	89.763 ± 0.017	89.950 ± 0.017	89.7630 ± 0.017	89.645 ± 0.017
MobileNetV2	91.398 ± 0.003	91.456 ± 0.003	91.398 ± 0.003	91.405 ± 0.003
DenseNet	92.109 ± 0.005	92.213 ± 0.004	92.109 ± 0.005	92.047 ± 0.005
ResNet-18	91.919 ± 0.003	92.010 ± 0.003	91.919 ± 0.003	91.893 ± 0.003
VGG-16	92.109 ± 0.01	92.150 ± 0.01	92.109 ± 0.01	92.084 ± 0.01
Inception	89.004 ± 0.003	89.161 ± 0.004	89.004 ± 0.003	88.978 ± 0.003
DenseNet-LSHACSO-ELM	**92.417 ± 0.007**	**92.463 ± 0.006**	**92.417 ± 0.007**	**92.398 ± 0.007**

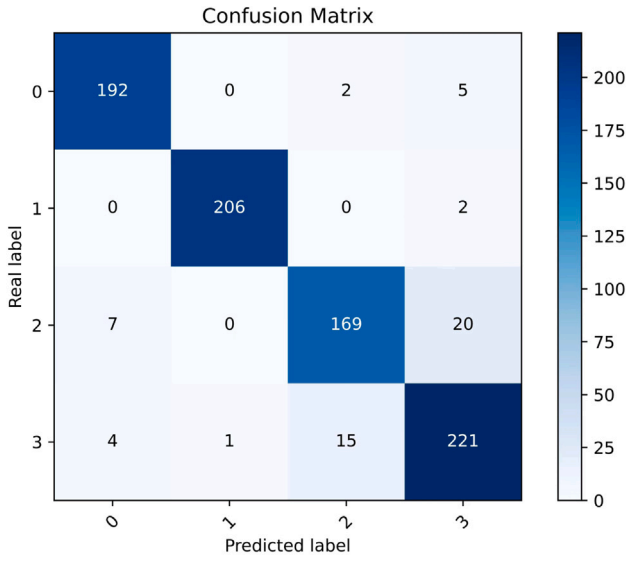

Fig. 22. The confusion matrix of the DenseNet-L-SHACSO-ELM model.

the weights and biases of ELM and enabling it to generalize effectively across diverse disease categories and varying images.

6. Conclusion

This paper introduces L-SHACSO, an efficient and effective optimizer for functional and engineering optimization. Motivated by the impressive performance of L-SHADE in various optimization challenges, we incorporate the success history adaptation strategy into CSO to intelligently adjust the hyper-parameter φ during optimization. Additionally, we integrate a linear population reduction scheme into CSO to better balance exploitation and exploration across different optimization phases. To thoroughly evaluate L-SHACSO, we conduct numerical experiments on CEC2017, CEC2020, CEC2022, and eight engineering optimization problems. We compare our proposal with state-of-the-art optimizers such as CMA-ES, jSO, and L-SHADE-cnEpSin, as well as recently proposed MAs including RIME and PO. The experimental results and rigorous statistical analysis confirm the efficiency and effectiveness of L-SHACSO across a range of optimization tasks, which demonstrates its potential for addressing real-world applications. Furthermore, sensitivity experiments examining the initial setting of μ_φ reveal the remarkable robustness of L-SHACSO to hyper-parameter settings. Furthermore, we present a hybrid deep learning model named DenseNet-L-SHACSO-ELM for eye disease detection, where the DenseNet is responsible for feature selection and the L-SHACSO-optimized ELM for classification. Comprehensive experiments and analysis underscore the potential of DenseNet-L-SHACSO-ELM for practical deployment in real-world scenarios. This model provides a valuable tool for medical practitioners to support timely interventions and increase the recovery rate of patients.

However, L-SHACSO still has limitations that warrant further investigation and improvement. For instance, L-SHACSO failed to consistently achieve the global optimum for many test instances in the CEC benchmarks. Moreover, its application potential remains constrained, with limited adaptability to broader and more challenging real-world scenarios. In future research, we will continue to focus on developing efficient variants of CSO to tackle various optimization challenges, including but not limited to large-scale expensive optimization, dynamic optimization, and discrete optimization.

CRediT authorship contribution statement

Rui Zhong: Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Conceptualization. **Zhongmin Wang:** Writing – review & editing, Writing – original draft, Investigation. **Abdelazim G. Hussien:** Writing – review & editing, Data curation. **Essam H. Houssein:** Writing – review & editing, Formal analysis. **Ibrahim Al-Shourbaji:** Writing – review & editing, Investigation. **Mohamed A. Elseify:** Writing – review & editing, Methodology, Conceptualization. **Jun Yu:** Writing – review & editing, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by JST SPRING Grant Number JPMJSP2119 and the project of the School of Tropical Crops, Yunnan Agricultural University (No. 2023RYYB003).

Appendix A. Details of CEC benchmarks

The details of CEC2017, CEC2020, and CEC2022 are summarized in Tables A.16, A.17, and A.18.

Appendix B. Details of engineering problems

B.1. Cantilever Beam Design Problem (CBDP)

The demonstration and the mathematical model of CBDP are presented in Fig. 23 and Eq. (10).

$$\begin{aligned} &\textbf{min } f(X) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5) \\ &\textbf{s.t. } g(X) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0 \\ &\textbf{where } 0.01 \le x_i \le 100,\ i \in \{1,2,3,4,5\} \end{aligned} \tag{10}$$

Table A.16
The CEC2017 benchmark: Uni.=Unimodal function, Multi.=Simple multimodal function, Hybrid.=Hybrid function, Comp.=Composition function.

No.	Func.	Feature	Optimum
f_1	Shifted and Rotated Bent Cigar function	Uni.	100
f_3	Shifted and Rotated Rosenbrock's function	Multi.	300
f_4	Shifted and Rotated Rastrigin's function		400
f_5	Shifted and Rotated Expanded Scaffer's F6 function		500
f_6	Shifted and Rotated Lunacek Bi_Rastrigin function		600
f_7	Shifted and Rotated Non-Continuous Rastrigin's function		700
f_8	Shifted and Rotated Levy function		800
f_9	Shifted and Rotated Schwefel's function		900
f_{10}	Hybrid function 1 (N = 3)	Hybrid.	1000
f_{11}	Hybrid function 2 (N = 3)		1100
f_{12}	Hybrid function 3 (N = 3)		1200
f_{13}	Hybrid function 4 (N = 4)		1300
f_{14}	Hybrid function 5 (N = 4)		1400
f_{15}	Hybrid function 6 (N = 4)		1500
f_{16}	Hybrid function 6 (N = 5)		1600
f_{17}	Hybrid function 6 (N = 5)		1700
f_{18}	Hybrid function 6 (N = 5)		1800
f_{19}	Hybrid function (N = 6)		1900
f_{20}	Composition function 1 (N = 3)	Comp.	2000
f_{21}	Composition function 2 (N = 3)		2100
f_{22}	Composition function 3 (N = 4)		2200
f_{23}	Composition function 4 (N = 4)		2300
f_{24}	Composition function 5 (N = 5)		2400
f_{25}	Composition function 6 (N = 5)		2500
f_{26}	Composition function 7 (N = 6)		2600
f_{27}	Composition function 8 (N = 6)		2700
f_{28}	Composition function 9 (N = 3)		2800
f_{29}	Composition function 10 (N = 3)		2900
f_{30}	Composition function 11 (N = 3)		3000
Search range: $[-100, 100]^D$			

Table A.17
The CEC2020 benchmark: Uni.=Unimodal function, Multi.=Multimodal function, Hybrid.=Hybrid function, Comp.=Composition function.

No.	Func.	Feature	Optimum
f_1	Shifted and Rotated Bent Cigar Function	Uni.	100
f_2	Shifted and Rotated Schwefel's function	Multi.	1100
f_3	Shifted and Rotated Lunacek bi-Rastrigin function		700
f_4	Expanded Rosenbrock's plus Griewangk's function		1900
f_5	Hybrid function 1 (N = 3)	Hybrid.	1700
f_6	Hybrid function 2 (N = 4)		1600
f_7	Hybrid function 3 (N = 5)		2100
f_8	Composition function 1 (N = 3)	Comp.	2200
f_9	Composition function 2 (N = 4)		2400
f_{10}	Composition function 3 (N = 5)		2500
Search range: $[-100, 100]^D$			

Table A.18
The CEC2022 benchmark: Uni.=Unimodal function, Basic.=Basic function, Hybrid.=Hybrid function, Comp.=Composition function.

No.	Func.	Feature	Optimum
f_1	Shifted and full Rotated Zakharov Function	Uni.	300
f_2	Shifted and full Rotated Rosenbrock's Function	Basic.	400
f_3	Shifted and full Rotated Expanded Schaffer's f_6 Function		600
f_4	Shifted and full Rotated Non-Continuous Rastrigin's Function		800
f_5	Shifted and full Rotated Levy Function		900
f_6	Hybrid function 1 (N = 3)	Hybrid.	1800
f_7	Hybrid function 2 (N = 6)		2000
f_8	Hybrid function 3 (N = 5)		2200
f_9	Composition function 1 (N = 5)	Comp.	2300
f_{10}	Composition function 2 (N = 4)		2400
f_{11}	Composition function 3 (N = 5)		2600
f_{12}	Composition function 3 (N = 6)		2700
Search range: $[-100, 100]^D$			

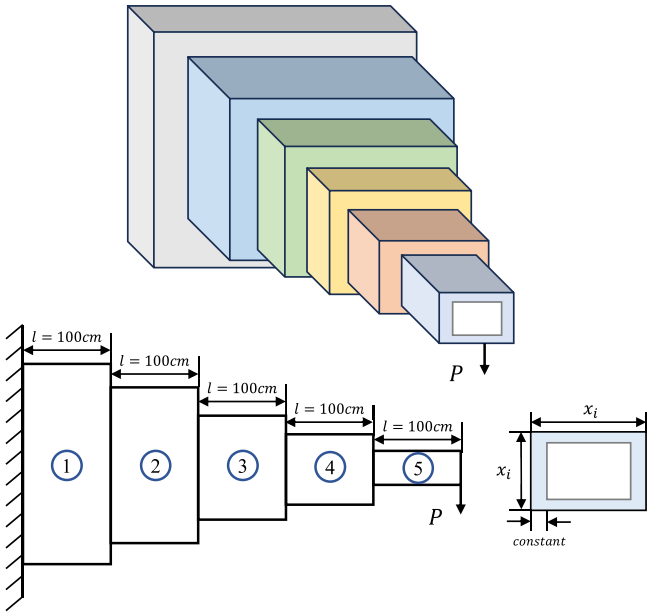

Fig. 23. A demonstration of CBDP.

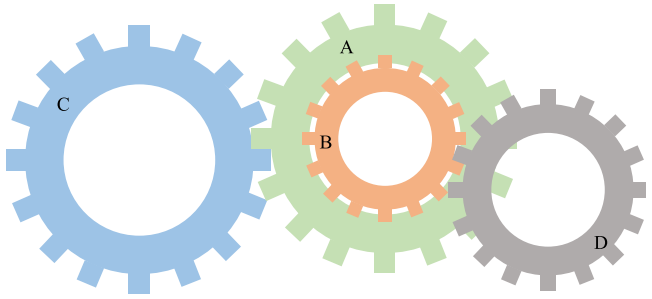

Fig. 24. A demonstration of GTDP.

B.2. Corrugated Bulkhead Design Problem (CBHDP)

The mathematical model of CBHDP is formulated in Eq. (11).

$$
\begin{aligned}
&\textbf{min } f(X) = \frac{5.885x_4(x_1+x_3)}{x_1+\sqrt{|x_3^2-x_2^2|}}\\
&\textbf{s.t. } g_1(X) = -x_4x_2(0.4x_1+\frac{x_3}{6})+8.94(x_1+\sqrt{|x_3^2-x_2^2|})\le 0\\
&\quad g_2(X) = -x_4x_2^2(0.2x_1+\frac{x_3}{12})+2.2(8.94(x_1+\sqrt{|x_3^2-x_2^2|}))^{4/3}\le 0\\
&\quad g_3(X) = -x_4+0.0156x_1+0.15\le 0\\
&\quad g_4(X) = -x_4+0.0156x_3+0.15\le 0\\
&\quad g_5(X) = -x_4+1.05\le 0\\
&\quad g_6(X) = -x_3+x_2\le 0\\
&\textbf{where } 0\le x_1,x_2,x_3\le 100\\
&\quad\ 0\le x_4\le 5
\end{aligned}
\tag{11}
$$

B.3. Gear Train Design Problem (GTDP)

The demonstration and the mathematical model of GTDP are presented in Fig. 24 and Eq. (12).

$$
\begin{aligned}
&\textbf{min } f(X) = \left(\frac{1}{6.931}-\frac{x_3x_2}{x_1x_4}\right)^2\\
&\textbf{where} x_1,x_2,x_3,x_4\in\{12,13,14,\dots,60\}
\end{aligned}
\tag{12}
$$

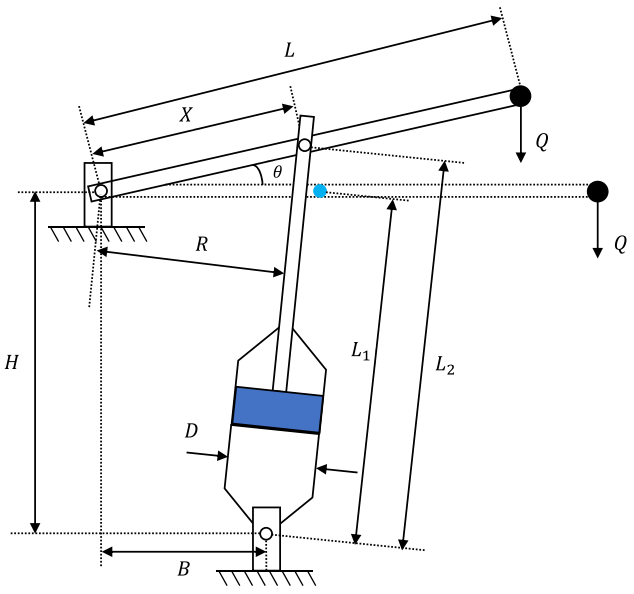

Fig. 25. A demonstration of PLDP.

