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.

Dual-Stage Guided Artificial Lemming Algorithm (DALA) For Solving Engineering Application Optimization Problems

Jiayi Gao
Northeast Forestry University
Yu Zhang
zhangyu0902@163.com

Northeast Forestry University

Research Article

Keywords: Artificial Lemming Algorithm, Swarm intelligence, Multi-strategy, Engineering problems

Posted Date: July 17th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10257444/v1

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

Additional Declarations: No competing interests reported.

Dual-Stage Guided Artificial Lemming Algorithm (DALA) For Solving Engineering Application Optimization Problems

Jiayi Gao[a], Yu Zhang[a,*]

[a]*Northeast Forestry University, , Harbin, 150000, , China*

Abstract

The Artificial Lemming Algorithm (ALA) is a metaheuristic optimization algorithm inspired by four typical behaviors of lemmings. The traditional ALA often suffers from confined local search and limited global exploration when solving high-dimensional complex problems, primarily due to a lack of solution diversity and individual cooperation. This paper proposes a two-stage guided artificial lemming algorithm (DALA) aimed at efficiently solving single-objective, high-dimensional complex optimization problems. The core contribution of the research lies in a triple innovation mechanism: designing an adaptive probability mechanism based on ranking, integrating it into strategy selection; integrating a two-stage greedy strategy that combines adversarial learning and topological neighborhood learning to deeply explore the optimal solution; finally, integrating the mutation operation of JADE differential evolution, significantly enhancing the algorithm's ability to escape local optima. Using classical test functions from CEC 2017 and CEC 2020, we evaluated DALA's optimization performance through comparative experiments with eight other optimization algorithms, demonstrating enhanced optimization capability and superior global search performance. Additional tests on five constrained optimization problems with practical engineering significance confirm its broad applicability and engineering potential.This study further investigates the application of DALA in optimizing the hyperparameters of a multi-layer LSTM network for power load forecasting, thereby providing additional validation of its potential in practical engineering applications characterized by high dimensionality and complexity.

Keywords:
Artificial Lemming Algorithm, Swarm intelligence, Multi-strategy, Engineering problems

*Corresponding author
Email address: zhangyu0902@163.com (Yu Zhang)

1. Introduction

With the rapid development of modern science and technology, many complex engineering problems urgently need to be solved. In the past, such problems were mainly addressed using exact optimization algorithms. However, these algorithms require the problem to satisfy strict mathematical properties, and once the problem becomes nonlinear, non-convex, multimodal, discrete, or high-dimensional, the computational complexity of exact algorithms increases exponentially[1]. Metaheuristic algorithms are an efficient class of algorithmic frameworks designed to solve complex optimization problems. Their core idea lies in mimicking various efficient evolutionary processes found in nature, such as biological evolution, group cooperative behavior, or dynamic changes in physical systems. These algorithms can effectively handle complex problems where information is incomplete or computational resources are limited [2, 3, 4].

A single metaheuristic optimization algorithm may suffer from drawbacks such as being easily trapped in local optima, premature convergence, or a lack of robustness in high-dimensional and non-convex search spaces. Researchers have attempted to address novel and complex engineering problems by incorporating different strategies into metaheuristic algorithms or combining multiple algorithms[5]. The Improved Whale Optimization Algorithm (IWOA)[6] , for instance, enhances the algorithm's exploration and exploitation capabilities through new strategies, and has been applied to optimal power flow problems in power systems to minimize objectives such as generation cost and power loss. Based on the hybrid particle swarm optimization and sparrow search algorithm (PSO-SSA)[7], it mainly focuses on using the hybrid swarm intelligence algorithm to optimize deep learning models, thereby improving the accuracy and efficiency of software defect prediction. The Adaptive Gorilla Troops Optimizer (AGTO)[8] introduces adaptive mechanisms to improve search efficiency and demonstrates excellent performance in structural optimization problems, enabling the discovery of lighter and more stable structural designs. The Modified Butterfly Optimization Algorithm (MBOA)[9] improved algorithm incorporates an adaptive mechanism to adjust its key parameters. The enhanced MBOA, when combined with the support vector machine, utilizes the powerful optimization ability of MBOA to optimize the hyperparameters of SVM, thereby enhancing the performance of the classification model. The improved dung beetle optimization algorithm[10]introduces a novel hybrid model that integrates multiple algorithms and a bidirectional long short-term memory network with an attention mechanism (BiLSTM-A).

In recent years, meta-heuristic algorithms have been extensively deployed to address complex engineering problems characterized by high nonlinearity, stringent constraints, and non-convexity, emerging as a pivotal tool for solving optimization challenges in physical systems. In the field of renewable energy systems, these algorithms have been successfully applied to Maximum Power Point Tracking for wind turbines and the hydrodynamic performance optimization of wave energy converter arrays, significantly enhancing energy conversion efficiency. Within structural optimization, they have enabled precise control over the weight and layout of complex mechanisms, such as bridge trusses and aerospace thin-walled components, while strictly adhering to mechanical strength requirements. Furthermore, in addressing large-scale combinatorial optimization tasks—including urban traffic route planning and Optimal Power Flow calculations in power systems—these algorithms effectively overcome the engineering bottlenecks associated with the tedious parameter tuning of traditional gradient-based methods[11]. The demand for optimization algorithms is widely present in numerous cutting-edge areas of machine learning and control. In the field of deep neural network hyperparameter optimization, the emergence of complex variants of Transformer architectures and graph neural networks has led to a dramatic increase in the dimensionality of hyperparameter space, posing significant challenges to the performance of optimization algorithms[12]. For high-degree-of-freedom robot trajectory planning problems, optimization algorithms must satisfy dynamic constraints and obstacle avoidance requirements while solving for smooth, energy-optimal motion trajectories, creating a search space characterized by strong constraints, non-convexity, and multiple local optima[13]. In multi-agent cooperative control systems, optimization algorithms need to address task allocation, path planning, and formation control under distributed constraints, with the decision space growing exponentially with the number of agents[14]. These emerging problems share common challenges of high dimensionality, non-convexity, strong constraints, and multi-scale complexity, placing extremely high demands on the convergence speed, solution quality, and robustness of optimization algorithms. It is applied to optimize the hyperparameters of deep learning models, aiming to minimize prediction error and enhance model stability.

Inspired by the swarm intelligence behavior of lemmings in nature, the Artificial Lemmings Algorithm (ALA) has been proposed to enrich the algorithmic toolbox. The Artificial Lemmings Algorithm emulates four typical behaviors of lemmings in their natural environment: long-distance migration, burrowing, foraging, and evading predators[15]. To improve the performance of the algorithm, researchers have proposed various enhancements to ALA. For example, Zhu et al. proposed an Enhanced ALA (EALA)[16], which integrates Kent chaotic mapping, an adaptive

perturbation mechanism, and a differential mutation operator. This approach was applied to unmanned aerial vehicle (UAV) path planning problems and significantly improved both search accuracy and robustness. In addition, a multi-strategy improved algorithm,(MsIALA)[17], was proposed, which incorporates chaotic mechanisms, guided search strategies, and multiple individual update methods to further enhance the algorithm's ability to escape local optima. The motivations of this study are as follows:

- Optimization problems are foundational to science and engineering, particularly with the increasing complexity of machine learning and artificial intelligence. Guided by the No Free Lunch (NFL) theorem [18, 19], it is recognized that no single algorithm excels across all scenarios. Empirical observations indicate that for high-dimensional complex functions, the performance of the original ALA algorithm deteriorates significantly, as the gap between its average fitness and the theoretical global optimum widens as dimensionality increases.

- In meta-heuristic algorithms, the balance between exploration and exploitation is critical to the overall optimization performance. The original ALA algorithm primarily relies on preset, decreasing weights and lacks adaptive feedback regulation based on the search process. This fixed parameter update mechanism often limits the algorithm's flexibility, resulting in insufficient local exploitation precision during the later stages of the search.

To address the aforementioned issues, this paper proposes a Dual-phase Artificial Lemmings Algorithm (DALA) with a guided mechanism. The proposed DALA aims to efficiently solve complex optimization problems that are single-objective, high-dimensional, and multimodal. Such problems are prevalent in fields such as engineering design and machine learning, posing significant challenges to the global exploration and local exploitation capabilities of optimization algorithms.

In DALA, this paper proposes and designs an adaptive probability mechanism based on ranking, which is integrated into the strategy selection process by utilizing a quadratic curve function to replace the original strategy selection controlled by fixed parameters. This mechanism dynamically adjusts the operation probabilities through fitness ranking, adaptively allocating and employing different strategies for both elite and bottom-level individuals.[20, 21, 22]. A U-shaped trigger probability mechanism is designed to screen certain individuals and incorporate a greedy mechanism that includes Opposition-Based Learning[23, 24] and topological neighborhood learning[25, 26, 27], simulating the learning and thinking processes performed by each individual based on the population-led foraging or burrowing in the original

lemming algorithm.The inclusion of the greedy mechanism balances global exploration and local exploitation capabilities, utilizing the communicative and learning behaviors among lemming individuals to achieve collective optimization. JADE (Adaptive Differential Evolution with Optional External Archive) [28, 29, 30] is a stochastic optimization algorithm based on evolutionary mechanisms, which generates offspring through mutation and crossover operators and retains superior solutions using selection pressure based on fitness comparison. Simulated the behavior of certain individual lemmings deviating from the established evolutionary path, improving the algorithm's ability to avoid local optima.The contribution of this study lies not in the introduction of individual components, but in the implementation of a coordinated feedback architecture. Within this framework, a ranking-based probability mechanism dynamically allocates search tasks based on individual performance, and integrates strategies such as differential evolution to increase solution diversity, providing a more robust solution for complex engineering optimization.

To evaluate the optimization performance of the proposed DALA, comprehensive experiments are conducted on the IEEE CEC2017 and CEC2020 benchmark suites, which encompass a diverse range of test functions including unimodal, multimodal, hybrid, and composition functions. The efficacy of the integrated improvement strategies is rigorously validated through both parameter sensitivity analysis and ablation studies. To establish a robust performance baseline, DALA is benchmarked against the original ALA, recently developed high-performing variants, and well-established classical metaheuristic algorithms, with the statistical significance of the results assessed via the Friedman test and Wilcoxon signed-rank test. Furthermore, the practical utility of the algorithm is demonstrated through five constrained engineering design problems—comprising pressure vessel design, three-bar truss design, welded beam design, and gear train design—as well as a real-world application involving the hyperparameter optimization of a multi-layer LSTM network for power load forecasting.

2. Artificial Lemming Algorithm (ALA)

Lemmings are small rodents that are primarily distributed in Arctic regions, including tundra, forest-tundra, and mountainous tundra habitats. Lemmings possess one of the highest reproductive capacities among known mammalian species, which implies that their population expansion demands more food and living space. When their population increases, some lemmings begin to search for and migrate to new resource-rich territories. Due to their position at the bottom of the food chain, lemmings have evolved excellent burrowing abilities to evade predators. They also rely on their acute sense of smell and hearing to locate food and often forage and store food underground. The Artificial Lemming Algorithm (ALA) abstracts four primary behavioral traits of lemmings, long-distance migration, burrowing, foraging, and predator avoidance, and models these behaviors mathematically to construct the optimization process[15].

2.1. Population Initialization

The Lemming Algorithm (ALA) is a population-based algorithm. Before entering the iterative process, it is necessary to initialize the positions of all search agents (i.e., lemming individuals). The set of all initial candidate solutions forms a matrix $\mathbf{Z}$ consisting of N (population size) rows and Dim columns (problem dimensions), with values distributed between the lower and upper bounds of the given problem, as shown in Equation (1). In each iteration, the best position obtained so far is considered the optimal solution or an approximation thereof. The decision variable $z_{i,j}$ in each dimension is calculated according to Equation (2).

$$\mathbf{Z} = \begin{pmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,\mathrm{Dim}-1} & z_{1,\mathrm{Dim}} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,\mathrm{Dim}-1} & z_{2,\mathrm{Dim}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ z_{N-1,1} & z_{N-1,2} & \cdots & z_{N-1,\mathrm{Dim}-1} & z_{N-1,\mathrm{Dim}} \\ z_{N,1} & z_{N,2} & \cdots & z_{N,\mathrm{Dim}-1} & z_{N,\mathrm{Dim}} \end{pmatrix} \tag{1}$$

$$z_{i,j} = \mathrm{LB}_j + \mathrm{rand} \times (\mathrm{UB}_j - \mathrm{LB}_j), \quad i = 1, 2, \ldots, N, \ j = 1, 2, \ldots, \mathrm{Dim} \tag{2}$$

where rand represents a random value in the range $[0, 1]$, LB_j denotes the lower bound of the dimension, and UB_j denotes the upper bound of the dimension.

2.2. Long-Distance Migration

In the first behavioral pattern, when food becomes scarce due to overpopulation, lemmings engage in random long-distance migration. In this phase, lemmings explore the search space based on their current position and the position of a randomly

selected individual within the population, in search of habitats with abundant food resources to achieve better living conditions and resource access. It is important to note that due to various factors such as ecological conditions, the direction and distance of lemming migration are not fixed. To simulate this behavior, Equation (3) is proposed as follows:

$$Z_i(t+1) = \begin{array}{l} Z_{\text{best}}(t) + F \times \mathbf{BM} \times (\mathbf{R} \times (Z_{\text{best}}(t) - Z_i(t))) \\ + (Z_i(t) - Z_a(t)) \end{array} \tag{3}$$

where $Z_i(t+1)$ denotes the position of the $(t+1)$-th time search agent at iteration t, $Z_{\text{best}}(t)$ represents the current best solution, and F functions as a direction regulator for dynamically changing an individual's movement relative to the target. **BM** models biological Brownian motion to incorporate random walk properties into the algorithm. **R** is a vector of size $1 \times$ Dim with elements that are random numbers uniformly distributed in the interval $[-1, 1]$. $Z_i(t)$ represents the current position of the i-th search agent, and $Z_a(t)$ denotes a search agent randomly selected from the population, where a is an integer index ranging from 1 to N.

2.3. Burrowing

The second behavioral pattern of lemmings is burrowing in their habitats to form complex tunnels, which provide safe shelters and spaces for food storage. Lemmings dig new burrows randomly based on the position of their current burrow and the position of a randomly selected individual from the population. This behavior is mathematically expressed by Equation (4):

$$Z_i(t+1) = Z_i(t) + F \times L \times (Z_{\text{best}}(t) - Z_b(t)) \tag{4}$$

where L is a random number related to the current iteration, Z_b represents a search agent randomly selected from the population, where b is a random integer index ranging from 1 to N. L and Z_b are used to describe the interaction between lemming individuals during the excavation of new burrows.

2.4. Foraging

In the third behavioral pattern, lemmings move extensively and randomly within the burrows of their habitat, relying on their acute sense of smell and hearing to locate food sources. Lemmings typically establish a relatively small foraging area within their habitat. In order to consume as much food as possible, they wander randomly within this area, simulating the random foraging behavior of lemmings. A

spiral mechanism is used to model this phase, which is mathematically expressed by Equation (5) as follows:

$$Z_i(t+1) = Z_{\text{best}}(t) + F \times \text{spiral} \times \text{rand} \times Z_i(t) \tag{5}$$

where *spiral* denotes the spiral shape of the random search during foraging, calculated by Eqs. (12 and 13).

$$spiral = radius \times (\sin(2 \times \pi \times rand) + \cos(2 \times \pi \times rand)) \tag{6}$$

$$radius = \sqrt{\sum_{j=1}^{Dim} (z_{best,j}(t) - z_{i,j}(t))^2} \tag{7}$$

where *radius* means the radius of the foraging scope, which is the Euclidean distance between the current position and the optimal solution.

2.5. Predator Avoidance

In the final phase, the modeling focuses on the lemmings' avoidance and protective behaviors when facing danger. The burrow serves as a refuge for lemmings. Upon detecting a predator, lemmings use their exceptional running ability to escape back to their burrows. At the same time, they may also perform deceptive maneuvers to evade pursuit by predators. The corresponding mathematical expression is given in Equation (8).

$$Z_i(t+1) = Z_{\text{best}}(t) + F \times G \times \text{Levy(Dim)} \times (Z_{\text{best}}(t) - Z_i(t)) \tag{8}$$

$$G = 2 \times \left(1 - \frac{t}{T_{\max}}\right) \tag{9}$$

Where G is the lemming escape coefficient, representing the ability to escape, which decreases as the number of iterations increases. $T_{\max}$ denotes the maximum number of iterations. Levy(Dim) is the Lévy flight function, used to simulate the deceptive movements of lemmings during the escape process.

2.6. Implementation of the ALA Algorithm

In the initialization phase of ALA, control parameters are set, such as population size N, maximum number of iterations T, and problem dimension Dim. Meanwhile, the position of each search agent is generated using a uniform random distribution within the search space. In the subsequent iterations, ALA introduces an energy factor E, whose value gradually decreases with the increasing number of iterations.

