Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Multi-Strategy Enhanced Artificial Lemming Optimization Algorithm for Engineering Optimization Problems

Jia Fan

[illegible]

Teng Gao

189456484@qq.com

[illegible]

Research Article

Keywords: Hybrid Meta-heuristic, Artificial Lemming Algorithm, Population Diversity, Robustness, Engineering Optimization

Posted Date: May 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9548142/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Multi-Strategy Enhanced Artificial Lemming Optimization Algorithm for Engineering Optimization Problems

Abstract

To address the limitations of the Artificial Lemming Algorithm (ALA) in complex continuous optimization problems, such as uneven population initialization, slow convergence, insufficient global search capability, and susceptibility to local optima, this paper proposes a multi-strategy enhanced Historical-Cognitive Hybrid Artificial Lemming Algorithm (HCALA). A unified enhanced optimization framework is established, systematically integrating multiple improvement strategies to boost overall algorithm performance. Building upon ALA's original global exploration and local exploitation capabilities, HCALA introduces historical memory mechanisms, dynamic selection strategies, and a late-stage hybrid exploitation approach to increase population diversity, optimize search paths, and improve the reliability of search directions, thereby enhancing adaptability and stability in high-dimensional and multimodal continuous optimization tasks.To validate its effectiveness, extensive experiments are conducted on 30 benchmark functions, as well as the CEC2017 and CEC2022 test suites. Moreover, HCALA is applied to typical engineering optimization problems. The results reveal that the proposed approach consistently surpasses existing advanced algorithms in convergence precision, stability, and robustness, and shows considerable adaptability and applicability in real-world engineering optimization scenarios.

Keywords: Hybrid Meta-heuristic, Artificial Lemming Algorithm, Population Diversity, Robustness, Engineering Optimization

1 Introduction

Meta-heuristic algorithms represent a class of optimization methods inspired by heuristic principles, which are well-suited for addressing uncertainty and complexity in the search process. Due to their flexibility and robustness, they have been widely utilized in solving diverse and challenging optimization problems. Typical representatives include the Wild Horse Optimizer (WHO) [1], Artificial Hummingbird Algorithm (AHA) [2], Dragonfly Algorithm (DA) [3], Chimp Optimization Algorithm (ChOA) [4], and Sand Cat Swarm Optimization (CSO) [5]. These methods are notable for their simple structures, computational efficiency, and minimal parameter requirements.

In addition, algorithms inspired by mathematical formulations have also shown promising performance. For example, the Sine Cosine Algorithm (SCA) [6] can be easily embedded into other optimization frameworks as an operator, which constitutes its major advantage. Physics-based algorithms, derived from physical laws through mathematical modeling, also play an important role, including Simulated Annealing (SA) [7] and the Gravitational Search Algorithm (GSA) [8]. These methods have demonstrated effectiveness in various applications such as path planning [9], structural design [10], unmanned aerial vehicle (UAV) systems [11], image matching [12], and power system control [13]. However, as stated by the No Free Lunch (NFL) theorem [14], no single optimization algorithm can consistently achieve the best performance across all problem domains. Algorithms with relatively simple mechanisms, such as PSO and DA, are particularly suitable for specific tasks (e.g., PID controller parameter tuning), where they offer ease of implementation, low computational cost, and satisfactory accuracy. Nevertheless, when dealing with high-dimensional problems and complex constraints, these simple models may fail to provide satisfactory solutions, which necessitates the development of more advanced intelligent optimization algorithms. This limitation has motivated continuous research toward more efficient and adaptive optimization strategies.

The Artificial Lemming Algorithm (ALA), proposed by Xiao [15] in 2025, is a bio-inspired meta-heuristic algorithm. It models four typical behaviors of lemmings—long-distance migration, burrowing, foraging, and predator avoidance—to effectively explore complex search spaces. In addition, ALA introduces an energy attenuation mechanism to dynamically balance exploration and exploitation, thereby enhancing its ability to escape from local optima. Although ALA offers benefits such as structural simplicity, a limited number of control parameters, and straightforward implementation, it exhibits certain shortcomings when tackling complex optimization tasks. In practical engineering design tasks, ALA tends to exhibit slow convergence in the early iterations and is prone to being trapped in local optima in the later stages. Moreover, the algorithm is sensitive to parameter settings and excessively relies on a single best individual to guide the search process. This leads to premature population convergence, weakens global exploration capability, and reduces the probability of finding the global optimum. These drawbacks become more pronounced in high-dimensional or multimodal optimization problems, thereby limiting its overall performance.

To overcome these shortcomings, numerous studies have focused on enhancing meta-heuristic algorithms through various strategies. Although these methods differ in their implementation mechanisms, they share a common objective: to improve population diversity, accelerate convergence, and maintain a proper balance between exploration and exploitation. For instance, He et al. [16] employed Sobol sequences for more uniform population initialization and incorporated a convex lens-based opposition learning mechanism to improve convergence speed. Wang et al. [17] introduced a role-based particle classification strategy combined with adaptive parameter tuning to enhance search efficiency. Li et al. [18] proposed a hybrid Grey Wolf Optimizer integrating teaching–learning strategies with PSO concepts to strengthen both global

and local search capabilities. In another study, Li et al. [19] improved SSA by adopting a refraction-based opposition learning strategy, effectively enhancing population diversity and expanding the search space. Hou et al. [20] utilized chaotic mapping for initialization and combined it with Gaussian mutation to better coordinate exploration and exploitation. Houssein et al. [21] introduced a dynamic opposition-based learning strategy during initialization to increase population diversity and prevent premature convergence. Lv et al. [22] developed a chaotic SSA variant to improve the ability to escape local optima. Safinaz et al. [23] proposed a binary clone flower pollination algorithm by integrating clone selection with pollination behavior to mitigate premature convergence. Lu et al. [24] incorporated Lévy flight perturbations along with a greedy selection strategy, effectively enhancing the ability to escape local optima. Peng et al. [25] improved the Grasshopper Optimization Algorithm by introducing position and velocity correction factors to enhance search performance. Overall, these studies indicate that multi-strategy integration has become an effective paradigm for improving optimization algorithms.

Despite these advancements, existing approaches still lack sufficient adaptability when addressing diverse and complex problem landscapes. To further improve the performance of ALA in continuous optimization problems and overcome its inherent deficiencies—such as uneven population distribution, slow convergence, limited accuracy, and susceptibility to local optima—this paper proposes a multi-strategy enhanced Historical-Cognitive Hybrid Artificial Lemming Algorithm (HCALA). The principal contributions and innovations of this study are summarized as follows:

- A dual-memory guidance mechanism is developed by combining an external elite historical archive with individual historical best information. This design enhances the reliability of search directions and strengthens local exploitation capability.
- A success-history-driven adaptive branch selection strategy is introduced, in which the selection probability of different search behaviors is dynamically adjusted according to their past performance. This mechanism improves resource allocation and reduces the risk of premature convergence.
- A hybrid exploitation strategy for the later stage is proposed, incorporating structured mutation, recombination, and adaptive local perturbation. By assigning higher weights to high-quality search behaviors, the algorithm can more effectively escape local optima and achieve higher solution precision.

The paper is structured as follows: Sections 1–2 introduce the background and preliminaries, Section 3 presents the HCALA algorithm, Sections 4–5 report benchmark experiments and engineering applications, Section 6 applies HCALA to UAV path planning, and Section 7 concludes with future directions.

2 Preliminaries

This section introduces the essential background required for the proposed HCALA. To maintain a clear and rigorous exposition, we first review the original Artificial Lemming Algorithm (ALA), including its biological inspiration, mathematical formulation, and overall workflow. Then, from the perspective of complex continuous

optimization, we summarize the main motivation for improving ALA and clarify why a historical-cognitive hybrid enhancement is necessary.

Unless otherwise stated, the optimization tasks considered in this paper are treated as minimization problems of the form

$$\min_{\mathbf{X}\in\Omega} f(\mathbf{X}), \qquad \Omega = \{\mathbf{X} \in \mathbb{R}^D \mid \mathbf{lb} \leq \mathbf{X} \leq \mathbf{ub}\}, \tag{1}$$

where $f(\cdot)$ denotes the objective function and Ω is the feasible search domain. Throughout the paper, the term *fitness* refers to the objective value under the above minimization setting, so a smaller fitness value indicates a better solution.

2.1 Artificial Lemming Algorithm

The Artificial Lemming Algorithm (ALA) is a bio-inspired meta-heuristic that models four representative survival behaviors of lemmings in nature, namely long-distance migration, digging holes, foraging, and evading predators. In natural environments, these behaviors are closely associated with habitat relocation, shelter construction, local food acquisition, and survival under predation pressure. Inspired by this process, ALA establishes a four-branch search framework, in which long-distance migration and digging holes mainly contribute to global exploration, whereas foraging and predator evasion mainly contribute to local exploitation. In addition, an energy-decreasing mechanism is introduced to regulate the transition between the two search phases during optimization.

Population initialization: As a population-based optimizer, ALA begins by generating an initial population within the feasible search domain. Let $\mathbf{X}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}]$ denote the position of the ith lemming individual in a D-dimensional search space, where $\mathbf{lb} = [lb_1, lb_2, \ldots, lb_D]$ and $\mathbf{ub} = [ub_1, ub_2, \ldots, ub_D]$ are the lower and upper bounds, respectively. The initialization rule is given by

$$x_{i,j}^0 = lb_j + r_{i,j}(ub_j - lb_j), \tag{2}$$

where $r_{i,j} \in (0, 1)$ is a uniformly distributed random number. After initialization, the fitness of all individuals is evaluated, and the current global-best solution is recorded as $\mathbf{X}_{best}^0 = \arg\min_{1\leq i\leq N} f(\mathbf{X}_i^0)$.

Behavior 1: Long-distance migration (Exploration): When food resources become insufficient, lemmings may migrate over long distances in search of more favorable habitats. Such movement is highly stochastic and is influenced by both environmental disturbance and interactions with neighboring individuals. In ALA, this phenomenon is used to model a long-range exploratory behavior, formulated as

$$\mathbf{X}_i^{t+1} = \mathbf{X}_{best}^t + F\,\mathbf{BM}_i^t \odot \left[\mathbf{r}_1 \odot (\mathbf{X}_{best}^t - \mathbf{X}_i^t) + (\mathbf{1} - \mathbf{r}_1) \odot (\mathbf{X}_i^t - \mathbf{X}_a^t)\right], \tag{3}$$

where $\odot$ denotes element-wise multiplication, $\mathbf{X}_a^t$ is a randomly selected individual, $\mathbf{r}_1 \in [-1, 1]^D$ is a random vector, and $F \in \{-1, 1\}$ is a directional flag. $\mathbf{BM}_i^t$ denotes a D-dimensional Brownian-motion increment vector, which introduces stochastic step perturbation and thus enhances the roaming ability of the population in the search

space. By jointly involving the current global best solution, the current individual, and a random peer, this behavior encourages broad exploration while preserving population diversity.

Behavior 2: Digging holes (Exploration): Lemmings dig holes and construct tunnel systems to obtain shelter and reserve resources. Compared with migration, this behavior is more localized, but it still belongs to the exploration phase because it does not directly force the individual to converge to the best position. In ALA, the digging-hole behavior is modeled as

$$\mathbf{X}_i^{t+1} = \mathbf{X}_i^t + F\, r_2^t \left(\mathbf{X}_{best}^t - \mathbf{X}_b^t\right), \tag{4}$$

where $\mathbf{X}_b^t$ is another randomly selected individual and r_2^t is a time-varying random coefficient defined by

$$r_2^t = \xi_2^t \left(1 + \sin(0.5t)\right), \qquad \xi_2^t \sim U(0,1). \tag{5}$$

The sinusoidal modulation changes the movement amplitude over the search process and helps the algorithm generate diversified exploratory steps.

Behavior 3: Foraging (Exploitation): When lemmings move within a favorable habitat, they search for food around promising local regions. This behavior mainly serves local exploitation. In ALA, foraging is modeled by a spiral search around the current global best solution:

$$\mathbf{X}_i^{t+1} = \mathbf{X}_{best}^t + F\, spiral_i^t\, \xi_3^t\, \mathbf{X}_i^t, \tag{6}$$

where $\xi_3^t \sim U(0,1)$ and the spiral factor is given by

$$spiral_i^t = \left\|\mathbf{X}_{best}^t - \mathbf{X}_i^t\right\|_2 \left[\sin(2\pi r_3) + \cos(2\pi r_3)\right], \tag{7}$$

and $r_3 \in (0,1)$ is a random number. Here $spiral_i^t$ is a scalar and $\|\cdot\|_2$ denotes the Euclidean norm. The spiral movement enables each individual to search around the best region from different directions, thereby improving local refinement.

Behavior 4: Predator evasion (Exploitation): When facing predators, lemmings usually escape through rapid and irregular movement. This behavior provides another exploitation-oriented operator, but unlike foraging, it also retains a certain ability to jump out of deceptive local regions. In ALA, predator evasion is described as

$$\mathbf{X}_i^{t+1} = \mathbf{X}_{best}^t + F\, G^t\, \mathbf{L}^t \odot \left(\mathbf{X}_{best}^t - \mathbf{X}_i^t\right), \tag{8}$$

where G^t is an escape coefficient and $\mathbf{L}^t$ is a Lévy-flight vector. Following the original implementation, the escape coefficient is defined as

$$G^t = 2\,\mathrm{sign}(r_4 - 0.5)\left(1 - \frac{t}{T_{\max}}\right), \tag{9}$$

where $r_4 \in (0,1)$ is a random number. Therefore, the magnitude of G^t gradually decreases with the iteration process, allowing predator-evasion behavior to shift from coarser perturbation to more localized exploitation in the later stage. The Lévy-flight step is defined as

$$\mathbf{L}^t = 0.01 \times \frac{\mathbf{u}\sigma}{|\mathbf{v}|^{1/\beta}}, \qquad \beta = 1.5, \qquad \sigma = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right)\beta 2^{(\beta-1)/2}} \right)^{1/\beta}, \tag{10}$$

where $\mathbf{u}$ and $\mathbf{v}$ are D-dimensional standard Gaussian random vectors, $\Gamma(\cdot)$ denotes the Gamma function, and the absolute value, division, and power operations are applied element-wise. Because Lévy flight combines many short steps with occasional long jumps, this behavior helps the algorithm maintain local exploitation while preserving escape capability.

Energy-decreasing mechanism: The four behaviors in ALA are coordinated by an iteration-dependent energy factor, which controls whether the algorithm emphasizes exploration or exploitation at the current stage. The energy factor is defined as

$$E(t) = 2\theta(t)\ln\left(\frac{1}{\xi_E^t}\right), \qquad \theta(t) = 2\arctan\left(1 - \frac{t}{T_{\max}}\right), \tag{11}$$

where $\xi_E^t \sim U(0,1)$, t is the current iteration number, and $T_{\max}$ is the maximum number of iterations. As the search proceeds, $\theta(t)$ gradually decreases, and the search tendency shifts from exploration to exploitation. Specifically, when $E(t) > 1$, ALA enters the exploration phase and executes either long-distance migration or digging holes; otherwise, it enters the exploitation phase and executes either foraging or predator evasion.

Greedy selection and workflow: After a new candidate position is generated, ALA performs boundary repair and evaluates the new solution. If a newly generated candidate exceeds the feasible search range, its violated components are corrected by projecting them back to the corresponding lower or upper bounds. If the new candidate is better than its parent, the parent is replaced. Meanwhile, the current global best solution $\mathbf{X}_{best}$ is updated whenever a superior individual is found. Therefore, ALA can be viewed as a four-behavior population-based search framework coordinated by an energy-decreasing controller.

In summary, the pseudocode of ALA is presented in Algorithm 1.

2.2 Research Background and Motivation for Improving ALA

Although ALA provides a concise four-behavior search framework, its performance may still deteriorate on complex continuous optimization problems with high dimensionality, multimodality, and irregular landscape structures. In such cases, the algorithm is expected not only to maintain sufficient search diversity in the early stage, but also to reuse valuable search information across generations, adapt its behavioral preference according to the online optimization status, and provide reliable local refinement in the later stage. However, the original ALA mainly relies on the

Algorithm 1 Pseudocode of ALA

Require: Population size N, dimension D, maximum iteration $T_{\max}$, lower bound $\mathbf{lb}$, upper bound $\mathbf{ub}$
Ensure: Best solution $\mathbf{X}_{best}$

function ALA($N, D, T_{\max}, \mathbf{lb}, \mathbf{ub}$)
  Initialize the population $\mathbf{X}$ using Eq. (2)
  Evaluate all individuals and record $\mathbf{X}_{best}$
  for $t = 1$ to $T_{\max}$ **do**
    Compute $\theta(t)$ and $E(t)$ using Eq. (11)
    for $i = 1$ to N **do**
      if $E(t) > 1$ **then**
        if $rand < 0.3$ **then**
          Generate $\mathbf{X}_i^{t+1}$ using Eq. (3)
        else
          Generate $\mathbf{X}_i^{t+1}$ using Eqs. (4)–(5)
        end if
      else
        if $rand < 0.5$ **then**
          Generate $\mathbf{X}_i^{t+1}$ using Eqs. (6)–(7)
        else
          Generate $\mathbf{X}_i^{t+1}$ using Eqs. (8)–(10)
        end if
      end if
      Repair boundary violations and evaluate the offspring
      Apply greedy selection and update $\mathbf{X}_{best}$
    end for
  end for
  return $\mathbf{X}_{best}$
end function

current population state and the current global best solution, which may limit cross-generational information reuse, adaptive coordination among behavioral branches, and terminal exploitation accuracy on difficult landscapes.

Recent improvements to ALA have mainly focused on three aspects of the search process: enhancing the quality of population initialization, strengthening the reuse of useful search information, and improving late-stage refinement. Existing studies have shown that a more uniform initial population can improve early search diversity, that historical or elite information can enhance the continuity of guidance during evolution, and that hybrid or local refinement operators can further improve convergence accuracy in the later stage. These advances confirm that ALA can benefit substantially from multi-strategy enhancement. However, the existing improvements are still relatively fragmented, and a more coordinated framework that jointly exploits persistent historical guidance, online branch adaptation, and structured late-stage exploitation remains insufficiently explored.

These observations motivate the proposed Historical-Cognitive Hybrid Artificial Lemming Algorithm (HCALA). The basic idea is to retain the original four-behavior search backbone of ALA while introducing three tightly coupled enhancements, namely

persistent historical guidance, online branch self-adaptation, and stronger late-stage hybrid exploitation.