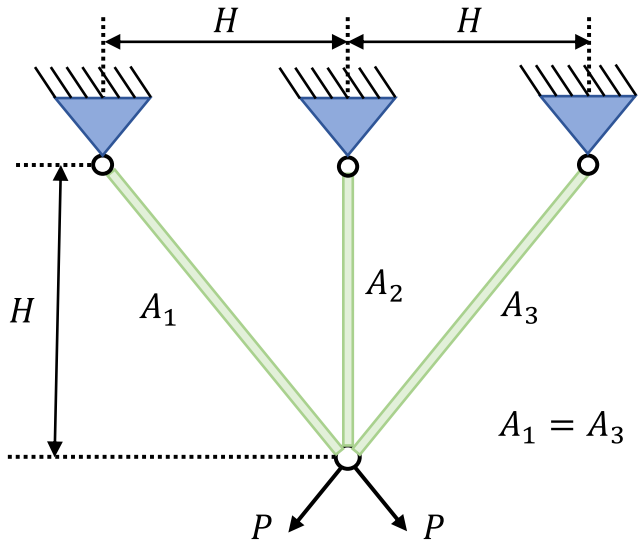

Fig. 26. A demonstration of TBTDP.

B.4. Piston Lever Design Problem (PLDP)

The demonstration and the mathematical model of TBTDP are presented in Fig. 25 and Eq. (13). More detailed constraints of this model can be found in [67].

$$
\begin{aligned}
&\textbf{min } f(X) = \frac{1}{4}\pi x_3^2(L_2-L_1)\\
&\textbf{s.t. } g_1(X) = QL\cos(\theta)-R\times F\le 0\\
&\quad g_2(X) = Q(L-x_4)-M_{max}\le 0\\
&\quad g_3(X) = 1.2(L_2-L_1)-L_1\le 0\\
&\quad g_4(X) = \frac{x_3}{2}-x_2\le 0\\
&\textbf{where } 0.05\le x_1,x_2,x_4\le 500\\
&\quad\ 0.05\le x_3\le 120
\end{aligned}
\tag{13}
$$

B.5. Speed Reducer Design Problem (SRDP)

The mathematical model of SRDP is formulated and Eq. (14).

$$
\begin{aligned}
\textbf{min}\ & f(X) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0924) - 1.508x_1(x_6^2 + x_7^2) + \\
& \quad 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \\
\textbf{s.t.}\ & g_1(X) = \frac{27}{x_1x_2^2x_3} - 1 \le 0 \\
& g_2(X) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0 \\
& g_3(X) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0 \\
& g_4(X) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0 \\
& g_5(X) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 16900000}}{110x_6^3} - 1 \le 0 \\
& g_6(X) = \frac{\sqrt{(745x_5/x_2x_3)^2 + 157500000}}{85x_7^3} - 1 \le 0 \\
& g_7(X) = \frac{x_2x_3}{40} - 1 \le 0 \\
& g_8(X) = \frac{5x_2}{x_1} - 1 \le 0 \\
& g_9(X) = \frac{x_1}{12x_2} - 1 \le 0 \\
& g_{10}(X) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \\
& g_{11}(X) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0 \\
\textbf{where}\ & 2.6 \le x_1 \le 2.6 \\
& 0.7 \le x_2 \le 0.8 \\
& 17 \le x_3 \le 28 \\
& 7.3 \le x_4x_5 \le 8.3 \\
& 2.9 \le x_6 \le 3.9 \\
& 5 \le x_7 \le 5.5
\end{aligned}
\tag{14}
$$

B.6. Three Bar Truss Design Problem (TBTDP)

The demonstration and the mathematical model of TBTDP are presented in Fig. 26 and Eq. (15).

$$
\begin{aligned}
\textbf{min}\ & f(X) = (2\sqrt{2}x_1 + x_2) \cdot l \\
\textbf{s.t.}\ & g_1(X) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \\
& g_2(X) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \\
& g_3(X) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \le 0 \\
& l = 100\ cm, P = 2kN/cm^3, \sigma = 2kN/cm^3 \\
\textbf{where}\ & 0 \le x_1, x_2 \le 1
\end{aligned}
\tag{15}
$$

B.7. Tubular Column Design Problem (TCDP)

The demonstration and the mathematical model of TCDP are presented in Fig. 27 and Eq. (16). More details about constraints in TCDP can be referred to [67].

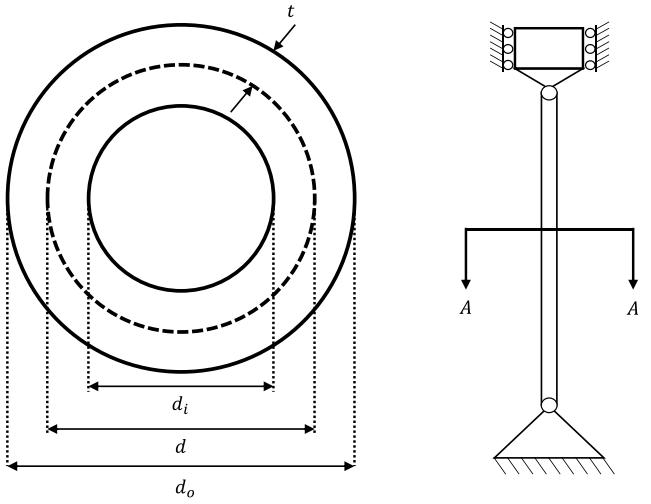

Fig. 27. A demonstration of TCDP.

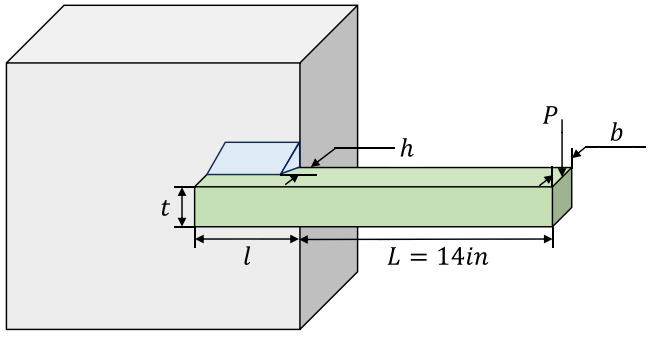

Fig. 28. A demonstration of WBDP.

$$
\begin{aligned}
\textbf{min}\ & f(X) = 9.8x_1x_2 + 2x_1 \\
\textbf{s.t.}\ & g_1(X) = \frac{P}{\pi x_1x_2\sigma_y} - 1 \le 0 \\
& g_2(X) = \frac{8PL^2}{\pi^3Ex_1x_2(x_1^2 + x_2^2)} - 1 \le 0 \\
\textbf{where}\ & 2 \le x_1 \le 14 \\
& 0.2 \le x_2 \le 0.8
\end{aligned}
\tag{16}
$$

B.8. Welded Beam Design Problem (WBDP)

The demonstration and the mathematical model of TCDP are presented in Fig. 28 and Eq. (17). More detailed constraints of this model can be found in [67].

$$
\begin{aligned}
\textbf{min}\ & f(X) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \\
\textbf{s.t.}\ & g_1(X) = \tau(X) - \tau_{max} \le 0 \\
& g_2(X) = \sigma(X) - \sigma_{max} \le 0 \\
& g_3(X) = \delta(X) - \delta_{max} \le 0 \\
& g_4(X) = x_1 - x_4 \le 0 \\
& g_5(X) = P - P_c(X) \le 0 \\
& g_6(X) = 0.125 - x_1 \le 0 \\
& g_7(X) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0 \\
\textbf{where}\ & 0.1 \le x_1, x_4 \le 2 \\
& 0.1 \le x_2, x_3 \le 10
\end{aligned}
\tag{17}
$$

Table C.19
Results of comparison experiments in 30-D CEC2017.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	3.056e+09 +	3.510e+07 +	5.304e+03 +	5.947e+03 +	7.978e+08 +	3.585e+06 +	1.825e+08 +	2.699e+09 +	2.903e+09 +	5.977e+07 +	1.772e+08 +	**2.497e+03**
	std	5.832e+08	9.878e+06	2.792e+03	2.161e+03	1.295e+08	4.131e+06	9.994e+07	1.896e+09	1.940e+09	2.951e+07	6.480e+07	1.622e+03
f_3	mean	8.717e+04 +	3.218e+04 ≈	7.843e+04 +	4.815e+04 +	4.466e+04 +	3.900e+04 +	3.523e+04 ≈	4.187e+04 +	4.301e+04 +	6.548e+04 +	6.206e+04 +	**3.164e+04**
	std	1.443e+04	4.850e+03	1.801e+04	8.699e+03	6.863e+03	6.272e+03	5.966e+03	8.760e+03	8.147e+03	1.657e+04	1.032e+04	6.902e+03
f_4	mean	7.245e+02 +	5.283e+02 +	**5.041e+02** ≈	5.149e+02 ≈	6.725e+02 +	5.510e+02 +	6.195e+02 +	6.591e+02 +	6.646e+02 +	5.335e+02 +	5.930e+02 +	5.158e+02
	std	5.063e+01	6.317e+00	1.839e+01	7.237e+00	3.496e+01	2.855e+01	8.301e+01	1.391e+02	1.147e+02	2.657e+01	4.272e+01	2.104e+00
f_5	mean	7.684e+02 +	7.980e+02 +	6.443e+02 +	5.985e+02 +	7.513e+02 +	8.000e+02 +	8.015e+02 +	7.222e+02 +	7.444e+02 +	6.128e+02 +	7.360e+02 +	**5.113e+02**
	std	1.111e+01	1.077e+01	1.180e+01	2.793e+01	1.625e+01	4.455e+00	3.548e+00	3.556e+01	4.257e+01	1.904e+01	3.570e+01	2.705e+00
f_6	mean	6.398e+02 +	6.666e+02 +	6.013e+02 +	6.001e+02 +	6.508e+02 +	6.651e+02 +	6.657e+02 +	6.569e+02 +	6.614e+02 +	6.074e+02 +	6.652e+02 +	**6.000e+02**
	std	3.450e+00	1.825e+00	2.479e−01	1.082e−02	1.562e+01	1.072e+00	6.671e−01	6.291e+00	1.281e+01	2.558e+00	7.431e+00	1.305e−03
f_7	mean	1.155e+03 +	1.220e+03 +	8.746e+02 +	8.920e+02 +	9.751e+02 +	1.303e+03 +	1.313e+03 +	1.191e+03 +	1.085e+03 +	8.963e+02 +	1.116e+03 +	**7.522e+02**
	std	2.813e+01	1.420e+02	1.275e+01	9.505e+00	1.170e+01	1.497e+01	9.855e+00	7.187e+01	8.188e+01	2.552e+01	6.001e+01	1.884e+01
f_8	mean	1.075e+03 +	9.956e+02 +	9.425e+02 +	8.921e+02 +	1.060e+03 +	9.970e+02 +	9.986e+02 +	9.665e+02 +	9.876e+02 +	9.134e+02 +	9.962e+02 +	**8.118e+02**
	std	1.862e+01	8.017e+00	1.192e+01	3.991e+01	1.457e+01	3.524e+00	4.035e+00	1.908e+01	3.383e+01	2.053e+01	2.453e+01	2.890e+00
f_9	mean	5.025e+03 +	8.762e+03 +	1.265e+03 +	9.000e+02 +	3.620e+03 +	6.737e+03 +	7.441e+03 +	5.253e+03 +	5.576e+03 +	1.796e+03 +	7.265e+03 +	**9.000e+02**
	std	5.320e+02	4.724e+02	1.035e+02	4.014e−03	1.302e+03	6.779e+02	4.306e+02	8.970e+02	7.284e+02	6.540e+02	2.183e+03	3.189e−05
f_{10}	mean	8.671e+03 +	5.262e+03 +	6.228e+03 +	6.443e+03 +	5.549e+03 +	5.321e+03 +	5.386e+03 +	5.784e+03 +	6.116e+03 +	5.352e+03 +	6.363e+03 +	**2.578e+03**
	std	3.065e+02	1.431e+02	2.581e+02	5.709e+02	3.301e+02	1.945e+02	2.148e+02	4.639e+02	7.652e+02	5.747e+02	4.890e+02	3.821e+02
f_{11}	mean	1.355e+03 +	1.349e+03 +	1.285e+03 +	1.264e+03 +	1.655e+03 +	1.218e+03 +	1.283e+03 +	1.656e+03 +	1.922e+03 +	1.395e+03 +	1.642e+03 +	**1.178e+03**
	std	2.324e+01	3.200e+01	2.186e+01	4.844e+01	9.386e+01	2.478e+01	8.177e+01	4.993e+02	7.285e+02	5.467e+01	1.471e+02	2.593e+01
f_{12}	mean	2.033e+07 +	1.007e+07 +	1.227e+06 +	8.239e+05 +	1.068e+08 +	9.175e+05 +	4.123e+06 +	3.356e+07 +	1.169e+08 +	2.518e+07 +	1.508e+08 +	**7.131e+05**
	std	2.791e+06	1.969e+06	4.593e+05	4.453e+05	2.504e+07	4.704e+05	3.574e+06	3.456e+07	8.284e+07	2.326e+07	1.776e+08	3.105e+05
f_{13}	mean	**7.850e+03** −	1.063e+06 +	9.929e+04 +	1.012e+04 ≈	2.182e+07 +	1.825e+04 +	3.753e+04 +	9.608e+05 +	3.475e+06 +	7.179e+05 +	8.800e+05 +	1.129e+04
	std	1.565e+03	4.897e+05	3.952e+04	3.660e+03	9.409e+06	6.180e+03	9.547e+03	2.669e+06	1.119e+07	1.101e+06	9.603e+05	2.627e+03
f_{14}	mean	**1.504e+03** −	1.065e+04 −	8.853e+03 −	2.332e+05 +	6.566e+04 −	1.907e+03 −	1.777e+03 −	4.640e+05 +	4.172e+05 +	1.420e+05 ≈	1.984e+05 ≈	1.191e+05
	std	8.476e+00	4.702e+03	9.614e+03	1.270e+05	3.700e+04	6.161e+02	3.500e+02	4.055e+05	5.685e+05	1.541e+05	1.714e+05	7.403e+04
f_{15}	mean	**1.836e+03** ≈	2.259e+04 +	1.815e+04 +	2.231e+03 ≈	3.366e+05 +	2.542e+03 +	3.863e+03 +	7.258e+04 +	2.734e+05 +	5.572e+04 +	6.523e+04 +	1.995e+03
	std	4.015e+01	5.983e+03	1.416e+04	6.819e+02	1.402e+05	5.142e+02	8.163e+02	8.592e+04	5.269e+05	3.525e+04	4.230e+04	5.094e+02
f_{16}	mean	3.620e+03 +	3.073e+03 +	2.672e+03 +	2.125e+03 +	3.506e+03 +	3.156e+03 +	3.006e+03 +	3.468e+03 +	3.359e+03 +	2.582e+03 +	3.500e+03 +	**1.947e+03**
	std	1.550e+02	1.975e+02	1.562e+02	2.597e+02	1.643e+02	2.405e+02	1.885e+02	2.782e+02	3.040e+02	2.770e+02	3.302e+02	1.908e+02

Table C.20
Results of comparison experiments in 30-D CEC2017 (Continued).