This mechanism allows the search agents to transition smoothly from exploration to exploitation. The threshold of the energy factor is set to 1: when $E \geq 1$, lemmings perform exploration behaviors (migration or burrowing); when $E < 1$, lemmings engage in exploitation behaviors (foraging and predator avoidance). Once the termination condition is met, the process ends and the global optimal solution is returned. The energy factor is calculated using Equation (10) as follows:

$$E(t) = 4 \times \arctan\left(1 - \frac{t}{T_{\max}}\right) \times \ln\left(\frac{1}{\text{land}}\right) \tag{10}$$

The pseudocode of the ALA algorithm is presented as follows:

Input: Maximum iteration number T_{max}, population size N, and dimension size Dim
Output: Global optimal solution $Z_{\text{best}}(t)$ and its fitness value

Randomly initialize positions Z_i, evaluate fitness, and record optimal solution $Z_{\text{best}}(t)$
$t = 1$
while $t \leq T_{max}$ **do**
  Calculate the value of E using Equation (10)
  for each search individual Z_i **do**
    if $E > 1$ **then** (Exploration phase)
      Update location using Equation (3) if rand < 0.3, else use Equation (4)
    else (Exploitation phase)
      Update location using Equation (5) if rand < 0.5, else use Equation (8)
    end
  end
  Recalculate fitness values and update the optimal solution $Z_{\text{best}}(t)$
  $t = t + 1$
end

Algorithm 1: Pseudocode of the Artificial Lemming Algorithm (ALA)

3. Dual-Stage Guided Artificial Lemming Algorithm

3.1. Rank-based Adaptive Probability Mechanism

In the original ALA, individuals within the lemming population are randomly selected to perform different exploration or exploitation actions. Inspired by the hierarchical survival structure in lemming colonies, we introduce a Rank-based Adaptive Probability strategy.

Specifically, the rank-based adaptive probability P_i is employed to replace the fixed parameters (such as $rand < 0.3$ or 0.5) in the original main loop. First, all individuals in the population $\mathbf{X} = \{X_1, X_2, \ldots, X_N\}$ are sorted in ascending order based on their fitness values $f(X)$, as defined in Equation (11) :

$$\text{Sort}(\mathbf{X}): \quad f(X_{best}) \leq f(X_2) \leq \cdots \leq f(X_{worst}) \tag{11}$$

This sorting ensures that superior individuals maintain lower P_i values, whereas poorly performing individuals are assigned higher P_i values. Subsequently, exploration or exploitation strategies are adaptively allocated to each individual X_i according to the probability P_i calculated as:

$$P_i = \left(\frac{Rank_i}{N}\right)^2 \tag{12}$$

where N is the population size, and $Rank_i$ represents the rank position of individual X_i in the sorted sequence.

Individuals with higher ranks are typically positioned within potential optimal regions. By assigning them a significantly low P_i value, the probability of triggering exploitation strategies is increased, effectively reducing the search radius to facilitate intensive refinement within their current local vicinity. Conversely, maintaining a high fixed exploration probability would introduce excessive stochastic perturbations, potentially causing these individuals to diverge from the identified superior solutions.For underperforming individuals, higher P_i values are assigned because their current fitness indicates minimal potential for high-quality solutions in their original regions. Granting these individuals a high exploration probability aims to drive them toward large-scale global search actions. The adoption of a quadratic mapping, rather than a linear one, is justified by the fact that the quadratic function grows slowly in the high-rank interval, thereby shielding a substantial proportion of elite individuals to focus on localized exploitation. Meanwhile, the probability increases sharply for lower-ranked individuals, ensuring the necessary intensity for exploration. This adaptive adjustment effectively mitigates the algorithm's reliance on manual hyperparameter tuning and leverages intrinsic population information to enhance the coverage of complex search spaces.

3.2. Greedy Mechanism

To address the issue that the original algorithm lacks collaborative utilization among individuals, leading to insufficient global and local exploration capabilities, this paper introduces a greedy mechanism for improvement.

3.2.1. Mathematics

The core concept of Opposition-Based Learning (OBL) is to generate an opposite solution for the current solution by reflecting it symmetrically with respect to the center of the defined search space. For the i-th individual $Z(i,:)$ in the search space, its opposite solution is mathematically expressed by Equation (13):

$$Z_{\text{alt},i} = lb + ub - Z_i \tag{13}$$

To prevent the opposite solution Z_{alt} generated by the strategy from exceeding the defined search boundaries $[lb, ub]$, boundary handling is applied to constrain its value within the permissible search space, thereby ensuring its validity. The boundary handling is mathematically expressed by Equation (14):

$$Z_{\text{alt},i} = \min\left(\max\left(Z_{\text{alt},i}, lb\right), ub\right) \tag{14}$$

where $\mathbf{Z}_i$ is the current value of the individual,lb is the lower bound of each dimension, and ub is the upper bound of each dimension.

3.2.2. Topological Neighborhood Perturbation

For each lemming individual, $K = \max(2, [0.1N])$ neighboring individuals are randomly selected from its population vicinity. The one with the best fitness among them is identified, and a biased perturbation is constructed based on its position. The individual then moves toward this locally optimal solution with a certain perturbation step size. The mathematical formulation is given by Equation (15).

$$Z_{\text{alt},i} = Z_i + F \times \gamma \times (Z_{\text{nbest},i} - \mathbf{Z}_i) \tag{15}$$

If all neighboring individuals have worse fitness than the current individual, the current individual is retained by default: $Z_{\text{nbest},i} = Z_i$. Here, $\gamma \in (0, 1)$, denotes the perturbation step size, and $F \in \{-1, +1\}$, is the directional perturbation factor.

3.2.3. Selective Greedy Mechanism for Strategy Selection

First, the population is sorted based on fitness values, where $Rank_i$ is defined as the rank of the i-th individual (1 representing the best and N the worst). A U-shaped trigger probability model is constructed, as expressed in Equation (16), to facilitate

the flexible activation of corresponding mechanisms for both elite individuals with superior fitness and bottom-level individuals with poor fitness. This model ensures that computational resources are concentrated on the fine-grained exploitation of high-performing individuals while subjecting underperforming ones to opposition-based learning to escape local optima; meanwhile, middle-tier individuals ($Rank_i \to N/2$) primarily follow the original algorithm's evolution to conserve computational costs.The triggering probability model is shown in Figure 1.

$$P_{trig,i} = \left(\frac{2 \times Rank_i}{N} - 1 \right)^2 \tag{16}$$

where t represents the iteration count, N is the population size, and $Rank_i$ denotes the rank of individual X_i in the sorted sequence

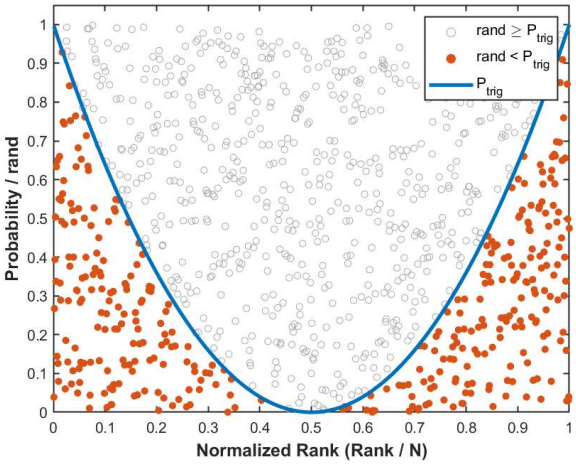


Figure 1: Distribution Diagram of U-shaped Trigger Probability

Based on this probability, the algorithm adopts a selective greedy mechanism for position updates. When the trigger condition ($rand < P_{trig,i}$) is satisfied, the individual further generates an alternative solution $Z_{alt,i}$ (as shown in Equation (17) according to the selection probability P_i (as shown in Equation (12)). To maintain consistency with the main loop, the probability selection algorithm utilizes the rank-based adaptive probability to dynamically regulate the trigger probabilities of the opposition-based learning strategy and the topological neighborhood search strategy:

$$Z_{alt,i} = \begin{cases} lb + ub - Z_i, & rand < P_i \\ \mathbf{Z}_i + F \times \gamma \times (\mathbf{Z}_{nbest,i} - \mathbf{Z}_i), & \text{otherwise} \end{cases} \tag{17}$$

Upon determining the candidate positions, the algorithm executes the selective greedy mechanism to decide the location for the next generation $Z_{t+1,i}$. If the generation of an alternative solution is triggered, the optimal one is selected among the current solution, the basic new solution, and the alternative solution; otherwise, the selection is performed only between the current and basic new solutions. This ensures that computational power is focused on the most promising updates, as formulated in Equation (18):

$$Z_{t+1,i} = \begin{cases} \arg\min\{f(Z_i), f(Z_{new,i}), f(Z_{alt,i})\}, & \text{if } rand < P_{trig,i} \\ \arg\min\{f(Z_i), f(Z_{new,i})\}, & \text{otherwise} \end{cases} \tag{18}$$

The mechanism defines a quadratic mapping function that is symmetric about the median of the population rank. In contrast to monotonically increasing or decreasing selection pressures, the U-shaped mechanism assigns higher strategy triggering probabilities to individuals at both extremes of the ranking. The quadratic term in the formula ensures that the triggering probability reaches its minimum at the population median ($Rank_i \approx N/2$), while increasing sharply toward both ends.Theoretically, this configuration shields the majority of the population, allowing them to maintain stable position updates under the original strategies, while adaptively allocating computational resources to individuals with either exceptional or poor fitness. By doing so, the mechanism effectively perturbs and adjusts the original search trajectories while maintaining a balance between exploration and exploitation. Ultimately, this enhances the algorithm's capability to escape local optima within complex solution spaces.

3.3. Adaptive Differential Evolution Hybrid Module

To address the issues of population diversity loss and premature convergence that algorithms may encounter in the later stages of iteration, this paper embeds the adaptive differential evolution (JADE) strategy within the ALA main loop. This module introduces a mutation mechanism based on $(current - to - pbest)$. Different from the classic DE algorithm, this strategy p is set to 30%, meaning that Zpbest is first randomly selected from the top 30% of the population based on fitness ranking as the guiding vector, and randomly selects two different individual indices $r1, r2 \neq i$ from the population to construct the mutation vector V_i, as expressed in Equation (19). Among them, the scaling factor F_i follows a normal distribution with a mean of 0.5 and a standard deviation of 0.3, clipped within the interval $[0.1, 1.0]$, as shown in Equation (20).

$$V_i = Z_i + F_i \times (Z_{pbest} - Z_i) + F_i \times (Z_{r1} - Z_{r2}) \tag{19}$$

$$F_i = \text{clip}(0.5 + randn \times 0.3, [0.1, 1.0]) \tag{20}$$

After differential mutation, a binomial crossover operation is used to generate new candidate individuals. Specifically, according to the crossover rate $CR = 0.9$, the current individual Z_i and the mutation vector V_i are combined through crossover to generate a trial solution U_i, as expressed in Equation (21). Finally, the individual with the best fitness is selected to update the global record.

$$U_{ij} = \begin{cases} V_{ij}, & \text{if } rand < CR \text{ or } j = j_{rand} \\ Z_{ij}, & \text{otherwise} \end{cases} \quad \forall j = 1, \ldots, d \tag{21}$$

Where CR represents the crossover rate, and the randomly selected dimension $j_{rand} \in \{1, 2, \ldots, d\}$ ensures that at least one dimension comes from the mutation vector.

By incorporating the adaptive differential evolution (JADE) module, the algorithm dynamically updates the mutation factor and crossover probability based on population feedback. This adaptive mechanism helps maintain population diversity during the early stages of the search, reducing the likelihood of premature convergence in complex landscapes. In the later stages, the module enhances local exploitation precision near the optimal solution by adaptively refining the search step size.

3.4. Implementation of the DALA Algorithm

Initialization Phase: Input parameters include the population size N, the maximum number of iterations T_{max}, the search space boundaries $[lb, ub]$, and the dimension Dim. The initial positions of the lemming population are generated through random initialization (as formulated in Equation (2)), followed by the calculation of the initial fitness values for all individuals.

Iteration Phase: Within each iteration of the main loop, the algorithm first computes the rank-based adaptive selection probability $Rank_i$ and the U-shaped trigger probability $P_{trigger}$ for strategy linkage. Subsequently, the process enters the first-stage basic update, where the individual energy factor E determines the search state: when $E > 1$, the individual enters the high-energy exploration phase; when $E < 1$, it transitions to the low-energy exploitation phase, resulting in the generation of a basic candidate solution X_{new}. Then, the selective greedy mechanism is invoked: a random number $rand$ is generated; if $rand < P_{trig}$, an alternative solution X_{alt} is produced by executing either the Opposition-Based Learning (OBL) strategy or the topological neighborhood search strategy according to the value of P_i. The optimal position for the next generation is then determined by evaluating

the fitness among the original position, X_{new}, and X_{alt}. If the trigger condition is not met, a binary greedy selection is performed between the original position and X_{new}, with the global optimum being updated synchronously. Simultaneously, the JADE differential evolution hybrid module allows 30% of the individuals to deviate from the current strategy by incorporating mutation perturbations to explore new regions of the search space. If the fitness of the perturbed solution is superior to the original one, the fitness and the global best value are updated, as mathematically expressed in Equation (19).The flowchart of DALA is shown in Fig. 2, and The pseudocode of the ALA algorithm is presented as follows:

Input: Maximum iteration number T_{max}, population size N, and dimension size Dim
Output: Global optimal solution $Z_{\text{best}}(t)$ and its fitness value f_x

Randomly initialize positions Z_i and evaluate initial fitness
$t = 1$
while $t \leq T_{max}$ **do**
  Calculate the value of E using Equation (10) and monotonic selection probability P_i using Equation (12)
  for each search individual Z_i **do**
    if $E > 1$ **then** (Exploration phase)
      Update location using Equation (3) if rand $< P_i$, else use Equation (4)
    else (Exploitation phase)
      Update location using Equation (5) if rand $< P_i$, else use Equation (8)
    end
    Calculate U-shaped trigger probability P_{trig} using Equation (16)
    if rand $< P_{trig}$ **then** (Activate greedy mechanism)
      Generate $Z_{alt,i}$ using Equation (14) if rand $< P_i$, else use Equation (15)
      The fitness of the three is compared, and the best fitness is selected
    end
    if rand < 0.3 **then** Execute differential evolution module Equation (21)
  end
  The current optimal value is recorded
  $t = t + 1$
end

Algorithm 2: Pseudocode of the Dual-Stage Guided Artificial Lemming Algorithm

(DALA)

3.5. Complexity Analysis

The time complexity of the DALA algorithm primarily stems from population initialization, adaptive ranking, position update, and hybrid strategies. Let N be the population size, d the dimensionality, T the maximum number of iterations, and C the cost of a single function evaluation.

Building upon the base ALA algorithm complexity $O(ALA) = O(N \times (1 + T \times d + T))$, the DALA algorithm incorporates the following enhancement strategies:

1. **Adaptive Ranking and Sorting:** In each iteration, sorting the population to compute the rank-based selection probability $Rank_i$ adds a complexity of $O(T \times N \log N)$;
2. **Strategy Linkage:** The hybrid strategy incorporating topological neighborhood search requires calculating the distance matrix among individuals, resulting in a time complexity of $O(T \times N^2)$;
3. **JADE and Selective Greedy Mechanism:** The JADE differential evolution module and the multiple fitness evaluations in the selective greedy mechanism contribute approximately $O(T \times N \times d + 2.3 \times T \times N \times C)$.

Considering the core position update process and the aforementioned enhancements, the overall time complexity of the DALA algorithm is:

$$O(DALA) = O(N \times d + T \times (N^2 + N \log N + N \times d + 2.3 \times N \times C))$$

Simplifying by retaining the dominant terms yields:

$$O(DALA) = O(T \times N^2 + T \times N \times (d + C))$$

Although the computational complexity of DALA is higher than that of the basic ALA due to the introduced enhancement strategies, experimental results demonstrate that the optimization performance and convergence speed of the DALA algorithm are significantly improved, which justifies the additional computational cost. The proposed hybrid strategy and DE module effectively enhance the algorithm's exploration and exploitation capabilities, facilitating escape from local optima and enabling the attainment of superior solutions.Figure 2 shows the workflow of the DALA algorithm.