3 Our Proposal: Historical-Cognitive Hybrid Artificial Lemming Algorithm

This section presents the proposed Historical-Cognitive Hybrid Artificial Lemming Algorithm (HCALA). Although the original ALA provides a concise four-behavior search backbone, its performance on difficult continuous optimization problems may still be limited by three closely related issues: insufficient cross-generational reuse of valuable search information, inflexible allocation of search effort among different behavioral branches, and inadequate terminal refinement in the later stage. Accordingly, HCALA is designed around three tightly coupled components, namely elite historical archive-guided dual-memory learning, success-memory-driven branch self-adaptation, and a late-stage hybrid exploitation kernel.

The overall workflow of HCALA is as follows. After initialization, the algorithm records the historical-best solution of each individual and constructs an elite historical archive. During the main search stage, the original ALA behaviors are retained, but each newly generated candidate is further corrected by dual-memory guidance, and the branch probabilities are updated online according to success feedback. When the search enters the later stage, a dedicated hybrid exploitation kernel is activated on elite individuals to reinforce structured recombination and terminal refinement. In this way, HCALA forms a coherent optimization pipeline from historical guidance, to adaptive branch coordination, and finally to late-stage hybrid exploitation.

For convenience, the normalized search progress is defined as

$$\tau_t = \frac{t}{T_{\max}}, \tag{12}$$

where t is the current iteration number and $T_{\max}$ is the maximum number of iterations.

3.1 Elite Historical Archive-Guided Dual-Memory Learning

The first difficulty addressed by HCALA is guidance continuity. In the original ALA, the search is mainly driven by the current population and the current best solution. Although this design is simple, it does not explicitly preserve useful structural information discovered in previous iterations. As a result, once the population moves away from a historically promising region, valuable search experience may be discarded. Meanwhile, excessive reliance on the current best solution may induce overly concentrated attraction, thereby weakening diversity and reducing guidance stability on complex landscapes.

To alleviate this limitation, HCALA introduces a dual-memory learning mechanism that combines population-level historical guidance with individual-level historical guidance. At the population level, an elite historical archive is maintained to preserve high-quality solutions across iterations. At the individual level, each search agent records its own historical-best position. Initially, $\mathbf{P}_i^0 = \mathbf{X}_i^0$. At iteration t, the current

global-best solution is defined by

$$\mathbf{X}_{best}^{t} = \arg\min_{1\le i\le N} f(\mathbf{P}_i^t), \tag{13}$$

that is, it is the best element in the current historical-best memory. The archive capacity $A_{\max}$ and elite subset size N_e are listed in Table 1.

Let $\mathcal{A}^t$ denote the archive at iteration t, and let $\mathcal{E}_X^t$ and $\mathcal{E}_P^t$ denote the elite subsets extracted from the current population and the historical-best memory, respectively. HCALA updates the archive by

$$\mathcal{A}^{t+1} = \mathrm{Top}_{A_{\max}}\left(\mathcal{A}^t \cup \mathcal{E}_X^t \cup \mathcal{E}_P^t\right), \tag{14}$$

where $\mathrm{Top}_{A_{\max}}(\cdot)$ retains the best $A_{\max}$ solutions according to ascending fitness.

Suppose $\mathcal{A}^t = \{\mathbf{A}_m^t\}_{m=1}^{M_t}$, where $M_t = |\mathcal{A}^t|$. Let $f_{\min}^t = \min_{1\le m\le M_t} f(\mathbf{A}_m^t)$. HCALA computes the inverse-shift archive weights and the weighted archive mean by

$$w_m^t = \frac{\left(f(\mathbf{A}_m^t) - f_{\min}^t + \varepsilon\right)^{-1}}{\sum_{q=1}^{M_t}\left(f(\mathbf{A}_q^t) - f_{\min}^t + \varepsilon\right)^{-1}}, \qquad \bar{\mathbf{A}}^t = \sum_{m=1}^{M_t} w_m^t \mathbf{A}_m^t, \tag{15}$$

where ε is a small positive constant.

Rather than directly using the archive mean as the sole search target, HCALA samples one archive elite $\mathbf{A}_s^t$ according to the probability distribution $\{w_m^t\}_{m=1}^{M_t}$ and constructs the archive guidance point by

$$\mathbf{G}^t = \eta_t\, \mathbf{A}_s^t + (1-\eta_t)\, \bar{\mathbf{A}}^t, \tag{16}$$

where the stage-dependent schedule of η_t is given in Table 1. For each individual, the historical-best memory is updated by greedy improvement, that is, $\mathbf{P}_i^{t+1}$ is replaced by $\mathbf{X}_i^{t+1}$ only if $f(\mathbf{X}_i^{t+1}) < f(\mathbf{P}_i^t)$.

Let $\mathbf{X}_{i,\mathrm{base}}^{t+1}$ denote the candidate generated by one of the four ALA behaviors. HCALA further corrects it through dual-memory guidance as

$$\mathbf{X}_{i,\mathrm{cand}}^{t+1} = \mathbf{X}_{i,\mathrm{base}}^{t+1} + k_{\mathrm{arch}}^t\left(\mathbf{G}^t - \mathbf{X}_i^t\right) + k_{\mathrm{pbest}}^t\left(\mathbf{P}_i^t - \mathbf{X}_i^t\right), \tag{17}$$

where the schedules of k_{arch}^t and k_{pbest}^t are listed in Table 1. The correction terms are anchored at the current parent $\mathbf{X}_i^t$ rather than at $\mathbf{X}_{i,\mathrm{base}}^{t+1}$ so that the archive-guided and personal-best-guided displacements can be interpreted as historical directional cues relative to the present search state. In this way, the archive guidance preserves population-level historical structure, whereas the personal-best memory retains individualized successful experience. Their combination improves guidance continuity without forcing the population to collapse prematurely toward a single attractor.

If a generated candidate violates the search boundary, HCALA applies reflection-type correction followed by clipping. For each dimension j, where x_j denotes the jth

component of $\mathbf{X}_{i,\mathrm{cand}}^{t+1}$,

$$\tilde{x}_j = \begin{cases} lb_j + 0.5(lb_j - x_j), & x_j < lb_j, \\ ub_j - 0.5(x_j - ub_j), & x_j > ub_j, \\ x_j, & \text{otherwise}, \end{cases} \qquad x_j^{\mathrm{rep}} = \min\{\max\{\tilde{x}_j, lb_j\}, ub_j\}. \tag{18}$$

The repaired candidate is denoted by $\mathbf{X}_{i,\mathrm{rep}}^{t+1}$.

The rationale behind this component is that archive guidance preserves population-level historical structure, whereas individual historical-best memory preserves personalized successful experience. Their joint action allows HCALA to reuse historical information from complementary perspectives without collapsing prematurely toward a single attractor.

3.2 Success-Memory-Driven Branch Self-Adaptation

The second difficulty addressed by HCALA is branch-level search allocation. In the original ALA, the energy mechanism determines whether the search enters exploration or exploitation, but the selection of the internal search branch remains fixed within each phase. In practice, however, different branches may exhibit different usefulness at different stages and on different landscapes. A fixed branch-allocation rule may therefore spend a non-negligible amount of search budget on relatively unproductive behaviors.

To improve stage-wise branch coordination, HCALA introduces a success-memory-driven branch self-adaptation mechanism. The four ALA behaviors are denoted as Branch A (long-distance migration), Branch B (digging holes), Branch C (foraging), and Branch D (predator evasion). The initial exploration-phase and exploitation-phase branch probabilities are both set to $[0.5, 0.5]$, as summarized in Table 1.

Whenever a candidate generated by one branch is accepted, the corresponding branch receives a success gain measured by

$$g_i^t = f(\mathbf{X}_i^t) - f(\mathbf{X}_{i,\mathrm{cand}}^{t+1}) + \varepsilon, \tag{19}$$

where ε is a small positive constant. Thus, a larger g_i^t corresponds to a larger accepted improvement under the minimization setting. Let $\mathcal{I}_A^t$, $\mathcal{I}_B^t$, $\mathcal{I}_C^t$, and $\mathcal{I}_D^t$ denote the accepted offspring sets generated by Branch A, B, C, and D, respectively. HCALA accumulates the branch success statistics as

$$S_A^t = \varepsilon + \sum_{i \in \mathcal{I}_A^t} g_i^t, \quad S_B^t = \varepsilon + \sum_{i \in \mathcal{I}_B^t} g_i^t, \quad S_C^t = \varepsilon + \sum_{i \in \mathcal{I}_C^t} g_i^t, \quad S_D^t = \varepsilon + \sum_{i \in \mathcal{I}_D^t} g_i^t. \tag{20}$$

The learned preferences are then estimated by

$$\hat{p}_A^t = \frac{S_A^t}{S_A^t + S_B^t}, \qquad \hat{p}_C^t = \frac{S_C^t}{S_C^t + S_D^t}. \tag{21}$$

In addition, the clipping operator used below is defined by

$$\mathrm{clip}_{[a,b]}(x) = \min\{\max\{x, a\}, b\}. \tag{22}$$

Instead of introducing separate equations for prior preference, prior–feedback fusion, clipping, and complementary probabilities, HCALA directly updates the primary branch probabilities in the two phases by

$$p_{\mathrm{exp},1}^{t+1} = \mathrm{clip}_{[p_{\min},p_{\max}]}\Big[(1-\rho)p_{\mathrm{exp},1}^{t} + \rho\Big(0.5(0.65 - 0.30\tau_t) + 0.5\hat{p}_A^t\Big)\Big], \tag{23}$$

$$p_{\mathrm{dev},1}^{t+1} = \mathrm{clip}_{[p_{\min},p_{\max}]}\Big[(1-\rho)p_{\mathrm{dev},1}^{t} + \rho\Big(0.5(0.30 + 0.45\tau_t) + 0.5\hat{p}_C^t\Big)\Big], \tag{24}$$

where ρ, $p_{\min}$, and $p_{\max}$ are listed in Table 1. The complementary probabilities are then set by $p_{\mathrm{exp},2}^{t+1} = 1 - p_{\mathrm{exp},1}^{t+1}$ and $p_{\mathrm{dev},2}^{t+1} = 1 - p_{\mathrm{dev},1}^{t+1}$.

Consequently, the proposed success-memory mechanism transforms branch selection from a fixed rule into an adaptive decision process. The first component stabilizes *where* candidate solutions are guided, whereas the present component improves *how* search effort is allocated during the generation of those candidates. In other words, branch selection becomes jointly determined by recent success statistics and the expected role of each branch at the current stage, rather than by a fixed phase-internal probability.

3.3 Late-Stage Hybrid Exploitation Kernel

The third difficulty addressed by HCALA is terminal refinement. Although the original ALA can gradually shift toward exploitation through the energy mechanism, its later-stage search still mainly depends on the original foraging and predator-evasion behaviors. On difficult landscapes, this may be insufficient to provide the structured refinement required for high-accuracy convergence. In particular, the original local search lacks explicit elite recombination and targeted terminal polishing. To strengthen exploitation in the final stage, HCALA introduces a dedicated late-stage hybrid exploitation kernel that operates on the elite subset.

Let $\mathcal{E}^t = \{\mathbf{X}_{(1)}^t, \ldots, \mathbf{X}_{(N_e)}^t\}$ denote the elite subset containing the best N_e individuals in ascending order of fitness. The kernel is activated when the normalized progress exceeds the threshold specified in Table 1, and its execution frequency also follows the schedule listed there. The elite dispersion is estimated by

$$\boldsymbol{\sigma}_e^t = \max\big(\mathrm{Std}(\mathcal{E}^t),\, \sigma_{\min}\big), \qquad \sigma_{\min} = 10^{-3}(\mathbf{ub} - \mathbf{lb}), \tag{25}$$

where $\mathrm{Std}(\mathcal{E}^t)$ returns the component-wise standard-deviation vector of the elite subset and $\max(\cdot,\cdot)$ is applied element-wise. This dispersion estimate provides a scale-adaptive local search range for the later-stage kernel.

In each kernel call, a target solution $\mathbf{X}_{\mathrm{tar}}^t$ is selected from the elite subset. Meanwhile, three parent solutions $\mathbf{X}_1^t$, $\mathbf{X}_2^t$, and $\mathbf{X}_3^t$ are chosen from elite individuals; if the number of distinct elite samples is insufficient or repeated selections occur, fallback

substitution by the current global-best solution or the archive mean is used. HCALA first generates a mutant vector by

$$\mathbf{V}^t = \mathbf{X}^t_{\text{tar}} + F_1^t \left(\mathbf{X}^t_{best} - \mathbf{X}^t_{\text{tar}}\right) + F_2^t \left(\mathbf{X}^t_1 - \mathbf{X}^t_2\right) + 0.08 \left(\bar{\mathbf{A}}^t - \mathbf{X}^t_{\text{tar}}\right), \tag{26}$$

where the sampling ranges of F_1^t and F_2^t are given in Table 1. This mutation step jointly exploits current-best attraction, elite-difference perturbation, and archive guidance, thereby producing a more structured search direction than ordinary late-stage random perturbation.

After mutation, HCALA performs binomial crossover between the target vector and the mutant vector. Let $\mathbf{U}^t$ denote the trial vector. For each dimension j,

$$u_j^t = \begin{cases} v_j^t, & \text{if } r_j \le CR_K \text{ or } j = j_{\text{rand}}, \\ x^t_{\text{tar},j}, & \text{otherwise}, \end{cases} \tag{27}$$

where v_j^t is the jth component of $\mathbf{V}^t$, $r_j \sim U(0,1)$, j_{rand} is a randomly chosen index that guarantees at least one inherited mutant dimension, and CR_K is listed in Table 1.

After crossover, HCALA performs terminal refinement around the trial vector:

$$\begin{aligned} \mathbf{X}^t_{\text{child}} = \mathbf{U}^t &+ \alpha_t \boldsymbol{\sigma}_e^t \odot \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &+ 0.10 \left(\mathbf{X}^t_{best} - \mathbf{U}^t\right) \\ &+ 0.04 \left(\bar{\mathbf{A}}^t - \mathbf{U}^t\right) \\ &+ 0.04 \left(\mathbf{X}^t_1 - \mathbf{X}^t_3\right) \end{aligned} \tag{28}$$

where the schedule of α_t is listed in Table 1, $\mathcal{N}(\mathbf{0}, \mathbf{I})$ denotes a D-dimensional standard Gaussian random vector, and $\odot$ denotes element-wise multiplication. Thus, terminal refinement is not purely random, but jointly driven by current-best information, archive guidance, elite-difference structure, and adaptive local perturbation.

In addition, HCALA performs an extra intra-elite polishing step. Let $\mathbf{X}^t_{\text{ref}}$ be a selected elite reference solution and $\mathbf{X}^t_{\text{mate}}$ be another elite mate. The local polished candidate is generated by

$$\begin{aligned} \mathbf{X}^t_{\text{local}} = \mathbf{X}^t_{\text{ref}} &+ 0.08 \left(\mathbf{X}^t_{best} - \mathbf{X}^t_{\text{ref}}\right) \\ &+ 0.05 \left(\bar{\mathbf{A}}^t - \mathbf{X}^t_{\text{ref}}\right) \\ &+ 0.06\, \mathbf{r} \odot \left(\mathbf{X}^t_{\text{mate}} - \mathbf{X}^t_{\text{ref}}\right) \\ &+ \beta_t \boldsymbol{\sigma}_e^t \odot \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{aligned} \tag{29}$$

where $\mathbf{r} \in (0,1)^D$ is a random vector and the schedule of β_t is given in Table 1. This additional polishing step is lightweight, but it further improves the precision and stability of late-stage local search around promising elite regions.

Regarding replacement, HCALA more often challenges elite-edge individuals in the later stage. Specifically, with probability p_{rep}, the refined child competes with an individual sampled from the elite-edge range listed in Table 1; otherwise, it competes with the current worst individual in the population.

3.4 Integrated Workflow and Main Parameter Settings

The above three components should not be understood as isolated add-on strategies. Instead, they are integrated into a progressive optimization pipeline. More specifically, the elite historical archive-guided dual-memory learning mechanism first improves guidance continuity during candidate generation; on this basis, the success-memory-driven branch self-adaptation mechanism further improves branch-level search allocation during the main search stage; finally, once the search has entered promising regions, the late-stage hybrid exploitation kernel is activated to strengthen terminal refinement.

For clarity, the main control parameters involved in the above components are summarized in Table 1. Fixed schedules and implementation thresholds are intentionally collected in the parameter table so that the main text can focus on the principal computational operators.

Algorithm 2 summarizes the complete workflow of HCALA.

3.5 Computational Complexity Analysis

Computational complexity is an important indicator for evaluating the efficiency and scalability of a population-based optimizer. Let N denote the population size, D the problem dimension, and $T_{\max}$ the maximum number of iterations. In addition, let $C_f(D)$ denote the average computational cost of one objective-function evaluation. Since HCALA is designed for black-box continuous optimization, it is more rigorous to separate the optimizer overhead from the fitness evaluation cost during complexity analysis.

Initialization stage: The initialization stage generates the initial population, initializes the historical-best memory, and performs the first-round fitness evaluation. The position generation and memory initialization require $\mathcal{O}(ND)$ operations, while the first fitness computation costs $\mathcal{O}(N\,C_f(D))$. Therefore, the total complexity of the initialization stage is

$$\mathcal{O}_{\text{init}} = \mathcal{O}(ND) + \mathcal{O}(N\,C_f(D)). \tag{30}$$

Main search stage: In each iteration, every individual executes one ALA branch to generate a base candidate, after which dual-memory correction, boundary repair, greedy selection, and personal-best updating are carried out. These vector-level operations are all linear in the problem dimension, so the algorithmic overhead of the population update is $\mathcal{O}(ND)$ per iteration. The corresponding fitness evaluation of the N candidates contributes $\mathcal{O}(N\,C_f(D))$ per iteration. In addition, the success-memory-driven branch self-adaptation only involves branch-wise score accumulation, normalization, and probability updating, whose cost is at most $\mathcal{O}(N)$ per iteration and is therefore lower-order than the population update. Hence, the overall complexity of the main search stage is

$$\mathcal{O}_{\text{main}} = \mathcal{O}(T_{\max}ND) + \mathcal{O}(T_{\max}N\,C_f(D))\,. \tag{31}$$

Archive maintenance and elite selection: HCALA maintains an elite historical archive and repeatedly extracts the elite subset from the current population and the

Algorithm 2 Pseudocode of HCALA

Require: Population size N, dimension D, maximum iteration $T_{\max}$, lower bound $\mathbf{lb}$, upper bound $\mathbf{ub}$
Ensure: Best solution $\mathbf{X}_{best}$

function HCALA($N, D, T_{\max}, \mathbf{lb}, \mathbf{ub}$)
  Initialize population $\mathbf{X}$ and evaluate fitness
  Initialize individual historical-best memory $\mathbf{P}$
  Construct elite historical archive $\mathcal{A}$ and initialize branch probabilities according to Table 1
  for $t = 1$ to $T_{\max}$ **do**
    Compute normalized progress τ_t using Eq. (12)
    Update the current best solution from the historical-best memory
    Update the elite archive using Eq. (14)
    Compute $\bar{\mathbf{A}}^t$ and $\mathbf{G}^t$ using Eqs. (15)–(16)
    for $i = 1$ to N **do**
      Compute the ALA energy factor and determine the current phase
      Select one search branch according to the current adaptive branch probabilities
      Generate the base candidate using the selected ALA behavior
      Correct the base candidate using Eq. (17)
      Repair boundary violations using Eq. (18)
      Evaluate the candidate and perform greedy selection
      if the candidate is accepted **then**
        Accumulate branch success gain using Eq. (19)
      end if
      Update the individual historical-best memory if improved
    end for
    Compute learned branch preferences using Eqs. (20)–(21)
    Update adaptive branch probabilities using Eqs. (23)–(24)
    if the late-stage kernel activation condition in Table 1 is satisfied **then**
      Compute elite dispersion using Eq. (25)
      for $k = 1$ to n_K^t **do**
        Generate the mutant vector using Eq. (26)
        Perform crossover using Eq. (27)
        Conduct terminal refinement using Eq. (28)
        Evaluate offspring and apply replacement
        Perform intra-elite polishing using Eq. (29)
      end for
    end if
    Refresh archive records and store the current best-so-far solution
  end for
  return $\mathbf{X}_{best}$
end function

historical-best memory. Since $A_{\max} = \mathcal{O}(N)$ and $N_e = \mathcal{O}(N)$ according to Table 1, the archive merge and top-set selection can be implemented by sorting or partial ranking over a set of size proportional to N. Consequently, the associated worst-case overhead is

$$\mathcal{O}_{\text{rank}} = \mathcal{O}(T_{\max} N \log N)\,. \tag{32}$$

Table 1 Main control parameters and schedules used in HCALA.