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_{17}	mean	2.618e+03 +	2.238e+03 +	2.000e+03 +	1.856e+03 +	2.376e+03 +	2.201e+03 +	2.602e+03 +	2.603e+03 +	2.422e+03 +	2.072e+03 +	2.607e+03 +	**1.790e+03**
	std	1.028e+02	1.719e+02	7.343e+01	1.188e+02	8.144e+01	2.745e+02	2.711e+02	3.009e+02	2.197e+02	1.898e+02	2.025e+02	7.607e+01
f_{18}	mean	**4.026e+03** −	2.942e+05 ≈	4.495e+05 ≈	4.422e+05 ≈	1.027e+06 +	4.936e+04 −	4.744e+04 −	1.425e+06 +	2.647e+06 +	1.775e+06 +	3.060e+06 +	3.843e+05
	std	7.610e+02	1.279e+05	2.324e+05	2.703e+05	4.393e+05	2.799e+04	2.161e+04	1.204e+06	4.610e+06	1.897e+06	2.600e+06	2.209e+05
f_{19}	mean	**2.144e+03** −	1.827e+04 +	1.881e+04 +	4.812e+03 ≈	4.754e+05 +	4.773e+03 ≈	4.042e+03 ≈	6.555e+05 +	1.950e+06 +	5.976e+04 +	5.977e+06 +	4.594e+03
	std	7.188e+01	9.001e+03	1.168e+04	1.522e+03	1.948e+05	1.354e+03	1.341e+03	5.718e+05	1.854e+06	5.194e+04	4.329e+06	1.198e+03
f_{20}	mean	2.898e+03 +	3.035e+03 +	2.399e+03 +	2.174e+03 +	2.643e+03 +	3.114e+03 +	3.114e+03 +	2.885e+03 +	2.650e+03 +	2.435e+03 +	2.657e+03 +	**2.153e+03**
	std	1.230e+02	1.102e+02	8.926e+01	6.010e+01	1.239e+02	3.619e+01	3.614e+01	2.807e+02	1.779e+02	1.586e+02	1.510e+02	6.374e+00
f_{21}	mean	2.556e+03 +	2.490e+03 +	2.444e+03 +	2.354e+03 +	2.526e+03 +	2.655e+03 +	2.659e+03 +	2.521e+03 +	2.519e+03 +	2.413e+03 +	2.524e+03 +	**2.310e+03**
	std	1.351e+01	8.928e+01	1.337e+01	2.835e+01	2.259e+01	5.309e+01	2.476e+01	5.462e+01	4.350e+01	2.833e+01	3.810e+01	2.363e+00
f_{22}	mean	9.977e+03 +	7.437e+03 +	2.483e+03 +	2.300e+03 +	2.460e+03 +	7.710e+03 +	7.827e+03 +	6.334e+03 +	4.128e+03 +	5.094e+03 +	5.404e+03 +	**2.300e+03**
	std	3.264e+02	1.867e+02	1.663e+02	6.019e−02	3.009e+01	2.433e+02	3.046e+02	1.634e+03	1.984e+03	2.035e+03	2.731e+03	9.366e−03
f_{23}	mean	2.921e+03 +	3.980e+03 +	2.794e+03 +	2.669e+03 +	2.895e+03 +	4.553e+03 +	4.629e+03 +	3.266e+03 +	2.908e+03 +	2.773e+03 +	3.105e+03 +	**2.662e+03**
	std	2.048e+01	6.078e+02	1.080e+01	6.705e+00	1.364e+01	1.248e+02	1.270e+02	1.360e+02	5.817e+01	2.325e+01	6.935e+01	5.139e+00
f_{24}	mean	3.082e+03 +	3.029e+03 +	2.964e+03 +	2.831e+03 +	3.061e+03 +	3.088e+03 +	3.112e+03 +	3.419e+03 +	3.055e+03 +	2.940e+03 +	3.215e+03 +	**2.825e+03**
	std	1.895e+01	8.878e+00	1.377e+01	3.903e+00	1.517e+01	1.551e+02	1.101e+02	1.161e+02	6.660e+01	2.892e+01	1.279e+02	4.362e+00
f_{25}	mean	3.151e+03 +	2.924e+03 +	2.892e+03 +	2.888e+03 +	2.982e+03 +	2.978e+03 +	2.980e+03 +	3.008e+03 +	3.085e+03 +	2.933e+03 +	3.005e+03 +	**2.888e+03**
	std	4.919e+01	1.159e+01	9.878e+00	1.143e+00	1.263e+01	2.310e+01	2.547e+01	4.034e+01	7.972e+01	2.499e+01	2.815e+01	1.427e+00
f_{26}	mean	6.362e+03 +	**3.081e+03** −	5.081e+03 +	3.647e+03 ≈	3.696e+03 ≈	7.146e+03 +	6.753e+03 +	7.757e+03 +	6.496e+03 +	4.854e+03 +	6.617e+03 +	3.523e+03
	std	1.514e+02	6.194e+01	1.257e+02	2.981e+02	7.515e+01	1.254e+03	1.088e+03	9.535e+02	7.746e+02	4.970e+02	1.819e+03	3.657e+02
f_{27}	mean	3.242e+03 +	5.801e+03 +	3.227e+03 +	**3.219e+03** ≈	3.295e+03 +	6.681e+03 +	6.645e+03 +	3.435e+03 +	3.345e+03 +	3.233e+03 +	3.478e+03 +	3.219e+03
	std	1.338e+01	1.073e+03	4.674e+00	5.871e+00	1.495e+01	1.197e+02	2.723e+02	1.123e+02	5.741e+01	9.279e+00	1.090e+02	6.588e+00
f_{28}	mean	3.470e+03 +	3.283e+03 +	3.249e+03 +	3.252e+03 +	3.393e+03 +	3.329e+03 +	3.358e+03 +	3.418e+03 +	3.502e+03 +	3.331e+03 +	3.377e+03 +	**3.225e+03**
	std	4.634e+01	1.344e+01	1.905e+01	1.753e+01	2.879e+01	1.905e+01	3.970e+01	6.975e+01	1.353e+02	5.913e+01	3.886e+01	1.608e+01
f_{29}	mean	4.381e+03 +	4.069e+03 +	3.745e+03 +	3.519e+03 +	4.375e+03 +	3.944e+03 +	4.357e+03 +	4.862e+03 +	4.597e+03 +	3.933e+03 +	4.933e+03 +	**3.402e+03**
	std	1.925e+02	1.415e+02	7.037e+01	1.233e+02	1.302e+02	1.475e+02	2.563e+02	4.872e+02	3.202e+02	2.718e+02	4.393e+02	8.793e+01
f_{30}	mean	8.254e+04 +	8.012e+05 +	1.819e+05 +	1.114e+04 +	6.507e+06 +	3.832e+04 +	2.556e+05 +	6.360e+06 +	1.748e+07 +	8.003e+05 +	2.670e+07 +	**7.791e+03**
	std	3.088e+04	2.567e+05	7.965e+04	2.118e+03	2.714e+06	3.697e+04	2.111e+05	7.292e+06	8.837e+06	5.809e+05	3.146e+07	7.588e+02
+ /≈/−:		24/1/4	25/2/2	26/2/1	22/7/0	27/1/1	26/2/1	25/2/2	29/0/0	29/0/0	28/1/0	28/1/0	–
Avg. ranks:		7.7	6.6	4.3	3.2	7.8	6.7	7.6	8.8	8.6	5.4	9.1	**1.6**

Appendix C. Detailed results on CEC2017

The detailed experimental results of comparison on CEC2017 are summarized in Tables C.19, C.20, C.21, and C.22.

Appendix D. Detailed results on CEC2020

The detailed experimental results of comparison on CEC2020 are summarized in Tables D.23 and D.24.

Appendix E. Detailed results on CEC2022

The detailed experimental results of comparison on CEC2022 are summarized in Tables E.25 and E.26.

Appendix F. Detailed results of sensitivity analysis

The detailed experimental results for sensitivity analysis of the hyper-parameter μ_φ in {0.1, 0.15, 0.2, 0.25, 0.3} are summarized in Tables F.27, F.28, F.29, F.30, F.31, and F.32.

Table C.21
Results of comparison experiments in 50-D CEC2017.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	1.156e+10 +	2.848e+08 +	**1.135e+03** ≈	2.000e+03 ≈	2.490e+08 +	5.024e+07 +	1.549e+09 +	1.104e+10 +	1.201e+10 +	2.127e+08 +	5.535e+08 +	1.381e+03
	std	2.874e+09	3.530e+07	1.475e+03	2.312e+03	6.902e+07	3.921e+07	8.469e+08	5.646e+09	4.804e+09	7.250e+07	1.690e+08	1.182e+03
f_3	mean	2.578e+05 +	1.005e+05 +	1.566e+05 +	1.152e+05 +	1.228e+05 +	1.082e+05 +	1.043e+05 +	1.465e+05 +	9.640e+04 +	1.530e+05 +	1.561e+05 +	**8.211e+04**
	std	3.211e+04	1.262e+04	6.412e+04	2.217e+04	9.228e+03	8.865e+03	1.148e+04	2.476e+04	1.844e+04	2.592e+04	1.921e+04	1.152e+04
f_4	mean	1.749e+03 +	6.909e+02 +	**5.320e+02** −	5.877e+02 ≈	7.005e+02 +	6.696e+02 +	9.285e+02 +	1.946e+03 +	1.989e+03 +	6.851e+02 +	8.822e+02 +	5.810e+02
	std	4.939e+02	2.820e+01	3.817e+01	3.343e+01	2.626e+01	5.401e+01	1.814e+02	1.031e+03	9.890e+02	3.667e+01	1.213e+02	2.970e+01
f_5	mean	1.010e+03 +	8.768e+02 +	7.761e+02 +	5.251e+02 ≈	9.809e+02 +	8.793e+02 +	8.824e+02 +	8.864e+02 +	9.030e+02 +	7.264e+02 +	9.128e+02 +	**5.233e+02**
	std	2.008e+01	8.642e+00	1.780e+01	1.022e+01	2.994e+01	8.001e+00	7.204e+00	3.235e+01	5.407e+01	2.745e+01	5.060e+01	3.710e+00
f_6	mean	6.567e+02 +	6.719e+02 +	6.004e+02 +	6.000e+02 +	6.557e+02 +	6.691e+02 +	6.698e+02 +	6.625e+02 +	6.703e+02 +	6.138e+02 +	6.736e+02 +	**6.000e+02**
	std	7.412e+00	1.279e+00	1.063e−01	2.097e−03	1.684e+01	7.676e−01	8.488e−01	6.751e+00	4.570e+00	5.197e+00	8.604e+00	6.336e−04
f_7	mean	1.502e+03 +	1.818e+03 +	1.027e+03 +	9.816e+02 +	1.178e+03 +	1.772e+03 +	1.780e+03 +	1.655e+03 +	1.587e+03 +	1.097e+03 +	1.688e+03 +	**7.682e+02**
	std	5.569e+01	3.544e+01	1.423e+01	3.708e+01	1.627e+01	9.296e+00	6.502e+00	1.107e+02	1.436e+02	4.800e+01	1.534e+02	5.000e+00
f_8	mean	1.321e+03 +	1.212e+03 +	1.079e+03 +	8.267e+02 ≈	1.331e+03 +	1.218e+03 +	1.220e+03 +	1.172e+03 +	1.224e+03 +	1.033e+03 +	1.235e+03 +	**8.234e+02**
	std	2.017e+01	6.009e+00	1.918e+01	8.594e+00	3.550e+01	7.386e+00	6.410e+00	3.654e+01	3.567e+01	3.453e+01	4.386e+01	4.811e+00
f_9	mean	1.374e+04 +	2.992e+04 +	3.410e+03 +	9.002e+02 ≈	1.157e+04 +	1.799e+04 +	2.181e+04 +	1.377e+04 +	1.881e+04 +	4.407e+03 +	2.344e+04 +	**9.002e+02**
	std	4.109e+03	1.599e+03	9.531e+02	2.464e−01	8.711e+03	2.173e+03	1.692e+03	1.849e+03	3.203e+03	2.074e+03	3.018e+03	1.940e−01
f_{10}	mean	1.509e+04 +	8.461e+03 +	1.024e+04 +	5.756e+03 +	1.247e+04 +	7.984e+03 +	8.116e+03 +	9.280e+03 +	9.797e+03 +	8.864e+03 +	1.068e+04 +	**3.232e+03**
	std	4.528e+02	4.321e+02	2.913e+02	1.336e+03	3.241e+02	3.931e+02	3.735e+02	7.192e+02	6.905e+02	9.058e+02	1.276e+03	5.056e+02
f_{11}	mean	1.640e+03 ≈	2.193e+03 +	1.598e+03 −	2.009e+03 ≈	1.945e+03 ≈	**1.442e+03** −	1.650e+03 ≈	3.810e+03 +	4.835e+03 +	1.853e+03 ≈	3.336e+03 +	1.842e+03
	std	6.951e+01	1.662e+02	7.179e+01	4.302e+02	4.042e+02	7.135e+01	1.399e+02	1.740e+03	1.938e+03	1.281e+02	6.335e+02	3.768e+02
f_{12}	mean	2.622e+08 +	1.123e+08 +	2.960e+06 +	2.074e+06 ≈	1.464e+08 +	4.057e+06 +	2.849e+07 +	9.371e+08 +	1.001e+09 +	1.845e+08 +	5.433e+08 +	**1.902e+06**
	std	5.878e+07	1.953e+07	8.409e+05	9.878e+05	3.675e+07	1.616e+06	2.128e+07	1.337e+09	9.817e+08	1.026e+08	3.088e+08	5.854e+05
f_{13}	mean	2.080e+05 +	8.140e+06 +	3.595e+04 +	2.400e+03 ≈	1.738e+07 +	1.016e+04 +	3.820e+04 +	6.131e+07 +	2.140e+08 +	2.482e+06 +	5.382e+06 +	**2.074e+03**
	std	5.460e+04	2.454e+06	1.806e+04	1.108e+03	7.898e+06	3.819e+03	1.423e+04	8.938e+07	2.436e+08	2.896e+06	3.651e+06	5.063e+02
f_{14}	mean	**1.613e+03** −	2.706e+05 ≈	5.499e+04 −	5.367e+05 +	9.266e+04 −	4.333e+04 −	1.495e+04 −	7.312e+05 +	1.285e+06 +	7.753e+05 +	2.020e+06 +	3.402e+05
	std	1.970e+01	1.041e+05	8.307e+04	2.007e+05	1.045e+05	3.610e+04	8.928e+03	4.865e+05	1.132e+06	6.141e+05	1.532e+06	1.792e+05
f_{15}	mean	**3.595e+03** −	1.914e+05 +	1.914e+04 +	5.290e+03 ≈	4.816e+05 +	1.526e+04 +	2.265e+04 +	2.719e+06 +	1.512e+07 +	3.132e+05 +	1.637e+05 +	6.631e+03
	std	2.665e+02	1.435e+05	9.697e+03	2.901e+03	2.809e+05	3.612e+03	2.289e+03	4.865e+06	2.822e+07	2.381e+05	9.642e+04	1.759e+03
f_{16}	mean	5.434e+03 +	4.235e+03 +	3.436e+03 +	2.223e+03 ≈	4.961e+03 +	3.116e+03 +	3.220e+03 +	4.421e+03 +	4.557e+03 +	3.761e+03 +	5.052e+03 +	**2.156e+03**
	std	1.978e+02	3.210e+02	2.813e+02	1.737e+02	3.060e+02	1.948e+02	1.994e+02	8.142e+02	4.459e+02	4.545e+02	7.290e+02	1.557e+02

Table C.22
Results of comparison experiments in 50-D CEC2017 (Continued).