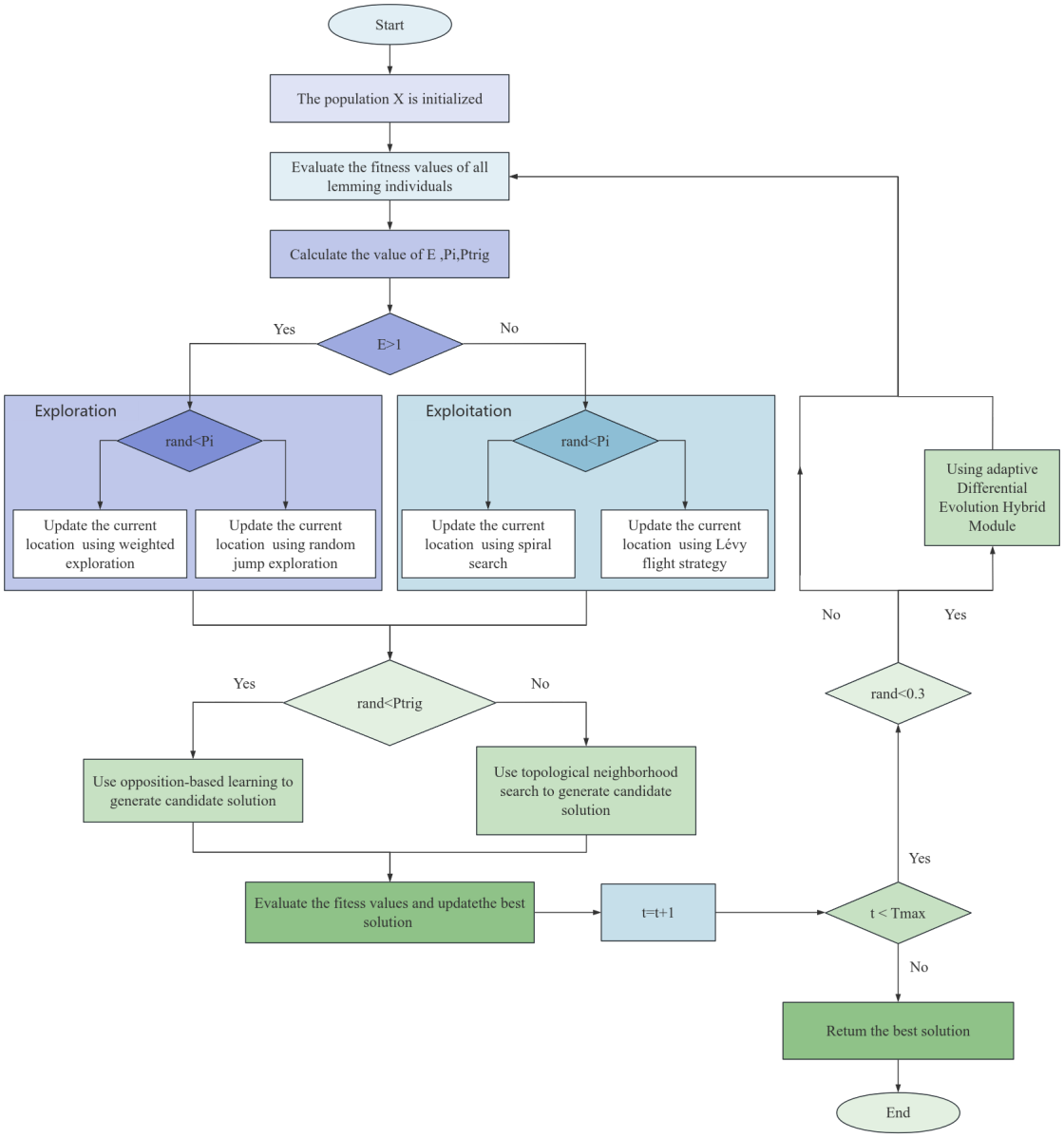


Figure 2: DALA algorithm flowchart

4. Experimental results and discussion

All experiments in this study were conducted on a laptop running the Windows 11 (64-bit) operating system, equipped with an Intel(R) Core(TM) i7-14650HX processor at 2.20 GHz and 16 GB of RAM, utilizing MATLAB R2022b software. To comprehensively evaluate the proposed DALA algorithm, this section is divided into two main parts. First, analytical experiments on the algorithm itself are conducted, which include parameter sensitivity analysis and ablation studies. Subsequently, the optimization performance of DALA is systematically verified through rigorous comparisons with various competitive algorithms on standard benchmark test suites.

For each algorithm, the number of individuals is set to N = 30, the maximum number of iterations T = 500, and the problem dimension dim = 30. Each of the nine algorithms is independently executed 30 times to obtain the mean and standard deviation, and rankings are determined based on these performance metrics.

4.1. Algorithm analysis and verification

4.1.1. Parameter sensitivity analysis

Detailed parameter sensitivity experiments were performed on the critical rank mapping exponent (Rank Power) and the JADE mutation trigger probability. Ten representative functions were selected from the IEEE CEC2017 benchmark suite to evaluate the experimental efficacy, including two unimodal functions (F1, F3) to assess exploitation and convergence speed, four multimodal functions (F5, F8, F10, F19) to examine the algorithm's capability in escaping local optima, and four composition functions (F22, F24, F28, F30) to investigate performance in complex hybrid scenarios. By systematically adjusting the exponent k of the rank mapping function $f(r) = r^k$ within the interval $[1, 2, 3]$ and changing the p parameter of the JADE operator, which controls the mutation weight, within the range of $[0.3, 0.6, 0.9]$, the sensitivity of convergence accuracy and success rate to these parameter fluctuations was rigorously evaluated, the results of which are summarized in Table (1).

The algorithm adopts the fourth parameter configuration ($\alpha = 0.3, \lambda = 2$) as the optimal setting. In the JADE mutation operator, the parameter p determines the selection range of the guiding vector $x^p_{best,g}$. As p approaches 0, the operator degrades into the $DE/current-to-best/1$ strategy, which increases the convergence pressure toward the current global best solution and facilitates intensive local exploitation. Conversely, as p increases, the expanding pool of candidate individuals enhances the diversity of the mutant vector $v_{i,g}$, thereby assisting the algorithm in escaping local stagnation. However, an excessively large p may hinder the convergence process. Experimental results indicate that setting $\alpha = 0.3$ achieves an effective balance between introducing potential solutions and ensuring local exploitation, yielding the optimal

Table 1: Parameter sensitivity analysis of different algorithm configurations

F	**Metric**	$\alpha = 0.3$ $\lambda = 1$	$\alpha = 0.6$ $\lambda = 1$	$\alpha = 0.9$ $\lambda = 1$	$\alpha = 0.3$ $\lambda = 2$	$\alpha = 0.6$ $\lambda = 2$	$\alpha = 0.9$ $\lambda = 2$	$\alpha = 0.3$ $\lambda = 3$	$\alpha = 0.6$ $\lambda = 3$	$\alpha = 0.9$ $\lambda = 3$
F1	Avg	5.00E+03	3.96E+03	3.45E+03	2.94E+03	3.83E+03	5.14E+03	**1.97E+03**	3.18E+03	4.35E+03
	Std	4.58E+03	4.60E+03	3.58E+03	3.71E+03	4.11E+03	5.69E+03	1.68E+03	3.37E+03	3.78E+03
	Rank	8	6	4	2	5	9	1	3	7
F3	Avg	4.46E+03	6.32E+03	6.35E+03	3.56E+03	3.28E+03	4.68E+03	**3.22E+03**	4.38E+03	4.82E+03
	Std	2.38E+03	5.42E+03	3.83E+03	2.27E+03	1.46E+03	4.11E+03	2.08E+03	2.36E+03	2.78E+03
	Rank	5	8	9	3	2	6	1	4	7
F5	Avg	5.85E+02	5.73E+02	**5.70E+02**	5.79E+02	5.77E+02	5.83E+02	5.91E+02	5.84E+02	5.83E+02
	Std	2.54E+01	2.68E+01	1.77E+01	2.09E+01	2.10E+01	2.22E+01	2.01E+01	2.21E+01	2.41E+01
	Rank	8	2	1	3	5	9	7	6	4
F8	Avg	8.71E+02	**8.70E+02**	8.70E+02	8.81E+02	8.73E+02	8.74E+02	8.84E+02	8.80E+02	8.79E+02
	Std	2.18E+01	2.28E+01	1.73E+01	1.35E+01	2.23E+01	1.61E+01	1.88E+01	2.35E+01	1.98E+01
	Rank	3	1	2	8	4	5	9	7	6
F10	Avg	**4.74E+03**	4.99E+03	5.15E+03	4.87E+03	5.06E+03	5.04E+03	5.04E+03	5.08E+03	4.90E+03
	Std	7.23E+02	7.32E+02	8.62E+02	6.13E+02	6.40E+02	7.21E+02	7.58E+02	7.23E+02	7.81E+02
	Rank	1	4	9	2	7	6	5	8	3
F19	Avg	2.45E+03	3.45E+03	2.93E+03	2.17E+03	3.07E+03	2.25E+03	2.34E+03	2.52E+03	**2.17E+03**
	Std	7.24E+02	4.50E+03	2.02E+03	5.95E+02	3.68E+03	6.38E+02	1.24E+03	1.59E+03	3.04E+02
	Rank	5	9	7	2	8	3	4	6	1
F22	Avg	**3.92E+03**	5.06E+03	4.44E+03	4.07E+03	4.53E+03	4.33E+03	4.33E+03	4.08E+03	4.50E+03
	Std	2.03E+03	2.24E+03	2.25E+03	2.05E+03	2.07E+03	2.10E+03	2.28E+03	1.88E+03	2.18E+03
	Rank	1	9	6	2	8	4	5	3	7
F24	Avg	2.90E+03	2.90E+03	**2.89E+03**	2.90E+03	2.89E+03	2.90E+03	2.90E+03	2.90E+03	2.92E+03
	Std	2.94E+01	2.81E+01	2.03E+01	1.83E+01	2.88E+01	2.18E+01	2.39E+01	3.06E+01	4.02E+01
	Rank	5	4	1	3	2	7	6	8	9
F28	Avg	**3.21E+03**	3.22E+03	3.22E+03	3.22E+03	3.22E+03	3.21E+03	3.22E+03	3.22E+03	3.22E+03
	Std	3.58E+01	3.29E+01	3.93E+01	2.13E+01	2.66E+01	2.01E+01	1.89E+01	3.37E+01	2.10E+01
	Rank	1	4	9	8	5	2	3	6	7
F30	Avg	1.07E+04	1.11E+04	1.06E+04	1.00E+04	1.02E+04	9.44E+03	**9.34E+03**	1.01E+04	1.05E+04
	Std	3.64E+03	3.76E+03	3.56E+03	3.66E+03	3.67E+03	2.86E+03	3.70E+03	3.47E+03	3.27E+03
	Rank	8	9	7	5	2	1	1	4	6
Average rank		4.500	5.600	5.500	**3.500**	5.200	4.900	4.600	5.300	5.900
Final rank		2	8	7	**1**	5	4	3	6	9

average rank of 3.500. Furthermore, the rank mapping exponent $\lambda = 2$ outperforms other settings. Compared to a linear mapping ($\lambda = 1$), the quadratic distribution more effectively differentiates the search weights among individuals, providing the necessary convergence drive during the early stages of optimization. Unlike higher-order exponents, the quadratic term ensures that mid-ranking individuals retain reasonable trigger probabilities, maintaining search flexibility for individuals across different fitness levels.

4.1.2. Ablation Study

To validate the individual contribution of each enhancement to the overall optimization performance of the proposed algorithm, a comprehensive ablation study was conducted. This analysis independently evaluates the efficacy of the three core components integrated into DALA: the adaptive selection probability, the selective greedy mechanism, and the JADE-based differential evolution hybrid module. Specifically, three distinct algorithm variants were developed for comparison: EALA (incorporating the adaptive selection probability into the baseline ALA), CALA (embedding the selective greedy mechanism), and JALA (integrating the JADE hybrid module). These variants were rigorously evaluated using unimodal, multimodal, and composition functions from the IEEE CEC2017 benchmark suite. An ablation factor γ is defined to quantitatively evaluate the importance of the omitted strategy, and its calculation is given in Equation (22):

$$\gamma = \frac{f_{\mathrm{ALA}}}{f_{\mathrm{ALAi}}} \tag{22}$$

In the equation, f_{ALA} and f_{ALAi} represent the average best fitness values of ALA and the partially improved versions of ALA, respectively. A larger γ indicates better optimization performance of the compared algorithm. The results of the ablation experiment are shown in Table (2). The γ values of EALA, CALA, and JALA are generally greater than 1, indicating that any combination of two strategies can improve the optimization performance of ALA.

Overall, EALA demonstrates superior robustness among the four compared variants, with its ablation factors consistently stabilizing within the 10^{-1} magnitude. The nonlinear selection pressure model constructed via $P_i = (Rank_i/N)^2$ assigns distinct computational strategies to individuals, validating that such an adaptive mechanism allocates computational resources more rationally than fixed-probability approaches and thereby enhances optimization performance. However, in more complex composition functions such as F19, F22, F24, and F28, other strategies exhibit improvements that significantly outperform EALA, suggesting that the effectiveness of the adaptive selection mechanism alone is relatively limited in irregular and

Table 2: Ablation Experiment Results

ψ	ALA/DALA	ALA/EALA	ALA/CALA	ALA/JALA
F1	2.87E+02	3.20E+00	2.66E+02	2.44E+02
F3	5.79E+00	1.33E+00	2.37E+00	1.90E+00
F5	1.04E+00	1.01E+00	9.71E-01	1.05E+00
F8	1.04E+00	1.02E+00	1.02E+00	1.05E+00
F10	1.11E+00	1.03E+00	1.03E+00	1.02E+00
F19	5.93E+00	1.25E+00	1.92E+00	1.01E+01
F22	1.70E+00	1.02E+00	1.36E+00	1.42E+00
F24	1.02E+00	1.00E+00	1.01E+00	1.02E+00
F28	1.08E+00	1.02E+00	1.07E+00	1.07E+00
F30	7.78E+00	2.56E+00	5.30E+00	4.80E+00

complex search spaces. In contrast, while CALA and JALA suffer from fluctuating convergence and lower robustness across diverse functions, they achieve superior results in composition functions.

Due to the inherent limitations of their respective operators, only offer improvements on specific problems. This study integrates multiple enhancement strategies into DALA, the core of which lies in the establishment of a coordinated feedback architecture. The ranking-based probability strategy is utilized for the allocation of search tasks: assigning exploitation-type tasks to elite individuals within high-potential regions, while individuals in low-potential regions execute random search behaviors. However, intensive local exploitation behaviors inevitably lead to population aggregation. To address this, the integrated JADE mutation operator introduces stochastic perturbations to break the limitations of the original search paths within a localized scope, thereby preventing premature convergence. Experimental results demonstrate that this cooperative mode of search task allocation and correction significantly enhances the global robustness of the algorithm, enabling it to exhibit superior search capabilities compared to single-strategy variants when tackling complex non-linear problems.

4.2. Benchmark Function Test Analysis

This paper first analyzes the effectiveness of DALA by visualizing its exploration and exploitation processes. To comparatively evaluate its optimization performance, eight representative intelligent optimization algorithms were selected: PSO (Particle Swarm Optimization, Kennedy and Eberhart, 1995), SSA (Sparrow Search Algorithm, Xue and Shen, 2020), ABC (Artificial Bee Colony, Karaboga, 2005), GWO (Grey Wolf Optimizer, Mirjalili et al., 2014), MVO (Multi-Verse Optimizer, Mirjalili et al., 2016), DOA (Dhole Optimization Algorithm, Hamad and Rashid, 2025), DMOA (Dwarf Mongoose Optimization Algorithm, Agushaka et al., 2022), and ALA (Artificial Lemming Algorithm, Naderi et al., 2025). The chosen benchmark algorithms include well-established classical algorithms (e.g., PSO, SSA, and ABC), popular algorithms in recent years (e.g., GWO and MVO), as well as newer algorithms (e.g., DOA, DMOA, and ALA). These algorithms have been widely applied to continuous optimization and engineering design problems, demonstrating strong representativeness and comparative value.

4.2.1. Experiments on CEC2017 Benchmark Functions

To verify the optimization performance of the DALA algorithm, the IEEE CEC2017 benchmark test functions are used in this paper. The IEEE CEC2017 benchmark set includes various types of optimization problems such as unimodal, multimodal, high-dimensional, and noisy functions, which allows for comprehensive coverage of scenarios that optimization algorithms may encounter, as well as standardized evaluation of algorithm applicability and robustness.As shown in Table 3, the CEC2017 benchmark includes a total of 29 functions (with function F2 removed due to instability in high-dimensional cases). Among them, F1 and F3 are unimodal functions; F4–F10 are basic multimodal functions; F11–F20 are hybrid functions; and F21–F30 are composition functions. The dimensionality can be selected from 10, 30, 50, or 100 dimensions. As dimensionality increases, these functions become extremely difficult to solve, and the runtime also increases significantly.