Symbol / item	Setting / schedule	Role
$A_{\max}$	$\max(12, \min(24, \mathrm{round}(0.6N)))$	archive capacity
N_e	$\max(4, \min(N, \mathrm{round}(0.25N)))$	elite subset size
$\mathbf{p}_{\mathrm{exp}}^0$	[0.5, 0.5]	initial exploration-branch probabilities
$\mathbf{p}_{\mathrm{dev}}^0$	[0.5, 0.5]	initial exploitation-branch probabilities
η_t	0.75 if $\tau_t < 0.5$, else 0.90	archive-guide mixture coefficient
k_{arch}^t	$0.04 + (0.15 - 0.04)\tau_t$	archive-guidance strength
k_{pbest}^t	$\begin{cases} 0.01 + 0.04\tau_t, & \tau_t < 0.5 \\ 0.04 + (0.14 - 0.04)\dfrac{\tau_t - 0.5}{0.5}, & \tau_t \geq 0.5 \end{cases}$	historical-best guidance strength
ρ	0.18	probability update rate
$p_{\min}$, $p_{\max}$	0.15, 0.85	clipping bounds for branch probabilities
τ_K	0.42	late-stage kernel trigger threshold
n_K^t	$4 + \mathbb{I}(\tau_t > 0.65) + \mathbb{I}(\tau_t > 0.80)$	kernel call frequency
F_1^t	sampled in [0.55, 0.70]	current-best attraction in mutation
F_2^t	sampled in [0.35, 0.45]	elite-difference perturbation in mutation
CR_K	0.92	crossover rate in kernel
α_t	$0.06(1 - \tau_t) + 0.004$	child perturbation scale
β_t	$0.03(1 - \tau_t) + 0.002$	intra-elite polishing scale
$c_{\mathrm{arch}}^{\mathrm{mut}}$	0.08	archive attraction in mutation
$\mathbf{c}_{\mathrm{child}}$	[0.10, 0.04, 0.04]	best / archive / elite-difference pulls
$\mathbf{c}_{\mathrm{local}}$	[0.08, 0.05, 0.06]	best / archive / mate-difference pulls
p_{rep}	0.60	elite-edge replacement preference
$\mathcal{R}_{\mathrm{edge}}$	$[\max(2, \mathrm{round}(0.55N_e)),\ N_e]$	replacement candidate range
$\sigma_{\min}$	$10^{-3}(\mathbf{ub} - \mathbf{lb})$	lower bound of elite dispersion

Late-stage hybrid exploitation kernel: Once $\tau_t > \tau_K$, HCALA activates the late-stage hybrid exploitation kernel. In each kernel call, elite-parent selection, mutation, crossover, and local polishing are dominated by vector operations on a small number of elite individuals, which gives $\mathcal{O}(n_K^t D + n_K^t C_f(D))$ at iteration t. Because n_K^t is bounded by a small constant according to Table 1, the total kernel cost over the whole

run can be upper-bounded by

$$\mathcal{O}_{\text{kernel}} = \mathcal{O}(T_{\max} D) + \mathcal{O}(T_{\max} C_f(D)) \,, \tag{33}$$

which does not change the leading-order complexity with respect to N.

By combining the above components, the total complexity of HCALA can be written as

$$\mathcal{O}(\text{HCALA}) = \mathcal{O}(ND) + \mathcal{O}(T_{\max} ND) + \mathcal{O}(T_{\max} N\, C_f(D)) + \mathcal{O}(T_{\max} N \log N) \,. \tag{34}$$

For the CEC benchmark functions considered in this paper, the objective evaluation cost is typically linear in the dimension, i.e., $C_f(D) = \mathcal{O}(D)$. Under this common setting, Eq. (34) can be simplified to

$$\mathcal{O}(\text{HCALA}) = \mathcal{O}(T_{\max} ND + T_{\max} N \log N) \,, \tag{35}$$

and it is usually dominated by $\mathcal{O}(T_{\max} ND)$ when the fitness evaluation cost is the principal runtime component. In addition, the memory complexity of HCALA is $\mathcal{O}(ND + A_{\max} D) = \mathcal{O}(ND)$, since both the population-level memories and the archive store only a linear number of D-dimensional vectors. Therefore, the proposed historical-cognitive enhancement improves guidance, adaptation, and late-stage refinement without altering the linear-in-population, linear-in-dimension storage order of the original population-based search framework.

4 Numerical Experiments and Statistical Analysis

This section conducts numerical experiments in CEC benchmarks to investigate the performance of HCALA. Section 4.1 introduces the experimental environment and implementation details. Section 4.2 describes the benchmarks, competitor algorithms, and evaluation protocol. The comparison experiments and statistical analysis in CEC2017 and CEC2022 are presented in Sections 4.3 and 4.4, respectively. To ensure that the conclusions are transparent and easy to verify, the analysis is carried out jointly from four aspects, namely summary statistics, function-level mean/std results, representative convergence curves, and boxplots.

4.1 Experimental Environment and Implementation Details

All experiments were carried out under the same software and hardware settings to ensure a fair comparison of algorithm performance. The experimental conditions were standardized to minimize external interferences and enhance the repeatability and reliability of the results. Specifically, the experiments were performed on a system configured as follows:

- Operating System: Windows 11
- Processor: AMD Ryzen 9 7940HX with Radeon Graphics
- Memory: 16 GB RAM
- Programming Language: MATLAB R2024b

To reduce potential biases caused by differences in hardware or software, all experiments were conducted within a consistent computing environment. This ensures that comparisons among different optimization algorithms remain fair, reliable, and objective. Unless otherwise stated, all algorithms are executed with a unified population size of $N = 100$ and a unified maximum iteration budget of $T_{\max} = 500$. The compared baseline methods follow the parameter settings recommended in their original papers, except for the common population size and stopping budget, which are fixed for all methods to ensure a fair comparison.

For each benchmark function, 30 independent runs are conducted, and the final results are summarized by the mean and standard deviation. The convergence figures report the median best-so-far fitness over 500 iterations, whereas the boxplots present the distribution of the final solutions over repeated runs. Pairwise significance is assessed by the two-sided Wilcoxon signed-rank test at the 0.05 level, and the overall ranking is evaluated by the Friedman test. In the detailed tables, only the mean and standard deviation are reported so that the function-level numerical results can be checked directly against the source spreadsheets.

4.2 Benchmarks, Competitor Algorithms, and Evaluation Protocol

To comprehensively evaluate the overall performance of the proposed HCALA, three benchmark settings are considered in this study, namely CEC2017 with 30 dimensions, CEC2017 with 50 dimensions, and CEC2022 with 20 dimensions [26, 27]. These test suites contain shifted, rotated, hybrid, and composition landscapes with very different modality, variable coupling, and basin structures, and therefore provide a suitable basis for examining the optimization capability, stability, and scalability of HCALA.

Tables 2 and 3 summarize the basic information of the CEC2017 and CEC2022 benchmark suites used in this paper. Following the standard protocol adopted in the source experimental setting, the CEC2017 suite is evaluated on 29 functions and covers one unimodal function, seven simple multimodal functions, ten hybrid functions, and eleven composition functions. By contrast, CEC2022 contains twelve functions, including one unimodal, four basic, three hybrid, and four composite functions. Consequently, the former suite emphasizes broad landscape diversity and dimensional scalability, whereas the latter provides a more compact but still highly challenging test set with strong structural coupling. The joint use of these two suites makes it possible to assess whether HCALA is only strong on one specific landscape type or can maintain competitive behavior across heterogeneous optimization scenarios.

Additionally, twelve representative optimizers are selected as competitors, including PSO [28], DE [29], HHO [30], WOA [31], GWO [32], SCA [33], TLCO [34], SOA [35], PLO [36], ALA [37], EALA [38], and IALA [39]. Among them, ALA serves as the original baseline, whereas EALA and IALA are improved variants within the same algorithmic family. Consequently, the comparison not only evaluates whether HCALA is competitive against widely used population-based optimizers, but also directly verifies whether the proposed historical-cognitive hybrid design provides consistent gains over the original ALA framework and its improved derivatives.

Table 2 CEC2017 benchmark functions. Uni.=Unimodal, Multi.=Simple multimodal, Hybrid.=Hybrid, Comp.=Composite.

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

Table 3 CEC2022 benchmark functions. Uni.=Unimodal, Basic.=Basic, Hybrid.=Hybrid, Comp.=Composite.

No.	Func.	Feature	Optimum
f_1	Shifted and full Rotated Zakharov	Uni.	300
f_2	Shifted and full Rotated Rosenbrock	Basic.	400
f_3	Shifted and full Rotated Expanded Schaffer f_6		600
f_4	Shifted and full Rotated Non-Continuous Rastrigin		800
f_5	Shifted and full Rotated Levy		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: $[-100, 100]^D$			

The reported results are organized at two levels. First, summary tables provide the $+/\approx/-$ counts, average Friedman ranks, and average CPU time, thereby showing the overall behavior of each method on a complete benchmark set. Second, the detailed mean/std tables list the final objective values on every individual function, which makes the strengths and weaknesses of HCALA easier to trace at the function level. The representative convergence curves and boxplots are plotted for functions that cover different difficulty characteristics within each benchmark suite, so the final conclusion is based on both global statistics and fine-grained visual evidence.

4.3 Comparison Experiments and Analysis on CEC2017

Table 4 summarizes the Wilcoxon significance results, average ranks, and average CPU time on CEC2017 with 30D and 50D. A first important observation is that HCALA achieves the best overall Friedman rank in both settings, namely 2.452 in 30D and 2.075 in 50D. The second-best ranks are 4.118 and 3.846, respectively, which means that HCALA reduces the rank value by about 40.5% on 30D and 46.0% on 50D relative to the closest competitor in each setting. Therefore, the superiority of HCALA is not a marginal or accidental improvement on a few isolated functions, but a stable rank-level advantage over the entire benchmark suite.

The pairwise Wilcoxon results further clarify the source of this advantage. In the 30D setting, HCALA obtains perfect 29/0/0 superiority counts against HHO, WOA, SCA, and ALA, and still preserves strong win counts against PSO (26/0/3), DE (27/0/2), GWO (27/0/2), TLCO (26/0/3), SOA (28/0/1), and PLO (28/0/1). The

competition is noticeably tighter against EALA and IALA, with counts of 22/0/7 and 24/0/5, respectively, which is a reasonable outcome because these methods belong to the same algorithmic lineage as ALA. Even under this more demanding intra-family comparison, HCALA still retains the best overall Friedman rank, indicating that the proposed historical guidance, adaptive branch coordination, and late-stage exploitation kernel provide a real algorithmic gain rather than a cosmetic parameter-level adjustment.

The convergence and distribution figures are consistent with the above statistical evidence. In Figure 1, HCALA usually shows a steeper early descent and, more importantly, a lower late-stage convergence plateau than the compared methods on the representative 30D functions. This indicates that the proposed method does not merely enter promising regions quickly, but can also continue refining the solution after several baselines have already stagnated. Figure 2 provides complementary evidence from the final-value distributions: the boxes of HCALA are generally more concentrated and exhibit fewer high-end outliers, which means that the algorithm is not relying on occasional lucky runs to achieve its mean performance. In other words, the summary superiority in Table 4 is well aligned with the run-level robustness shown in the boxplots.

When the dimensionality increases from 30D to 50D, the advantage of HCALA becomes even more pronounced rather than deteriorating. This is a particularly important result, because many meta-heuristics that are competitive in moderate dimensions suffer a clear loss of ranking when the search space becomes more difficult. In contrast, HCALA remains the top-ranked method and its pairwise superiority becomes stronger on several comparisons: it achieves 29/0/0 against HHO, WOA, SCA, TLCO, SOA, and PLO; 28/0/1 against PSO, DE, and ALA; and still maintains clear advantages over GWO and IALA with 25/0/4. Even against EALA, which is the strongest same-family rival in this setting, HCALA still records 22/0/7. These results show that

Table 4 Wilcoxon significance results, average Friedman ranks, and average CPU time on CEC2017.

Algorithm	CEC2017-30D			CEC2017-50D		
	$+/\approx/-$	Rank	Time(s)	$+/\approx/-$	Rank	Time(s)
PSO	26/0/3	6.199	0.0894	28/0/1	6.341	0.1522
DE	27/0/2	6.976	0.1542	28/0/1	6.979	0.2138
HHO	29/0/0	9.974	0.2042	29/0/0	9.038	0.3363
WOA	29/0/0	11.885	0.0808	29/0/0	11.639	0.1362
GWO	27/0/2	6.337	0.1184	25/0/4	5.846	0.1994
SCA	29/0/0	12.054	0.1004	29/0/0	12.243	0.1720
TLCO	26/0/3	6.538	0.1724	29/0/0	7.301	0.2577
SOA	28/0/1	6.875	0.1297	29/0/0	7.129	0.2025
PLO	28/0/1	7.345	0.1655	29/0/0	8.364	0.2577
ALA	29/0/0	4.118	0.0991	28/0/1	4.408	0.1622
EALA	22/0/7	5.722	0.0992	22/0/7	5.791	0.1618
IALA	24/0/5	4.526	0.1968	25/0/4	3.846	0.3268
HCALA	–	2.452	0.2078	–	2.075	0.2879

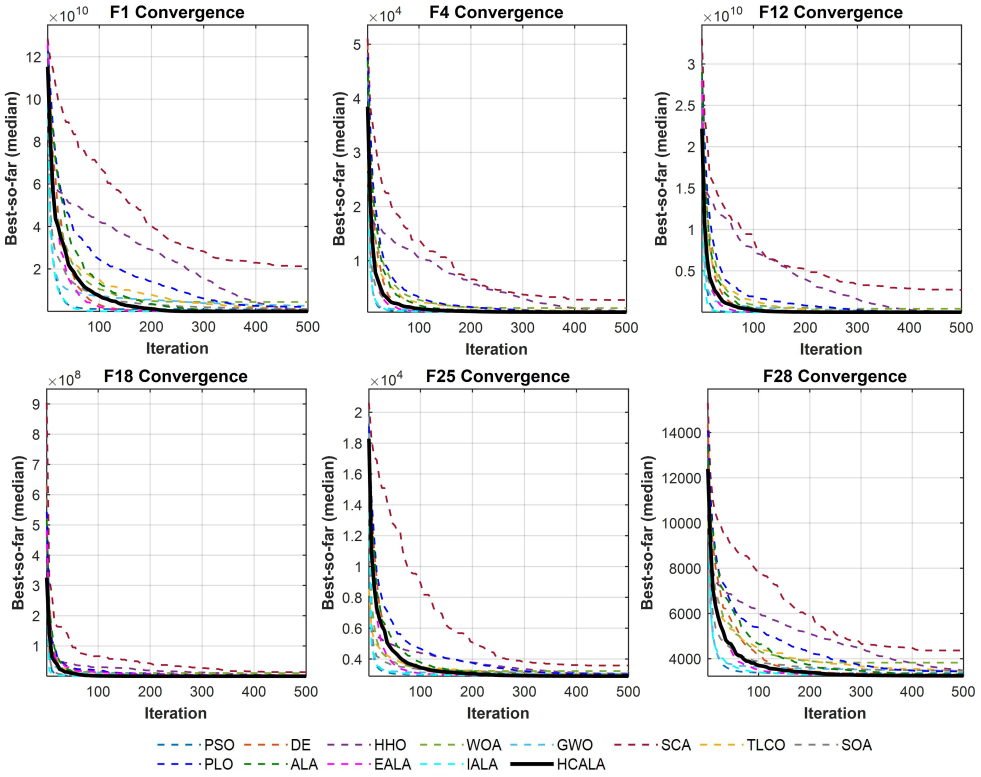


Fig. 1 Representative convergence curves of optimizers on CEC2017 with 30D.

the proposed method is not only effective in a lower-dimensional regime, but also retains strong competitiveness when variable interaction and deceptive basin structure become more severe.

The visual evidence in the 50D case further strengthens this conclusion. In Figure 3, several competing methods flatten out or oscillate around suboptimal levels in the middle or later iterations, whereas HCALA typically preserves visible descent for a longer period and reaches lower terminal objective values. This pattern is exactly what one would expect from the combination of the three proposed components: the dual-memory mechanism helps retain useful directional information, the success-memory scheme reduces ineffective branch usage, and the late-stage kernel improves the final refinement once the search has already entered a promising region. The corresponding boxplots in Figure 4 again show narrower distributions and better lower-tail behavior for HCALA on representative functions, confirming that its advantage is not only reflected in mean values but also in stability.