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_{17}	mean	4.220e+03 +	3.516e+03 +	2.971e+03 +	2.156e+03 ≈	3.714e+03 +	3.072e+03 +	3.097e+03 +	3.698e+03 +	3.665e+03 +	3.069e+03 +	3.830e+03 +	**2.066e+03**
	std	1.355e+02	1.281e+02	1.354e+02	1.818e+02	2.076e+02	1.647e+02	1.219e+02	3.599e+02	5.025e+02	3.801e+02	2.723e+02	1.314e+02
f_{18}	mean	**6.202e+04** −	1.693e+06 ≈	1.374e+06 ≈	2.153e+06 +	6.734e+05 −	5.613e+05 −	2.992e+05 −	2.259e+06 +	1.260e+07 +	4.623e+06 +	6.256e+06 +	1.635e+06
	std	1.364e+04	3.829e+05	7.872e+05	8.401e+05	1.027e+06	3.121e+05	2.192e+05	1.621e+06	2.027e+07	3.537e+06	4.427e+06	5.152e+05
f_{19}	mean	**5.412e+03** −	3.784e+05 +	2.135e+04 +	1.550e+04 ≈	7.804e+05 +	1.964e+04 +	2.593e+04 +	6.849e+05 +	3.659e+06 +	3.081e+05 +	6.874e+06 +	1.688e+04
	std	1.067e+03	1.371e+05	6.102e+03	4.474e+03	2.522e+05	3.336e+03	8.875e+03	4.548e+05	3.112e+06	1.913e+05	7.766e+06	2.732e+03
f_{20}	mean	4.177e+03 +	3.677e+03 +	3.219e+03 +	2.176e+03 ≈	3.595e+03 +	3.661e+03 +	3.672e+03 +	3.651e+03 +	3.350e+03 +	3.174e+03 +	3.648e+03 +	**2.164e+03**
	std	1.739e+02	6.830e+01	1.596e+02	1.195e+02	1.602e+02	6.559e+01	7.833e+01	3.599e+02	4.115e+02	2.343e+02	2.927e+02	1.203e+02
f_{21}	mean	2.823e+03 +	2.726e+03 +	2.571e+03 +	2.326e+03 +	2.720e+03 +	2.790e+03 +	2.761e+03 +	2.802e+03 +	2.796e+03 +	2.528e+03 +	2.791e+03 +	**2.322e+03**
	std	2.902e+01	1.647e+01	1.647e+01	9.619e+00	1.702e+01	5.021e+01	4.873e+01	6.943e+01	7.647e+01	3.669e+01	6.960e+01	3.176e+00
f_{22}	mean	1.647e+04 +	1.243e+04 +	1.069e+04 +	3.083e+03 +	1.086e+04 +	1.301e+04 +	1.318e+04 +	1.117e+04 +	1.163e+04 +	1.035e+04 +	1.235e+04 +	**2.300e+03**
	std	3.640e+02	3.168e+02	2.727e+03	1.612e+03	5.035e+03	2.565e+02	2.772e+02	9.463e+02	9.570e+02	2.041e+03	9.591e+02	1.071e−02
f_{23}	mean	3.302e+03 +	4.895e+03 +	2.997e+03 +	2.757e+03 ≈	3.168e+03 +	5.169e+03 +	5.229e+03 +	3.980e+03 +	3.298e+03 +	2.995e+03 +	3.749e+03 +	**2.753e+03**
	std	4.082e+01	3.727e+02	1.820e+01	8.028e+00	2.138e+01	1.853e+02	6.543e+01	1.793e+02	1.092e+02	3.040e+01	2.280e+02	1.259e+01
f_{24}	mean	3.471e+03 +	3.316e+03 +	3.163e+03 +	2.923e+03 +	3.338e+03 +	3.180e+03 +	3.314e+03 +	4.112e+03 +	3.459e+03 +	3.153e+03 +	3.864e+03 +	**2.908e+03**
	std	3.984e+01	2.034e+01	2.206e+01	6.174e+00	1.235e+01	8.097e+01	7.301e+01	2.154e+02	1.018e+02	4.108e+01	2.042e+02	7.549e+00
f_{25}	mean	3.959e+03 +	3.166e+03 +	3.097e+03 ≈	**3.084e+03** ≈	3.188e+03 +	3.209e+03 +	3.419e+03 +	3.777e+03 +	3.947e+03 +	3.162e+03 +	3.338e+03 +	3.088e+03
	std	1.996e+02	2.559e+01	1.437e+01	1.446e+01	3.288e+01	2.737e+01	1.528e+02	4.420e+02	4.388e+02	4.403e+01	1.090e+02	1.911e+01
f_{26}	mean	9.303e+03 +	3.702e+03 −	6.530e+03 +	4.013e+03 +	**3.681e+03** −	1.287e+04 +	1.179e+04 +	1.275e+04 +	1.022e+04 +	6.561e+03 +	1.115e+04 +	3.984e+03
	std	2.726e+02	9.232e+01	2.160e+02	1.041e+02	8.947e+01	5.321e+02	8.607e+02	1.804e+03	2.066e+03	4.964e+02	2.496e+03	3.075e+02
f_{27}	mean	3.500e+03 +	3.609e+03 +	3.360e+03 +	3.350e+03 +	3.651e+03 +	1.091e+04 +	1.143e+04 +	4.142e+03 +	4.044e+03 +	3.476e+03 +	4.477e+03 +	**3.313e+03**
	std	1.098e+02	5.037e+01	2.778e+01	3.539e+01	4.755e+01	1.189e+03	6.395e+02	4.431e+02	2.266e+02	8.918e+01	4.120e+02	2.284e+01
f_{28}	mean	3.969e+03 +	3.496e+03 +	3.366e+03 ≈	3.373e+03 ≈	3.552e+03 +	3.630e+03 +	3.874e+03 +	4.487e+03 +	4.462e+03 +	3.541e+03 +	3.687e+03 +	**3.361e+03**
	std	2.896e+02	3.708e+01	2.020e+01	2.457e+01	4.607e+01	4.632e+01	1.357e+02	3.754e+02	2.838e+02	6.985e+01	1.607e+02	1.398e+01
f_{29}	mean	5.936e+03 +	5.310e+03 +	4.132e+03 +	3.563e+03 ≈	5.438e+03 +	4.243e+03 +	4.849e+03 +	6.446e+03 +	6.418e+03 +	4.728e+03 +	7.829e+03 +	**3.550e+03**
	std	2.539e+02	1.596e+02	1.658e+02	1.117e+02	1.688e+02	2.904e+02	3.702e+02	7.397e+02	6.613e+02	3.824e+02	1.090e+03	1.457e+02
f_{30}	mean	9.628e+06 +	3.345e+07 +	4.111e+06 +	9.580e+05 +	3.783e+07 +	1.576e+06 +	1.342e+07 +	6.143e+07 +	1.318e+08 +	3.593e+07 +	2.297e+08 +	**9.092e+05**
	std	2.176e+06	5.734e+06	7.871e+05	8.606e+04	1.073e+07	3.853e+05	4.743e+06	2.549e+07	6.301e+07	1.047e+07	6.841e+07	7.287e+04
+ /≈/−:		24/1/4	26/2/1	22/4/3	12/17/0	25/1/3	26/0/3	26/1/2	29/0/0	29/0/0	28/1/0	29/0/0	–
Avg. ranks:		8.0	7.2	3.9	2.9	7.1	5.9	7.2	9.2	9.4	5.4	9.7	**1.8**

Table D.23
Results of comparison experiments in 30-D CEC2020.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	3.102e+09 +	3.890e+07 +	**2.333e+03** ≈	5.947e+03 +	7.978e+08 +	3.585e+06 +	1.825e+08 +	2.429e+09 +	2.903e+09 +	5.977e+07 +	1.742e+08 +	2.497e+03
	std	4.696e+08	8.236e+06	1.667e+03	2.161e+03	1.295e+08	4.131e+06	9.994e+07	2.085e+09	1.940e+09	2.951e+07	6.431e+07	1.622e+03
f_2	mean	3.137e+11 +	5.315e+09 +	3.957e+05 ≈	6.467e+05 +	1.051e+11 +	6.468e+08 +	2.285e+10 +	3.129e+11 +	2.783e+11 +	6.283e+09 +	2.194e+10 +	**2.121e+05**
	std	5.127e+10	1.042e+09	2.963e+05	2.930e+05	1.764e+10	1.645e+09	1.746e+10	3.048e+11	1.575e+11	3.249e+09	8.535e+09	1.900e+05
f_3	mean	9.860e+10 +	1.509e+09 +	8.227e+04 ≈	2.469e+05 +	3.225e+10 +	3.016e+08 +	9.901e+09 +	6.931e+10 +	1.131e+11 +	2.180e+09 +	7.771e+09 +	**7.132e+04**
	std	1.395e+10	2.227e+08	4.626e+04	9.284e+04	7.181e+09	3.996e+08	7.906e+09	4.866e+10	7.259e+10	1.151e+09	3.423e+09	5.243e+04
f_4	mean	2.127e+03 +	1.919e+03 +	1.914e+03 +	1.915e+03 +	1.942e+03 +	1.925e+03 +	1.960e+03 +	2.766e+03 +	2.499e+03 +	1.916e+03 +	1.950e+03 +	**1.904e+03**
	std	1.058e+02	1.379e+00	8.549e−01	8.352e−01	8.894e+00	4.886e+00	4.486e+01	2.381e+03	9.635e+02	2.628e+00	1.542e+01	8.004e−01
f_5	mean	6.409e+05 +	6.367e+05 +	2.739e+05 +	1.302e+05 +	3.483e+06 +	8.842e+04 ≈	1.333e+05 +	6.463e+05 +	1.003e+06 +	8.382e+05 +	6.140e+06 +	**7.442e+04**
	std	2.105e+05	1.892e+05	6.464e+04	3.985e+04	1.098e+06	4.616e+04	5.897e+04	3.613e+05	5.951e+05	5.549e+05	5.216e+06	1.057e+04
f_6	mean	7.308e+03 ≈	1.663e+04 +	1.048e+04 +	8.221e+03 ≈	4.823e+04 +	1.026e+04 +	8.362e+03 +	2.528e+04 +	6.469e+04 +	5.571e+04 +	4.450e+04 +	**6.377e+03**
	std	2.884e+03	3.265e+03	4.002e+03	3.505e+03	2.123e+04	3.498e+03	2.903e+03	1.158e+04	8.925e+04	4.236e+04	2.435e+04	2.290e+03
f_7	mean	5.557e+05 +	6.335e+05 +	3.624e+05 ≈	3.650e+05 ≈	5.348e+06 +	**2.000e+05** −	2.635e+05 ≈	1.172e+06 +	1.359e+06 +	1.704e+06 +	3.379e+06 +	3.001e+05
	std	1.374e+05	1.729e+05	1.070e+05	1.374e+05	2.356e+06	1.056e+05	9.668e+04	7.539e+05	7.416e+05	1.126e+06	2.256e+06	1.242e+05
f_8	mean	2.452e+03 +	2.718e+03 +	2.363e+03 +	2.367e+03 +	2.436e+03 +	5.084e+03 +	5.357e+03 +	2.891e+03 +	2.473e+03 +	2.386e+03 +	2.547e+03 +	**2.347e+03**
	std	7.759e+00	6.166e+02	3.030e+00	3.494e+00	5.285e+00	2.198e+02	2.946e+02	3.831e+02	4.219e+01	8.790e+00	8.244e+01	2.331e+01
f_9	mean	6.334e+03 +	3.169e+03 +	**2.600e+03** ≈	2.604e+03 +	5.469e+03 +	2.788e+03 +	4.125e+03 +	7.555e+03 +	7.327e+03 +	3.220e+03 +	3.445e+03 +	2.600e+03
	std	2.877e+02	9.132e+01	4.581e−02	4.737e−01	4.320e+02	1.852e+02	5.044e+02	2.783e+03	1.834e+03	1.837e+02	2.126e+02	1.806e−01
f_{10}	mean	3.105e+03 +	2.932e+03 +	2.922e+03 +	2.922e+03 +	3.012e+03 +	2.984e+03 +	3.041e+03 +	3.095e+03 +	3.222e+03 +	2.944e+03 +	3.123e+03 +	**2.922e+03**
	std	4.707e+01	2.045e+00	3.540e−01	4.789e−01	2.136e+01	1.598e+01	5.766e+01	1.141e+02	1.431e+02	1.422e+01	1.049e+02	6.666e−01
+ /≈/−:		9/1/0	10/0/0	5/5/0	8/2/0	10/0/0	8/1/1	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	–
Avg. ranks:		8.6	5.8	2.8	3.1	8.8	4.7	7.1	9.8	10.5	6.7	8.7	**1.4**