Statistical and Convergence Curve Analysis Using CEC2017 Test Functions. Table 4 presents the comparison results of the DALA algorithm and six other algorithms on the IEEE CEC2017 benchmark test functions. The data in the table represent the average and standard deviation obtained over 30 independent runs, with the best average value highlighted in bold. From the test results shown in the table, it can be observed that the DALA algorithm demonstrates strong optimization performance on the CEC2017 benchmark functions. The mean results indicate that DALA achieved the optimal fitness value in 24 out of 29 test functions. Among the unimodal functions, DALA's mean value ranked second only to SSA on F1, while being significantly

smaller than other algorithms, and secured the first place on F3.On the multimodal functions F4-F10, DALA obtained the minimum value in 6 functions, demonstrating its excellent local exploitation capability in handling complex multimodal problems. For the hybrid functions F11-F20, DALA achieved the smallest fitness values across all 10 functions. Particularly on F14, DALA's mean value (1.52E+03) maintained a clear advantage over the second-best algorithm ALA (1.85E+03), proving the algorithm's robustness in handling hybrid characteristic problems.On the composition functions F21-F30, DALA achieved optimal results in 8 functions, slightly underperforming compared to DMOA and SSA only on F27 and F28 respectively, while maintaining overall leading performance. Regarding standard deviation, DALA exhibited relatively small values across most functions, such as a mere 1.42E+00 on F6, indicating excellent algorithm stability and convergence consistency with reduced susceptibility to local optima.

Figure 5 presents the convergence curves of DALA and other comparison algorithms. It can be observed that DALA delivers favorable convergence speed and accuracy across these test cases. For the unimodal functions F1 and F3, both DALA and PSO converge rapidly in the early iterations, while the remaining algorithms offer similar convergence precision. On the multimodal functions F4–F10, DMOA, ABC, and MVO show poor convergence performance. Although DALA initially converges similarly to other algorithms, a noticeable inflection point emerges in its curve as the search strategy shifts from opposition-based learning to a topological neighborhood strategy in the later phase, leading to significantly accelerated convergence speed and ultimately a far better mean fitness than the other algorithms. For the hybrid functions F11–F20, DALA demonstrates stable performance and the most superior solution quality across different functions, while PSO also exhibits competitive search efficiency. On the complex composition functions F21–F30, DALA, ALA, GWO, and MVO all achieve relatively good results, with the effectiveness of the improved strategy in DALA over the later stage of ALA being clearly evident. Overall, the results indicate that DALA successfully maintains exploration and exploitation in the early search phase and, by switching strategies in the later stage, escapes local optima to achieve higher-precision convergence, validating the effectiveness of the incorporated strategies.

Table 3: CEC2017 Test Functions

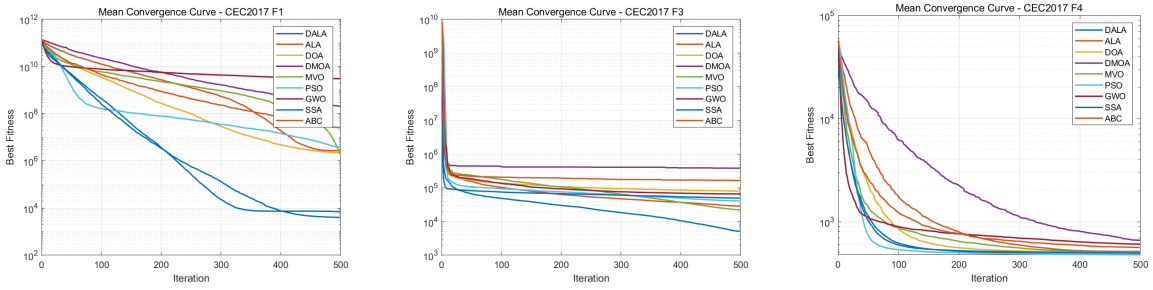
Function	Function Name	Optimal Value
F1	Shifted and Rotated Bent Cigar Function	100
F3	Shifted and Rotated Zakharov Function	300
F4	Shifted and Rotated Rosenbrock's Function	400
F5	Shifted and Rotated Rastrigin's Function	500
F6	Shifted and Rotated Expanded Scaffer's F6 Function	600
F7	Shifted and Rotated Lunacek Bi-Rastrigin Function	700
F8	Shifted and Rotated Non-Continuous Rastrigin's Function	800
F9	Shifted and Rotated Lévy Function	900
F10	Shifted and Rotated Schwefel's Function	1000
F11	Hybrid Function 1 (N = 3)	1100
F12	Hybrid Function 2 (N = 3)	1200
F13	Hybrid Function 3 (N = 3)	1300
F14	Hybrid Function 4 (N = 4)	1400
F15	Hybrid Function 5 (N = 4)	1500
F16	Hybrid Function 5 (N = 4)	1600
F17	Hybrid Function 6 (N = 4)	1700
F18	Hybrid Function 6 (N = 5)	1800
F19	Hybrid Function 6 (N = 5)	1900
F20	Hybrid Function 6 (N = 6)	2000
F21	Composition Function 1 (N = 3)	2100
F22	Composition Function 2 (N = 3)	2200
F23	Composition Function 3 (N = 4)	2300
F24	Composition Function 4 (N = 4)	2400
F25	Composition Function 5 (N = 5)	2500
F26	Composition Function 6 (N = 5)	2600
F27	Composition Function 7 (N = 6)	2700
F28	Composition Function 8 (N = 6)	2800
F29	Composition Function 9 (N = 3)	2900
F30	Composition Function 10 (N = 3)	3000

Table 4: Statistical Results of Experiments Using CEC2017 Test Functions

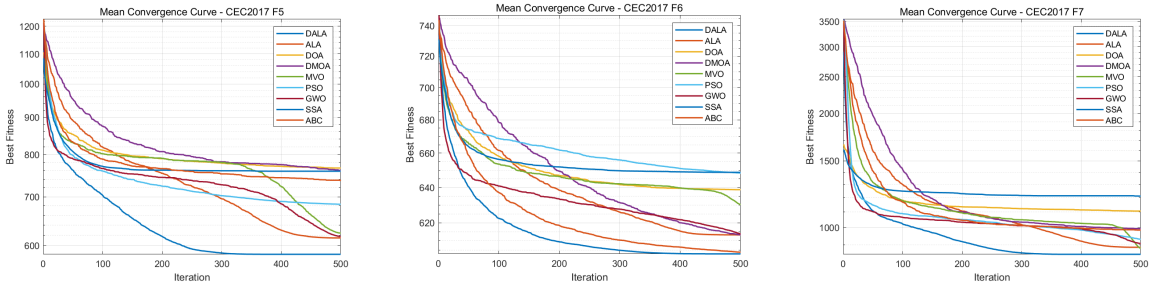
Function	**DALA** Mean Std	**ALA** Mean Std	**DOA** Mean Std	**DMOA** Mean Std	**MVO** Mean Std	**PSO** Mean Std	**GWO** Mean Std	**SSA** Mean Std	**ABC** Mean Std
F1	**4.31E+03** 4.73E+03	2.11E+06 2.03E+06	4.84E+10 9.40E+09	2.12E+08 1.25E+08	2.24E+06 8.51E+05	3.22E+06 1.26E+06	3.40E+09 2.28E+09	5.20E+03 5.27E+03	2.88E+07 1.56E+07
F3	**6.08E+03** 3.32E+03	2.79E+04 7.43E+03	7.78E+04 1.15E+04	3.99E+05 1.30E+05	2.25E+04 1.09E+04	4.33E+04 1.35E+04	6.37E+04 1.27E+04	4.69E+04 8.32E+03	1.58E+05 2.41E+04
F4	4.81E+02 2.48E+01	5.25E+02 3.92E+01	1.14E+04 2.50E+03	6.60E+02 8.41E+01	5.03E+02 1.60E+01	**4.73E+02** 2.01E+01	6.77E+02 1.27E+02	5.00E+02 3.31E+01	5.68E+02 2.06E+01
F5	**5.80E+02** 2.30E+01	6.10E+02 2.57E+01	8.68E+02 4.96E+01	7.56E+02 2.18E+01	6.31E+02 4.01E+01	6.99E+02 3.05E+01	6.31E+02 4.24E+01	7.50E+02 4.64E+01	7.36E+02 1.07E+01
F6	**6.03E+02** 2.66E+00	6.12E+02 4.95E+00	6.77E+02 1.14E+01	6.13E+02 3.96E+00	6.32E+02 1.14E+01	6.48E+02 7.22E+00	6.13E+02 5.25E+00	6.45E+02 9.71E+00	6.04E+02 1.00E+00
F7	**8.44E+02** 3.35E+01	8.98E+02 4.21E+01	1.38E+03 8.39E+01	9.99E+02 1.58E+01	8.76E+02 3.59E+01	9.42E+02 3.84E+01	9.30E+02 5.23E+01	1.17E+03 1.15E+02	9.84E+02 1.37E+01
F8	**8.89E+02** 2.01E+01	9.18E+02 4.05E+01	1.11E+03 3.75E+01	1.06E+03 1.73E+01	9.36E+02 3.81E+01	9.49E+02 2.56E+01	9.06E+02 4.02E+01	9.83E+02 2.31E+01	1.04E+03 1.27E+01
F9	**1.35E+03** 3.70E+02	2.76E+03 9.11E+02	1.06E+04 2.15E+03	5.18E+03 1.67E+03	6.80E+03 3.08E+03	6.59E+03 1.67E+03	2.87E+03 1.16E+03	5.31E+03 3.99E+02	2.09E+03 4.06E+02
F10	5.11E+03 7.08E+02	6.01E+03 9.46E+02	8.69E+03 6.86E+02	9.36E+03 3.06E+02	**4.74E+03** 7.18E+02	4.89E+03 6.37E+02	5.14E+03 1.19E+03	5.30E+03 7.00E+02	8.53E+03 3.27E+02
F11	**1.20E+03** 4.44E+01	1.26E+03 4.05E+01	7.40E+03 4.01E+03	1.10E+04 3.85E+03	1.33E+03 5.38E+01	1.25E+03 5.07E+01	3.05E+03 2.03E+03	1.30E+03 7.72E+01	1.61E+03 1.19E+02
F12	**1.45E+05** 1.82E+05	2.87E+06 2.04E+06	8.12E+09 5.04E+09	3.70E+08 1.43E+08	1.25E+07 9.68E+06	2.30E+06 1.87E+06	6.57E+07 4.78E+07	1.21E+06 1.05E+06	1.77E+07 7.35E+06
F13	**1.46E+04** 1.49E+04	5.26E+04 4.15E+04	3.17E+09 3.59E+09	3.87E+06 4.60E+06	1.68E+05 1.07E+05	3.43E+04 1.80E+04	6.70E+07 1.52E+08	2.21E+04 1.79E+04	1.64E+04 1.14E+04
F14	**1.59E+03** 8.07E+01	1.86E+03 2.13E+02	4.06E+05 7.02E+05	4.13E+05 2.68E+05	5.91E+04 4.61E+04	3.28E+04 3.05E+04	6.37E+05 6.42E+05	6.59E+04 6.12E+04	1.54E+05 1.35E+05
F15	**5.87E+03** 7.88E+03	2.61E+04 1.35E+04	1.06E+07 5.08E+07	6.99E+05 4.81E+05	6.41E+04 3.81E+04	7.68E+03 7.54E+03	2.43E+06 5.88E+06	1.39E+04 1.02E+04	6.53E+03 4.75E+03

Table 5: Statistical Results of Experiments Using CEC2017 Test Functions (continued)

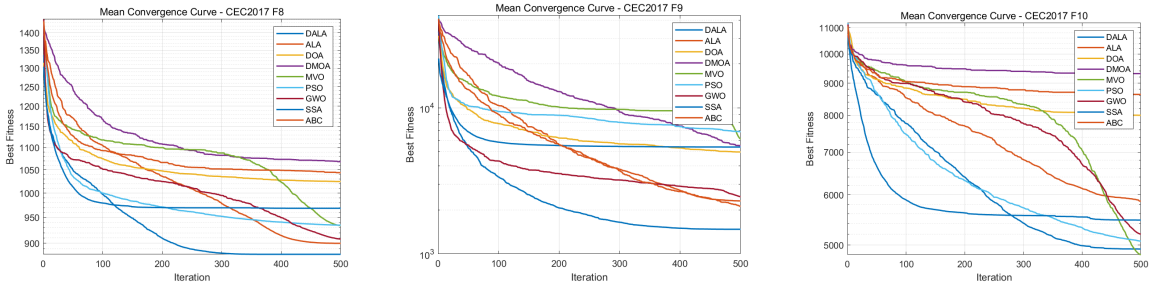
Function	DALA	ALA	DOA	DMOA	MVO	PSO	GWO	SSA	ABC
F16	**2.44E+03**	2.73E+03	4.53E+03	4.08E+03	2.90E+03	2.79E+03	2.65E+03	2.89E+03	3.49E+03
	2.76E+02	2.27E+02	9.20E+02	2.44E+02	3.03E+02	2.98E+02	4.16E+02	3.69E+02	1.38E+02
F17	**1.99E+03**	2.16E+03	2.74E+03	3.04E+03	2.28E+03	2.42E+03	2.17E+03	2.57E+03	2.43E+03
	1.38E+02	1.93E+02	3.70E+02	1.32E+02	2.10E+02	2.90E+02	1.82E+02	2.57E+02	1.16E+02
F18	**3.89E+04**	8.81E+04	1.87E+06	1.94E+07	9.32E+05	1.29E+06	2.94E+06	8.43E+05	6.36E+06
	3.31E+04	6.01E+04	4.37E+06	9.33E+06	7.68E+05	1.29E+06	4.77E+06	9.89E+05	2.62E+06
F19	**3.39E+03**	2.25E+04	3.59E+07	1.35E+05	2.15E+06	1.25E+04	1.64E+06	9.94E+03	1.34E+04
	3.42E+03	1.94E+04	7.81E+07	1.62E+05	1.71E+06	8.41E+03	1.54E+06	1.32E+04	2.16E+04
F20	**2.36E+03**	2.56E+03	2.88E+03	3.16E+03	2.61E+03	2.63E+03	2.50E+03	2.77E+03	2.82E+03
	1.68E+02	2.01E+02	2.34E+02	1.80E+02	1.97E+02	2.15E+02	1.72E+02	2.34E+02	1.12E+02
F21	2.38E+03	2.40E+03	2.67E+03	2.56E+03	2.42E+03	2.50E+03	**2.40E+03**	2.50E+03	2.52E+03
	2.00E+01	2.55E+01	7.17E+01	1.44E+01	4.31E+01	4.10E+01	3.12E+01	5.91E+01	1.44E+01
F22	**4.18E+03**	6.53E+03	8.09E+03	1.06E+04	5.29E+03	5.41E+03	5.45E+03	5.76E+03	5.03E+03
	2.10E+03	2.05E+03	1.19E+03	3.04E+02	1.94E+03	2.14E+03	2.04E+03	2.20E+03	2.78E+03
F23	**2.72E+03**	2.78E+03	3.33E+03	2.93E+03	2.78E+03	3.16E+03	2.81E+03	2.92E+03	2.88E+03
	2.19E+01	3.91E+01	1.50E+02	1.79E+01	4.59E+01	1.59E+02	5.64E+01	7.66E+01	1.15E+01
F24	**2.90E+03**	2.96E+03	3.56E+03	3.10E+03	2.93E+03	3.19E+03	2.97E+03	3.09E+03	3.04E+03
	1.89E+01	3.76E+01	1.88E+02	1.71E+01	3.77E+01	9.24E+01	7.10E+01	7.73E+01	1.39E+01
F25	**2.89E+03**	2.91E+03	4.95E+03	2.97E+03	2.90E+03	2.91E+03	3.00E+03	2.89E+03	2.92E+03
	9.17E+00	1.74E+01	7.99E+02	2.57E+01	1.96E+01	2.28E+01	3.81E+01	1.33E+01	1.05E+01
F26	**4.53E+03**	4.94E+03	1.02E+04	5.96E+03	4.81E+03	6.46E+03	5.05E+03	6.12E+03	5.95E+03
	5.29E+02	5.70E+02	1.37E+03	1.59E+02	7.70E+02	1.88E+03	3.83E+02	1.28E+03	1.33E+02
F27	3.23E+03	3.23E+03	3.82E+03	**3.20E+03**	3.24E+03	3.33E+03	3.27E+03	3.27E+03	3.24E+03
	2.36E+01	1.95E+01	4.22E+02	5.93E-05	2.32E+01	1.66E+02	2.73E+01	3.78E+01	6.68E+00
F28	**3.22E+03**	3.38E+03	6.57E+03	3.30E+03	3.26E+03	3.25E+03	3.46E+03	3.23E+03	3.33E+03
	1.67E+01	4.99E+02	9.56E+02	2.16E-01	4.42E+01	3.32E+01	8.25E+01	2.34E+01	2.97E+01
F29	**3.76E+03**	4.01E+03	5.75E+03	5.06E+03	4.09E+03	4.31E+03	3.90E+03	4.18E+03	4.52E+03
	3.02E+02	2.86E+02	9.21E+02	3.49E+02	2.52E+02	2.93E+02	2.02E+02	2.69E+02	1.75E+02
F30	**1.01E+04**	5.67E+04	4.01E+08	2.71E+05	4.32E+06	1.76E+05	1.24E+07	2.01E+04	3.08E+05
	3.43E+03	7.20E+04	8.23E+08	2.70E+05	3.02E+06	1.30E+05	1.05E+07	1.43E+04	2.32E+05
+/−/=		27/2/0	29/0/0	24/4/1	24/4/1	26/2/1	26/2/1	25/4/0	27/2/0