From the efficiency perspective, HCALA is not the fastest algorithm in Table 4. Its average CPU time is 0.2078 s on 30D and 0.2879 s on 50D, which is higher than that of the original ALA (0.0991 s and 0.1622 s) and some lightweight baselines such as WOA. However, the absolute overhead remains small, while the gain in average rank

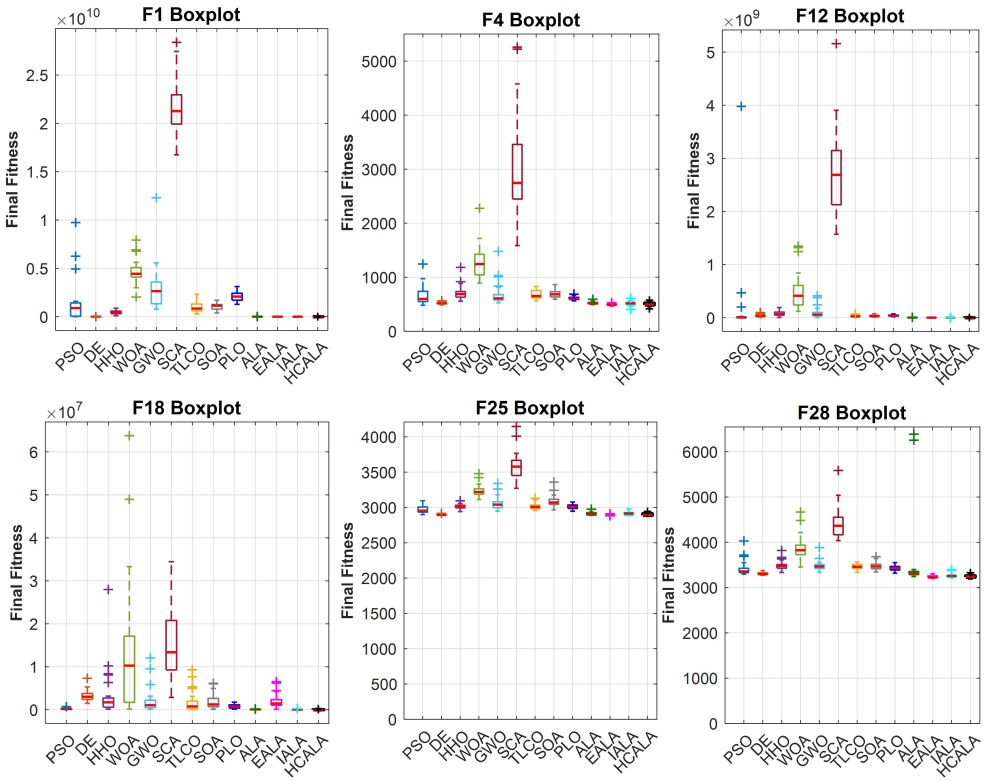


Fig. 2 Representative boxplots of optimizers on CEC2017 with 30D.

is substantial and consistent across both dimensions. Therefore, the CEC2017 results indicate that HCALA achieves a favorable accuracy–robustness–efficiency trade-off: it pays a moderate runtime premium in exchange for much stronger overall search quality and markedly better scalability.

4.4 Comparison Experiments and Analysis on CEC2022

Table 5 summarizes the Wilcoxon significance results, average ranks, and average CPU time on CEC2022 with 20D. HCALA still achieves the best overall average rank on this benchmark group, with an average rank of 2.853. The second-best average rank is 4.747, so HCALA improves the global rank value by about 39.9% relative to the closest competitor. Compared with the CEC2017 results, the absolute rank gap is slightly smaller, which indicates that the competition on CEC2022 is more intense and that strong baselines such as DE, EALA, and IALA remain competitive on several individual functions. Even so, HCALA still preserves the best overall ranking, which is strong evidence that the proposed method maintains balanced performance even when the benchmark size is smaller and each function has a larger influence on the final rank.

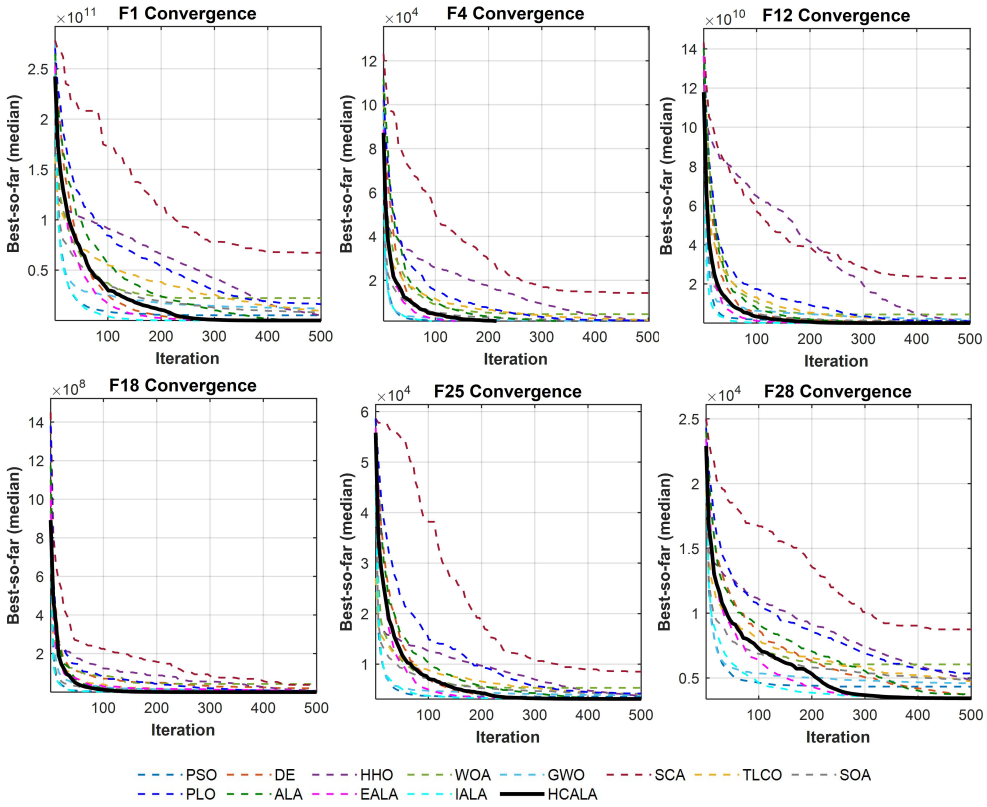


Fig. 3 Representative convergence curves of optimizers on CEC2017 with 50D.

The pairwise Wilcoxon statistics provide a more fine-grained explanation. HCALA achieves perfect 12/0/0 superiority counts against PSO, HHO, WOA, and GWO. It also records 11/0/1 against SCA, TLCO, SOA, PLO, and ALA, which means that on these comparisons HCALA is better on almost the whole benchmark suite and never loses. The strongest challenges come from DE, IALA, and especially EALA, against which HCALA obtains 10/0/2, 9/0/3, and 7/0/5, respectively. This result is important for interpretation: on CEC2022, the difference among top-performing algorithms is not that HCALA dominates every single function without resistance, but rather that it sustains a higher and more stable performance level across the entire suite. In other words, its superiority comes from consistency, not from a few extreme wins.

From the detailed results, HCALA attains the best mean value on 4 out of 12 functions. At first glance, this count may seem less dominant than the CEC2017 case. However, this is precisely why the Friedman rank is informative here: it shows that HCALA remains in the top tier on most of the remaining functions and avoids severe performance degradation on any category listed in Table 3. Several baseline algorithms exhibit stronger function-type dependence, performing well on part of the suite but

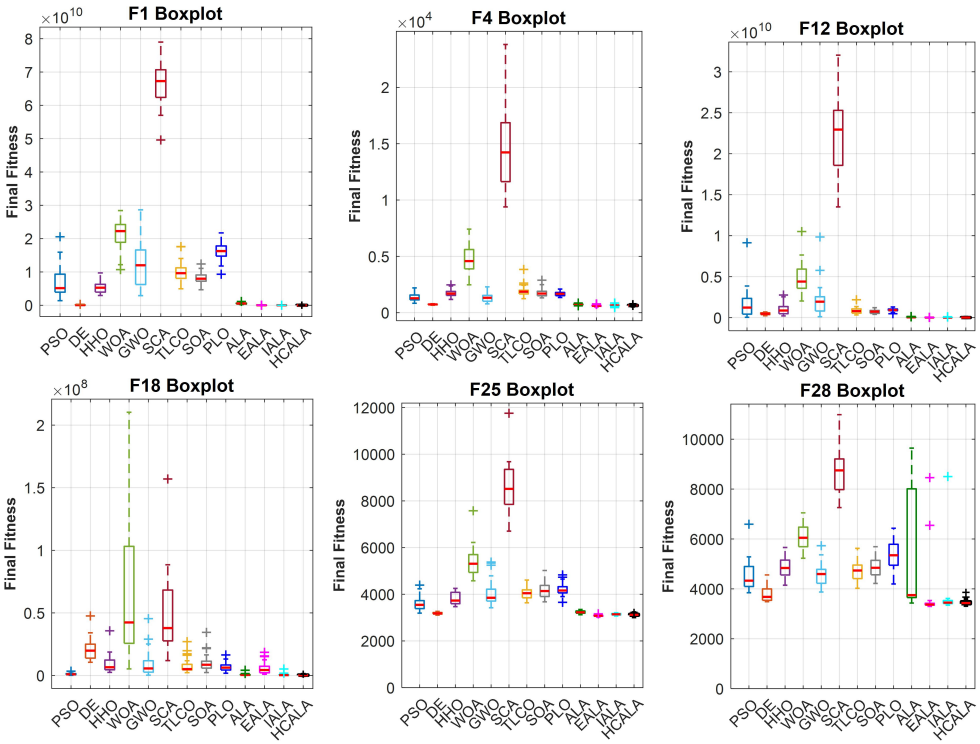


Fig. 4 Representative boxplots of optimizers on CEC2017 with 50D.

dropping more visibly on other functions. Therefore, HCALA does not win by over-specializing to a narrow landscape type; instead, it benefits from a more even balance among exploration, guidance continuity, and terminal exploitation.

The dynamic behavior shown in Figure 5 is consistent with this interpretation. On representative difficult functions such as f_5 and f_{11}, HCALA continues to improve in the later iterations after a number of baseline curves have already become nearly flat. This late-stage separation is particularly important on CEC2022 because several functions in this suite contain strong rotations, hybrid structures, or composite interactions, so an optimizer that cannot maintain effective refinement often ends with a noticeably inferior final objective value. The boxplots in Figure 6 provide the same message from the repeated-run perspective: the distribution of HCALA is usually more concentrated and contains fewer severe outliers, showing that the improved final performance is reproducible rather than occasional.

From the efficiency viewpoint, HCALA again incurs a higher runtime than the original ALA and some lightweight baselines: its average CPU time is 0.2177 s, compared with 0.0835 s for ALA and 0.0681 s for WOA. Nevertheless, the absolute runtime still remains low, and the additional cost is accompanied by a clear improvement in overall solution quality and rank stability. Therefore, the CEC2022 results reinforce the conclusion already observed on CEC2017: although HCALA is not designed to

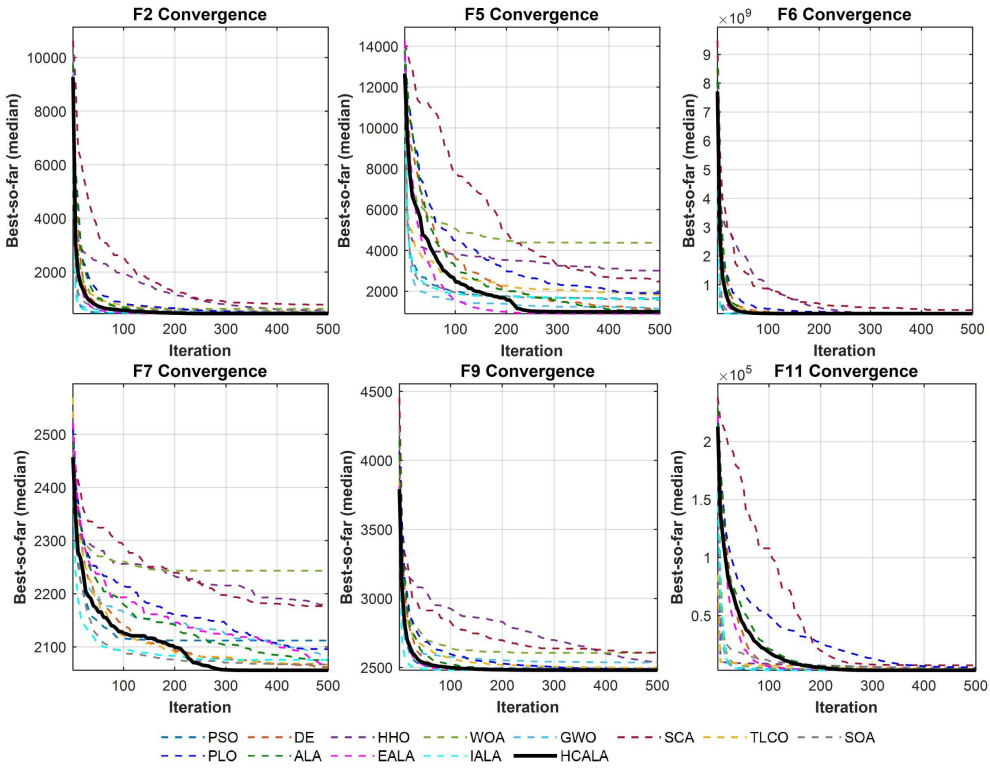


Fig. 5 Representative convergence curves of optimizers on CEC2022 with 20D.

Table 5 Wilcoxon significance results, average Friedman ranks, and average CPU time on CEC2022-20D.

Algorithm	$+/\approx/-$	Avg. Rank	Avg. CPU Time (s)
PSO	12/0/0	6.922	0.0742
DE	10/0/2	5.678	0.1612
HHO	12/0/0	10.108	0.1722
WOA	12/0/0	11.406	0.0681
GWO	12/0/0	7.064	0.0983
SCA	11/0/1	11.175	0.0799
TLCO	11/0/1	6.803	0.1624
SOA	11/0/1	6.758	0.1183
PLO	11/0/1	7.667	0.1481
ALA	11/0/1	<u>4.747</u>	0.0835
EALA	7/0/5	4.961	0.0821
IALA	9/0/3	4.858	0.1635
HCALA	–	**2.853**	0.2177

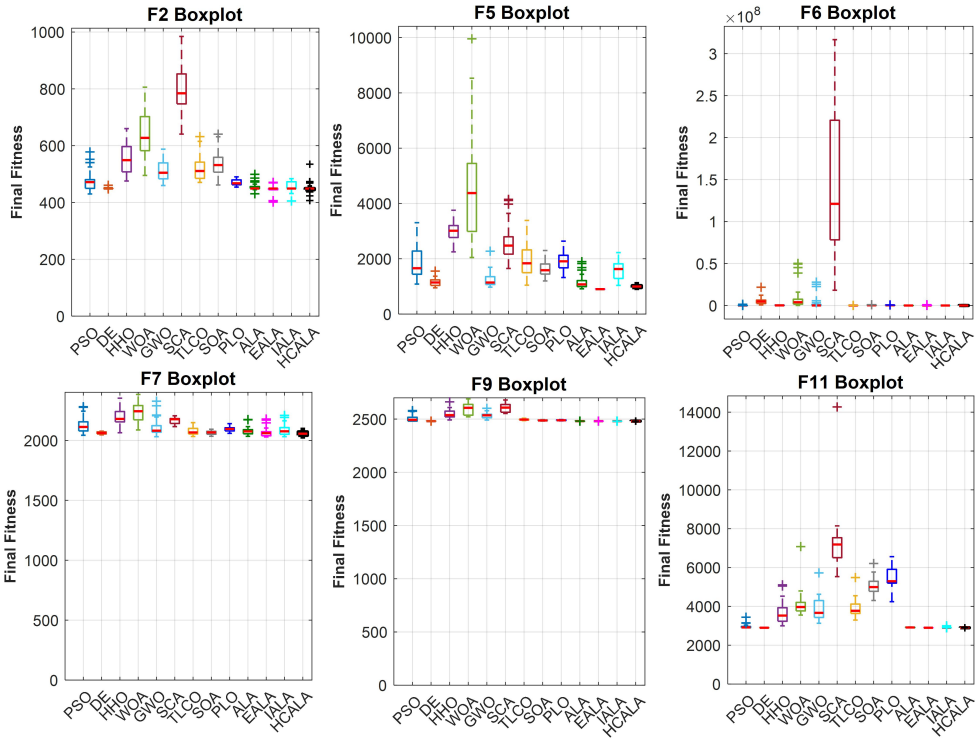


Fig. 6 Representative boxplots of optimizers on CEC2022 with 20D.

be the cheapest optimizer in raw runtime, it delivers the most favorable overall balance among convergence quality, robustness, and cross-function consistency within the considered benchmark setting.

4.5 Discussion

Combining the three benchmark settings, four observations can be drawn. First, HCALA obtains the best average Friedman rank in all benchmark groups considered in this study, and the rank is consistently below 3 (2.452 on CEC2017-30D, 2.075 on CEC2017-50D, and 2.853 on CEC2022-20D). This indicates that the improvement brought by HCALA is not restricted to one specific dimensionality or one specific benchmark suite. Second, its advantage becomes more evident on CEC2017 when the dimensionality increases from 30D to 50D, which suggests that the historical-cognitive hybrid design has favorable scalability under more difficult search spaces. Third, the competition on CEC2022 is relatively tighter, yet HCALA still preserves the best overall rank and strong Wilcoxon superiority counts, implying that its strength lies in stable cross-function behavior rather than in occasional wins on only a few functions. Fourth, HCALA is not the cheapest method in terms of CPU time, but the additional

computational cost remains acceptable in view of the clearly improved convergence quality, final accuracy, and run-to-run stability.

Overall, the experimental evidence matches the design intention of HCALA. The historical-elite guidance helps preserve and reuse valuable search information across generations, the success-memory-driven branch coordination reduces ineffective search allocation, and the late-stage hybrid exploitation kernel strengthens terminal refinement after promising regions have been reached. Their combined effect explains why HCALA can maintain both competitive convergence speed and reliable final-solution quality across different CEC benchmark settings.

4.6 Detailed Mean/Std Tables

For completeness and reproducibility, the detailed mean/std results are listed below. These tables are placed after the main discussion so that the central narrative of the experimental section remains compact, while the full function-level evidence is still retained for verification and further comparison. To keep the presentation concise and consistent with the source spreadsheets, these tables report only the mean and standard deviation.

5 Practical Engineering Applications of the HCALA Algorithm

To further assess the effectiveness and real-world applicability of the HCALA algorithm, three representative engineering optimization problems are selected as test cases: the pressure vessel design problem [40], the compression spring design problem [41], and the welded beam design problem [42]. Notably, the welded beam problem includes up to eleven constraints, presenting considerable complexity and making it an ideal benchmark to examine the algorithm's performance in tackling constrained optimization scenarios.