Table D.24
Results of comparison experiments in 50-D CEC2020.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	1.025e+10 +	2.994e+08 +	**1.086e+03** ≈	2.000e+03 ≈	2.490e+08 +	5.024e+07 +	1.549e+09 +	1.387e+10 +	1.201e+10 +	2.127e+08 +	5.535e+08 +	1.381e+03
	std	2.171e+09	4.582e+07	1.132e+03	2.312e+03	6.902e+07	3.921e+07	8.469e+08	4.746e+09	4.804e+09	7.250e+07	1.690e+08	1.182e+03
f_2	mean	1.183e+12 +	3.231e+10 +	3.744e+05 ≈	4.064e+05 ≈	3.152e+10 +	4.769e+09 +	1.714e+11 +	1.298e+12 +	1.459e+12 +	2.507e+10 +	6.988e+10 +	**3.424e+05**
	std	3.294e+11	5.118e+09	3.671e+05	2.335e+05	8.564e+09	2.428e+09	8.858e+10	5.536e+11	5.361e+11	9.225e+09	3.239e+10	1.600e+05
f_3	mean	4.099e+11 +	9.403e+09 +	4.580e+04 ≈	2.234e+04 ≈	8.627e+09 +	1.197e+09 +	3.979e+10 +	3.968e+11 +	3.439e+11 +	8.205e+09 +	2.338e+10 +	**1.985e+04**
	std	1.036e+11	9.672e+08	7.396e+04	3.029e+04	1.897e+09	1.147e+09	2.486e+10	2.005e+11	1.537e+11	2.701e+09	7.665e+09	2.883e+04
f_4	mean	8.157e+03 +	1.941e+03 +	1.928e+03 +	1.916e+03 +	1.944e+03 +	1.962e+03 +	2.167e+03 +	5.421e+03 +	7.311e+03 +	1.937e+03 +	2.036e+03 +	**1.905e+03**
	std	6.214e+03	2.736e+00	2.609e+00	3.552e+00	3.646e+00	9.634e+00	1.669e+02	3.281e+03	7.027e+03	5.156e+00	6.868e+01	8.331e−01
f_5	mean	2.502e+06 +	3.994e+06 +	1.400e+06 +	1.120e+06 ≈	3.404e+06 +	**7.323e+05** ≈	7.871e+05 ≈	4.664e+06 +	6.709e+06 +	6.227e+06 +	1.214e+07 +	8.617e+05
	std	6.141e+05	9.556e+05	5.829e+05	5.670e+05	1.787e+06	2.283e+05	3.687e+05	2.389e+06	3.573e+06	4.018e+06	5.823e+06	2.810e+05
f_6	mean	3.632e+04 +	4.905e+04 +	5.307e+03 ≈	5.110e+03 ≈	1.133e+05 +	7.084e+03 +	1.301e+04 +	6.569e+05 +	1.747e+06 +	4.318e+05 +	1.114e+06 +	**4.998e+03**
	std	9.732e+03	1.040e+04	2.007e+03	2.162e+03	5.503e+04	1.996e+03	5.613e+03	6.807e+05	2.309e+06	4.834e+05	8.568e+05	1.282e+03
f_7	mean	1.393e+07 +	1.017e+07 +	2.493e+06 +	2.004e+06 ≈	1.358e+07 +	1.965e+06 ≈	2.555e+06 +	1.063e+07 +	1.679e+07 +	1.119e+07 +	3.568e+07 +	**1.846e+06**
	std	5.010e+06	3.043e+06	6.549e+05	7.798e+05	5.173e+06	8.051e+05	1.585e+06	1.039e+07	1.236e+07	6.311e+06	1.707e+07	7.304e+05
f_8	mean	2.650e+03 +	7.136e+03 +	2.398e+03 +	2.395e+03 +	2.509e+03 +	8.472e+03 +	9.026e+03 +	6.441e+03 +	2.886e+03 +	2.475e+03 +	3.115e+03 +	**2.352e+03**
	std	4.045e+01	1.293e+03	4.528e+00	3.845e+00	9.862e+00	4.089e+02	4.339e+02	1.680e+03	1.937e+02	1.545e+01	3.840e+02	3.881e+01
f_9	mean	1.135e+04 +	3.610e+03 +	**2.600e+03** −	2.601e+03 ≈	4.037e+03 +	3.228e+03 +	6.678e+03 +	2.361e+04 +	1.972e+04 +	3.874e+03 +	4.464e+03 +	2.606e+03
	std	1.388e+03	3.906e+02	7.359e−03	5.210e−02	8.451e+02	1.579e+02	1.479e+03	8.992e+03	5.525e+03	2.919e+02	2.791e+02	2.600e+01
f_{10}	mean	4.596e+03 +	3.540e+03 +	**3.330e+03** ≈	3.336e+03 ≈	3.588e+03 +	3.561e+03 +	4.045e+03 +	4.607e+03 +	4.844e+03 +	3.612e+03 +	4.407e+03 +	3.337e+03
	std	3.495e+02	4.371e+01	3.480e+01	4.340e+01	6.155e+01	4.725e+01	2.485e+02	6.573e+02	3.420e+02	1.529e+02	3.624e+02	1.964e+01
+ /≈/−:		10/0/0	10/0/0	4/5/1	2/8/0	10/0/0	8/2/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0	–
Avg. ranks:		9.2	6.6	2.6	2.5	6.6	4.6	7.7	10.2	10.8	6.3	9.2	**1.7**

Table E.25
Results of comparison experiments in 10-D CEC2022.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	4.267e+02 +	3.001e+02 +	3.896e+02 +	3.117e+02 +	8.824e+02 +	4.562e+02 +	3.877e+02 +	3.340e+02 +	4.864e+02 +	3.471e+02 +	3.300e+02 +	**3.000e+02**
	std	4.489e+01	5.615e−02	4.677e+01	5.425e+00	1.547e+02	2.597e+02	1.415e+02	3.445e+01	3.173e+02	3.463e+01	2.704e+01	4.597e−04
f_2	mean	4.149e+02 +	4.109e+02 +	4.076e+02 +	4.115e+02 +	4.790e+02 +	4.290e+02 +	4.405e+02 +	4.212e+02 +	4.271e+02 +	4.242e+02 +	4.426e+02 +	**4.001e+02**
	std	2.379e+00	2.042e+01	4.015e+00	1.353e+01	1.819e+01	3.101e+01	3.179e+01	2.768e+01	3.073e+01	2.775e+01	3.478e+01	1.692e−01
f_3	mean	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	**6.000e+02**
	std	5.867e−04	6.836e−07	1.695e−06	3.515e−06	6.329e−03	1.786e−02	9.314e−03	5.160e−04	3.943e−03	2.100e−04	5.624e−04	7.438e−09
f_4	mean	8.010e+02 +	8.004e+02 +	8.004e+02 +	8.006e+02 +	8.008e+02 +	8.004e+02 +	8.004e+02 +	8.003e+02 +	8.003e+02 +	8.003e+02 +	8.004e+02 +	**8.001e+02**
	std	1.920e−01	4.040e−02	8.212e−02	2.309e−01	1.591e−01	8.085e−02	7.520e−03	2.228e−01	1.617e−01	1.237e−01	1.240e−01	6.040e−02
f_5	mean	9.004e+02 +	9.000e+02 +	9.001e+02 +	9.000e+02 +	9.003e+02 +	9.002e+02 +	9.002e+02 +	9.010e+02 +	9.007e+02 +	9.003e+02 +	9.006e+02 +	**9.000e+02**
	std	9.080e−02	3.621e−05	3.750e−02	4.002e−04	7.500e−02	1.395e−01	9.615e−02	1.112e+00	7.786e−01	3.090e−01	5.553e−01	2.129e−06
f_6	mean	**4.115e+03** −	6.996e+03 −	1.604e+04 +	2.196e+04 +	1.413e+06 +	2.586e+05 +	9.453e+03 ≈	3.518e+04 +	4.015e+04 +	3.971e+04 +	4.492e+04 +	1.009e+04
	std	7.331e+02	2.322e+03	5.423e+03	5.857e+03	1.024e+06	1.068e+06	3.257e+03	1.520e+04	2.064e+04	2.187e+04	1.668e+04	3.093e+03
f_7	mean	2.075e+03 +	2.034e+03 +	2.034e+03 +	2.035e+03 +	2.103e+03 +	2.032e+03 ≈	2.057e+03 +	2.110e+03 +	2.094e+03 +	2.030e+03 ≈	2.079e+03 +	**2.029e+03**
	std	1.399e+01	3.543e+00	2.860e+00	4.327e+00	2.101e+01	5.296e+00	2.848e+01	8.047e+01	4.781e+01	8.248e+00	2.972e+01	2.582e+00
f_8	mean	2.229e+03 −	**2.226e+03** −	2.229e+03 ≈	2.265e+03 +	2.544e+03 +	2.231e+03 ≈	2.228e+03 −	2.573e+03 +	2.486e+03 +	2.250e+03 +	3.130e+03 +	2.233e+03
	std	2.103e+00	2.569e+00	3.963e+00	4.072e+01	2.770e+02	2.155e+01	7.993e+00	5.051e+02	5.228e+02	3.153e+01	6.977e+02	6.696e+00
f_9	mean	2.588e+03 +	2.432e+03 +	2.315e+03 +	2.323e+03 +	2.566e+03 +	2.477e+03 +	2.564e+03 +	2.647e+03 +	2.597e+03 +	2.562e+03 +	2.605e+03 +	**2.300e+03**
	std	1.308e+02	1.798e+02	1.253e+01	8.054e+01	1.151e+02	2.041e+02	1.926e+02	1.182e+02	1.674e+02	1.671e+02	1.935e+02	8.341e−03
f_{10}	mean	2.606e+03 +	3.071e+03 +	2.606e+03 +	2.609e+03 +	2.725e+03 +	3.683e+03 +	3.446e+03 +	2.712e+03 +	2.633e+03 +	2.631e+03 +	2.606e+03 +	**2.600e+03**
	std	9.026e−01	5.750e+02	1.449e+00	2.574e+01	6.225e+01	5.261e+02	5.143e+02	2.772e+02	5.435e+01	5.863e+01	2.329e+00	2.004e+00
f_{11}	mean	2.777e+03 +	2.600e+03 +	2.600e+03 +	2.601e+03 +	2.622e+03 +	2.607e+03 +	2.612e+03 +	2.692e+03 +	2.607e+03 +	2.603e+03 +	2.602e+03 +	**2.600e+03**
	std	3.434e+02	3.335e−02	1.154e−01	1.210e−01	4.884e+00	7.269e+00	2.309e+01	2.630e+02	7.801e+00	3.142e+00	2.119e+00	2.790e−03
f_{12}	mean	2.868e+03 +	2.887e+03 +	**2.867e+03** ≈	2.867e+03 ≈	2.885e+03 +	4.125e+03 +	3.775e+03 +	2.879e+03 +	2.870e+03 ≈	2.867e+03 ≈	2.880e+03 +	2.867e+03
	std	5.323e−01	8.718e+01	5.208e−01	4.206e−01	8.799e+00	3.549e+02	5.746e+02	1.950e+01	1.012e+01	1.372e+00	4.325e+01	6.526e−01
+ /≈/−:		10/0/2	10/0/2	10/2/0	11/1/0	12/0/0	10/2/0	10/1/1	12/0/0	11/1/0	10/2/0	12/0/0	–
Avg. ranks:		7.0	4.2	3.9	5.0	10.3	8.0	7.3	8.3	8.4	5.5	8.2	**1.7**

Table E.26
Results of comparison experiments in 20-D CEC2022.