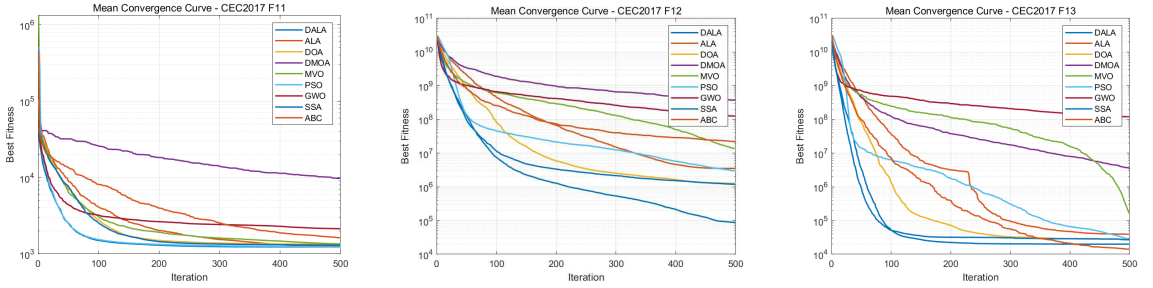
F1 F3 F4



F5 F6 F7

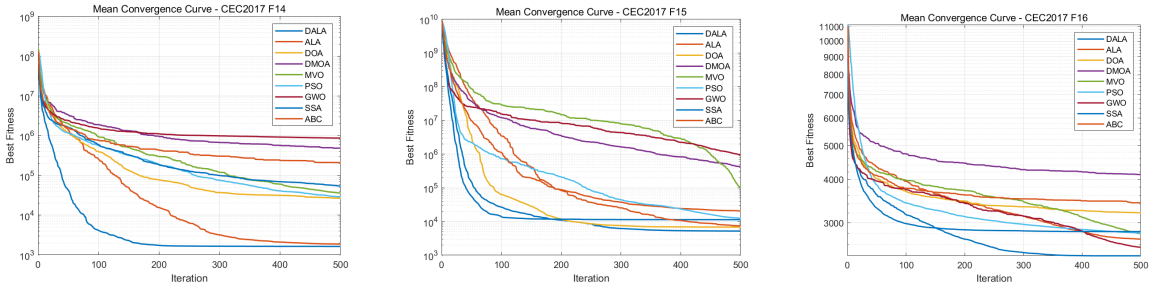


F8 F9 F10

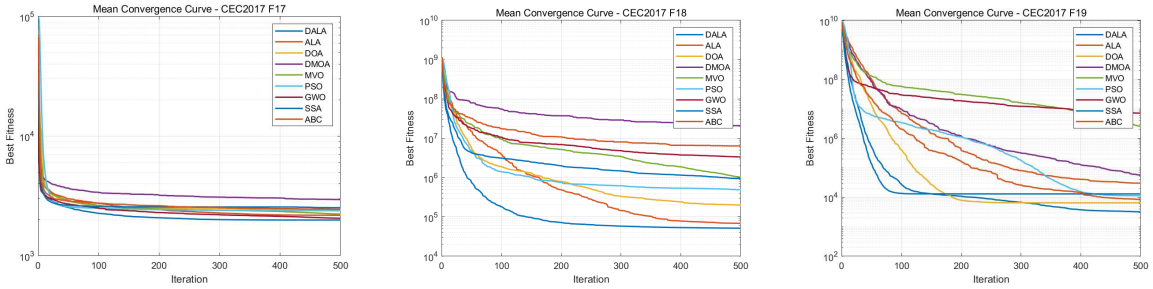


F11 F12 F13

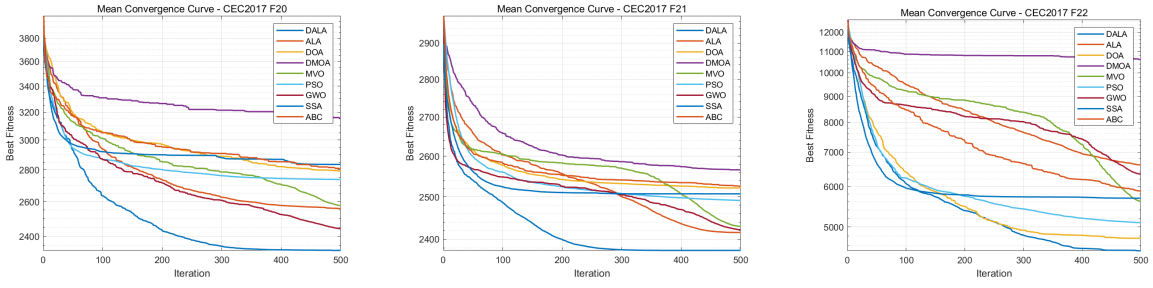
Figure 3: Convergence Curves of Experiments Using CEC2017 Test Functions



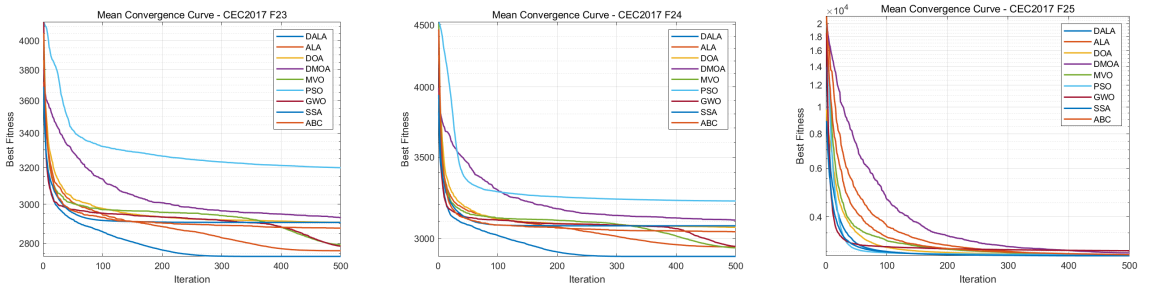
F14 F15 F16



F17 F18 F19

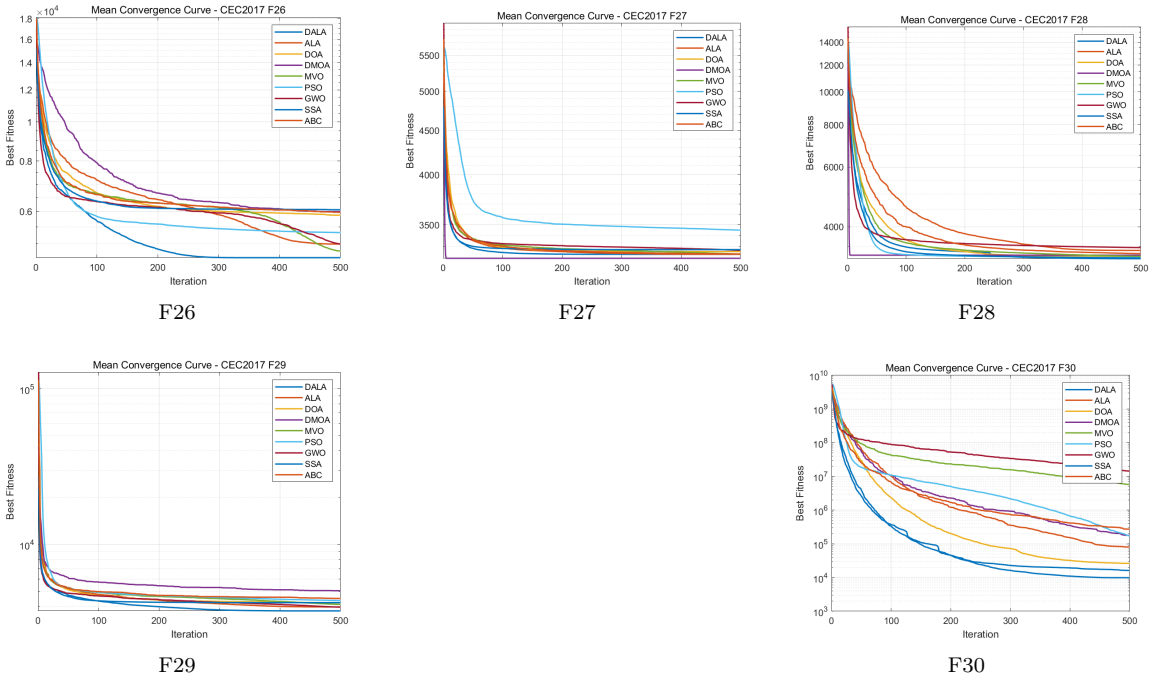


F20 F21 F22



F23 F24 F25

Figure 4: Convergence Curves of Experiments Using CEC2017 Test Functions



F26 F27 F28

F29 F30

Figure 5: Convergence Curves of Experiments Using CEC2017 Test Functions

*Friedman Test.*The Friedman test is a non-parametric statistical method commonly used at a significance level of 0.05 to determine whether performance differences among multiple algorithms across multiple problems are statistically significant. This test ranks the performance of each algorithm on each problem, with a smaller rank indicating better performance. A key indicator in the test results is the p-value; the smaller the p-value, the more significant the differences among the algorithms. Additionally, the test table includes SS (sum of squares), df (degrees of freedom), MS (mean square, SS divided by df), and the chi-square statistic, which are used to assess the significance of performance differences among algorithms.As shown in the results in Table 6, the p-value obtained is 5.4012×10^{-5}, which is far below 0.05, indicating that there are significant differences in both accuracy and computational time.

Table 6: Friedman Analysis of Variance

Source	**SS**	**df**	**MS**	**Chi-sp**	**Prob>Chi-sp**
Columns	102.1	8	12.7625	33.3161	5.4012×10^{-5}
Error	633.4	232	2.7302	-	-
Total	735.5	269	-	-	-

Table 7 shows that DALA achieved the highest ranking with an average rank of 3.67. ALA and MVO secured the second and third positions with average ranks of 4.53 and 4.77, respectively. SSA, PSO, GWO, and ABC followed in the fourth, fifth, sixth, and seventh places, while DOA and DMOA ranked last with average ranks of 5.55 and 5.85.

Table 7: Algorithm Rankings Based on Friedman Test

Algorithms	DALA	ALA	DOA	DMOA	MVO	PSO	GWO	SSA	ABC
Avg. Rank	1.13	3.63	8.6	7.2	4.17	4.67	5.4	4.7	5.5
Rank	1	2	9	8	3	4	6	5	7

Analysis of Wilcoxon Signed-Rank Test Results. Table 4 presents the results of the Wilcoxon signed-rank test for the DALA algorithm compared to eight other contrasting algorithms on the IEEE CEC2017 benchmark functions. The significance level is set at 5%. The symbol "+" indicates that DALA significantly outperforms the contrasting algorithms on that function, "-" indicates significant inferiority, and "=" indicates no significant difference. The above experiments demonstrate that DALA has better optimization performance and improvement compared to the original algorithm and other algorithms.

Table 8: Wilcoxon signed-rank test results for CEC2017 functions

Function	**ALA**	**DOA**	**DMOA**	**MVO**	**PSO**	**GWO**	**SSA**	**ABC**
F1	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	5.71E-02	1.73E-06
F2	1.73E-06	1.92E-06	1.73E-06	6.34E-06	2.83E-04	1.73E-06	2.05E-04	1.73E-06
F3	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06
F4	2.16E-05	5.32E-03	1.73E-06	1.32E-02	4.65E-01	1.92E-06	1.96E-02	1.73E-06
F5	1.48E-03	1.73E-06	1.73E-06	2.83E-04	1.73E-06	3.72E-05	1.73E-06	1.73E-06
F6	2.88E-06	1.73E-06	2.88E-06	1.92E-06	1.73E-06	2.13E-06	1.73E-06	9.84E-03
F7	2.77E-03	1.73E-06	1.73E-06	3.61E-03	6.98E-06	8.31E-04	1.73E-06	1.73E-06
F8	8.73E-03	1.73E-06	1.73E-06	3.88E-06	1.92E-06	5.22E-06	1.73E-06	1.73E-06
F9	1.74E-04	3.18E-06	1.73E-06	2.35E-06	1.73E-06	3.72E-05	1.73E-06	1.74E-04
F10	1.96E-03	1.73E-06	1.73E-06	4.05E-01	3.82E-01	7.50E-01	8.22E-03	1.73E-06
F11	1.40E-02	1.49E-05	1.73E-06	3.88E-06	7.04E-01	1.73E-06	3.32E-04	1.73E-06
F12	1.73E-06	1.92E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06
F13	2.11E-03	2.45E-01	1.73E-06	1.73E-06	5.19E-02	1.73E-06	1.92E-01	6.14E-01
F14	1.80E-05	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06
F15	5.31E-05	3.87E-02	1.73E-06	1.73E-06	4.11E-03	1.73E-06	7.71E-03	1.17E-02
F16	3.16E-03	3.18E-06	1.73E-06	1.64E-05	1.06E-04	1.53E-01	8.92E-05	1.73E-06
F17	1.36E-04	1.73E-06	1.73E-06	4.20E-04	1.24E-05	1.59E-01	2.13E-06	1.92E-06
F18	8.22E-02	5.22E-06	1.73E-06	1.73E-06	3.52E-06	1.73E-06	2.35E-06	1.73E-06
F19	6.34E-06	1.57E-02	1.73E-06	1.73E-06	3.32E-04	1.73E-06	1.49E-05	3.11E-05
F20	9.71E-05	1.73E-06	1.73E-06	1.36E-04	5.22E-06	7.27E-03	1.92E-06	1.92E-06

Table 9: Wilcoxon signed-rank test results for CEC2017 functions (continued)

Function	**ALA**	**DOA**	**DMOA**	**MVO**	**PSO**	**GWO**	**SSA**	**ABC**
F21	1.49E-05	1.73E-06	1.73E-06	2.60E-06	1.80E-05	1.97E-05	1.73E-06	1.73E-06
F22	2.22E-04	6.58E-01	1.73E-06	3.87E-02	2.99E-01	2.56E-02	3.33E-02	2.56E-02
F23	2.77E-03	1.73E-06	1.73E-06	1.13E-05	1.73E-06	1.13E-05	1.73E-06	1.73E-06
F24	6.32E-05	1.73E-06	1.73E-06	6.89E-05	1.73E-06	1.13E-05	1.73E-06	1.73E-06
F25	1.97E-05	1.97E-05	1.73E-06	2.07E-02	1.96E-03	1.73E-06	4.17E-01	2.35E-06
F26	8.94E-04	5.29E-04	1.73E-06	1.20E-01	1.48E-02	8.22E-03	3.88E-04	1.73E-06
F27	9.92E-01	8.73E-03	1.73E-06	4.53E-01	1.29E-03	5.29E-04	8.92E-05	3.82E-01
F28	1.73E-06	1.02E-05	1.73E-06	4.73E-06	2.22E-04	1.73E-06	7.19E-02	1.73E-06
F29	8.94E-04	1.97E-05	1.73E-06	1.64E-05	4.73E-06	7.27E-03	1.49E-05	1.73E-06
F30	2.35E-06	3.88E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	2.41E-04	1.73E-06
+/−/=	27/2/0	27/2/0	29/0/0	27/2/0	24/4/1	26/3/0	23/4/2	27/2/0

4.2.2. Experiments on CEC2020 Benchmark Functions

To further verify the optimization performance of the DALA algorithm, the IEEE CEC2020 benchmark test functions are used in this paper. The IEEE CEC2020 benchmark function set is an important standard for evaluating the performance of single-objective constrained optimization algorithms. As shown in Table 10, this test set includes 10 complex continuous optimization problems, covering various function types such as unimodal functions (F1–F3), multimodal functions (F4–F7), and composition and rotation functions (F8–F10), designed to assess the algorithm's search ability and robustness in high-dimensional spaces (dimensions of 5, 10, 15, and 20).

For each algorithm, the number of individuals is set to $N = 30$, the maximum number of iterations is $T = 500$, and the problem dimension is dim = 20. Each of the nine algorithms is independently executed 30 times to obtain the average and standard deviation for each, and rankings are determined based on these performance metrics.