When solving such constrained optimization problems, it is essential to adopt appropriate constraint-handling techniques. Common approaches include penalty function methods [44, 45], feasibility rule-based methods [43], and multi-objective optimization strategies [46, 47].Owing to its straightforward concept and relatively simple implementation, the penalty function approach has become a popular choice in engineering optimization tasks. Therefore, this study employs a penalty-based approach to manage constraints. Specifically, when a candidate solution violates any constraint, a penalty factor is incorporated to significantly increase its objective function value, thereby reducing its likelihood of selection. This mechanism effectively guides the algorithm toward convergence within the feasible region and ultimately to an optimal solution that satisfies all constraints.

5.1 Pressure Vessel Design Optimization

Pressure vessel design is a classical problem in engineering optimization. The primary objective is to minimize the manufacturing cost of the vessel while satisfying specified pressure requirements. A schematic illustration of the pressure vessel is shown in Figure 7.

This design optimization task considers four decision variables: L, R, T_s, and T_h. In this context, L represents the length of the cylindrical section, R denotes the inner radius of the shell, T_s indicates the thickness of the cylindrical body, and T_h corresponds to the head thickness.

The problem's objective function is formulated in Eq. (36) and Eq. (37), while the relevant constraints are expressed in inequalities from Eq. (38) through Eq. (41).

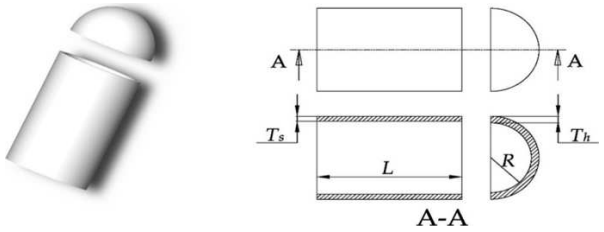


Fig. 7 Schematic model of the pressure vessel design problem.

$$\mathcal{X} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \tag{36}$$

$$\mathcal{F}(\mathcal{X}) = 0.6224\, x_1 x_3 x_4 + 1.7781\, x_2 x_3^2 + 3.1661\, x_1^2 x_4 + 19.84\, x_1^2 x_3 \tag{37}$$

$$\mathcal{G}_1(\mathcal{X}) = -x_1 + 0.0193\, x_3 \leq 0 \tag{38}$$

$$\mathcal{G}_2(\mathcal{X}) = -x_2 + 0.00954\, x_3 \leq 0 \tag{39}$$

$$\mathcal{G}_3(\mathcal{X}) = -\pi x_3^2 x_4 - \frac{4\pi x_3^3}{3} + 1296000 \leq 0 \tag{40}$$

$$\mathcal{G}_4(\mathcal{X}) = x_4 - 240 \leq 0 \tag{41}$$

where the variable bounds are $0 \leq x_1 \leq 99,\ 0 \leq x_2 \leq 99,\ 10 \leq x_3 \leq 200,\ 10 \leq x_4 \leq 200$.

For a fair comparison, the above thirteen algorithms were independently executed 100 times under the same experimental conditions. The average results were recorded as the final performance metrics. The optimization results are presented in Table 6.

As shown in Table 6, the performance of thirteen optimization algorithms on the pressure vessel design problem is comprehensively compared. It can be observed that HCALA achieves the best overall performance with the lowest mean objective value of 6105.72889, indicating its superior optimization capability. The proposed ALA

Table 6 Comparison results on pressure vessel design problem

Algorithm	Mean	Std	Best	Worst	Rank	X_1	X_2	X_3	X_4
HCALA	6105.72889	216.51611	6058.71988	7332.61936	1	0.80621	0.41651	42.11476	176.43449
ALA	6132.65617	188.57635	6058.71988	7332.61933	2	0.83998	0.46433	42.11478	176.43434
EALA	6257.70061	345.29978	6058.71988	7332.61933	3	0.82154	0.41481	42.11478	176.43433
GWO	6258.2157	395.92391	6058.79579	7407.33607	4	0.81302	0.42048	42.11667	176.41352
PLO	6294.42371	123.50303	6072.46633	6687.66705	5	0.83291	0.44869	42.1086	177.09262
DE	6319.3711	348.17142	6058.92012	7332.62401	6	0.81493	0.45513	42.11098	176.48756
TLCO	6389.86208	269.87674	6058.72097	7273.28998	7	0.81577	0.46722	42.11435	176.43962
PSO	6419.22614	299.92784	6058.71988	7332.61946	8	0.84238	0.4203	42.1148	176.43407
HHO	6726.96007	342.16603	6098.04673	7792.54628	9	0.90558	0.40752	45.271	140.92528
IALA	6783.73456	516.26822	6058.74164	8017.28164	10	0.78158	0.45937	42.11236	176.46424
SOA	6828.10346	423.35686	6091.93654	7552.59203	11	0.87261	0.4631	45.37578	139.86422
SCA	7307.47618	660.44465	6134.17566	8758.27361	12	0.81108	0.44495	42.13188	179.40121
WOA	8276.31443	1490.7558	6062.89645	13494.9139	13	0.80769	0.4383	42.14825	176.02048

also demonstrates competitive performance, ranking second with a mean value of 6132.65617, which confirms the effectiveness of the algorithm design.

Although EALA obtains a slightly higher mean value (6257.70061), it still outperforms several classical algorithms such as PSO, DE, and SCA, demonstrating its robustness in solving constrained engineering optimization problems. In contrast, traditional algorithms such as WOA and SCA exhibit relatively poor performance, with significantly higher mean values and larger standard deviations, indicating weaker stability and convergence performance.

From the perspective of stability, PLO achieves the smallest standard deviation (123.50303), suggesting that it has strong robustness across multiple independent runs. However, its mean performance is inferior to HCALA and ALA, indicating a trade-off between stability and solution quality. Furthermore, most algorithms, including HCALA, ALA, and PSO, are able to obtain similar best values (approximately 6058.72), implying that the global optimum can be reached by multiple methods, but their convergence reliability differs significantly.

Overall, HCALA demonstrates the best balance between convergence accuracy and robustness, making it the most effective algorithm among all compared methods for the pressure vessel design problem.

5.2 The compression spring design problem

The objective of compression spring optimization is to obtain a minimum-weight design while ensuring compliance with constraints associated with deformation, dynamic response, and stress levels.

The design problem is governed by three decision variables, including the wire diameter d, the mean coil diameter D, and the number of active coils P. The objective model is described in Eq. (42) and Eq. (43), whereas the constraint conditions are specified from Eq. (44) to Eq. (45).

$$\mathcal{X} = [x_1, x_2, x_3] = [d, D, P] \tag{42}$$

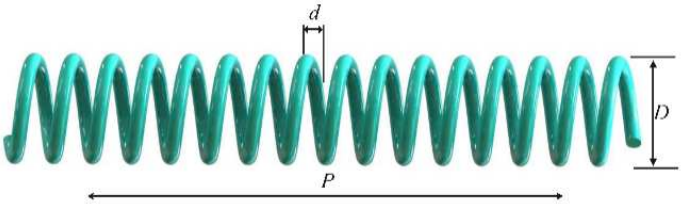


Fig. 8 Pressure vessel problem model.

$$\mathcal{F}(\mathcal{X}) = (x_3 + 2)\, x_2 x_1^2 \tag{43}$$

$$\mathcal{G}_1(\mathcal{X}) = 1 - \frac{x_3 x_2^2}{71785\, x_1^4} \leq 0 \tag{44}$$

$$\mathcal{G}_2(\mathcal{X}) = \frac{4x_2^2 - x_1 x_2}{12566\,(x_2 x_1^3 - x_1^4)} + \frac{1}{5108\, x_1^2} - 1 \leq 0 \tag{45}$$

$$\mathcal{G}_3(\mathcal{X}) = 1 - \frac{140.45\, x_1}{x_2^2 x_3} \leq 0 \tag{46}$$

$$\mathcal{G}_4(\mathcal{X}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \tag{47}$$

where the variable bounds are $0.05 \leq x_1 \leq 2,\ 0.25 \leq x_2 \leq 1.3,\ 2 \leq x_3 \leq 15$.

For a fair comparison, the above thirteen algorithms were independently executed 100 times under the same experimental conditions. The average results were recorded as the final performance metrics. The optimization results are presented in Table 7.

Table 7 Optimal design results obtained by different algorithms for the compression spring problem

Algorithm	Mean	Std	Best	Worst	Rank	X_1	X_2	X_3
HCALA	0.01269	4.6E-05	0.01267	0.0129	1	0.05168	0.35641	11.30711
EALA	0.01269	1.8E-05	0.01267	0.01277	2	0.05185	0.36064	11.06414
ALA	0.01272	0.00013	0.01267	0.01377	3	0.05163	0.35520	11.3788
GWO	0.0128	0.00015	0.01268	0.01355	4	0.05136	0.34873	11.77933
DE	0.01291	0.0002	0.01268	0.01358	5	0.05214	0.36753	10.69101
PLO	0.01304	0.00027	0.01274	0.01534	6	0.05	0.31723	14.0651
SCA	0.01324	0.00049	0.01276	0.01559	7	0.05216	0.36667	10.7915
HHO	0.0136	0.00105	0.01267	0.01777	8	0.05146	0.35132	11.6125
WOA	0.01369	0.00117	0.01267	0.01779	9	0.05163	0.35519	11.37952
PSO	0.01378	0.00143	0.01267	0.01777	10	0.05149	0.35183	11.58114
TLCO	0.0138	0.00105	0.01267	0.01732	11	0.05202	0.36474	10.83396
IALA	0.01444	0.0018	0.01267	0.01777	12	0.05187	0.36096	11.04475
SOA	0.01484	0.00185	0.01267	0.01987	13	0.05126	0.34636	11.92769

Table 7 presents the comparative results of thirteen algorithms on the compression spring design problem. It can be observed that HCALA and EALA achieve the best performance with identical mean objective values of 0.01269, indicating their strong optimization capability. Among them, EALA exhibits a significantly smaller standard deviation (1.8×10^{-5}) compared to HCALA (4.6×10^{-5}), demonstrating superior stability and robustness.

The proposed ALA ranks third with a mean value of 0.01272, which is very close to the best results, confirming its effectiveness in handling constrained optimization problems. In contrast, classical algorithms such as PSO, WOA, and SCA show relatively inferior performance, with higher mean values and larger standard deviations, indicating weaker convergence accuracy and stability.

From the perspective of convergence reliability, most algorithms are able to reach a similar best value (0.01267), suggesting that the global optimum is attainable. However, the differences in mean and standard deviation values reveal that HCALA and EALA provide more consistent performance across multiple independent runs. Notably, PLO achieves relatively stable performance with a moderate standard deviation, but its optimization accuracy is inferior to the top-performing algorithms.

Overall, HCALA and EALA demonstrate the best balance between solution accuracy and robustness, with EALA showing a slight advantage in stability. Therefore, the proposed improved algorithms exhibit superior performance compared to both classical and recent metaheuristic algorithms in solving the compression spring design problem.

5.3 The welded beam design problem

The welded beam optimization problem is frequently adopted as a representative example in engineering design studies, particularly for evaluating the performance of optimization algorithms under practical constraints. Its primary objective is to reduce the overall production cost while ensuring that the structure operates safely within prescribed limits.

To achieve this, several critical mechanical conditions must be satisfied, including limitations on shear stress, normal stress induced by bending, structural stability against buckling, and allowable deformation at the beam tip. These requirements collectively define a constrained design space with strong nonlinearity.

The system configuration is characterized by four independent design parameters, namely the weld thickness (h), weld length (l), beam depth (t), and beam width (b). The optimization process seeks a feasible combination of these variables that yields the lowest possible cost without violating any constraint conditions. The detailed mathematical representation of the objective function is given in Eqs. (48) and (49), while the full set of constraint expressions can be found in Eqs. (50)–(62).

$$\mathcal{X} = [x_1, x_2, x_3, x_4] = [h, l, t, b] \tag{48}$$

$$\mathcal{F}(\mathcal{X}) = 1.10471\, x_1^2 x_2 + 0.04811\, x_3 x_4 (14 + x_2) \tag{49}$$

$$\mathcal{G}_1(\mathcal{X}) = \tau(\mathcal{X}) - \tau_{\max} \leq 0 \tag{50}$$

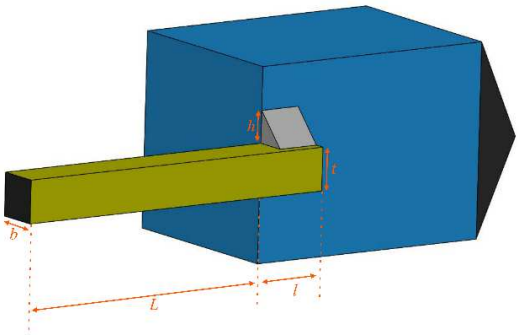


Fig. 9 Pressure vessel problem model.

$$\mathcal{G}_2(\mathcal{X}) = \sigma(\mathcal{X}) - \sigma_{\max} \leq 0 \tag{51}$$

$$\mathcal{G}_3(\mathcal{X}) = \delta(\mathcal{X}) - \delta_{\max} \leq 0 \tag{52}$$

$$\mathcal{G}_4(\mathcal{X}) = x_1 - x_4 \leq 0 \tag{53}$$

$$\mathcal{G}_5(\mathcal{X}) = P - P_c(\mathcal{X}) \leq 0 \tag{54}$$

$$\mathcal{G}_6(\mathcal{X}) = 0.125 - x_1 \leq 0 \tag{55}$$

$$\mathcal{G}_7(\mathcal{X}) = 1.10471\, x_1 + 0.04811\, x_3 x_4 (14 + x_2) - 5 \leq 0 \tag{56}$$

$$0.1 \leq x_1 \leq 2, \quad 0.1 \leq x_2 \leq 10, \quad 0.1 \leq x_3 \leq 10, \quad 0.1 \leq x_4 \leq 2 \tag{57}$$

$$\tau(\mathcal{X}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2} \tag{58}$$

$$\tau' = \frac{P}{\sqrt{2}\, x_1 x_2} \tag{59}$$

$$\tau'' = \frac{P\left(L + \frac{x_2}{2}\right)\sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}}{2\left\{\sqrt{2}\, x_1 x_2 \left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}} \tag{60}$$

$$\sigma(\mathcal{X}) = \frac{6PL}{x_4 x_3^2}, \qquad \delta(\mathcal{X}) = \frac{6PL^3}{E x_3^2 x_4} \tag{61}$$

$$P_c(\mathcal{X}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right) \tag{62}$$

where the variable bounds are $\delta_{\max} = 0.25$ in, $\tau_{\max} = 13600$ psi, $\sigma_{\max} = 30000$ psi, $P = 6000$ lb, $L = 14$ in, $E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi.

For a fair comparison, the above thirteen algorithms were independently executed 100 times under the same experimental conditions. The average results were recorded as the final performance metrics. The optimization results are presented in Table 8.

Table 8 Optimal solutions of each algorithm in the welded beam design problem

Algorithm	Mean	Std	Best	Worst	Rank	X_1	X_2	X_3	X_4
HCALA	1.69277	0	1.69277	1.69277	1	0.20573	3.23492	9.03662	0.20573
EALA	1.69277	1.5E-05	1.69277	1.69291	2	0.20573	3.23492	9.03662	0.20573
ALA	1.69351	0.00476	1.69277	1.73412	3	0.20573	3.23492	9.03662	0.20573
GWO	1.69737	0.00258	1.69431	1.71045	4	0.20523	3.2457	9.04122	0.20573
TLCO	1.82758	0.14366	1.6937	2.20991	5	0.20532	3.24339	9.03698	0.20577
PSO	1.8519	0.2226	1.69277	2.86731	6	0.20573	3.23492	9.03662	0.20573
SCA	1.86716	0.0578	1.73774	2.05059	7	0.20787	3.38143	9.0038	0.20936
PLO	1.88824	0.0685	1.74531	2.14477	8	0.20398	3.35539	9.16933	0.20782
DE	1.92201	0.14662	1.71643	2.34807	9	0.20837	3.26864	8.97842	0.20909
HHO	2.04238	0.35852	1.70297	3.71348	10	0.20446	3.21725	9.13643	0.20539
SOA	2.38987	0.33625	1.77631	3.39689	11	0.21837	3.19052	8.67273	0.22422
IALA	2.61253	0.70541	1.69277	4.4025	12	0.20573	3.23492	9.03662	0.20573
WOA	1.2E+99	9.1E+99	1.77195	8E+100	13	0.19208	3.42382	9.58631	0.20314

Table 8 presents the comparative performance of thirteen algorithms on the welded beam design problem. It is evident that HCALA achieves the best performance with the lowest mean objective value of 1.69277 and a standard deviation of zero, indicating that it consistently converges to the same optimal solution in all independent runs. This demonstrates its excellent robustness and convergence reliability.

EALA also demonstrates highly competitive performance, attaining the same best objective value of 1.69277 and a mean value nearly identical to HCALA, with an exceptionally small standard deviation of only 1.5×10^{-5}. These results indicate that EALA can consistently approach the global optimum, although its stability is slightly lower than that of HCALA.

The proposed ALA ranks third with a mean value of 1.69351, which is very close to the optimal value, confirming its effectiveness in solving constrained engineering design problems. In contrast, classical algorithms such as PSO, SCA, and DE exhibit relatively higher mean values and larger standard deviations, indicating poorer convergence accuracy and stability.

It is also noteworthy that the Whale Optimization Algorithm (WOA) fails to yield meaningful results for the welded beam design problem, producing extremely large objective values on the order of 10^{99}. This indicates that WOA struggles to effectively handle the problem's constraints, underscoring the critical importance of robust constraint-handling mechanisms in engineering optimization.

Overall, HCALA demonstrates the best performance in terms of both accuracy and robustness, while EALA and ALA also show strong competitiveness. The results

confirm the superiority of the proposed algorithms over classical and several recent metaheuristic methods in solving the welded beam design problem.

6 UAV Trajectory Optimization

Unmanned Aerial Vehicles (UAVs) have been widely applied in civil and military domains due to their flexibility, low cost, and high maneuverability. With the development of autonomous control and artificial intelligence, UAVs are evolving toward fully autonomous systems. Path planning is a key technology in UAV autonomy, aiming to generate a collision-free and feasible trajectory from the start point to the target while satisfying constraints such as path length, safety, smoothness, and dynamic limits.

However, UAV path planning in complex environments faces significant challenges, including obstacle avoidance, dynamic constraints, and the need to balance multiple conflicting objectives. Traditional methods, such as Dijkstra and A*, suffer from high computational cost in complex or continuous spaces, while sampling-based methods like RRT often produce suboptimal or non-smooth paths.