	Func.	CMA-ES	JADE	L-SHADE	CSO	MPEDE	jSO	L-SHADE-cnEpSin	INFO	SCSO	RIME	PO	L-SHACSO
f_1	mean	2.666e+03 +	4.576e+02 +	3.752e+02 +	3.079e+02 +	5.931e+03 +	4.094e+02 +	1.013e+03 +	1.746e+03 +	3.043e+03 +	5.287e+02 +	6.562e+02 +	**3.000e+02**
	std	3.590e+02	5.353e+01	3.812e+01	4.269e+00	1.327e+03	1.691e+02	4.229e+02	1.174e+03	2.152e+03	1.291e+02	1.887e+02	7.654e−04
f_2	mean	5.101e+02 +	4.567e+02 +	4.528e+02 ≈	4.521e+02 ≈	5.281e+02 +	5.248e+02 +	5.290e+02 +	5.441e+02 +	5.383e+02 +	4.845e+02 +	5.150e+02 +	**4.511e+02**
	std	1.090e+01	1.011e+01	6.968e+00	6.723e+00	1.623e+01	2.932e+01	4.121e+01	7.131e+01	5.631e+01	3.397e+01	4.514e+01	4.997e+00
f_3	mean	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	**6.000e+02**
	std	4.264e−03	4.424e−05	4.468e−08	3.985e−07	1.090e−02	5.186e−04	5.858e−03	1.153e−02	2.909e−02	6.116e−04	1.224e−03	1.222e−09
f_4	mean	8.040e+02 +	8.005e+02 +	8.016e+02 +	8.005e+02 +	8.020e+02 +	8.005e+02 +	8.005e+02 +	8.008e+02 +	8.012e+02 +	8.009e+02 +	8.013e+02 +	**8.003e+02**
	std	3.612e−01	9.317e−02	2.449e−01	2.473e−01	3.548e−01	8.843e−02	1.497e−01	3.136e−01	5.782e−01	1.813e−01	4.903e−01	7.849e−02
f_5	mean	9.032e+02 +	9.015e+02 +	9.005e+02 +	9.000e+02 +	9.016e+02 +	9.020e+02 +	9.019e+02 +	9.029e+02 +	9.023e+02 +	9.014e+02 +	9.030e+02 +	**9.000e+02**
	std	5.582e−01	6.526e−01	1.375e−01	8.750e−05	6.034e−01	6.547e−01	2.054e−01	1.784e+00	1.276e+00	9.316e−01	1.886e+00	6.008e−06
f_6	mean	4.882e+06 +	1.508e+06 +	1.277e+06 +	2.903e+05 +	4.088e+07 +	**2.454e+04** ≈	3.525e+04 +	7.454e+04 +	2.302e+05 +	3.747e+05 +	2.694e+05 +	2.558e+04
	std	2.579e+06	5.285e+05	5.129e+05	1.123e+05	1.615e+07	5.284e+03	6.929e+03	2.219e+04	5.186e+05	3.337e+05	2.127e+05	3.613e+03
f_7	mean	2.301e+03 +	2.112e+03 +	2.091e+03 +	2.041e+03 +	2.271e+03 +	2.170e+03 +	2.318e+03 +	2.513e+03 +	2.269e+03 +	2.078e+03 +	2.443e+03 +	**2.026e+03**
	std	5.996e+01	2.868e+01	1.525e+01	2.928e+00	8.259e+01	1.417e+02	1.658e+02	4.234e+02	1.811e+02	2.431e+01	2.170e+02	1.928e+00
f_8	mean	**2.431e+03** −	2.675e+03 −	2.762e+03 −	3.624e+03 +	3.952e+04 +	2.501e+03 −	2.501e+03 −	3.602e+03 +	4.285e+03 +	5.206e+03 +	4.994e+03 +	3.134e+03
	std	6.664e+01	1.994e+02	2.486e+02	3.409e+02	1.057e+05	2.163e+02	1.711e+02	6.161e+02	1.279e+03	1.983e+03	1.139e+03	2.217e+02
f_9	mean	2.649e+03 +	2.650e+03 +	2.653e+03 +	2.654e+03 +	2.760e+03 +	**2.597e+03** −	2.682e+03 +	2.699e+03 +	2.768e+03 +	2.660e+03 +	2.897e+03 +	2.646e+03
	std	4.166e+00	5.282e+00	5.680e+00	3.248e+00	3.783e+01	1.471e+02	8.934e+01	5.999e+01	8.191e+01	1.416e+01	1.759e+02	2.473e+00
f_{10}	mean	3.861e+03 +	5.613e+03 +	2.804e+03 +	2.788e+03 ≈	2.838e+03 +	5.776e+03 +	5.778e+03 +	4.398e+03 +	3.831e+03 +	3.077e+03 +	2.833e+03 +	**2.787e+03**
	std	1.677e+03	1.502e+02	2.853e+01	4.954e+01	8.677e+01	1.916e+02	2.197e+02	1.150e+03	1.189e+03	6.314e+02	1.086e+02	4.636e+01
f_{11}	mean	2.620e+03 +	2.601e+03 +	2.600e+03 +	2.600e+03 +	2.634e+03 +	2.602e+03 +	2.612e+03 +	2.694e+03 +	2.740e+03 +	2.608e+03 +	2.612e+03 +	**2.600e+03**
	std	4.019e+00	2.408e−01	2.574e−02	2.316e−02	4.821e+00	3.024e+00	6.250e+00	2.928e+02	3.604e+02	5.520e+00	1.044e+01	1.353e−03
f_{12}	mean	2.954e+03 +	5.143e+03 +	**2.947e+03** ≈	2.950e+03 ≈	4.705e+03 +	5.195e+03 +	5.204e+03 +	3.134e+03 +	3.033e+03 +	2.958e+03 +	3.127e+03 +	2.950e+03
	std	3.088e+00	9.311e+01	3.601e+00	6.729e+00	9.579e+02	2.870e+01	3.733e+01	1.281e+02	6.681e+01	1.343e+01	6.961e+01	8.474e+00
+ /≈/−:		11/0/1	11/0/1	9/2/1	9/3/0	12/0/0	9/1/2	11/0/1	12/0/0	12/0/0	12/0/0	12/0/0	–
Avg. ranks:		8.0	5.5	4.2	3.7	9.6	5.4	7.9	8.7	8.8	6.2	8.2	**1.6**

Table F.27
Results of sensitivity analysis in 30-D CEC2017.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	2.575e+03 ≈	1.275e+03	1.752e+03 ≈	8.538e+02	**1.572e+03** ≈	1.255e+03	1.941e+03 ≈	1.365e+03	2.497e+03	1.622e+03
f_3	3.315e+04 ≈	5.504e+03	3.226e+04 ≈	6.665e+03	3.452e+04 ≈	7.868e+03	3.307e+04 ≈	7.384e+03	**3.164e+04**	6.902e+03
f_4	5.156e+02 ≈	2.213e+00	5.132e+02 ≈	8.669e+00	**5.132e+02** ≈	8.114e+00	5.156e+02 ≈	2.052e+00	5.158e+02	2.104e+00
f_5	5.121e+02 ≈	2.739e+00	5.123e+02 ≈	2.295e+00	5.129e+02 ≈	3.775e+00	5.114e+02 ≈	2.856e+00	**5.113e+02**	2.705e+00
f_6	6.000e+02 ≈	9.032e−04	**6.000e+02** −	9.252e−04	6.000e+02 ≈	1.160e−03	6.000e+02 ≈	1.580e−03	6.000e+02	1.305e−03
f_7	7.404e+02 ≈	6.297e+00	**7.396e+02** ≈	4.914e+00	7.396e+02 ≈	2.593e+00	7.427e+02 ≈	4.410e+00	7.522e+02	1.884e+01
f_8	8.116e+02 ≈	2.684e+00	8.119e+02 ≈	2.783e+00	8.123e+02 ≈	3.135e+00	**8.115e+02** ≈	2.638e+00	8.118e+02	2.890e+00
f_9	**9.000e+02** ≈	1.099e−05	9.000e+02 ≈	1.341e−05	9.000e+02 ≈	2.008e−04	9.000e+02 ≈	1.951e−02	9.000e+02	3.189e−05
f_{10}	2.646e+03 ≈	4.153e+02	2.566e+03 ≈	3.699e+02	2.624e+03 ≈	4.410e+02	**2.512e+03** ≈	3.844e+02	2.578e+03	3.821e+02
f_{11}	1.190e+03 ≈	2.934e+01	1.201e+03 ≈	4.686e+01	1.206e+03 +	5.423e+01	1.190e+03 ≈	3.037e+01	**1.178e+03**	2.593e+01
f_{12}	**6.262e+05** ≈	2.792e+05	7.749e+05 ≈	4.278e+05	6.630e+05 ≈	2.938e+05	7.617e+05 ≈	3.410e+05	7.131e+05	3.105e+05
f_{13}	**9.615e+03** ≈	2.888e+03	1.004e+04 ≈	2.979e+03	1.008e+04 ≈	3.090e+03	1.144e+04 ≈	4.396e+03	1.129e+04	2.627e+03
f_{14}	1.135e+05 ≈	6.462e+04	**1.113e+05** ≈	3.978e+04	1.180e+05 ≈	5.697e+04	1.399e+05 ≈	8.538e+04	1.191e+05	7.403e+04
f_{15}	2.294e+03 ≈	8.345e+02	**1.950e+03** ≈	3.776e+02	2.211e+03 ≈	7.166e+02	2.219e+03 ≈	5.985e+02	1.995e+03	5.094e+02
f_{16}	**1.943e+03** ≈	1.659e+02	1.965e+03 ≈	1.827e+02	1.988e+03 ≈	2.346e+02	1.954e+03 ≈	1.501e+02	1.947e+03	1.908e+02
f_{17}	**1.763e+03** ≈	4.058e+01	1.797e+03 ≈	8.444e+01	1.808e+03 ≈	7.556e+01	1.791e+03 ≈	8.881e+01	1.790e+03	7.607e+01
f_{18}	**2.466e+05** ≈	1.646e+05	3.024e+05 ≈	1.861e+05	3.588e+05 ≈	2.246e+05	3.682e+05 ≈	2.097e+05	3.843e+05	2.209e+05
f_{19}	**4.559e+03** ≈	1.024e+03	4.655e+03 ≈	1.077e+03	4.932e+03 ≈	1.336e+03	4.991e+03 ≈	1.509e+03	4.594e+03	1.198e+03
f_{20}	2.159e+03 +	6.516e+00	2.158e+03 +	5.953e+00	2.157e+03 +	6.260e+00	2.168e+03 ≈	4.552e+01	**2.153e+03**	6.374e+00
f_{21}	2.312e+03 ≈	4.942e+00	2.312e+03 ≈	4.600e+00	2.313e+03 ≈	6.018e+00	2.312e+03 ≈	4.173e+00	**2.310e+03**	2.363e+00
f_{22}	**2.300e+03** ≈	4.060e−03	2.300e+03 ≈	6.865e−03	2.300e+03 ≈	9.153e−03	2.300e+03 ≈	7.197e−03	2.300e+03	9.366e−03
f_{23}	2.663e+03 ≈	7.986e+00	2.665e+03 ≈	7.698e+00	2.664e+03 ≈	6.698e+00	**2.659e+03** ≈	6.874e+00	2.662e+03	5.139e+00
f_{24}	2.826e+03 ≈	3.247e+00	2.827e+03 ≈	3.908e+00	**2.825e+03** ≈	3.799e+00	2.825e+03 ≈	4.845e+00	2.825e+03	4.362e+00
f_{25}	2.889e+03 +	2.480e+00	2.890e+03 +	2.507e+00	2.889e+03 +	1.838e+00	2.888e+03 ≈	2.069e+00	**2.888e+03**	1.427e+00
f_{26}	3.692e+03 ≈	2.333e+02	3.532e+03 ≈	4.319e+02	3.630e+03 ≈	3.614e+02	3.723e+03 +	2.396e+02	**3.523e+03**	3.657e+02
f_{27}	3.223e+03 +	6.992e+00	3.221e+03 ≈	5.490e+00	3.225e+03 +	6.609e+00	3.222e+03 ≈	7.392e+00	**3.219e+03**	6.588e+00
f_{28}	3.244e+03 +	1.971e+01	3.252e+03 +	1.709e+01	3.244e+03 +	1.803e+01	3.241e+03 +	1.583e+01	**3.225e+03**	1.608e+01
f_{29}	**3.388e+03** ≈	4.895e+01	3.416e+03 ≈	7.869e+01	3.406e+03 ≈	6.980e+01	3.409e+03 ≈	8.273e+01	3.402e+03	8.793e+01
f_{30}	8.819e+03 +	1.617e+03	8.158e+03 ≈	1.005e+03	8.919e+03 +	1.348e+03	8.157e+03 ≈	1.127e+03	**7.791e+03**	7.588e+02
+ /≈/−:	5/24/0		3/25/1		6/23/0		2/27/0		–	
Avg. ranks:	2.7		2.9		3.5		3.3		**2.5**	

Table F.28
Results of sensitivity analysis in 50-D CEC2017.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	1.784e+03 ≈	1.644e+03	1.375e+03 ≈	1.316e+03	**1.134e+03** ≈	9.970e+02	1.920e+03 ≈	1.854e+03	1.381e+03	1.182e+03
f_3	8.976e+04 ≈	1.387e+04	8.894e+04 ≈	1.249e+04	8.773e+04 ≈	1.238e+04	**8.097e+04** ≈	1.103e+04	8.211e+04	1.152e+04
f_4	5.821e+02 ≈	3.420e+01	5.851e+02 ≈	3.475e+01	5.775e+02 ≈	3.083e+01	**5.662e+02** ≈	3.286e+01	5.810e+02	2.970e+01
f_5	5.257e+02 ≈	3.975e+00	5.262e+02 +	4.669e+00	5.261e+02 +	5.282e+00	5.258e+02 +	3.581e+00	**5.233e+02**	3.710e+00
f_6	6.000e+02 ≈	3.014e−04	**6.000e+02** ≈	3.510e−04	6.000e+02 ≈	4.214e−04	6.000e+02 ≈	5.959e−04	6.000e+02	6.336e−04
f_7	7.647e+02 −	2.086e+00	**7.639e+02** −	2.574e+00	7.644e+02 −	3.761e+00	7.670e+02 ≈	6.851e+00	7.682e+02	5.000e+00
f_8	8.264e+02 +	4.002e+00	8.270e+02 +	5.697e+00	8.272e+02 +	4.710e+00	8.275e+02 +	5.172e+00	**8.234e+02**	4.811e+00
f_9	9.003e+02 +	2.947e−01	9.002e+02 ≈	5.946e−01	**9.001e+02** ≈	1.568e−01	9.002e+02 ≈	2.185e−01	9.002e+02	1.940e−01
f_{10}	3.406e+03 ≈	5.509e+02	3.268e+03 ≈	5.110e+02	3.500e+03 ≈	6.030e+02	3.324e+03 ≈	5.266e+02	**3.232e+03**	5.056e+02
f_{11}	1.953e+03 ≈	4.095e+02	1.930e+03 ≈	4.149e+02	**1.779e+03** ≈	2.631e+02	1.861e+03 ≈	2.211e+02	1.842e+03	3.768e+02
f_{12}	2.335e+06 ≈	7.470e+05	2.085e+06 ≈	5.486e+05	2.336e+06 ≈	9.203e+05	2.339e+06 ≈	7.491e+05	**1.902e+06**	5.854e+05
f_{13}	2.920e+03 +	9.764e+02	3.025e+03 +	9.091e+02	2.628e+03 +	1.040e+03	2.535e+03 +	8.028e+02	**2.074e+03**	5.063e+02
f_{14}	3.444e+05 ≈	1.690e+05	4.136e+05 ≈	2.576e+05	4.030e+05 ≈	1.929e+05	4.079e+05 ≈	1.971e+05	**3.402e+05**	1.792e+05
f_{15}	**4.836e+03** −	1.226e+03	6.294e+03 ≈	1.611e+03	6.225e+03 ≈	1.554e+03	6.078e+03 ≈	1.627e+03	6.631e+03	1.759e+03
f_{16}	2.168e+03 ≈	1.546e+02	2.208e+03 ≈	1.895e+02	**2.109e+03** ≈	1.326e+02	2.182e+03 ≈	1.989e+02	2.156e+03	1.557e+02
f_{17}	2.147e+03 ≈	1.933e+02	2.227e+03 +	1.804e+02	2.143e+03 ≈	1.498e+02	2.156e+03 +	1.598e+02	**2.066e+03**	1.314e+02
f_{18}	1.694e+06 ≈	4.448e+05	1.840e+06 ≈	5.100e+05	1.730e+06 ≈	5.006e+05	1.844e+06 ≈	7.509e+05	**1.635e+06**	5.152e+05
f_{19}	1.646e+04 ≈	3.296e+03	1.626e+04 ≈	2.840e+03	**1.589e+04** ≈	2.429e+03	1.686e+04 ≈	3.042e+03	1.688e+04	2.732e+03
f_{20}	2.186e+03 ≈	1.557e+02	2.196e+03 ≈	1.259e+02	2.183e+03 ≈	1.275e+02	2.231e+03 +	1.607e+02	**2.164e+03**	1.203e+02
f_{21}	2.325e+03 +	3.756e+00	2.326e+03 +	4.281e+00	2.324e+03 +	3.007e+00	2.324e+03 ≈	2.999e+00	**2.322e+03**	3.176e+00
f_{22}	2.300e+03 ≈	8.467e−03	**2.300e+03** ≈	1.195e−02	2.300e+03 ≈	1.494e−02	2.449e+03 ≈	6.498e+02	2.300e+03	1.071e−02
f_{23}	2.760e+03 ≈	9.607e+00	2.759e+03 ≈	1.049e+01	2.758e+03 ≈	1.314e+01	2.753e+03 ≈	8.932e+00	**2.753e+03**	1.259e+01
f_{24}	2.917e+03 +	8.837e+00	2.917e+03 +	7.105e+00	2.914e+03 +	7.178e+00	2.913e+03 ≈	9.384e+00	**2.908e+03**	7.549e+00
f_{25}	3.101e+03 +	1.675e+01	3.106e+03 +	1.644e+01	3.103e+03 +	1.630e+01	3.098e+03 ≈	1.370e+01	**3.088e+03**	1.911e+01
f_{26}	4.140e+03 +	1.206e+02	4.302e+03 +	2.136e+02	4.193e+03 +	2.091e+02	4.236e+03 +	2.843e+02	**3.984e+03**	3.075e+02
f_{27}	3.346e+03 +	3.052e+01	3.350e+03 +	3.014e+01	3.350e+03 +	2.929e+01	3.348e+03 +	3.065e+01	**3.313e+03**	2.284e+01
f_{28}	3.409e+03 +	3.068e+01	3.400e+03 +	1.532e+01	3.393e+03 +	1.268e+01	3.384e+03 +	2.614e+01	**3.361e+03**	1.398e+01
f_{29}	3.632e+03 ≈	1.663e+02	3.648e+03 ≈	1.553e+02	3.546e+03 ≈	1.664e+02	**3.544e+03** ≈	1.713e+02	3.550e+03	1.457e+02
f_{30}	1.068e+06 +	9.361e+04	1.091e+06 +	1.346e+05	1.010e+06 +	8.002e+04	9.849e+05 +	5.905e+04	**9.092e+05**	7.287e+04
+ /≈/−:	10/17/2		11/17/1		10/18/1		9/20/0		–	
Avg. ranks:	3.3		3.8		2.7		3.3		**1.9**	