Table 10: CEC2020 Test Functions

Function	**Function Name**	**Optimal Value**
F1	Shifted Sphere Function	100
F2	Shifted Schwefel's 1.2 Function	200
F3	Shifted Rosenbrock Function	300
F4	Shifted Rastrigin Function	400
F5	Shifted Griewank Function	500
F6	Shifted Ackley Function	600
F7	Shifted Weierstrass Function	700
F8	Shifted HappyCat Function	800
F9	Shifted HGBat Function	900
F10	Expanded Schaffer's F6 Function	1000

Statistical and Convergence Curve Analysis Using CEC2020 Test Functions. Table 11 presents the comparison results of the DALA algorithm and six other algorithms on the IEEE CEC2020 benchmark test functions. The data in the table represent the mean and standard deviation of 30 independent processes, with the best mean values highlighted in bold. From the test results presented in the table, it can be observed that the DALA algorithm exhibits commendable optimization capabilities in the IEEE CEC2020 benchmark function tests. The mean results indicate that DALA achieves optimal values for the average fitness on seven functions. Compared to the original ALA algorithm, DALA outperforms ALA in terms of average fitness on nine functions, excluding F5. For the unimodal functions F1 and

F3, the optimization performance of DALA and ALA significantly surpasses that of the other comparative algorithms, demonstrating the effectiveness of the original algorithm's strategy on simple functions. Among the multimodal functions F4-F7, DALA demonstrates a substantial advantage over other functions in solving problems F5 and F7, while both ALA and MVO can achieve high-precision convergence. For the hybrid functions F8-F10, DALA and MVO exhibit superior performance, with DALA achieving optimal values on two of these functions. Overall, the data in the table suggest that DALA is well-balanced in solving both simple and complex functions, exhibiting commendable performance across various function tests. Additionally, the standard deviation values of DALA are significantly lower than those of other algorithms, indicating stable algorithm results, effective solutions, and resistance to local optima.

Figure 6 shows the convergence curves of DALA and other comparison algorithms. It can be observed that in these test cases, DALA exhibits rapid convergence speed and high convergence accuracy. For unimodal functions, DALA's early convergence speed is not significantly different from the original algorithm, but in the latter half of the process, DALA switches strategies to accelerate local exploration, thereby significantly improving the convergence speed and ultimately achieving the best fitness among the nine algorithms. On unimodal functions F1–F3, Particle Swarm Optimization (PSO), Simulated Annealing (SSA), and DALA demonstrate rapid convergence speed in early iterations, while Grey Wolf Optimization (GWO), Artificial Bee Colony Algorithm (ABC), and Sparrow Swarm Optimization (MVO) provide similar convergence accuracy. On multimodal functions F4–F7, the charts show that on three functions (F5, F6, and F7), ALA and DALA exhibit significantly better convergence performance in the middle and later stages compared to other algorithms, while the Dynamic Multi-Objective Optimization Algorithm (DMOA) demonstrates poor convergence results. On hybrid functions F8–F10, ALA performs well in the middle and later stages, and GWO performs well in the early stages, making them competitive.

*Friedman Test.*The Friedman test is a non-parametric statistical method commonly used at a significance level of 0.05 to determine whether performance differences among multiple algorithms across multiple problems are statistically significant. This test ranks the performance of each algorithm on each problem, with a smaller rank indicating better performance. A key indicator in the test results is the p-value; the smaller the p-value, the more significant the differences among the algorithms. Additionally, the test table includes SS (sum of squares), df (degrees of freedom), MS (mean square, SS divided by df), and the chi-square statistic, which are used to assess the significance of performance differences among algorithms. As shown in

Table 11: Statistical Results of Experiments Using CEC2020 Test Functions

Function	**DALA**	**ALA**	**DOA**	**DMOA**	**MVO**	**PSO**	**GWO**	**SSA**	**ABC**
	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean
	Std	Std	Std	Std	Std	Std	Std	Std	Std
F1	**7.88E+02**	5.18E+03	4.92E+03	1.88E+06	2.75E+05	1.46E+05	8.42E+08	2.96E+03	3.20E+05
	8.27E+02	3.74E+03	3.88E+03	3.32E+06	8.80E+04	8.81E+04	7.68E+08	3.82E+03	4.08E+05
F2	**2.36E+03**	3.14E+03	5.09E+03	5.70E+03	3.09E+03	2.88E+03	2.84E+03	3.05E+03	4.98E+03
	4.91E+02	4.47E+02	4.68E+02	2.60E+02	5.57E+02	5.12E+02	4.63E+02	5.51E+02	2.62E+02
F3	**7.69E+02**	7.92E+02	8.94E+02	8.47E+02	7.91E+02	7.97E+02	7.84E+02	8.93E+02	8.42E+02
	1.35E+01	2.32E+01	3.80E+01	9.41E+00	1.74E+01	2.44E+01	2.23E+01	4.50E+01	1.03E+01
F4	**1.90E+03**	1.91E+03	1.91E+03	1.91E+03	1.91E+03	1.91E+03	1.99E+03	1.91E+03	1.91E+03
	1.27E+00	1.91E+00	3.27E+00	1.27E+00	2.12E+00	2.10E+00	2.50E+02	3.25E+00	9.35E-01
F5	1.03E+04	**9.11E+03**	1.44E+05	1.62E+06	2.55E+05	2.47E+05	1.08E+06	2.44E+05	1.40E+06
	1.54E+04	5.16E+03	8.75E+04	7.65E+05	2.40E+05	1.88E+05	9.83E+05	1.32E+05	6.79E+05
F6	**1.74E+03**	1.81E+03	2.04E+03	2.51E+03	2.01E+03	2.16E+03	2.04E+03	2.00E+03	1.77E+03
	1.23E+02	1.46E+02	2.20E+02	1.51E+02	1.66E+02	2.39E+02	2.36E+02	2.31E+02	8.60E+01
F7	**2.79E+03**	4.11E+03	1.77E+05	5.09E+05	1.00E+05	1.23E+05	2.66E+05	1.20E+05	2.20E+05
	5.36E+02	8.44E+02	1.93E+05	2.99E+05	1.08E+05	1.04E+05	5.06E+05	9.81E+04	1.29E+05
F8	2.85E+03	3.61E+03	2.94E+03	6.61E+03	3.68E+03	3.27E+03	2.91E+03	4.00E+03	**2.31E+03**
	1.15E+03	1.39E+03	1.52E+03	1.39E+03	1.28E+03	1.34E+03	1.10E+03	1.59E+03	1.08E+01
F9	**2.85E+03**	2.87E+03	2.92E+03	2.94E+03	2.86E+03	3.07E+03	2.87E+03	2.94E+03	2.91E+03
	2.22E+01	2.03E+01	4.83E+01	9.23E+00	2.41E+01	5.95E+01	3.45E+01	6.33E+01	9.69E+00
F10	2.95E+03	2.94E+03	2.95E+03	2.94E+03	**2.92E+03**	2.95E+03	3.01E+03	2.95E+03	2.93E+03
	3.40E+01	3.00E+01	3.16E+01	3.98E+01	2.13E+01	3.31E+01	5.19E+01	3.11E+01	1.97E+01

the results in Table 12, the p-value = 9.893e-06, which is far below 0.05, indicating that there are significant differences in both accuracy and computational time.

Table 12: Friedman Analysis of Variance

Source	**SS**	**df**	**MS**	**Chi-sp**	**Prob>Chi-sp**
Columns	263.6	8	32.95	37.357	9.893e-06
Error	300.9	72	4.1792	–	–
Total	564.5	89	–	–	–

Table 13 shows that DALA achieved the highest ranking with an average rank of 1.05. ALA, MVO, and SSA secured the second, third, and fourth positions with average ranks of 4.2, 4.35, and 5.1, respectively. ABC, DOA, and PSO followed in the fifth, sixth, and seventh places, while GWO and DMOA ranked last in the eighth and ninth positions.

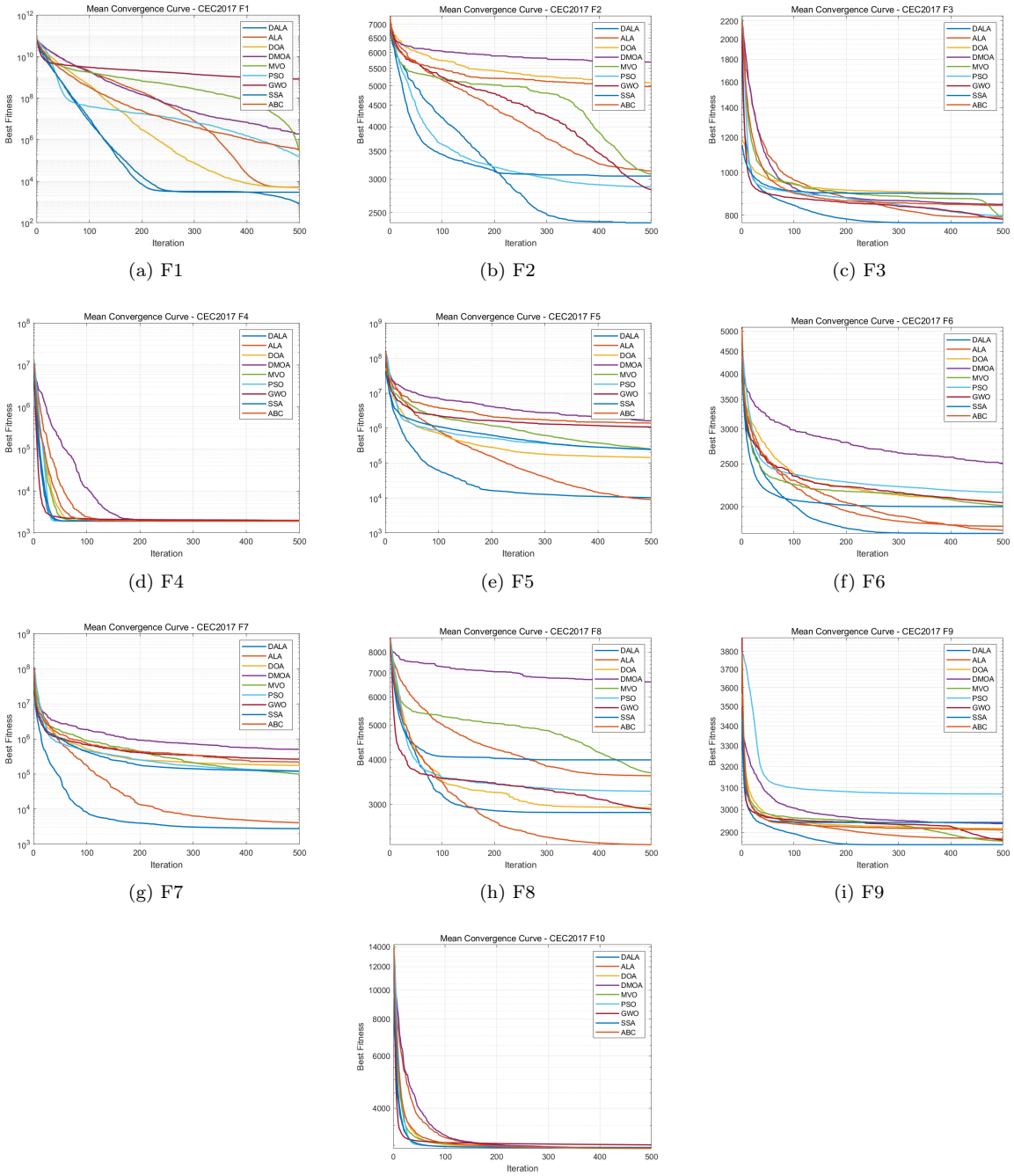
Table 13: Algorithm Rankings Based on Friedman Test

Algorithm	DALA	ALA	DOA	DMOA	MVO	PSO	GWO	SSA	ABC
Average Rank	1.05	4.2	5.35	7.7	4.35	5.60	6.35	5.1	5.3
Rank	1	2	6	9	3	7	8	4	5

*Wilcoxon Signed-Rank Test.*This study conducts the Wilcoxon signed-rank test to detect significant differences between the algorithms' behaviors. Table 14 presents the Wilcoxon signed-rank test results comparing the DALA algorithm with eight other contrasting algorithms on the IEEE CEC2020 benchmark functions. The data reveals that for the vast majority of functions, the p values are significantly below the 0.05 significance level, indicating substantial differences between these contrasting algorithms and DALA. These experiments demonstrate that DALA exhibits superior optimization performance and greater improvement compared to the original algorithm and other algorithms.

Table 14: Wilcoxon signed-rank test results for CEC2020 functions

Function	**ALA**	**DOA**	**DMOA**	**MVO**	**PSO**	**GWO**	**SSA**	**ABC**
F1	1.13E-05	6.98E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	2.43E-02	1.73E-06
F2	1.80E-05	1.73E-06	1.73E-06	2.37E-05	8.73E-03	2.58E-03	3.41E-05	1.73E-06
F3	2.41E-04	1.73E-06	1.73E-06	3.41E-05	3.41E-05	9.84E-03	1.73E-06	1.73E-06
F4	3.38E-03	1.73E-06	1.73E-06	4.45E-05	2.35E-06	1.73E-06	1.36E-05	1.73E-06
F5	2.06E-01	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06
F6	4.68E-03	2.37E-05	1.73E-06	7.69E-06	4.73E-06	6.89E-05	2.37E-05	2.37E-01
F7	1.97E-05	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06	1.73E-06
F8	7.27E-03	2.54E-01	1.73E-06	7.73E-03	1.32E-02	8.97E-02	2.58E-03	1.71E-01
F9	1.15E-04	1.73E-06	1.73E-06	1.66E-02	1.73E-06	1.57E-02	3.88E-06	1.92E-06
F10	3.39E-01	9.43E-01	1.65E-01	1.57E-02	8.77E-01	4.07E-05	8.29E-01	2.30E-02
+/−/=	7/2/1	8/2/0	9/1/0	9/0/1	9/1/0	9/1/0	9/1/0	7/2/1



(a) F1 (b) F2 (c) F3

(d) F4 (e) F5 (f) F6

(g) F7 (h) F8 (i) F9

(j) F10

Figure 6: Convergence Curves of Experiments Using CEC2020 Test Functions

4.3. Constrained Engineering Problem Solving

To evaluate the optimization performance of the DALA algorithm in practical applications, this paper selects five classic engineering problems for testing: the pressure vessel design problem, three-bar truss problem, welded beam design problem, gear train design problem and stepped pulley design problem [31, 32]. The results are as follows.

4.3.1. Pressure Vessel Design Problem

The pressure vessel design problem aims to adjust design variables such as material and wall thickness within specified requirements to minimize weight while ensuring sufficient strength, thereby guaranteeing safety in high-pressure operating environments. On this basis, the total manufacturing cost of the pressure vessel is minimized. This problem considers four variables: shell thickness T_s, head thickness T_h, inner radius R, and pipe length L. The design optimization is subject to constraints related to material properties, shape, and other factors. The specific structure of the pressure vessel is shown in Figure 7.

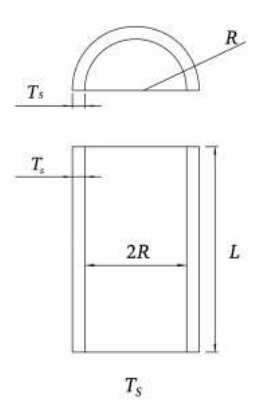


Figure 7: Schematic Diagram of the Pressure Vessel Design Problem

The mathematical formulation of the pressure vessel design problem is as follows:

$$\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \tag{23}$$

Objective function:

$$f(\overrightarrow{x}) = 0.6224x_1x_3 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{24}$$

Constraints:

$$g_1(\overrightarrow{x}) = -x_1 + 0.0193x_3 \leq 0 \tag{25}$$
$$g_2(\overrightarrow{x}) = -x_2 + 0.00954x_3 \leq 0 \tag{26}$$
$$g_3(\overrightarrow{x}) = -\pi x_3^2 x_4 + \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0 \tag{27}$$
$$g_4(\overrightarrow{x}) = x_4 - 240 \leq 0 \tag{28}$$

Variable bounds:

$$0 \leq x_1 \leq 99, \quad 0 \leq x_2 \leq 99, \quad 10 \leq x_3 \leq 200, \quad 10 \leq x_4 \leq 200 \tag{29}$$

The experimental results of the pressure vessel design problem are shown in Table 15. From the data in Table 15, it can be seen that the DALA algorithm obtained a shell thickness $T_s = 0.7782$, head thickness $T_h = 0.3846$, inner radius $R = 40.3196$, and length $L = 200$. At this point, the minimum cost is 5885.3, which is 3.7% higher than that obtained by the original ALA algorithm. Moreover, among the other six comparison algorithms tested on the pressure vessel design problem, four algorithms produced minimum costs greater than 6000. Compared to these six algorithms, DALA achieves the best optimization performance.This demonstrates that the algorithm can solve engineering problems with geometric constraints more efficiently than its counterparts, thereby identifying greater potential for weight reduction and cost savings while simultaneously guaranteeing the structural safety factor.