From an optimization perspective, UAV trajectory generation involves handling multiple constraints while minimizing a synthesized performance index. This index captures several critical aspects, such as flight distance, exposure to threats, altitude feasibility, and the smoothness of the resulting trajectory.To further improve performance, this paper proposes an improved Hybrid Cooperative Adaptive Learning Algorithm (HCALA), which enhances global exploration and local exploitation through multi-strategy cooperation and adaptive mechanisms. Finally, the proposed HCALA is compared with several representative algorithms, including ALA, GWO, SOA, and HHO, to evaluate its effectiveness and superiority in UAV path planning.

6.1 Safety and Feasibility Constraints

Threat Cost

Obstacles are modeled as vertical cylinders with center $\mathbf{c}_k$ and radius R_k. The threat set is defined as $\mathcal{O}$.

The distance between a path segment and obstacle center is denoted as d. The threat cost is defined as:

$$J_{\text{threat}} = \begin{cases} 0, & d > R_k + d_s \\ \alpha(R_k + d_s - d), & R_k < d \leq R_k + d_s \\ \infty, & d \leq R_k \end{cases} \tag{63}$$

where d_s is the safety buffer distance and α is a penalty coefficient.

Altitude Cost

The flight altitude is constrained within $[z_{\min}, z_{\max}]$. The altitude cost for waypoint $\mathbf{p}_i$ is defined as:

$$J_{\text{altitude},i} = \begin{cases} (z_i - z_{\text{ref}})^2, & z_{\min} \leq z_i \leq z_{\max} \\ \infty, & \text{otherwise} \end{cases} \tag{64}$$

The total altitude cost is:

$$J_{\text{altitude}} = \sum_{i=1}^{N} J_{\text{altitude},i} \tag{65}$$

where z_{ref} is the reference altitude.

Turning Angle

The turning angle between consecutive segments is defined as:

$$\theta_i = \arccos\left(\frac{(\mathbf{p}_i - \mathbf{p}_{i-1}) \cdot (\mathbf{p}_{i+1} - \mathbf{p}_i)}{\|\mathbf{p}_i - \mathbf{p}_{i-1}\| \cdot \|\mathbf{p}_{i+1} - \mathbf{p}_i\|}\right) \tag{66}$$

Climb Angle

The climb angle is defined as:

$$\phi_i = \arctan\left(\frac{z_{i+1} - z_i}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}}\right) \tag{67}$$

Smoothness Cost Function

The total smoothness cost is defined as:

$$J_{\text{smooth}} = \sum_{i=2}^{N-1} \left(\beta_1 \theta_i^2 + \beta_2 (\phi_i - \phi_{i-1})^2\right) \tag{68}$$

where β_1 and β_2 are penalty coefficients.

6.2 Analysis of Experimental Results

A complex UAV mission environment is constructed on a single map containing multiple threat regions, as depicted in Figure 10. Each region is defined parametrically by its geometric center and radius, which together determine its position and scale. Flight trajectories must navigate around these obstacles to ensure safety. Figure 11 indicates that HCALA outperforms conventional algorithms in convergence speed, global search capability, and path quality.

Based on the results in Table 9, the HCALA algorithm demonstrates excellent performance in UAV path optimization. It achieves the lowest best fitness value of

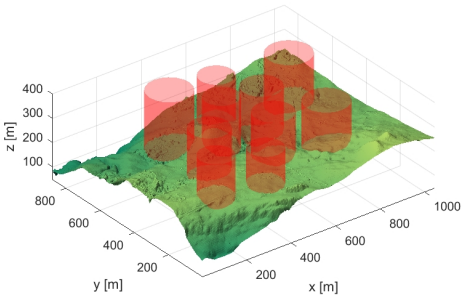


Fig. 10 UAV Path Planning Map.

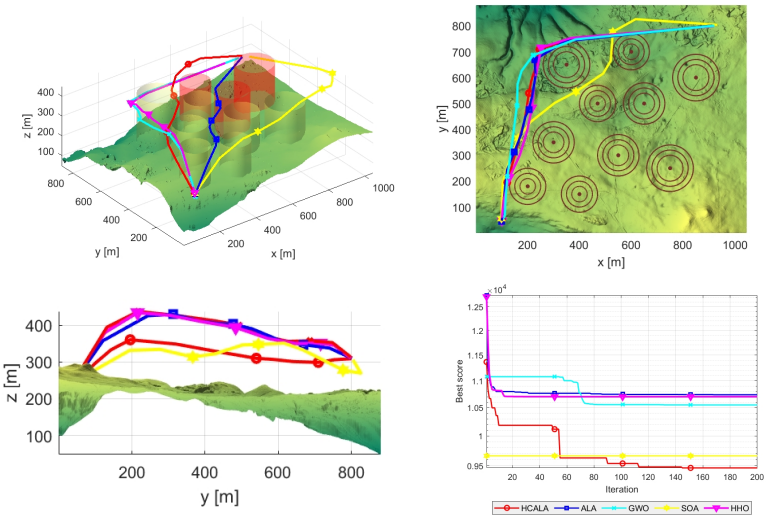


Fig. 11 Path Planning Results and Convergence Curves.

8432.21 and a mean fitness of 9379.70, both significantly better than ALA, GWO, SOA, and HHO. Even in the worst-case scenario, HCALA maintains a reasonable fitness of 10330.03, indicating strong robustness. These results show that HCALA can consistently generate high-quality and reliable flight paths, making it a highly effective algorithm for complex UAV path planning tasks.

Table 9 Statistics of Fitness Values over 30 Trials for Different Algorithms

Algorithm	Best	Mean	Std	Worst
HCALA	8432.2135	9379.7010	484.4596	10330.031
ALA	9758.0553	10587.8819	332.0230	10980.1847
GWO	9016.5731	Inf	NaN	Inf
SOA	9382.7052	10519.5741	507.3180	11244.7888
HHO	9719.4202	10643.4888	316.9370	11254.948

7 conclusion and prospect

This work focuses on overcoming several shortcomings of the conventional Artificial Lemming Algorithm (ALA), such as unevenly distributed initial populations, sluggish convergence, poor balance between global exploration and local exploitation, and a tendency to become trapped in local optima. A multi-strategy Historical-Cognitive Hybrid ALA (HCALA) is proposed. It integrates a dual-memory guidance mechanism combining external elite archives with individual historical bests to enhance search reliability and local refinement, a success-memory-driven adaptive branch selection mechanism to balance exploration and exploitation, and a late-stage hybrid exploitation strategy using structured mutation, recombination, and adaptive local perturbation to improve escape from local optima. Extensive experiments on 30 benchmark functions and CEC2022 test functions, supported by Wilcoxon rank-sum analysis, demonstrate that HCALA outperforms classical algorithms and other improved variants in convergence speed, solution accuracy, stability, and robustness. Applications to engineering problems further validate its practical effectiveness.

The HCALA algorithm demonstrates superior UAV path planning in complex 3D environments with multiple obstacles, consistently generating shorter, smoother, and safer trajectories. Its rapid convergence, reliable obstacle avoidance, and stable performance underscore its effectiveness and practical value for real-world UAV applications.Although HCALA maintains rapid convergence in high-dimensional multimodal problems, terminal solution precision can still be improved, motivating future integration with additional metaheuristic or adaptive strategies.

Supplementary information. If your article has accompanying supplementary file/s please state so here.

Authors reporting data from electrophoretic gels and blots should supply the full unprocessed scans for key as part of their Supplementary information. This may be requested by the editorial team/s if it is missing.

Please refer to Journal-level guidance for any specific requirements.

Acknowledgements. Acknowledgements are not compulsory. Where included they should be brief. Grant or contribution numbers may be acknowledged.

Please refer to Journal-level guidance for any specific requirements.

Declarations

Some journals require declarations to be submitted in a standardised format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading 'Declarations':

- Funding
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use)
- Ethics approval and consent to participate
- Consent for publication
- Data availability
- Materials availability
- Code availability
- Author contribution

If any of the sections are not relevant to your manuscript, please include the heading and write 'Not applicable' for that section.

Editorial Policies for:

Springer journals and proceedings: https://www.springer.com/gp/editorial-policies

Nature Portfolio journals: https://www.nature.com/nature-research/editorial-policies

Scientific Reports: https://www.nature.com/srep/journal-policies/editorial-policies

BMC journals: https://www.biomedcentral.com/getpublished/editorial-policies

Appendix A Section title of first appendix

All the data are placed at the end of this article.

References

[1] Naruei and F. Keynia, "Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems," *Engineering with Computers*, vol. 38, Suppl. 4, pp. 3025–3056, 2022.

[2] W. Zhao, L. Wang, and S. Mirjalili, "Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications," *Computer Methods in Applied Mechanics and Engineering*, vol. 388, p. 114194, 2022.

[3] Y. Meraihi, A. Ramdane-Cherif, D. Acheli, et al., "Dragonfly algorithm: A comprehensive review and applications," *Neural Computing and Applications*, vol. 32, pp. 16625–16646, 2020.

[4] X. Sun, J. Zhang, Y. Gu, and B. Zhang, "Discrete chimp optimization algorithm based on neighbor traction operator," *Control and Decision*, vol. 39, no. 4, pp. 1133–1141, 2024.

[5] A. Seyyedabbasi and F. Kiani, "Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems," *Engineering with Computers*, vol. 39, no. 4, pp. 2627–2651, 2023.

[6] L. Abualigah and A. Diabat, "Advances in sine cosine algorithm: A comprehensive survey," *Artificial Intelligence Review*, vol. 54, no. 4, pp. 2567–2608, 2021.

[7] Z. Mao, W. Wang, K. Fang, et al., "Research on the balance problem of human-machine collaborative assembly line based on simulated annealing algorithm," *Control and Decision*, 2024. [Online]. Available: https://doi.org/10.13195/j.kzyjc.2023.1068

[8] K. Fan, G. Yang, D. Wu, et al., "Improved gravitational search algorithm based on multi-strategy fusion," *Application Research of Computers*, vol. 40, no. 12, pp. 3592–3598, 2023.

[9] A. Tharwat, M. Elhoseny, A. E. Hassanien, T. Gabel, and A. Kumar, "Intelligent Bézier curve-based path planning model using chaotic particle swarm optimization algorithm," *Cluster Computing*, vol. 22, pp. 4745–4766, 2019.

[10] Y. Wang, G. Li, C. Wang, and S. Li, "Ship structure optimization method based on adaptive mutation particle swarm algorithm," *Chinese Journal of Ship Research*, vol. 17, no. 2, pp. 156–164, 2022.

[11] W. Yue, Y. Xi, and X. Guan, "A new searching approach using improved multi-ant colony scheme for multi-UAVs in unknown environments," *IEEE Access*, vol. 7, pp. 161094–161102, 2019.

[12] P. Jian and C. Zhongqing, "Image matching based on DCT transform and chicken swarm optimization algorithm," *Computer and Digital Engineering*, vol. 48, no. 6, pp. 1450–1468, 2020.

[13] X. Bao, H. Gao, R. Zhou, et al., "Design of power system low-frequency oscillation damping strategy based on grey wolf optimization algorithm," *China Electrical Equipment Industry*, no. 4, pp. 35–39, 2025.

[14] C. Lu, L. Gao, and J. Yi, "Grey wolf optimizer with cellular topological structure," *Expert Systems with Applications*, vol. 107, pp. 89–114, 2018.

[15] Y. Xiao, H. Cui, R. A. Khurma, et al., "Artificial lemming algorithm: A novel bionic meta-heuristic technique for solving real-world engineering optimization problems," *Artificial Intelligence Review*, vol. 58, no. 3, p. 84, 2025.

[16] Q. He and S. Li, "Chimp optimization algorithm with hybrid improvement strategy and its mechanical applications," *Control and Decision*, vol. 38, no. 2, pp. 354–364, 2023.

[17] W. Wang, Y. Jin, J. Chen, G. Li, M. Hu, and J. Deng, "Multi-role multi-strategy multi-objective particle swarm optimization algorithm," *Journal of Zhejiang University (Engineering Science)*, vol. 56, no. 3, pp. 531–541, 2022.

[18] Q. Li and Y. Sun, "A hybrid grey wolf optimization algorithm based on teaching-learning," *Control and Decision*, vol. 37, no. 12, pp. 3190–3196, 2022.

[19] A. Li, L. Qian, G. Chen, and S. Xu, "Sparrow search algorithm integrated with sine cosine and Cauchy mutation," *Computer Engineering and Applications*, vol. 58, no. 3, pp. 91–99, 2022.

[20] Y. Hou, H. Gao, Z. Wang, and C. Du, "Improved grey wolf optimization algorithm and application," *Sensors*, vol. 22, no. 10, 2022.

[21] E. H. Houssein, S. Abdel, F. Mahmoud, et al., "Dynamic Coati optimization algorithm for biomedical classification tasks," *Computers in Biology and Medicine*, vol. 164, p. 107237, 2023.

[22] X. Lyu, X. Mu, J. Zhang, and Z. Wang, "Chaos sparrow search optimization algorithm," *Journal of Beijing University of Aeronautics and Astronautics*, vol. 47, no. 8, pp. 1712–1720, 2021.

[23] S. A. F. Sayed, E. Nabil, and A. Badr, "A binary clonal flower pollination algorithm for feature selection," *Pattern Recognition Letters*, vol. 77, pp. 21–27, 2016.

[24] Y. Lu and J. Zhang, "Short-term load forecasting of VMD-KELM based on improved predator-prey algorithm," *Proceedings of the Chinese Society for Electrical Engineering*, vol. 18, no. 4, pp. 228–238, 2023.

[25] P. Qin, H. Hu, and Z. Yang, "The improved grasshopper optimization algorithm and its applications," *Scientific Reports*, vol. 11, no. 1, 2021.

[26] Awad N H, Ali M Z, Suganthan P N, Liang J J and Qu B 2016 *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization* Technical Report (Nanyang Technological University, Singapore)

[27] Kumar A, Price K V, Mohamed A W, Hadi A A and Suganthan P N 2022 *Problem Definitions and Evaluation Criteria for the CEC 2022 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization* Technical Report (Nanyang Technological University, Singapore)

[28] Kennedy J and Eberhart R 1995 Particle swarm optimization *Proc. IEEE Int. Conf. on Neural Networks* vol 4 pp 1942–1948

[29] Storn R and Price K 1997 Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces *Journal of Global Optimization* **11** 341–359

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

[31] Mirjalili S and Lewis A 2016 The whale optimization algorithm *Advances in Engineering Software* **95** 51–67

[32] Mirjalili S, Mirjalili S M and Lewis A 2014 Grey wolf optimizer *Advances in Engineering Software* **69** 46–61

[33] Mirjalili S 2016 SCA: A sine cosine algorithm for solving optimization problems *Knowledge-Based Systems* **96** 120–133

[34] Minh H-L, Sang-To T, Theraulaz G, Abdel Wahab M and Cuong-Le T 2023 Termite life cycle optimizer *Expert Systems with Applications* **213** 119211

[35] Dhiman G and Kumar V 2019 Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems *Knowledge-Based Systems* **165** 169–196

[36] Yuan C, Zhao D, Heidari A A, Liu L, Chen Y and Chen H 2024 Polar lights optimizer: Algorithm and applications in image segmentation and feature selection *Neurocomputing* **607** 128427

[37] Xiao Y, Cui H, Khurma R A, Castaño F and Castillo P A 2025 Artificial lemming algorithm: A novel bionic meta-heuristic technique for solving real-world engineering optimization problems *Artificial Intelligence Review* **58** 84

[38] Zhu X, Jia C, Zhao J, Xia C, Peng W, Huang J and Li L 2025 An enhanced artificial lemming algorithm and its application in UAV path planning *Biomimetics* **10** 377

[39] Zheng X, Liu R, Huang S and Duan Z 2026 IALA: An improved artificial lemming algorithm for unmanned aerial vehicle path planning *Technologies* **14** 91

[40] Mohammed H, Rashid T. A novel hybrid GWO with WOA for global numerical optimization and solving pressure vessel design[J]. Neural Computing and Applications, 2020, 32(18): 14701-14718.

[41] Tzanetos A, Blondin M. A qualitative systematic review of metaheuristics applied to tension/compression spring design problem: Current situation, recommendations, and research direction[J]. Engineering Applications of Artificial Intelligence, 2023, 118: 105521.

[42] Kamil A T, Saleh H M, Abd-Alla I H. A multi-swarm structure for particle swarm optimization: Solving the welded beam design problem[C]//Journal of Physics: Conference Series. IOP Publishing, 2021, 1804(1): 012012.

[43] Coello C A C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art[J]. Computer Methods in Applied Mechanics and Engineering, 2002, 191(11-12): 1245-1287.

[44] Dong M, Wang N, Cheng X, et al. Composite differential evolution with modified oracle penalty method for constrained optimization problems[J]. Mathematical Problems in Engineering, 2014, 2014.

[45] Fletcher R, Leyffer S. Nonlinear programming without a penalty function[J]. Mathematical Programming, 2002, 91: 239-269.

[46] Coello C A C. Evolutionary multi-objective optimization: a historical view of the field[J]. IEEE Computational Intelligence Magazine, 2006, 1(1): 28-36.

[47] Gunantara N. A review of multi-objective optimization: Methods and its applications[J]. Cogent Engineering, 2018, 5(1): 1502242.

Table A1 Detailed mean and standard deviation results on CEC2017 with 30D (Part 1/2).