Table F.29
Results of sensitivity analysis in 30-D CEC2020.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	2.575e+03 ≈	1.275e+03	1.752e+03 ≈	8.538e+02	**1.572e+03** ≈	1.255e+03	1.941e+03 ≈	1.365e+03	2.497e+03	1.622e+03
f_2	2.287e+05 ≈	1.505e+05	2.348e+05 ≈	1.800e+05	2.361e+05 ≈	1.289e+05	2.278e+05 ≈	1.419e+05	**2.121e+05**	1.900e+05
f_3	7.440e+04 ≈	4.503e+04	6.356e+04 ≈	4.786e+04	6.890e+04 ≈	5.228e+04	**5.521e+04** ≈	3.988e+04	7.132e+04	5.243e+04
f_4	**1.904e+03** ≈	6.762e−01	1.904e+03 ≈	8.088e−01	1.904e+03 ≈	1.149e+00	1.904e+03 ≈	1.575e+00	1.904e+03	8.004e−01
f_5	7.662e+04 ≈	2.004e+04	**7.076e+04** ≈	1.595e+04	7.417e+04 ≈	1.483e+04	7.164e+04 ≈	1.643e+04	7.442e+04	1.057e+04
f_6	6.428e+03 ≈	2.335e+03	7.832e+03 ≈	2.648e+03	7.179e+03 ≈	2.524e+03	7.732e+03 ≈	3.295e+03	**6.377e+03**	2.290e+03
f_7	3.130e+05 ≈	1.175e+05	3.087e+05 ≈	9.361e+04	**2.558e+05** ≈	6.849e+04	2.985e+05 ≈	9.104e+04	3.001e+05	1.242e+05
f_8	2.357e+03 ≈	1.331e+01	2.359e+03 ≈	3.197e+00	2.357e+03 ≈	1.334e+01	2.360e+03 +	3.096e+00	**2.347e+03**	2.331e+01
f_9	**2.600e+03** −	6.861e−02	2.600e+03 −	1.150e−01	2.600e+03 −	9.730e−02	2.600e+03 ≈	1.407e−01	**2.600e+03**	1.806e−01
f_{10}	2.922e+03 +	6.343e−01	2.922e+03 +	1.049e+00	2.923e+03 +	1.994e+00	2.922e+03 +	1.834e+00	2.922e+03	6.666e−01
+ /≈/−:	1/8/1		1/8/1		1/8/1		2/8/0		–	
Avg. ranks:	3.2		3.0		3.0		3.0		**2.8**	

Table F.30
Results of sensitivity analysis in 50-D CEC2020.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	1.784e+03 ≈	1.644e+03	1.375e+03 ≈	1.316e+03	**1.134e+03** ≈	9.970e+02	1.920e+03 ≈	1.854e+03	1.381e+03	1.182e+03
f_2	**3.154e+05** ≈	1.945e+05	3.409e+05 ≈	1.765e+05	3.527e+05 ≈	1.719e+05	3.516e+05 ≈	1.491e+05	3.424e+05	1.600e+05
f_3	2.500e+04 ≈	2.285e+04	2.173e+04 ≈	2.160e+04	**1.344e+04** ≈	1.429e+04	3.089e+04 ≈	2.897e+04	1.985e+04	2.883e+04
f_4	1.905e+03 ≈	8.198e−01	**1.904e+03** −	5.933e−01	1.904e+03 −	6.096e−01	1.904e+03 ≈	5.530e−01	1.905e+03	8.331e−01
f_5	1.094e+06 +	2.987e+05	9.920e+05 ≈	4.130e+05	1.089e+06 +	4.251e+05	1.141e+06 +	4.522e+05	**8.617e+05**	2.810e+05
f_6	5.693e+03 ≈	1.362e+03	5.112e+03 ≈	1.135e+03	5.686e+03 ≈	1.475e+03	**4.718e+03** ≈	1.111e+03	4.998e+03	1.282e+03
f_7	2.230e+06 +	6.558e+05	2.248e+06 +	5.658e+05	2.303e+06 +	7.297e+05	2.285e+06 +	7.779e+05	**1.846e+06**	7.304e+05
f_8	2.390e+03 +	3.863e+00	2.389e+03 +	4.492e+00	2.381e+03 +	2.162e+01	2.360e+03 ≈	3.995e+01	**2.352e+03**	3.881e+01
f_9	**2.600e+03** ≈	6.433e−02	2.600e+03 ≈	5.938e−02	2.606e+03 ≈	2.605e+01	2.606e+03 ≈	2.544e+01	2.606e+03	2.600e+01
f_{10}	3.374e+03 +	2.210e+01	3.384e+03 +	5.072e+01	3.373e+03 +	2.453e+01	3.358e+03 +	2.282e+01	**3.337e+03**	1.964e+01
+ /≈/−:	6/4/0		3/6/1		4/5/1		3/7/0		–	
Avg. ranks:	3.4		2.7		3.1		3.4		**2.4**	

Table F.31
Results of sensitivity analysis in 10-D CEC2022.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	3.000e+02 ≈	8.065e−04	3.000e+02 ≈	1.237e−03	3.000e+02 ≈	6.468e−04	3.000e+02 ≈	7.656e−04	**3.000e+02**	4.597e−04
f_2	4.003e+02 ≈	9.829e−01	**4.001e+02** ≈	1.232e−01	4.001e+02 ≈	1.913e−01	4.001e+02 ≈	1.122e−01	4.001e+02	1.692e−01
f_3	**6.000e+02** ≈	4.723e−09	6.000e+02 ≈	2.221e−08	6.000e+02 ≈	7.747e−09	6.000e+02 ≈	1.815e−08	6.000e+02	7.438e−09
f_4	8.002e+02 ≈	1.119e−01	8.002e+02 +	1.739e−01	8.002e+02 ≈	1.228e−01	8.002e+02 +	1.382e−01	**8.001e+02**	6.040e−02
f_5	9.000e+02 ≈	1.808e−06	**9.000e+02** ≈	9.668e−07	9.000e+02 ≈	4.029e−06	9.000e+02 ≈	1.510e−06	9.000e+02	2.129e−06
f_6	9.731e+03 ≈	1.539e+03	9.393e+03 ≈	1.842e+03	9.951e+03 ≈	3.818e+03	**9.283e+03** ≈	3.078e+03	1.009e+04	3.093e+03
f_7	2.030e+03 ≈	2.598e+00	2.029e+03 ≈	2.738e+00	**2.029e+03** ≈	3.169e+00	2.030e+03 ≈	3.139e+00	2.029e+03	2.582e+00
f_8	2.236e+03 ≈	1.052e+01	2.239e+03 ≈	1.375e+01	**2.233e+03** ≈	7.005e+00	2.241e+03 ≈	1.561e+01	2.233e+03	6.696e+00
f_9	**2.300e+03** ≈	6.518e−03	2.300e+03 ≈	1.184e−02	2.300e+03 ≈	8.455e−03	2.300e+03 ≈	8.426e−03	2.300e+03	8.341e−03
f_{10}	2.605e+03 ≈	2.647e+01	2.617e+03 +	4.223e+01	2.605e+03 ≈	2.601e+01	**2.599e+03** ≈	7.104e−01	2.600e+03	2.004e+00
f_{11}	**2.600e+03** ≈	2.640e−03	2.600e+03 ≈	4.808e−03	2.600e+03 ≈	3.303e−03	2.600e+03 ≈	4.647e−03	2.600e+03	2.790e−03
f_{12}	2.867e+03 ≈	4.809e−01	2.867e+03 ≈	1.072e+00	2.867e+03 ≈	6.870e−01	**2.867e+03** ≈	1.927e−01	2.867e+03	6.526e−01
+ /≈/−:	0/12/0		2/10/0		0/12/0		1/11/0		–	
Avg. ranks:	2.9		3.8		2.9		3.1		**2.3**	

Table F.32
Results of sensitivity analysis in 20-D CEC2022.

Func.	L-SHACSO (μ_φ=0.1)		L-SHACSO (μ_φ=0.15)		L-SHACSO (μ_φ=0.2)		L-SHACSO (μ_φ=0.25)		L-SHACSO (μ_φ=0.3)	
	Mean	std	Mean	std	Mean	std	Mean	std	Mean	std
f_1	3.000e+02 ≈	3.768e−04	3.000e+02 ≈	4.821e−04	3.000e+02 ≈	5.977e−04	**3.000e+02** ≈	5.438e−04	3.000e+02	7.654e−04
f_2	4.534e+02 +	7.058e+00	4.525e+02 +	5.503e+00	4.520e+02 +	5.480e+00	4.515e+02 ≈	4.211e+00	**4.511e+02**	4.997e+00
f_3	**6.000e+02** ≈	4.864e−10	6.000e+02 ≈	7.269e−10	6.000e+02 ≈	1.565e−09	6.000e+02 ≈	1.445e−09	6.000e+02	1.222e−09
f_4	8.003e+02 ≈	8.128e−02	8.003e+02 ≈	1.277e−01	8.003e+02 ≈	1.252e−01	8.003e+02 ≈	1.650e−01	**8.003e+02**	7.849e−02
f_5	**9.000e+02** ≈	4.086e−06	9.000e+02 ≈	6.429e−06	9.000e+02 ≈	6.470e−06	9.000e+02 ≈	1.657e−05	9.000e+02	6.008e−06
f_6	2.720e+04 ≈	3.578e+03	**2.524e+04** ≈	2.492e+03	2.605e+04 ≈	3.963e+03	2.603e+04 ≈	3.406e+03	2.558e+04	3.613e+03
f_7	2.025e+03 ≈	1.118e+00	**2.025e+03** ≈	1.569e+00	2.026e+03 ≈	1.386e+00	2.026e+03 ≈	1.335e+00	2.026e+03	1.928e+00
f_8	3.141e+03 ≈	2.352e+02	3.175e+03 ≈	2.415e+02	**3.075e+03** ≈	2.544e+02	3.161e+03 ≈	2.922e+02	3.134e+03	2.217e+02
f_9	2.651e+03 +	2.899e+00	2.651e+03 +	4.990e+00	2.652e+03 +	4.174e+00	2.648e+03 +	2.498e+00	**2.646e+03**	2.473e+00
f_{10}	2.771e+03 ≈	3.271e+01	2.782e+03 ≈	4.450e+01	**2.770e+03** ≈	3.400e+01	2.776e+03 ≈	4.024e+01	2.787e+03	4.636e+01
f_{11}	2.600e+03 ≈	5.679e−04	**2.600e+03** ≈	7.563e−04	2.600e+03 ≈	9.706e−04	2.600e+03 ≈	1.144e−03	2.600e+03	1.353e−03
f_{12}	**2.949e+03** ≈	6.292e+00	2.954e+03 ≈	7.835e+00	2.952e+03 ≈	7.986e+00	2.952e+03 ≈	7.154e+00	2.950e+03	8.474e+00
+ /≈/−:	2/10/0		2/10/0		2/10/0		1/11/0		–	
Avg. ranks:	**2.4**		3.3		3.0		3.3		3.0	

Data availability

The source code of this research can be downloaded from https://github.com/RuiZhong961230/L-SHACSO.

References

[1] T. Pratap, P. Kokil, Computer-aided diagnosis of cataract using deep transfer learning, Biomed. Signal Process. Control 53 (2019) 101533, http://dx.doi.org/10.1016/j.bspc.2019.04.010.

[2] M. Sudhan, M. Sinthuja, S. Pravinth Raja, J. Amutharaj, G. Charlyn Pushpa Latha, S. Sheeba Rachel, T. Anitha, T. Rajendran, Y.A. Waji, Segmentation and classification of glaucoma using U-net with deep learning model, J. Healthc. Eng. 2022 (1) (2022) 1601354, http://dx.doi.org/10.1155/2022/1601354.

[3] K. Uyar, M. Yurdakul, Ş. Taşdemir, Abc-based weighted voting deep ensemble learning model for multiple eye disease detection, Biomed. Signal Process. Control 96 (2024) 106617, http://dx.doi.org/10.1016/j.bspc.2024.106617.