Table 15: Experimental Results of the Pressure Vessel Design Problem

Algorithm	x_1	x_2	x_3	x_4	f_{best}
DALA	0.7782	0.3846	40.3196	200.0000	5885.3
ALA	0.7863	0.3887	40.7404	194.2245	5899.4
DOA	0.8198	0.4114	42.4010	173.1276	5994.5
DMOA	0.8664	0.4338	44.8833	144.9350	6074.9
MVO	1.2921	0.6334	66.0710	10.0000	7689.4
PSO	0.8161	0.4034	42.2867	174.3180	5953.4
GWO	0.7882	0.3944	40.8262	193.2213	5922.2
SSA	0.8566	0.4234	44.3809	150.2673	6033.3
ABC	1.0355	0.5207	53.5306	73.0062	6558.3

4.3.2. Three-Bar Truss Problem

The challenge of the three-bar truss problem is to create a truss composed of three bars, where the design optimization involves adjusting parameters such as the cross-sectional areas and lengths of the bars to achieve the lightest possible structure that meets strength and stiffness requirements. The structure must exhibit optimal strength and stiffness while bearing external loads. Stress, deflection, and buckling constitute three optimization constraints applied to the two optimization variables in this test, as expressed in Equations (32)–(34). Figure 8 depicts the overall structure of the three-bar truss. The mathematical formulation of the three-bar truss problem

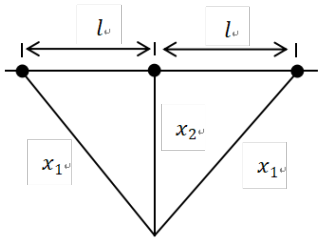


Figure 8: Schematic Diagram of the Three-Bar Truss Problem

is as follows:

$$\overrightarrow{x} = [x_1, x_2] = [S_1, S_2] \tag{30}$$

Objective function:

$$f(\overrightarrow{x}) = x_2 + 2\sqrt{2}x_1 \tag{31}$$

Constraints:

$$g_1(\overrightarrow{x}) = \frac{P(x_2 + \sqrt{2}x_1)}{2x_1x_2 + \sqrt{2}x_1^2} - \sigma \leq 0 \tag{32}$$

$$g_2(\overrightarrow{x}) = \frac{Px_2}{2x_1x_2 + \sqrt{2}x_1^2} - \sigma \leq 0 \tag{33}$$

$$g_3(\overrightarrow{x}) = \frac{P}{x_1 + \sqrt{2}x_2} - \sigma \leq 0 \tag{34}$$

Where:

$$10 < z_s < 200, \quad 10 < a_y < 200, \quad x_1, x_2 \leq 1$$
$$L = 100\,\text{cm}, \quad \sigma_{\max} = 2\,\text{kN/cm}^2, \quad P = 2\,\text{kN}$$

The experimental results of the three-bar truss problem are shown in Table 16. From the data presented in Table 16, it can be seen that the DALA algorithm obtained a cross-sectional area $S_1 = 0.7887$ for the two side bars and $S_2 = 0.4082$ for the vertical bar. The experimental results indicate that both DALA and ALA algorithms found the optimal solution for the three-bar truss problem, significantly outperforming the other algorithms. MVO, GJO, and MFO showed moderate performance, while WOA and GOOSE yielded poorer results, demonstrating clear differences in optimization capability among these algorithms for the three-bar truss problem.For large-scale truss systems, even marginal parameter optimizations can significantly reduce raw material procurement costs and structural self-weight through cumulative effects. Consequently, these results provide robust support for the automated optimization of complex engineering designs.

Table 16: Experimental Results of the Three-Bar Truss Problem

Algorithm	x_1	x_2	f_{best}
DALA	0.7887	0.4082	263.8958
ALA	0.7887	0.4083	263.8958
DOA	0.7888	0.4078	263.8959
DMOA	0.7889	0.4075	263.8960
MVO	0.7925	0.3976	263.9073
PSO	0.7917	0.3997	263.9025
GWO	0.7862	0.4153	263.9026
SSA	0.7882	0.4097	263.8960
ABC	0.7880	0.4102	263.9010

4.3.3. Welding Beam Design Problem

The welding beam design problem is a common and challenging issue in structural engineering. The objective of the welding beam design problem is to optimize the shape, size, and layout parameters of the welds under given constraints to achieve optimal structural performance while reducing weight. In this problem, four variables are considered: weld width h, connecting beam length l, beam height t, and connecting beam thickness b. In the design optimization of this engineering problem, these four decision variables, along with seven other different constraints, must be satisfied.

The mathematical formulation of the welding beam design problem is as follows. Consider the design vector as:

$$\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b] \tag{35}$$

Objective function:

$$f(\overrightarrow{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \tag{36}$$

Constraints:

$$g_1(\overrightarrow{x}) = \tau(\overrightarrow{x}) - \tau_{\max} \leq 0 \tag{37}$$
$$g_2(\overrightarrow{x}) = \sigma(\overrightarrow{x}) - \sigma_{\max} \leq 0 \tag{38}$$
$$g_3(\overrightarrow{x}) = \delta(\overrightarrow{x}) - \delta_{\max} \leq 0 \tag{39}$$
$$g_4(\overrightarrow{x}) = x_1 - x_4 \leq 0 \tag{40}$$
$$g_5(\overrightarrow{x}) = P - P_c(\overrightarrow{x}) \leq 0 \tag{41}$$
$$g_6(\overrightarrow{x}) = 0.125 - x_1 \leq 0 \tag{42}$$
$$g_7(\overrightarrow{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 0.5 \leq 0 \tag{43}$$

Where,

$$\tau(\overrightarrow{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2} \tag{44}$$

$$\tau' = \frac{P}{\sqrt{2x_1x_2}}, \quad \tau'' = \frac{MR}{J} \tag{45}$$

$$M = P\left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2} \tag{46}$$

$$\sigma(\overrightarrow{x}) = \frac{6PL}{x_4x_3^2} \tag{47}$$

$$J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \tag{48}$$

$$\delta(\overrightarrow{x}) = \frac{6PL^3}{x_4x_3^3} \tag{49}$$

$$P_c(\overrightarrow{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2} \cdot \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)^2 \tag{50}$$

Parameters:

$$\begin{aligned} P &= 6000\,\text{lb} \\ L &= 14\,\text{in} \\ \delta_{\max} &= 0.25\,\text{in} \\ E &= 30 \times 10^6\,\text{psi} \\ \tau_{\max} &= 13600\,\text{psi} \\ \sigma_{\max} &= 30000\,\text{psi} \end{aligned}$$

Variable bounds:

$$\begin{aligned} 0.1 \leq x_i \leq 2,\ i = 1, 4 \\ 0.1 \leq x_i \leq 10,\ i = 2, 3 \end{aligned}$$

The experimental results of the welding beam design problem are shown in Table 17.From the data presented in Table 17, it can be seen that the DALA algorithm obtained a weld width $h = 0.2057$, connecting beam length $l = 3.2549$, beam height $t = 9.03662$, and connecting beam thickness $b = 0.20574$. The final result is the same

as that of the original algorithm and ranks as the best among the seven algorithms tested, followed by GJO, MVO, and MFO, all of which obtained results less than 1.8. This indicates that ALA already has excellent capability in solving the welding beam design problem, and since DALA maintains consistency with ALA in the main search strategy, both DALA and ALA achieve identical search results.The results validate the robustness of the strategy, providing engineers with a standardized optimization paradigm for conducting stress analysis on complex welded structures.

Table 17: Experimental Results of the Welding Beam Design Problem

Algorithm	x_1	x_2	x_3	x_4	f_{best}
DALA	0.2413	2.6371	9.0366	0.2057	1.670216
ALA	0.2416	2.6327	9.0366	0.2057	1.670225
DOA	0.2401	2.6528	9.0366	0.2057	1.670238
DMOA	0.2409	2.6415	9.0366	0.2057	1.670255
MVO	0.2278	2.7721	9.2240	0.2048	1.688598
PSO	0.2449	2.5880	9.0366	0.2057	1.670432
GWO	0.2322	2.7728	9.0330	0.2059	1.673173
SSA	0.2424	2.6209	9.0366	0.2057	1.670239
ABC	0.2412	2.6385	9.0367	0.2057	1.670365

4.3.4. Gear Transmission Design Problem

Gear transmission systems are widely used in various machinery and power equipment, and their design quality directly affects the overall system efficiency, precision, and stability. The objective of this optimization problem is to adjust the number of teeth n_A, n_B, n_D, n_F on four gears to minimize the transmission ratio error, ensuring that the actual transmission ratio of the system is as close as possible to the target transmission ratio. This is a well-known engineering design challenge, as illustrated in Figure 9. The mathematical formulation of the gear transmission design problem

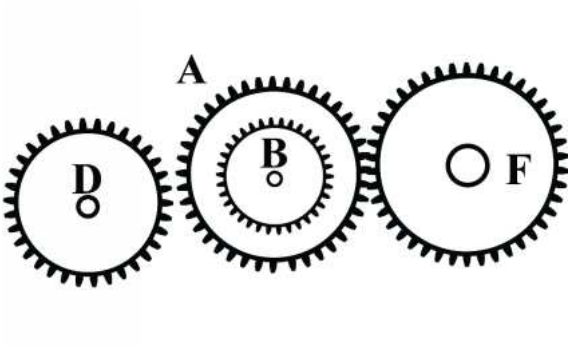


Figure 9: Schematic Diagram of the Gear Transmission Problem

is as follows. Consider the design vector as:

$$\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [n_A, n_B, n_D, n_F] \tag{51}$$

Objective function:

$$f(\overrightarrow{x}) = \left(\frac{1}{6.931} - \frac{x_2 x_3}{x_1 x_4}\right)^2 \tag{52}$$

Variable bounds:

$$12 \leq x_1, x_2, x_3, x_4 \leq 60, \quad x_i \in \mathbb{Z} \tag{53}$$

The experimental results of the gear transmission design problem are shown in Table 18.From the data in Table 18, it can be seen that the DALA algorithm obtained a number of teeth for the driving gears $n_A = 43$ and $n_B = 16$, and for the driven gears $n_D = 19$ and $n_F = 49$. At this point, the minimum cost is 2.7009×10^{-12}, with the transmission coefficient being 98% higher than that of the original ALA algorithm. In the gear transmission design problem, GJO performed relatively well among the six comparison algorithms, achieving a transmission coefficient of 2.3078×10^{-11}, while the other five algorithms yielded transmission coefficients greater than 1×10^{-10}. It can be observed that compared to the minimal costs calculated by the other six

algorithms, DALA achieves the best optimization performance for this problem.The optimal solution of DALA facilitates smoother torque fluctuations in practical manufacturing, thereby mitigating gear surface wear and reducing operational noise, which is of great significance for extending the fatigue life of mechanical systems.

Table 18: Experimental Results of the Gear Transmission Design Problem

Algorithm	x_1	x_2	x_3	x_4	f_{best}
DALA	52.7575	30.0929	13.2020	50.7445	2.3078×10^{-11}
ALA	59.9691	27.5691	17.3510	54.7314	1.3616×10^{-9}
DOA	49.4935	13.2411	31.3832	56.9027	9.9399×10^{-11}
DMOA	51.2812	25.6759	14.9582	52.5989	2.3078×10^{-11}
MVO	53.6093	12.0000	36.8944	57.1514	8.8876×10^{-10}
PSO	49.0434	16.3546	19.2851	42.6955	2.7009×10^{-12}
GWO	49.3194	18.9374	15.8704	42.9937	2.7009×10^{-12}
SSA	58.8943	20.4077	19.5165	46.8350	9.7457×10^{-10}
ABC	37.4186	23.4234	12.5406	55.6667	6.6021×10^{-10}

4.3.5. Stepped pulley design problem

The core objective of the stepped pulley optimization design problem is to maintain constant belt tension, as variations in belt length during shifts between different pulley steps would lead to slippage and accelerated wear. This engineering challenge aims to identify the optimal combination of four-level pulley working diameters d_1, d_2, d_3, d_4 and belt w width while satisfying geometric constraints of belt drive systems, wrap angle requirements, power transmission capacity, and consistency of center distance across all transmission stages. The ultimate goal is to minimize the total system mass to address this multi-speed transmission engineering problem.The specific structure of the Step-cone pulley is shown in Figure 10. The mathematical

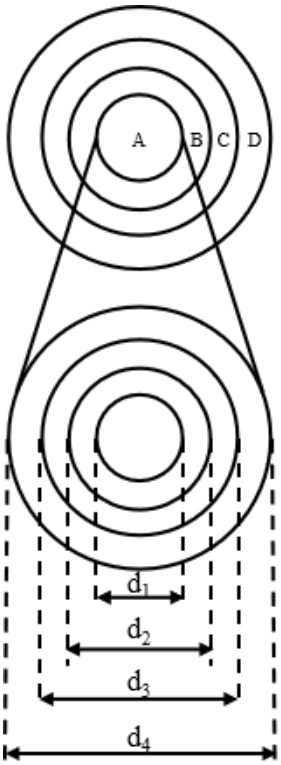


Figure 10: Schematic Diagram of the Step-cone pulley Problem

formulation of the stepped pulley problem is as follows:

Objective function:

$$\text{fitness} = f + \text{panalty} \tag{54}$$

$$f = \rho \cdot w \cdot \frac{\pi}{4}\left[d_1^2\left(1+\left(\frac{N_1}{N}\right)^2\right)+d_2^2\left(1+\left(\frac{N_2}{N}\right)^2\right)+d_3^2\left(1+\left(\frac{N_3}{N}\right)^2\right)+d_4^2\left(1+\left(\frac{N_4}{N}\right)^2\right)\right] \tag{55}$$

$$\text{panalty} = \text{panalty_factor}\left[\sum_{g_i>0} g_i^2 + \sum_{h_i\neq 0} h_i^2\right] \tag{56}$$

Belt length constraints:

$$C_1 = \frac{\pi d_1}{2}\left(1+\frac{N_1}{N}\right)+\frac{\left(\frac{N_1}{N}-1\right)^2 d_1^2}{4a}+2a \tag{57}$$

$$C_2 = \frac{\pi d_2}{2}\left(1+\frac{N_2}{N}\right)+\frac{\left(\frac{N_2}{N}-1\right)^2 d_2^2}{4a}+2a \tag{58}$$

$$C_3 = \frac{\pi d_3}{2}\left(1+\frac{N_3}{N}\right)+\frac{\left(\frac{N_3}{N}-1\right)^2 d_3^2}{4a}+2a \tag{59}$$

$$C_4 = \frac{\pi d_4}{2}\left(1+\frac{N_4}{N}\right)+\frac{\left(\frac{N_4}{N}-1\right)^2 d_4^2}{4a}+2a \tag{60}$$

Wrap angle coefficients:

$$R_1 = \exp\left[\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_1}{N}-1\right)d_1}{2a}\right)\right)\right] \tag{61}$$

$$R_2 = \exp\left[\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_2}{N}-1\right)d_2}{2a}\right)\right)\right] \tag{62}$$

$$R_3 = \exp\left[\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_3}{N}-1\right)d_3}{2a}\right)\right)\right] \tag{63}$$

$$R_4 = \exp\left[\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_4}{N}-1\right)d_4}{2a}\right)\right)\right] \tag{64}$$

Power constraints:

$$P_1 = s \cdot t \cdot w \cdot \left[1 - \exp\left(-\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_1}{N} - 1\right) d_1}{2a}\right)\right)\right)\right] \cdot \frac{\pi d_1 N_1}{60} \tag{65}$$

$$P_2 = s \cdot t \cdot w \cdot \left[1 - \exp\left(-\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_2}{N} - 1\right) d_2}{2a}\right)\right)\right)\right] \cdot \frac{\pi d_2 N_2}{60} \tag{66}$$

$$P_3 = s \cdot t \cdot w \cdot \left[1 - \exp\left(-\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_3}{N} - 1\right) d_3}{2a}\right)\right)\right)\right] \cdot \frac{\pi d_3 N_3}{60} \tag{67}$$

$$P_4 = s \cdot t \cdot w \cdot \left[1 - \exp\left(-\mu\left(\pi - 2\sin^{-1}\left(\frac{\left(\frac{N_4}{N} - 1\right) d_4}{2a}\right)\right)\right)\right] \cdot \frac{\pi d_4 N_4}{60} \tag{68}$$

Inequality constraints:

$$g(1) = -R_1 + 2 \leq 0 \tag{69}$$

$$g(2) = -R_2 + 2 \leq 0 \tag{70}$$

$$g(3) = -R_3 + 2 \leq 0 \tag{71}$$

$$g(4) = -R_4 + 2 \leq 0 \tag{72}$$

$$g(5) = -P_1 + 0.75 \times 745.6998 \leq 0 \tag{73}$$

$$g(6) = -P_2 + 0.75 \times 745.6998 \leq 0 \tag{74}$$

$$g(7) = -P_3 + 0.75 \times 745.6998 \leq 0 \tag{75}$$

$$g(8) = -P_4 + 0.75 \times 745.6998 \leq 0 \tag{76}$$

Equality constraints:

$$h(1) = C_1 - C_2 = 0 \tag{77}$$

$$h(2) = C_1 - C_3 = 0 \tag{78}$$

$$h(3) = C_1 - C_4 = 0 \tag{79}$$

The experimental results of the stepped pulley problem are shown in Table 19. From the data in Table 19, it can be seen that the DALA algorithm obtained working diameters of $d_1 = 38.4140$,$d_2 = 52.8586$,$d_3 = 70.4727$,$d_4 = 84.4957$. At this point, the minimum pulley width w=16.090, and the transmission coefficient is 30% higher than the original ALA algorithm. In the stepped pulley design problem, it can be observed that compared to the minimum costs calculated by the other 8 algorithms for this problem, DALA has the best optimization effect. Additionally, SSA and

ABC also show good computational performance, achieving optimal solutions below 16.5.For machining equipment requiring constant power or constant torque output, this optimization directly enhances the accuracy of the transmission ratio, thereby ensuring the surface quality of the processed workpieces and substantially reducing the costs associated with engineering simulation and physical prototype debugging.