Func.	Stat.	PSO	DE	HHO	WOA	GWO	SCA	TLCO	SOA	PLO	ALA	EALA	IALA	HCALA
f_1	mean	1.2447E+09	1.3250E+06	4.6911E+08	4.6378E+09	2.8378E+09	2.1518E+10	9.9954E+08	1.0695E+09	2.0761E+09	3.6581E+06	7.5305E+03	8.5221E+03	2.6371E+03
	std	2.1330E+09	8.4078E+05	2.1008E+08	1.1746E+09	2.1749E+09	3.0126E+09	5.3784E+08	3.2442E+08	4.5108E+08	3.5885E+06	7.3881E+03	8.1545E+03	3.7947E+03
f_3	mean	6.2212E+04	1.6620E+05	5.8080E+04	2.6070E+05	6.4775E+04	8.8550E+04	5.8711E+04	7.8626E+04	1.5939E+05	2.7997E+04	1.3777E+05	9.6464E+03	1.2602E+04
	std	3.1892E+04	2.9389E+04	7.8729E+03	8.4944E+04	1.1920E+04	2.0540E+04	1.2042E+04	2.1915E+04	2.2657E+04	8.7962E+03	3.1038E+04	3.4983E+03	6.2241E+03
f_4	mean	6.6775E+02	5.2148E+02	7.0658E+02	1.2772E+03	6.8102E+02	3.0189E+03	6.8214E+02	6.9034E+02	6.0993E+02	5.2175E+02	4.9567E+02	5.1244E+02	5.0403E+02
	std	1.7222E+02	1.8562E+01	1.2481E+02	2.8714E+02	1.9765E+02	9.1135E+02	7.8314E+01	6.2793E+01	2.9121E+01	2.1919E+01	1.3229E+01	3.8569E+01	3.0286E+01
f_5	mean	6.6573E+02	6.9703E+02	7.7280E+02	8.6361E+02	6.3638E+02	8.2416E+02	6.7268E+02	6.7625E+02	6.8874E+02	6.1246E+02	7.1621E+02	6.5616E+02	5.6959E+02
	std	3.9782E+01	1.1225E+01	3.2452E+01	6.5219E+01	4.6157E+01	2.9018E+01	3.4839E+01	2.2818E+01	1.3841E+01	3.3705E+01	3.8514E+01	4.3383E+01	1.6371E+01
f_6	mean	6.4086E+02	6.0019E+02	6.6776E+02	6.7850E+02	6.1214E+02	6.6425E+02	6.2412E+02	6.1720E+02	6.3233E+02	6.1275E+02	6.0116E+02	6.2896E+02	6.0921E+02
	std	8.4400E+00	5.0354E-02	6.0185E+00	1.4217E+01	4.6554E+00	5.5589E+00	6.2332E+00	3.8134E+00	4.8736E+00	4.8105E+00	9.0290E-01	7.9526E+00	2.5247E+00
f_7	mean	9.2125E+02	9.2920E+02	1.3269E+03	1.3033E+03	9.0497E+02	1.2565E+03	1.0116E+03	1.0471E+03	1.1007E+03	9.0053E+02	9.6848E+02	1.0481E+03	8.3640E+02
	std	5.1198E+01	1.6810E+01	6.5024E+01	8.9589E+01	4.1083E+01	5.2119E+01	3.4860E+01	5.7813E+01	3.9948E+01	4.5275E+01	2.7244E+01	9.6684E+01	2.7547E+01
f_8	mean	9.3189E+02	9.9330E+02	9.8213E+02	1.0799E+03	9.0351E+02	1.1039E+03	9.5180E+02	9.4719E+02	9.7976E+02	9.0598E+02	1.0295E+03	9.5154E+02	8.7671E+02
	std	3.8517E+01	1.2708E+01	2.9377E+01	6.6233E+01	2.9381E+01	2.2039E+01	2.5383E+01	2.2524E+01	1.7111E+01	3.1683E+01	2.4057E+01	3.6436E+01	1.8860E+01
f_9	mean	4.4325E+03	2.1930E+03	9.0994E+03	1.1835E+04	2.7954E+03	8.3458E+03	5.1272E+03	2.9339E+03	5.5306E+03	2.3941E+03	1.0197E+03	3.5912E+03	1.4090E+03
	std	1.3428E+03	5.4091E+02	1.0977E+03	4.6991E+03	9.7844E+02	1.6950E+03	1.5262E+03	5.5415E+02	1.2567E+03	1.1176E+03	2.6081E+02	1.0285E+03	2.1375E+02
f_{10}	mean	5.0364E+03	7.6940E+03	5.9475E+03	7.5779E+03	5.1900E+03	8.8852E+03	5.5505E+03	5.8815E+03	6.3166E+03	5.9817E+03	7.3631E+03	4.9257E+03	5.1818E+03
	std	7.5508E+02	2.5679E+02	5.9154E+02	4.9800E+02	1.6858E+03	3.4010E+02	5.8427E+02	6.6008E+02	3.9022E+02	9.2977E+02	1.6392E+03	6.8966E+02	7.9393E+02
f_{11}	mean	1.3037E+03	1.7863E+03	1.5978E+03	8.8498E+03	2.3475E+03	4.1499E+03	1.5313E+03	2.0644E+03	2.6144E+03	1.2772E+03	1.2847E+03	1.2912E+03	1.2385E+03
	std	6.4414E+01	3.5080E+02	1.5035E+02	5.0479E+03	1.0456E+03	1.1192E+03	1.9374E+02	5.7969E+02	5.0960E+02	4.4826E+01	6.6607E+01	8.7997E+01	4.9921E+01
f_{12}	mean	1.6027E+08	3.7057E+07	7.6670E+07	4.9019E+08	8.8633E+07	2.7509E+09	2.8716E+07	3.3496E+07	3.9891E+07	2.4571E+06	9.4896E+05	3.5062E+05	7.8145E+05
	std	7.2661E+08	2.0447E+07	4.7332E+07	3.3312E+08	9.6375E+07	8.0533E+08	1.5491E+07	1.7138E+07	1.3308E+07	1.7997E+06	8.2730E+05	4.2880E+05	5.8736E+05
f_{13}	mean	2.7338E+06	3.3130E+06	9.9293E+05	1.3311E+07	2.3199E+07	1.1369E+09	5.7453E+05	1.8485E+06	1.2764E+06	4.6540E+04	1.1724E+05	2.2501E+04	1.7138E+04
	std	1.3073E+07	2.2182E+06	5.7630E+05	1.1923E+07	4.9219E+07	4.4047E+08	1.6737E+06	1.5332E+06	6.0495E+05	3.0815E+04	2.2092E+05	1.9688E+04	2.1149E+04
f_{14}	mean	6.0576E+04	4.8600E+05	9.9186E+05	2.0335E+06	3.1491E+05	9.0483E+05	4.1633E+05	7.2881E+05	8.2612E+04	1.8622E+03	4.9991E+04	3.7102E+03	1.6078E+03
	std	9.1927E+04	2.9922E+05	9.3414E+05	1.7812E+06	4.4437E+05	6.0969E+05	4.6911E+05	8.4913E+05	5.9439E+04	2.0758E+02	2.9569E+04	6.5592E+03	6.9302E+01
f_{15}	mean	1.5570E+04	7.1040E+05	1.4674E+05	1.1090E+07	1.4087E+06	6.4645E+07	9.8942E+03	1.1652E+05	1.1749E+05	2.1158E+04	1.7214E+05	1.1488E+04	1.4680E+04
	std	1.2439E+04	3.5569E+05	9.0298E+04	1.7966E+07	2.1063E+06	5.1221E+07	7.8243E+03	9.5166E+04	4.5940E+04	1.1641E+04	2.7893E+05	9.8966E+03	1.1102E+04
f_{16}	mean	2.7349E+03	3.0331E+03	3.6264E+03	4.4674E+03	2.5314E+03	4.1055E+03	2.7806E+03	2.8260E+03	2.8177E+03	2.7151E+03	3.4162E+03	2.5847E+03	2.3978E+03
	std	2.8961E+02	2.0950E+02	3.6308E+02	7.0456E+02	3.1868E+02	2.5812E+02	2.1949E+02	2.9947E+02	2.1561E+02	3.4631E+02	3.8330E+02	3.0503E+02	2.9214E+02

Table A2 Detailed mean and standard deviation results on CEC2017 with 30D (Part 2/2).

Func.	Stat.	PSO	DE	HHO	WOA	GWO	SCA	TLCO	SOA	PLO	ALA	EALA	IALA	HCALA
f_{17}	mean	2.3076E+03	2.1730E+03	2.8317E+03	2.7638E+03	2.0586E+03	2.8308E+03	2.1288E+03	2.1617E+03	2.0862E+03	2.1950E+03	2.2518E+03	2.2980E+03	2.0283E+03
	std	2.7418E+02	1.3739E+02	2.9577E+02	2.6493E+02	1.6350E+02	1.7064E+02	1.8553E+02	1.7191E+02	1.0476E+02	1.8013E+02	2.5515E+02	2.4547E+02	1.2894E+02
f_{18}	mean	2.4681E+05	3.2645E+06	3.1975E+06	1.3366E+07	2.0036E+06	1.5191E+07	1.6696E+06	1.7479E+06	7.5256E+05	7.7950E+04	2.0082E+06	4.7170E+04	2.7608E+04
	std	1.9436E+05	1.2115E+06	5.3393E+06	1.5174E+07	2.6895E+06	8.7943E+06	2.3051E+06	1.5943E+06	4.4088E+05	4.0152E+04	1.6294E+06	5.4128E+04	4.0684E+04
f_{19}	mean	4.3554E+04	6.5379E+05	1.6803E+06	2.5098E+07	2.1223E+06	1.0416E+08	9.7437E+03	1.2267E+05	3.5417E+05	2.3439E+04	6.2393E+04	1.2272E+04	1.0463E+04
	std	1.7981E+05	4.7513E+05	1.4709E+06	3.3474E+07	6.6790E+06	5.3731E+07	8.9763E+03	1.4225E+05	1.9359E+05	2.1937E+04	1.4234E+05	1.0064E+04	1.1183E+04
f_{20}	mean	2.6143E+03	2.4925E+03	2.9158E+03	2.9441E+03	2.5285E+03	2.9425E+03	2.4315E+03	2.4032E+03	2.4569E+03	2.4884E+03	2.6336E+03	2.4946E+03	2.4207E+03
	std	1.8199E+02	1.0355E+02	1.7662E+02	2.3653E+02	2.1124E+02	1.5903E+02	2.0877E+02	1.6817E+02	1.2154E+02	1.7655E+02	2.7889E+02	1.7585E+02	1.9958E+02
f_{21}	mean	2.4570E+03	2.4954E+03	2.6020E+03	2.6466E+03	2.4074E+03	2.6077E+03	2.4454E+03	2.4450E+03	2.4727E+03	2.4168E+03	2.5212E+03	2.4446E+03	2.3776E+03
	std	3.9413E+01	1.3881E+01	5.4340E+01	7.0963E+01	3.0863E+01	2.8894E+01	4.5516E+01	2.8997E+01	1.7771E+01	3.1786E+01	2.1870E+01	3.3460E+01	2.3930E+01
f_{22}	mean	4.5472E+03	6.1803E+03	6.6264E+03	8.2134E+03	5.5135E+03	9.8461E+03	4.0164E+03	4.2068E+03	3.7815E+03	6.2067E+03	7.4992E+03	5.6504E+03	5.5060E+03
	std	2.1395E+03	1.9135E+03	1.8927E+03	1.7801E+03	1.8867E+03	1.4724E+03	2.3106E+03	2.4648E+03	1.6369E+03	2.1684E+03	3.2089E+03	1.7778E+03	2.1706E+03
f_{23}	mean	3.0224E+03	2.8394E+03	3.2750E+03	3.1540E+03	2.7945E+03	3.0808E+03	2.8628E+03	2.8138E+03	2.8104E+03	2.7686E+03	2.8384E+03	2.8671E+03	2.7277E+03
	std	1.1873E+02	1.1545E+01	1.3202E+02	1.1029E+02	5.0127E+01	5.0593E+01	3.8234E+01	2.7292E+01	2.1195E+01	3.2654E+01	5.4889E+01	6.5017E+01	2.1501E+01
f_{24}	mean	3.1665E+03	3.0332E+03	3.5337E+03	3.2542E+03	2.9680E+03	3.2449E+03	3.0513E+03	2.9908E+03	2.9801E+03	2.9526E+03	3.0385E+03	3.0212E+03	2.8935E+03
	std	9.3643E+01	1.7424E+01	1.2980E+02	8.7786E+01	6.5560E+01	3.9205E+01	4.2594E+01	2.5646E+01	1.7013E+01	3.7610E+01	4.7050E+01	5.6755E+01	1.4840E+01
f_{25}	mean	2.9729E+03	2.8966E+03	3.0127E+03	3.2332E+03	3.0550E+03	3.5757E+03	3.0142E+03	3.0830E+03	3.0096E+03	2.9119E+03	2.8882E+03	2.9138E+03	2.9006E+03
	std	5.6758E+01	5.7276E+00	3.2877E+01	8.2316E+01	8.6960E+01	1.9311E+02	4.2925E+01	7.7122E+01	3.3370E+01	2.0049E+01	4.4509E+00	2.7526E+01	1.5535E+01
f_{26}	mean	6.1647E+03	5.4856E+03	8.2693E+03	8.6354E+03	4.9723E+03	7.9521E+03	5.4240E+03	5.8036E+03	5.3412E+03	5.0554E+03	5.6759E+03	5.9306E+03	4.4184E+03
	std	1.2970E+03	1.4260E+02	1.0903E+03	7.9549E+02	4.1320E+02	4.4125E+02	1.0117E+03	8.3088E+02	3.9057E+02	3.6223E+02	5.9446E+02	1.0631E+03	2.5713E+02
f_{27}	mean	3.3387E+03	3.2295E+03	3.5878E+03	3.5342E+03	3.2684E+03	3.5536E+03	3.3267E+03	3.2663E+03	3.2460E+03	3.2359E+03	3.2224E+03	3.2724E+03	3.2299E+03
	std	9.0663E+01	4.5042E+00	1.9492E+02	1.6945E+02	3.2379E+01	6.5254E+01	3.5563E+01	1.7923E+01	7.1733E+00	1.7340E+01	1.7674E+01	3.1721E+01	1.4102E+01
f_{28}	mean	3.4146E+03	3.3048E+03	3.4877E+03	3.8709E+03	3.4763E+03	4.4314E+03	3.4625E+03	3.4750E+03	3.4326E+03	3.5151E+03	3.2390E+03	3.2579E+03	3.2490E+03
	std	1.5518E+02	2.4793E+01	1.0128E+02	2.4402E+02	1.0063E+02	3.3953E+02	6.4804E+01	7.4663E+01	5.7327E+01	7.6418E+02	2.8428E+01	4.1172E+01	2.7125E+01
f_{29}	mean	4.2250E+03	4.1943E+03	5.0235E+03	5.5336E+03	4.0271E+03	5.3076E+03	4.0112E+03	3.9189E+03	3.9795E+03	4.0183E+03	3.8359E+03	4.1576E+03	3.7851E+03
	std	2.3705E+02	1.1420E+02	4.5901E+02	5.5202E+02	1.7830E+02	3.3746E+02	2.6761E+02	1.9745E+02	1.4352E+02	2.1283E+02	2.5005E+02	2.9303E+02	1.7641E+02
f_{30}	mean	2.6592E+05	5.1061E+05	1.1585E+07	8.5130E+07	1.3453E+07	1.9283E+08	5.1274E+05	2.3422E+05	1.1536E+06	6.4341E+04	9.1695E+04	1.4030E+04	1.4896E+04
	std	4.3397E+05	3.4455E+05	1.0101E+07	7.4682E+07	1.3040E+07	9.1625E+07	4.0717E+05	1.3057E+05	5.0004E+05	8.8983E+04	1.0726E+05	9.1975E+03	6.6034E+03

Table A3 Detailed mean and standard deviation results on CEC2017 with 50D (Part 1/2).