[4] P. Hijma, S. Heldens, A. Sclocco, B. van Werkhoven, H.E. Bal, Optimization techniques for GPU programming, ACM Comput. Surv. 55 (11) (2023) http://dx.doi.org/10.1145/3570638.

[5] A. Mohammadi, F. Sheikholeslam, Intelligent optimization: Literature review and state-of-the-art algorithms (1965–2022), Eng. Appl. Artif. Intell. 126 (2023) 106959, http://dx.doi.org/10.1016/j.engappai.2023.106959.

[6] L. Pinciroli, P. Baraldi, E. Zio, Maintenance optimization in industry 4.0, Reliab. Eng. Syst. Saf. 234 (2023) 109204, http://dx.doi.org/10.1016/j.ress.2023.109204.

[7] M. Thirunavukkarasu, Y. Sawle, H. Lala, A comprehensive review on optimization of hybrid renewable energy systems using various optimization techniques, Renew. Sustain. Energy Rev. 176 (2023) 113192, http://dx.doi.org/10.1016/j.rser.2023.113192.

[8] R. Zhong, Q. Fan, C. Zhang, J. Yu, Hybrid remora crayfish optimization for engineering and wireless sensor network coverage optimization, Cluster Comput. (2024) 1–28, http://dx.doi.org/10.1007/s10586-024-04508-1.

[9] J. Shi, Y. Chen, C. Wang, A. Asghar Heidari, L. Liu, H. Chen, X. Chen, L. Sun, Multi-threshold image segmentation using new strategies enhanced whale optimization for lupus nephritis pathological images, Displays (2024) 102799, http://dx.doi.org/10.1016/j.displa.2024.102799.

[10] M. Han, Z. Du, K.F. Yuen, H. Zhu, Y. Li, Q. Yuan, Walrus optimizer: A novel nature-inspired metaheuristic algorithm, Expert Syst. Appl. 239 (2024) 122413, http://dx.doi.org/10.1016/j.eswa.2023.122413.

[11] M.A. Al-Betar, M.A. Awadallah, M.S. Braik, S. Makhadmeh, I.A. Doush, Elk herd optimizer: a novel nature-inspired metaheuristic algorithm, Artif. Intell. Rev. 57 (3) (2024) 48, http://dx.doi.org/10.1007/s10462-023-10680-4.

[12] W. chuan Wang, W. can Tian, D. mei Xu, H. fei Zang, Arctic puffin optimization: A bio-inspired metaheuristic algorithm for solving engineering design optimization, Adv. Eng. Softw. 195 (2024) 103694, http://dx.doi.org/10.1016/j.advengsoft.2024.103694.

[13] B. Alatas, ACROA: Artificial chemical reaction optimization algorithm for global optimization, Expert Syst. Appl. 38 (10) (2011) 13170–13180, http://dx.doi.org/10.1016/j.eswa.2011.04.126.

[14] S. Bechikh, A. Chaabani, L. Ben Said, An efficient chemical reaction optimization algorithm for multiobjective optimization, IEEE Trans. Cybern. 45 (10) (2015) 2051–2064, http://dx.doi.org/10.1109/TCYB.2014.2363878.

[15] A. Qi, D. Zhao, A.A. Heidari, L. Liu, Y. Chen, H. Chen, FATA: An efficient optimization method based on geophysics, Neurocomputing (2024) 128289, http://dx.doi.org/10.1016/j.neucom.2024.128289.

[16] R. Zhong, J. Yu, C. Zhang, M. Munetomo, SRIME: a strengthened RIME with latin hypercube sampling and embedded distance-based selection for engineering optimization problems, Neural Comput. Appl. 36 (12) (2024) 6721–6740, http://dx.doi.org/10.1007/s00521-024-09424-4.

[17] K. Rajwar, K. Deep, S. Das, An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges, Artif. Intell. Rev. 56 (11) (2023) 13187–13257, http://dx.doi.org/10.1007/s10462-023-10470-y.

[18] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204, http://dx.doi.org/10.1109/TCYB.2014.2322602.

[19] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948, http://dx.doi.org/10.1109/ICNN.1995.488968.

[20] X. Wang, K. Zhang, J. Wang, Y. Jin, An enhanced competitive swarm optimizer with strongly convex sparse operator for large-scale multiobjective optimization, IEEE Trans. Evol. Comput. 26 (5) (2022) 859–871, http://dx.doi.org/10.1109/TEVC.2021.3111209.

[21] J. Dong, W. Gong, F. Ming, A tri-stage competitive swarm optimizer for constrained multi-objective optimization, Appl. Intell. 53 (7) (2023) 7892–7916, http://dx.doi.org/10.1007/s10489-022-03874-w.

[22] R. Lan, Y. Zhu, H. Lu, Z. Liu, X. Luo, A two-phase learning-based swarm optimizer for large-scale optimization, IEEE Trans. Cybern. 51 (12) (2021) 6284–6293, http://dx.doi.org/10.1109/TCYB.2020.2968400.

[23] T. Ma, Y. Wang, X. Li, Convex combination multiple populations competitive swarm optimization for moving target search using UAVs, Inform. Sci. 641 (2023) 119104, http://dx.doi.org/10.1016/j.ins.2023.119104.

[24] Y. Tian, X. Zheng, X. Zhang, Y. Jin, Efficient large-scale multiobjective optimization based on a competitive swarm optimizer, IEEE Trans. Cybern. 50 (8) (2020) 3696–3708, http://dx.doi.org/10.1109/TCYB.2019.2906383.

[25] S. Qi, J. Zou, S. Yang, Y. Jin, J. Zheng, X. Yang, A self-exploratory competitive swarm optimization algorithm for large-scale multiobjective optimization, Inform. Sci. 609 (2022) 1601–1620, http://dx.doi.org/10.1016/j.ins.2022.07.110.

[26] F. Xue, T. Dong, S. You, Y. Liu, H. Tang, L. Chen, X. Yang, J. Li, A hybrid many-objective competitive swarm optimization algorithm for large-scale multirobot task allocation problem, Int. J. Mach. Learn. Cybern. 12 (2021) 1–15, http://dx.doi.org/10.1007/s13042-020-01213-4.

[27] M. Qaraad, A. Aljadania, M. Elhosseini, Large-scale competitive learning-based salp swarm for global optimization and solving constrained mechanical and engineering design problems, Mathematics 11 (6) (2023) http://dx.doi.org/10.3390/math11061362.

[28] R. Tanabe, A.S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: 2014 IEEE Congress on Evolutionary Computation, CEC, 2014, pp. 1658–1665, http://dx.doi.org/10.1109/CEC.2014.6900380.

[29] J. Brest, M.S. Maučec, B. Bošković, Single objective real-parameter optimization: Algorithm JSO, in: 2017 IEEE Congress on Evolutionary Computation, CEC, 2017, pp. 1311–1318, http://dx.doi.org/10.1109/CEC.2017.7969456.

[30] N.H. Awad, M.Z. Ali, P.N. Suganthan, Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving CEC2017 benchmark problems, in: 2017 IEEE Congress on Evolutionary Computation, CEC, 2017, pp. 372–379, http://dx.doi.org/10.1109/CEC.2017.7969336.

[31] H. Su, D. Zhao, A.A. Heidari, L. Liu, X. Zhang, M. Mafarja, H. Chen, RIME: A physics-based optimization, Neurocomputing 532 (2023) 183–214, http://dx.doi.org/10.1016/j.neucom.2023.02.010.

[32] J. Lian, G. Hui, L. Ma, T. Zhu, X. Wu, A.A. Heidari, Y. Chen, H. Chen, Parrot optimizer: Algorithm and applications to medical problems, Comput. Biol. Med. 172 (2024) 108064, http://dx.doi.org/10.1016/j.compbiomed.2024.108064.

[33] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman, S. Vimal, 25 years of particle swarm optimization: Flourishing voyage of two decades, Arch. Comput. Methods Eng. 30 (3) (2023) 1663–1725, http://dx.doi.org/10.1007/s11831-022-09849-x.

[34] P. Mohapatra, K. Nath Das, S. Roy, A modified competitive swarm optimizer for large scale optimization problems, Appl. Soft Comput. 59 (2017) 340–362, http://dx.doi.org/10.1016/j.asoc.2017.05.060.

[35] C. Huang, X. Zhou, X. Ran, Y. Liu, W. Deng, W. Deng, Co-evolutionary competitive swarm optimizer with three-phase for large-scale complex optimization problem, Inform. Sci. 619 (2023) 2–18, http://dx.doi.org/10.1016/j.ins.2022.11.019.

[36] J.-S. Pan, Q. Liang, S.-C. Chu, K.-K. Tseng, J. Watada, A multi-strategy surrogate-assisted competitive swarm optimizer for expensive optimization problems, Appl. Soft Comput. 147 (2023) 110733, http://dx.doi.org/10.1016/j.asoc.2023.110733.

[37] W. Huang, W. Zhang, Multi-objective optimization based on an adaptive competitive swarm optimizer, Inform. Sci. 583 (2022) 266–287, http://dx.doi.org/10.1016/j.ins.2021.11.031.

[38] Y. Ge, D. Chen, F. Zou, M. Fu, F. Ge, Large-scale multiobjective optimization with adaptive competitive swarm optimizer and inverse modeling, Inform. Sci. 608 (2022) 1441–1463, http://dx.doi.org/10.1016/j.ins.2022.07.018.

[39] S. Liu, Q. Lin, Q. Li, K.C. Tan, A comprehensive competitive swarm optimizer for large-scale multiobjective optimization, IEEE Trans. Syst. Man Cybern.: Syst. 52 (9) (2022) 5829–5842, http://dx.doi.org/10.1109/TSMC.2021.3131312.

[40] D. Chauhan, Shivani, R. Cheng, Competitive swarm optimizer: A decade survey, Swarm Evol. Comput. 87 (2024) 101543, http://dx.doi.org/10.1016/j.swevo.2024.101543.

[41] R. Tanabe, A. Fukunaga, How far are we from an optimal, adaptive DE? in: Parallel Problem Solving from Nature – PPSN XIV, Springer International Publishing, Cham, 2016, pp. 145–155.

[42] A.W. Mohamed, A.A. Hadi, K.M. Jambi, Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization, Swarm Evol. Comput. 50 (2019) 100455, http://dx.doi.org/10.1016/j.swevo.2018.10.006.

[43] P. Kolenovsky, P. Bujok, An adaptive variant of jSO with multiple crossover strategies employing eigen transformation, in: 2022 IEEE Congress on Evolutionary Computation, CEC, 2022, pp. 1–8, http://dx.doi.org/10.1109/CEC55065.2022.9870378.

[44] J. Zhang, A.C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958, http://dx.doi.org/10.1109/TEVC.2009.2014613.

[45] Y. Li, S. Wang, B. Yang, H. Chen, Z. Wu, H. Yang, Population reduction with individual similarity for differential evolution, Artif. Intell. Rev. 56 (5) (2023) 3887–3949, http://dx.doi.org/10.1007/s10462-022-10264-8.

[46] X. Zhang, Q. Liu, Y. Qu, An adaptive differential evolution algorithm with population size reduction strategy for unconstrained optimization problem, Appl. Soft Comput. 138 (2023) 110209, http://dx.doi.org/10.1016/j.asoc.2023.110209.
[47] Y. Sun, K. Zhang, Z. Li, Z. Liu, A cascaded differential evolution optimization framework with adaptive population allocation and reduction, Swarm Evol. Comput. 82 (2023) 101376, http://dx.doi.org/10.1016/j.swevo.2023.101376.
[48] M. Srinivas, L. Patnaik, Genetic algorithms: a survey, Computer 27 (6) (1994) 17–26, http://dx.doi.org/10.1109/2.294849.
[49] R. Storn, K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.
[50] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey Wolf Optimizer, Adv. Eng. Softw. 69 (2014) 46–61, http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.
[51] T. Nguyen, A framework of optimization functions using Numpy (OpFuNu) for optimization problems, 2020, http://dx.doi.org/10.5281/zenodo.3620960, Zenodo.
[52] N.V. Thieu, ENOPPY: A Python library for engineering optimization problems, 2023, http://dx.doi.org/10.5281/zenodo.7953206, Zenodo.
[53] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001) 159–195, http://dx.doi.org/10.1162/106365601750190398.
[54] G. Wu, R. Mallipeddi, P. Suganthan, R. Wang, H. Chen, Differential evolution with multi-population based ensemble of mutation strategies, Inform. Sci. 329 (2016) 329–345, http://dx.doi.org/10.1016/j.ins.2015.09.009, Special issue on Discovery Science.
[55] I. Ahmadianfar, A.A. Heidari, S. Noshadian, H. Chen, A.H. Gandomi, INFO: An efficient optimization algorithm based on weighted mean of vectors, Expert Syst. Appl. 195 (2022) 116516, http://dx.doi.org/10.1016/j.eswa.2022.116516.
[56] A. Seyyedabbasi, F. Kiani, Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems, Eng. Comput. 39 (4) (2023) 2627–2651, http://dx.doi.org/10.1007/s00366-022-01604-x.
[57] N. Van Thieu, S. Mirjalili, MEALPY: An open-source library for latest meta-heuristic algorithms in Python, J. Syst. Archit. 139 (2023) 102871, http://dx.doi.org/10.1016/j.sysarc.2023.102871.
[58] K. Zhou, S.-K. Oh, W. Pedrycz, J. Qiu, Data preprocessing strategy in constructing convolutional neural network classifier based on constrained particle swarm optimization with fuzzy penalty function, Eng. Appl. Artif. Intell. 117 (2023) 105580, http://dx.doi.org/10.1016/j.engappai.2022.105580.
[59] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82, http://dx.doi.org/10.1109/4235.585893.
[60] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), in: Advances in Neural Information Processing Systems, vol. 25, Curran Associates, Inc., 2012.
[61] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, 2019, arXiv:1801.04381.
[62] G. Huang, Z. Liu, L. van der Maaten, K.Q. Weinberger, Densely connected convolutional networks, 2018, arXiv:1608.06993.
[63] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2015, arXiv:1512.03385.
[64] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2015, arXiv:1409.1556.
[65] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, (no. 1) 2017.
[66] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, 32, Curran Associates, Inc., 2019.
[67] H. Bayzidi, S. Talatahari, M. Saraee, C.-P. Lamarche, Social network search for solving engineering optimization problems, Comput. Intell. Neurosci. 2021 (1) (2021) 8548639, http://dx.doi.org/10.1155/2021/8548639.