Table 19: Experimental Results of the Stepped Pulley Design Problem

Algorithm	d_1	d_2	d_3	d_4	f_{best}
DALA	38.4140	52.8586	70.4727	84.4957	16.090
ALA	39.6126	54.5093	72.6733	87.1319	16.591
DOA	40.9044	56.2883	75.0448	89.9727	17.131
DMOA	40.1778	55.2877	73.7130	88.3753	17.204
MVO	40.4124	55.6022	74.1356	88.8968	30.765
PSO	39.5789	54.4628	72.6113	87.0576	16.577
GWO	40.9150	56.2896	75.0492	90.0000	57.685
SSA	39.3129	54.0966	72.1231	86.4728	16.466
ABC	38.6354	53.1637	70.8793	84.9829	16.186

4.3.6. Power load forecasting problem using multi-layer LSTM

Short-term load forecasting is a critical technology for the safe, stable, and economical operation of smart grids. However, load sequences exhibit high levels of nonlinearity, non-stationarity, and multi-timescale characteristics, posing significant challenges to accurate prediction. Deep LSTM models are widely used in power load forecasting due to their powerful capability for temporal feature extraction. Compared to single-layer LSTM, a three-layer architecture can establish a hierarchical temporal feature extraction mechanism: the bottom LSTM layer captures short-term fluctuations such as hourly load variations, the middle layer learns daily cyclical patterns, and the top layer extracts long-term trends including seasonal patterns and holiday effects. This multi-scale modeling capability is particularly crucial for power load data [33, 34, 35].

This research aims to minimize the generalization error of the deep learning model in the task of power load forecasting by optimizing its hyperparameter configuration. Specifically, the design objective is to minimize the Root Mean Square Error (RMSE) of the model on the test set by adjusting nine key hyperparameters in the three-layer Long Short-Term Memory (LSTM) network. These hyperparameters include the number of neurons in each layer, learning rate, L2 regularization coefficient, dropout rate, batch size, and convolutional kernel multiplier, thereby enhancing prediction accuracy and model performance.

Decision Variables: The optimization involves nine key hyperparameters that define the architecture and training strategy of the deep learning model:

$$\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9] \tag{80}$$

Constraints:

$$x_1 : \text{L2 regularization coefficient} \qquad 1 \times 10^{-10} \leq x_1 \leq 1 \times 10^{-2} \tag{81}$$

$$x_2 : \text{Initial learning rate} \qquad 0.001 \leq x_2 \leq 0.01 \tag{82}$$

$$x_3 : \text{LSTM layer 1 neuron count} \qquad 20 \leq x_3 \leq 256 \tag{83}$$

$$x_4 : \text{LSTM layer 2 neuron count} \qquad 20 \leq x_4 \leq 256 \tag{84}$$

$$x_5 : \text{LSTM layer 3 neuron count} \qquad 20 \leq x_5 \leq 256 \tag{85}$$

$$x_6 : \text{Dropout rate} \qquad 0.1 \leq x_6 \leq 0.5 \tag{86}$$

$$x_7 : \text{Learning rate drop factor} \qquad 0.1 \leq x_7 \leq 0.9 \tag{87}$$

$$x_8 : \text{Batch size} \qquad 64 \leq x_8 \leq 512 \tag{88}$$

$$x_9 : \text{Convolutional kernel multiplier} \qquad 2 \leq x_9 \leq 8 \tag{89}$$

Objective function:

$$\min_{\mathbf{x}} f(\mathbf{x}) = \text{RMSE}\left(\mathcal{M}_{\text{LSTM}}(\mathbf{x}; \mathcal{D}_{\text{train}}), \mathcal{D}_{\text{test}}\right) \tag{90}$$

where, $\mathbf{x}$ represents the hyperparameter vector to be optimized, $\mathcal{M}_{\text{LSTM}}(\mathbf{x}; \mathcal{D}_{\text{train}})$ denotes the three-layer LSTM model trained with hyperparameters $\mathbf{x}$ on training dataset $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}$ is the test dataset, RMSE is the Root Mean Square Error metric.

The overall hyperparameter optimization problem can be formally stated as:

$$\begin{aligned}
\underset{\mathbf{x}\in\mathbb{R}^9}{\text{minimize}} \quad & f(\mathbf{x}) = \sqrt{\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \left(y_i - \hat{y}_i(\mathbf{x})\right)^2} && (91)\\
\text{subject to} \quad & x_3, x_4, x_5, x_8, x_9 \in \mathbb{Z}^+ && (92)\\
& \hat{y}_i(\mathbf{x}) = \mathcal{M}_{\text{LSTM}}(\mathbf{x}; \mathcal{D}_{\text{train}})(\mathbf{x}_i^{\text{test}}) && (93)
\end{aligned}$$

The four evaluation metrics selected in this study are calculated as follows: the Mean Square Error (MSE) computes the average of the squared differences between predicted and actual values; the Root Mean Square Error (RMSE) is the arithmetic square root of MSE, which shares the same unit of measurement as the target variable; the Mean Absolute Error (MAE) calculates the average of absolute errors; and the Mean Absolute Percentage Error (MAPE) measures the average of absolute errors relative to the actual values in percentage terms. These four formulas quantify prediction errors from different dimensions, the experimental results of the stepped pulley problem are shown in Table 20.

From the evaluation results, the DALA algorithm demonstrates the best performance across all four error metrics, with the lowest values of RMSE=2.3405, MSE=5.4778, MAE=1.8135, and MAPE=3.79%, indicating superior prediction accuracy and stability. The GWO and SSA algorithms perform moderately well, with relatively close and mid-to-high-level error metrics. In contrast, the ALA algorithm shows the worst performance among all compared methods, with significantly higher error values across all indicators, highlighting the effectiveness of the improvements in high-dimensional applications.

In terms of prediction accuracy and decision support, the reduction in RMSE and MAE indicates that the model captures non-linear load fluctuations more accurately. In power dispatch or resource planning, this improvement in precision leads to more reliable forecasting trajectories, thereby reducing the risks of reserve capacity waste or power supply shortages caused by prediction bias. This demonstrates

the reliability of the proposed adaptive mechanism in handling complex neural network parameter spaces and provides an effective optimization scheme for time-series forecasting tasks.

Table 20: Experimental Results of the Gear Transmission Design Problem

Algorithm	RMSE	MSE	MAE	MAPE
DALA	2.3405	5.4778	1.8135	3.79%
ALA	3.3767	11.4018	2.8131	5.9326%
DOA	2.9502	8.7037	2.2367	4.7721%
DMOA	2.8543	8.1472	2.4388	5.4535%
MVO	2.7887	7.7771	2.4411	5.2593%
PSO	3.0463	9.2802	2.7212	5.8237%
GWO	2.598	6.7496	2.2246	4.8592%
SSA	2.5841	6.6774	2.3842	5.1174%
ABC	2.8829	8.3109	2.4206	5.1172%

5. Conclusion and future work

The Artificial Lemming Algorithm (ALA) is inspired by the learning and simulation of lemming population behavior. However, ALA often suffers from insufficient final convergence precision in its later stages due to a lack of individual searchability. To further enhance performance and adapt to high-dimensional complex problems, this paper proposes a Dual-Stage Guided Artificial Lemming Algorithm (DALA), which introduces three strategies: a rank-based adaptive probability mechanism, a dual-stage greedy strategy combining opposition-based and topological neighborhood learning, and a mutation operator integrated from the JADE algorithm. The performance of DALA is validated through experiments on IEEE CEC2017 and CEC2020 benchmark functions (including unimodal, multimodal, hybrid, and fixed-dimension functions) against eight comparative algorithms, with results demonstrating DALA's advantages. To evaluate its practical problem-solving capability, simulation experiments were conducted on five constrained engineering design problems, yielding excellent results; additionally, DALA's application in optimizing multi-layer LSTM parameters for power load forecasting confirms its ability to obtain high-quality solutions for complex novel problems.

However, experimental findings reveal that DALA lacks sufficient intervention during the initialization phase, which leads to lower solution quality in the early

stages and a relatively slow initial convergence speed. Furthermore, the sensitivity of the adaptive switching between global exploration and local exploitation strategies remains inadequate; thus, more flexible strategy-switching mechanisms will be explored in future work. Current research primarily focuses on continuous optimization tasks. For more severe optimization scenarios—such as discrete-continuous mixed variables, large-scale combinatorial optimization, and dynamic multimodal environments—the performance efficacy and structural universality of DALA still require further mathematical verification and parameter tuning. Future plans involve applying DALA to a wider range of practical, complex engineering problems, including energy scheduling, structural design optimization, and manufacturing system layout, to thoroughly validate its pragmatism and robustness in industrial scenarios. By incorporating intelligent mechanisms such as co-evolution and transfer learning, the search efficiency and solution diversity of DALA are expected to be further enhanced.

Declarations

- **Funding**
  The authors did not receive support from any organization for the submitted work.

- **Conflict of interest/Competing interests**
  The authors have no relevant financial or non-financial interests to disclose.

- **Ethics approval and consent to participate**
  Not applicable. This study does not involve human participants or animals.

- **Consent for publication**
  All authors have read and approved the final manuscript for publication.

- **Data availability**
  The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

- **Dual publication**
  This manuscript has not been published previously and is not under consideration for publication elsewhere. All results, data, and figures are original to this work.

- **Authorship confirmation**
  The corresponding author confirms that she/he has read the journal's policies and that this submission complies with them.

- **Permission to use third-party material**
  Not applicable. All figures and tables are original and created by the authors. No third-party material has been reused.

- **Materials availability**
  Not applicable.

- **Code availability**
  The code for the LES mechanism and the improved algorithms will be made available upon reasonable request to the corresponding author.

- **Author contribution**
  J.G. and Y.Z. wrote the main manuscript text. J.G. performed the algorithmic modeling and implemented the LES mechanism. J.G. and Y.Z. designed the experimental framework and analyzed the data. J.G. prepared all figures and tables. All authors reviewed the manuscript.

References

[1] Z. Michalewicz, D. Dasgupta, R. G. Le Riche, et al. Evolutionary algorithms for constrained engineering problems. *Computers & Industrial Engineering*, 30(4):851–870, 1996.

[2] J. Tang, G. Liu, and Q. Pan. A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. *IEEE/CAA Journal of Automatica Sinica*, 8(10):1627–1643, 2021.

[3] H. Emami. Hazelnut tree search algorithm: a nature-inspired method for solving numerical and engineering problems. *Engineering with Computers*, 38(Suppl 4):3191–3215, 2022.

[4] H. Emami, M. Fardi, and B. Azarnavid. An enhanced seasons optimization algorithm for numerical optimization and engineering design. *Scientific Reports*, 15(1):25675, 2025.

[5] C. Yuan, D. Zhao, A. A. Heidari, et al. Polar lights optimizer: Algorithm and applications in image segmentation and feature selection. *Neurocomputing*, 607:128427, 2024.

[6] S. Mostafa Bozorgi and S. Yazdani. Iwoa: An improved whale optimization algorithm for optimization problems. *Journal of Computational Design and Engineering*, 6(3):243–259, 2019.

[7] L. Yang, Z. Li, D. Wang, et al. Software defects prediction based on hybrid particle swarm optimization and sparrow search algorithm. *IEEE Access*, 9:60865–60879, 2021.

[8] A. G. Hussien, A. Bouaouda, A. Alzaqebah, et al. An in-depth survey of the artificial gorilla troops optimizer: Outcomes, variations, and applications. *Artificial Intelligence Review*, 57(9):246, 2024.

[9] K. Hu, H. Jiang, C. G. Ji, et al. A modified butterfly optimization algorithm: An adaptive algorithm for global optimization and the support vector machine. *Expert Systems*, 38(3):e12642, 2021.

[10] Yanhui Li, Kaixuan Sun, Qi Yao, and Lin Wang. A dual-optimization wind speed forecasting model based on deep learning and improved dung beetle optimization algorithm. *Energy*, 286:129604, 2024.

[11] J. Faria, C. Marques, J. Pombo, et al. Optimal sizing of renewable energy communities: a multiple swarms multi-objective particle swarm optimization approach. *Energies*, 16(21):7227, 2023.

[12] Antonio R Moya and Sebastián Ventura. A multi-fidelity genetic algorithm for hyperparameter optimization of deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2025.

[13] Hao Liu, Yi Shen, Shuangjiang Yu, Zijun Gao, and Tong Wu. Deep reinforcement learning for mobile robot path planning. *arXiv preprint arXiv:2404.06974*, 2024.

[14] Bing Yan, Peng Shi, and Jonathon Chambers. Cooperative control for heterogeneous multi-agent systems: progress, applications, and challenges. *Science China Information Sciences*, 67(5):156201, 2024.

[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*, 58(3):84, 2025.

[16] X. Zhu, C. Jia, J. Zhao, et al. An enhanced artificial lemming algorithm and its application in uav path planning. *Biomimetics*, 10(6):377, 2025.

[17] Y. Xie, Z. Sun, K. Yuan, et al. 3d uav route optimization in complex environments using an enhanced artificial lemming algorithm. *Symmetry*, 17(6):946, 2025.

[18] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 2002.

[19] Stavros P Adam, Stamatios-Aggelos N Alexandropoulos, Panos M Pardalos, and Michael N Vrahatis. No free lunch theorem: A review. *Approximation and optimization: Algorithms, complexity and applications*, pages 57–82, 2019.

[20] Aldeida Aleti and Irene Moser. A systematic literature review of adaptive parameter control methods for evolutionary algorithms. *ACM Computing Surveys (CSUR)*, 49(3):1–35, 2016.

[21] Xi Jia, Tieyong Qin, Yu Lu, et al. Adaptive weighted ranking-oriented label distribution learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):11302–11316, 2023.

[22] M Ramachandran, Seyedali Mirjalili, Muthu MS Ramalingam, et al. A ranking-based fuzzy adaptive hybrid crow search algorithm for combined heat and power economic dispatch. *Expert Systems with Applications*, 197:116625, 2022.

[23] M. Khishe. Greedy opposition-based learning for chimp optimization algorithm. *Artificial Intelligence Review*, 56(8):7633–7663, 2023.

[24] Sedigheh Mahdavi, Shahryar Rahnamayan, and Kalyanmoy Deb. Opposition based learning: A literature review. *Swarm and evolutionary computation*, 39:1–23, 2018.

[25] A. J. Ma, P. C. Yuen, W. W. W. Zou, et al. Supervised spatio-temporal neighborhood topology learning for action recognition. *IEEE transactions on circuits and systems for video technology*, 23(8):1447–1460, 2013.

[26] A. Zhang, G. Sun, J. Ren, et al. A dynamic neighborhood learning-based gravitational search algorithm. *IEEE transactions on cybernetics*, 48(1):436–447, 2016.

[27] J. Ma, P. C. Yuen, W. Zou, et al. Supervised neighborhood topology learning for human action recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 476–481. IEEE, 2009.

[28] Jingqiao Zhang and Arthur C Sanderson. Jade: adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13(5):945–958, 2009.

[29] Janez Brest, Mirjam Sepesy Maučec, and Borko Bošković. Single objective real-parameter optimization: Algorithm jso. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1311–1318. IEEE, 2017.

[30] Chang Huang, Jie Zhu, and Ming Xu. Dys-mpade: A novel multipopulation adaptive differential evolution methodology based on dynamic subpopulation. *Journal of Computational Design and Engineering*, 12(3):204–225, 2025.

[31] Heming Jia, Jinrui Zhang, Honghua Rao, and Laith Abualigah. Improved sand-cat swarm optimization algorithm for solving global optimum problems. *Artificial Intelligence Review*, 58(1):5, 2024.

[32] João Luiz Junho Pereira, Guilherme Antônio Oliver, Matheus Brendon Francisco, Sebastiao Simoes Cunha Jr, and Guilherme Ferreira Gomes. A review of multi-objective optimization: methods and algorithms in mechanical engineering problems. *Archives of Computational Methods in Engineering*, 29(4):2285–2308, 2022.

[33] Xianlun Tang, Yuyan Dai, Ting Wang, and Yingjie Chen. Short-term power load forecasting based on multi-layer bidirectional recurrent neural network. *IET Generation, Transmission & Distribution*, 13(17):3847–3854, 2019.

[34] Chong Wang, Xiang Li, Yan Shi, Wenshuai Jiang, Qiong Song, and Xueling Li. Load forecasting method based on cnn and extended lstm. *Energy Reports*, 12:2452–2461, 2024.

[35] Ke Li, Wei Huang, Gaoyuan Hu, and Jiao Li. Ultra-short term power load forecasting based on ceemdan-se and lstm neural network. *Energy and Buildings*, 279:112666, 2023.