Func.	Stat.	PSO	DE	HHO	WOA	GWO	SCA	TLCO	SOA	PLO	ALA	EALA	IALA	HCALA
f_1	mean	6.7194E+09	6.3249E+07	5.4200E+09	2.1463E+10	1.2014E+10	6.6322E+10	9.9733E+09	8.0909E+09	1.6254E+10	5.2496E+08	5.6462E+06	7.4276E+05	8.2779E+06
	std	4.6071E+09	5.3883E+07	1.6873E+09	4.5791E+09	6.3185E+09	6.2670E+09	2.9327E+09	1.7257E+09	2.6481E+09	2.2844E+08	3.5312E+06	7.7831E+05	7.0794E+06
f_3	mean	2.1154E+05	3.4080E+05	1.7214E+05	3.0518E+05	1.6830E+05	2.1034E+05	1.6506E+05	1.9758E+05	3.1854E+05	1.1347E+05	3.4688E+05	8.3776E+04	7.1225E+04
	std	5.6690E+04	4.2151E+04	1.9766E+04	1.1355E+05	3.2400E+04	4.2965E+04	2.2135E+04	2.7168E+04	4.9755E+04	2.0032E+04	4.8689E+04	1.9623E+04	1.6284E+04
f_4	mean	1.3535E+03	7.3140E+02	1.7218E+03	4.6907E+03	1.3128E+03	1.4634E+04	1.9175E+03	1.7647E+03	1.6806E+03	7.2646E+02	6.1813E+02	6.5231E+02	6.5358E+02
	std	3.5283E+02	3.3974E+01	3.0736E+02	1.2548E+03	3.5814E+02	3.5232E+03	5.0380E+02	3.4084E+02	2.1537E+02	5.4736E+01	6.6270E+01	7.8256E+01	5.5279E+01
f_5	mean	8.2537E+02	9.2568E+02	9.2392E+02	1.1029E+03	7.6018E+02	1.1281E+03	9.2100E+02	9.1718E+02	9.5236E+02	7.8523E+02	9.4937E+02	8.1300E+02	6.4883E+02
	std	5.2988E+01	1.5613E+01	2.4868E+01	7.8711E+01	3.2922E+01	3.8298E+01	4.5133E+01	4.7651E+01	3.3188E+01	6.7424E+01	5.3388E+01	5.5720E+01	4.3433E+01
f_6	mean	6.5486E+02	6.0276E+02	6.7842E+02	6.9401E+02	6.2538E+02	6.8555E+02	6.4428E+02	6.3587E+02	6.5498E+02	6.2965E+02	6.0833E+02	6.4272E+02	6.1858E+02
	std	8.0722E+00	5.1786E-01	4.2026E+00	7.8472E+00	5.4978E+00	5.9698E+00	7.1139E+00	5.6563E+00	5.1502E+00	8.0125E+00	3.7658E+00	7.7064E+00	3.7658E+00
f_7	mean	1.3488E+03	1.2088E+03	1.9143E+03	1.9142E+03	1.1484E+03	1.8933E+03	1.4772E+03	1.5846E+03	1.9048E+03	1.2028E+03	1.2538E+03	1.3578E+03	1.1082E+03
	std	1.2133E+02	1.5169E+01	7.1737E+01	9.0754E+01	8.5437E+01	9.8895E+01	1.0624E+02	1.0395E+02	8.5988E+01	7.8192E+01	4.2112E+01	1.1385E+02	9.2048E+01
f_8	mean	1.1361E+03	1.2392E+03	1.2340E+03	1.3940E+03	1.0578E+03	1.4358E+03	1.2138E+03	1.1991E+03	1.2515E+03	1.0834E+03	1.2681E+03	1.1392E+03	9.3962E+02
	std	5.3732E+01	1.6212E+01	2.8631E+01	8.5558E+01	3.5022E+01	3.9768E+01	4.8684E+01	4.4690E+01	3.7316E+01	5.3199E+01	3.8286E+01	5.9902E+01	3.7499E+01
f_9	mean	1.4118E+04	9.1019E+03	3.1990E+04	4.0807E+04	1.1329E+04	3.3000E+04	1.9748E+04	1.4046E+04	2.4065E+04	8.7110E+03	4.8426E+03	1.2474E+04	3.4795E+03
	std	3.9807E+03	1.3491E+03	3.1021E+03	1.2282E+04	5.4355E+03	5.2130E+03	4.4620E+03	4.3454E+03	4.4880E+03	2.3113E+03	2.8653E+03	3.1952E+03	8.2087E+02
f_{10}	mean	8.1572E+03	1.4150E+04	1.0344E+04	1.3220E+04	8.7811E+03	1.5429E+04	1.0501E+04	1.0820E+04	1.1985E+04	1.1968E+04	1.4424E+04	8.1853E+03	8.7913E+03
	std	8.7023E+02	3.4484E+02	1.0967E+03	1.0409E+03	2.1892E+03	4.4319E+02	8.9352E+02	8.3783E+02	4.1585E+02	1.6419E+03	2.0948E+03	1.3084E+03	1.8955E+03
f_{11}	mean	1.8480E+03	7.0018E+03	2.9622E+03	8.6256E+03	6.8118E+03	1.2861E+04	4.0228E+03	5.4469E+03	1.2109E+04	1.8846E+03	2.4206E+03	1.4511E+03	1.4309E+03
	std	3.1820E+02	2.0816E+03	5.7241E+02	1.9851E+03	2.4805E+03	2.6824E+03	1.3113E+03	2.0658E+03	3.0321E+03	2.5074E+02	1.0895E+03	1.1013E+02	1.1061E+02
f_{12}	mean	1.6882E+09	4.7003E+08	1.0137E+09	4.6513E+09	2.1197E+09	2.2384E+10	8.4470E+08	7.3174E+08	9.3588E+08	4.5902E+07	1.2351E+07	1.5961E+07	1.0249E+07
	std	1.7937E+09	1.3720E+08	6.7929E+08	1.7437E+09	1.9041E+09	4.5565E+09	3.8880E+08	2.1673E+08	1.8085E+08	3.3715E+07	8.6592E+06	2.1854E+07	7.1984E+06
f_{13}	mean	6.8331E+08	1.3180E+07	3.2780E+07	5.2318E+08	5.3106E+08	6.2930E+09	3.6024E+07	4.3514E+07	5.5640E+07	4.4232E+04	1.8783E+04	1.6549E+04	1.1658E+04
	std	1.5438E+09	9.7842E+06	3.9707E+07	3.5989E+08	9.7029E+08	2.3069E+09	2.0005E+07	2.2214E+07	1.9679E+07	2.2042E+04	1.2447E+04	1.1015E+04	8.0054E+03
f_{14}	mean	4.7190E+05	3.2341E+06	5.1182E+06	6.9567E+06	1.2661E+06	7.9519E+06	2.6568E+06	2.6095E+06	1.1088E+06	7.2890E+04	4.0422E+05	3.4851E+04	3.1278E+04
	std	9.8511E+05	2.0216E+06	4.1390E+06	5.5347E+06	1.1186E+06	3.8933E+06	2.4232E+06	2.4666E+06	6.0691E+05	7.3210E+04	2.7829E+05	4.7064E+04	5.7008E+04
f_{15}	mean	2.7599E+04	1.6119E+06	1.4055E+06	6.9606E+07	2.5271E+07	9.9684E+08	5.6080E+05	6.7662E+06	2.3353E+06	2.5485E+04	3.7310E+04	1.1394E+04	1.4192E+04
	std	1.6629E+04	1.7320E+06	7.1539E+05	9.0021E+07	3.3356E+07	4.1924E+08	6.5337E+05	3.7233E+06	1.4471E+06	1.3401E+04	2.5678E+04	8.6214E+03	8.8117E+03
f_{16}	mean	3.6662E+03	4.9239E+03	4.9119E+03	6.3663E+03	3.3308E+03	6.4059E+03	3.8456E+03	3.6699E+03	4.1524E+03	3.9690E+03	4.9803E+03	3.6615E+03	3.3178E+03
	std	5.9017E+02	3.5638E+02	6.4468E+02	1.0034E+03	4.7098E+02	4.8288E+02	5.3232E+02	4.7582E+02	2.7331E+02	5.9609E+02	8.1359E+02	3.5958E+02	3.9950E+02

Table A4 Detailed mean and standard deviation results on CEC2017 with 50D (Part 2/2).

Func.	Stat.	PSO	DE	HHO	WOA	GWO	SCA	TLCO	SOA	PLO	ALA	EALA	IALA	HCALA
f_{17}	mean	3.4561E+03	3.6315E+03	3.7396E+03	4.7172E+03	3.1834E+03	5.1219E+03	3.3209E+03	3.2549E+03	3.6196E+03	3.4052E+03	4.2070E+03	3.4335E+03	3.0510E+03
	std	3.8208E+02	1.8383E+02	3.6400E+02	6.0651E+02	4.3918E+02	4.9975E+02	3.9278E+02	2.7989E+02	2.1023E+02	4.8217E+02	3.4641E+02	4.2899E+02	3.4911E+02
f_{18}	mean	1.1049E+06	2.1461E+07	9.1527E+06	6.3240E+07	8.8998E+06	4.9085E+07	7.6739E+06	1.0157E+07	6.8328E+06	7.7375E+05	5.7861E+06	5.9523E+05	2.9741E+05
	std	6.6158E+05	8.7441E+06	6.8167E+06	5.1305E+07	9.9106E+06	3.1231E+07	5.7725E+06	6.7289E+06	3.2473E+06	9.9304E+05	4.6597E+06	9.4609E+05	2.4491E+05
f_{19}	mean	4.8871E+05	7.4224E+05	1.7648E+06	2.0560E+07	1.0380E+07	7.0932E+08	3.8194E+05	5.6150E+05	2.4370E+06	1.9607E+04	2.0997E+04	1.9252E+04	1.8258E+04
	std	1.3086E+06	4.4949E+05	1.2422E+06	1.4784E+07	2.2412E+07	3.4132E+08	3.6621E+05	3.7306E+05	7.0278E+05	1.6851E+04	2.4119E+04	1.4734E+04	1.3860E+04
f_{20}	mean	3.3704E+03	3.6803E+03	3.4477E+03	4.0087E+03	3.0790E+03	4.3631E+03	3.2400E+03	3.2047E+03	3.4343E+03	3.6916E+03	4.3143E+03	3.2036E+03	3.1978E+03
	std	3.6429E+02	1.6874E+02	3.2882E+02	3.1307E+02	3.9084E+02	1.8549E+02	2.2820E+02	2.0909E+02	2.1121E+02	3.1571E+02	2.5411E+02	2.7371E+02	2.8630E+02
f_{21}	mean	2.6752E+03	2.7302E+03	2.9102E+03	3.0865E+03	2.5760E+03	2.9514E+03	2.7075E+03	2.6721E+03	2.7347E+03	2.5736E+03	2.7554E+03	2.6306E+03	2.4457E+03
	std	7.7388E+01	1.9906E+01	7.1257E+01	1.2042E+02	7.8422E+01	5.4522E+01	5.0598E+01	4.1602E+01	3.3219E+01	5.4547E+01	4.9038E+01	5.5097E+01	3.3246E+01
f_{22}	mean	1.0181E+04	1.5820E+04	1.2639E+04	1.4888E+04	1.0396E+04	1.7126E+04	1.1936E+04	1.2619E+04	1.3298E+04	1.3239E+04	1.6181E+04	1.0379E+04	1.0136E+04
	std	1.4509E+03	3.6794E+02	1.2088E+03	9.8943E+02	2.5131E+03	3.8567E+02	2.4573E+03	1.8499E+03	1.0857E+03	1.3160E+03	1.7329E+03	1.2471E+03	1.6433E+03
f_{23}	mean	3.6177E+03	3.1599E+03	4.0293E+03	3.8874E+03	3.0490E+03	3.7342E+03	3.2994E+03	3.1801E+03	3.1804E+03	3.0744E+03	3.2007E+03	3.2061E+03	2.9018E+03
	std	1.7056E+02	2.2843E+01	1.8898E+02	1.6615E+02	6.8890E+01	1.0033E+02	4.5533E+01	5.6823E+01	1.9311E+01	6.1674E+01	6.9231E+01	1.2452E+02	4.1032E+01
f_{24}	mean	3.6833E+03	3.3482E+03	4.3489E+03	3.9187E+03	3.2172E+03	3.9068E+03	3.4985E+03	3.3597E+03	3.3115E+03	3.2243E+03	3.3800E+03	3.2976E+03	3.0492E+03
	std	1.7115E+02	1.6915E+01	2.7969E+02	1.6739E+02	7.4925E+01	1.0341E+02	6.1074E+01	4.6395E+01	2.7829E+01	8.0558E+01	3.9599E+01	9.9755E+01	3.9791E+01
f_{25}	mean	3.6047E+03	3.1861E+03	3.8102E+03	5.3626E+03	4.0398E+03	8.6240E+03	4.0400E+03	4.1780E+03	4.1989E+03	3.2369E+03	3.0820E+03	3.1429E+03	3.1273E+03
	std	3.0097E+02	4.1481E+01	2.4941E+02	6.0495E+02	5.2738E+02	1.0297E+03	2.4801E+02	3.6433E+02	2.5436E+02	6.0433E+01	3.2837E+01	3.8929E+01	4.1513E+01
f_{26}	mean	9.9925E+03	8.0832E+03	1.1748E+04	1.4505E+04	7.1223E+03	1.4117E+04	8.2778E+03	1.0193E+04	8.3886E+03	7.3798E+03	8.0262E+03	9.1576E+03	5.6283E+03
	std	1.5525E+03	1.8529E+02	1.4660E+03	1.2451E+03	7.4735E+02	7.6139E+02	2.1093E+03	9.9507E+02	2.6573E+02	7.2483E+02	1.0254E+03	1.1298E+03	3.7747E+02
f_{27}	mean	3.8544E+03	3.5811E+03	5.2052E+03	4.9363E+03	3.7430E+03	5.0323E+03	4.0305E+03	3.6937E+03	3.6262E+03	3.5809E+03	3.4493E+03	3.7026E+03	3.4876E+03
	std	1.9692E+02	4.9727E+01	8.6153E+02	7.8375E+02	1.3202E+02	2.1664E+02	1.5940E+02	1.0208E+02	5.9853E+01	1.3130E+02	1.3573E+02	1.2612E+02	8.6857E+01
f_{28}	mean	4.5256E+03	3.8079E+03	4.8320E+03	6.1143E+03	4.5692E+03	8.7277E+03	4.7471E+03	4.8602E+03	5.2707E+03	5.2985E+03	3.6543E+03	3.6280E+03	3.4628E+03
	std	6.0249E+02	3.2196E+02	3.9128E+02	5.5065E+02	4.2379E+02	9.1657E+02	4.0530E+02	3.5644E+02	5.9191E+02	2.3412E+03	1.0769E+03	9.2334E+02	1.0505E+02
f_{29}	mean	5.5998E+03	5.5740E+03	7.3901E+03	9.3684E+03	4.9168E+03	8.9611E+03	4.9758E+03	4.9716E+03	5.1641E+03	4.8286E+03	5.2020E+03	5.0414E+03	4.5080E+03
	std	5.3364E+02	2.0719E+02	1.5698E+03	1.5114E+03	4.0858E+02	8.2779E+02	4.1517E+02	3.8469E+02	2.3425E+02	4.7013E+02	7.6649E+02	5.1163E+02	4.7633E+02
f_{30}	mean	2.9666E+07	3.4563E+07	1.1261E+08	3.0995E+08	1.4586E+08	1.2655E+09	3.9278E+07	1.3277E+07	4.2505E+07	3.6418E+06	2.5546E+06	1.6684E+06	1.3369E+06
	std	1.1801E+07	1.1943E+07	3.8900E+07	1.1883E+08	4.8752E+07	3.9098E+08	1.5120E+07	7.5834E+06	9.8512E+06	1.8694E+06	1.1931E+06	5.8899E+05	4.5433E+05

Table A5 Detailed mean and standard deviation results on CEC2022 with 20D (Part 1/1).

Func.	Stat.	PSO	DE	HHO	WOA	GWO	SCA	TLCO	SOA	PLO	ALA	EALA	IALA	HCALA
f_1	mean	6.9397E+03	4.6047E+04	2.3491E+04	3.8200E+04	1.5661E+04	2.1414E+04	1.1271E+04	1.7901E+04	4.2477E+04	3.6768E+03	3.1078E+04	3.4151E+02	5.9674E+02
	std	4.4184E+03	7.1121E+03	7.1208E+03	1.1025E+04	5.9076E+03	5.8046E+03	4.3936E+03	5.7507E+03	9.6032E+03	1.9761E+03	1.0360E+04	5.3376E+01	3.8655E+02
f_2	mean	4.7432E+02	4.4967E+02	5.5303E+02	6.3769E+02	5.1303E+02	7.9933E+02	5.1997E+02	5.3670E+02	4.7117E+02	4.5474E+02	4.4504E+02	4.5613E+02	4.5256E+02
	std	3.4010E+01	2.2295E+00	5.3705E+01	7.7749E+01	3.8421E+01	8.4897E+01	4.4906E+01	4.8997E+01	1.1220E+01	1.3943E+01	1.4920E+01	1.7114E+01	2.0580E+01
f_3	mean	6.2907E+02	6.0001E+02	6.6315E+02	6.6871E+02	6.0721E+02	6.4797E+02	6.1160E+02	6.0825E+02	6.1908E+02	6.0474E+02	6.0010E+02	6.1904E+02	6.0387E+02
	std	1.1059E+01	2.5384E-03	1.0376E+01	1.4015E+01	4.2147E+00	6.1070E+00	3.5728E+00	2.4559E+00	3.7641E+00	3.0477E+00	1.5688E-01	7.3096E+00	1.4889E+00
f_4	mean	8.6135E+02	9.2361E+02	8.8601E+02	9.3222E+02	8.6483E+02	9.5288E+02	8.7942E+02	8.7296E+02	8.7666E+02	8.5495E+02	9.2781E+02	8.7203E+02	8.4233E+02
	std	1.8122E+01	8.9975E+00	1.8015E+01	3.7003E+01	3.1773E+01	1.6884E+01	1.8545E+01	1.5773E+01	8.8832E+00	1.8568E+01	1.4252E+01	2.4881E+01	1.3591E+01
f_5	mean	1.8472E+03	1.1494E+03	3.0232E+03	4.3871E+03	1.2511E+03	2.6464E+03	1.8877E+03	1.6409E+03	1.9261E+03	1.1713E+03	9.0117E+02	1.5811E+03	9.9570E+02
	std	5.4281E+02	1.4248E+02	3.6357E+02	1.8033E+03	2.8436E+02	6.4279E+02	5.1044E+02	2.8575E+02	3.4094E+02	2.6462E+02	1.0530E+00	3.1658E+02	4.6182E+01
f_6	mean	5.6090E+04	5.3772E+06	2.3282E+05	9.6908E+06	3.0285E+06	1.4919E+08	3.9523E+04	3.2876E+05	2.9284E+05	1.5597E+04	9.2473E+04	9.3406E+03	7.4258E+03
	std	2.7087E+05	4.2284E+06	1.3046E+05	1.4911E+07	7.6525E+06	9.0775E+07	4.5530E+04	2.7504E+05	1.4330E+05	7.9954E+03	1.9554E+05	7.2991E+03	5.4144E+03
f_7	mean	2.1296E+03	2.0619E+03	2.1976E+03	2.2323E+03	2.1069E+03	2.1656E+03	2.0765E+03	2.0650E+03	2.0951E+03	2.0764E+03	2.0717E+03	2.0880E+03	2.0572E+03
	std	6.4416E+01	9.4186E+00	6.2862E+01	7.2840E+01	7.1192E+01	2.5323E+01	3.0716E+01	1.5720E+01	2.2397E+01	3.4267E+01	4.2223E+01	4.7785E+01	1.7459E+01
f_8	mean	2.2969E+03	2.2314E+03	2.3354E+03	2.2860E+03	2.2676E+03	2.2877E+03	2.2295E+03	2.2354E+03	2.2346E+03	2.2328E+03	2.2481E+03	2.2537E+03	2.2252E+03
	std	7.5501E+01	2.1275E+00	1.1040E+02	5.6046E+01	5.4813E+01	3.8170E+01	5.9392E+00	3.0544E+01	1.9322E+00	5.2899E+00	2.2079E+01	4.5174E+01	5.2946E+00
f_9	mean	2.5061E+03	2.4808E+03	2.5477E+03	2.5966E+03	2.5322E+03	2.6074E+03	2.4953E+03	2.4877E+03	2.4889E+03	2.4808E+03	2.4808E+03	2.4808E+03	2.4808E+03
	std	3.1331E+01	1.1639E-02	3.9636E+01	5.3269E+01	2.8283E+01	3.9218E+01	5.3835E+00	3.8602E+00	3.0031E+00	3.7929E-02	3.1503E-06	4.0764E-03	6.8331E-03
f_{10}	mean	3.8599E+03	2.5878E+03	4.2910E+03	5.1344E+03	3.6456E+03	3.3996E+03	2.7487E+03	2.7609E+03	2.6277E+03	4.1395E+03	3.5714E+03	3.6289E+03	2.9297E+03
	std	8.0308E+02	1.5715E+02	6.5023E+02	8.4018E+02	6.9370E+02	1.4908E+03	3.2600E+02	3.5118E+02	4.1860E+02	7.6051E+02	1.0734E+03	4.8075E+02	6.5293E+02
f_{11}	mean	2.9660E+03	2.9001E+03	3.6463E+03	4.1050E+03	3.8501E+03	7.2858E+03	3.8865E+03	5.0428E+03	5.4403E+03	2.9147E+03	2.9000E+03	2.9040E+03	2.9000E+03
	std	1.1247E+02	3.7525E-02	5.4822E+02	6.4062E+02	5.5866E+02	1.4692E+03	4.5095E+02	4.3130E+02	5.4401E+02	8.9271E+00	8.6980E-03	1.8543E+01	2.0446E-02
f_{12}	mean	3.0374E+03	2.9477E+03	3.2297E+03	3.1195E+03	2.9737E+03	3.0852E+03	3.0257E+03	2.9680E+03	2.9558E+03	2.9592E+03	2.9529E+03	2.9816E+03	2.9490E+03
	std	6.2940E+01	3.8598E+00	1.6970E+02	1.7115E+02	1.9638E+01	3.3560E+01	2.8065E+01	1.3210E+01	4.6092E+00	2.5193E+01	1.4496E+01	3.6770E+01	9.8915E+00