Scientific Reports

https://doi.org/10.1038/s41598-026-38028-2

Article in Press

Learning-aided Artificial Bee Colony with neural knowledge transfer for global optimization

Received: 26 September 2025

Accepted: 28 January 2026

Published online: 03 February 2026

Cite this article as: Saini G., Jadon S.S. & Chaube S. Learning-aided Artificial Bee Colony with neural knowledge transfer for global optimization. ***Sci Rep*** **(2026). https://doi.org/10.1038/s41598-026-38028-2**

Gurmeet Saini, Shimpi Singh Jadon & Shshank Chaube

Learning-Aided Artificial Bee Colony with Neural Knowledge Transfer for Global Optimization

Gurmeet Saini[a] (gurmeetsaini3397@gmail.com), Shimpi Singh Jadon[a] (shimpisingh2k6@gmail.com), and Shshank Chaube[b] (shshank.chaube@sithyd.siu.edu.in)

[a]Department of Applied Sciences, Rajkiya Engineering College Kannauj, Kannauj, India
[b]Symbiosis Institute of Technology, Hyderabad Campus, Symbiosis International (Deemed University),Pune,India.

Corresponding Authors:
1.) Shimpi Singh Jadon
Department of Applied Sciences
Rajkiya Engineering College Kannauj
Kannauj, 209732
India
Email: shimpisingh2k6@gmail.com
2.) Shshank Chaube
Department of Mathematical science
Symbiosis Institute of Technology, Hyderabad Campus
India
Email: shshank.chaube@sithyd.siu.edu.in

Learning-Aided Artificial Bee Colony with Neural Knowledge Transfer for Global Optimization

Gurmeet Saini[a], Shimpi Singh Jadon *[a], Shshank Chaube *[b]

[a]*Department of Applied Sciences, Rajkiya Engineering College Kannauj, Kannauj, India*
[b]*Symbiosis Institute of Technology, Hyderabad Campus, Symbiosis International (Deemed University),Pune,India.*
Email: gurmeetsaini3397@gmail.com (Gurmeet Saini)
Email: shimpisingh2k6@gmail.com (Shimpi Singh Jadon)
Email: shshank.chaube@sithyd.siu.edu.in (Shshank Chaube)

Abstract

A critical challenge in swarm intelligence is the effective utilization of knowledge gained during the search, a process often confounded by the risk of negative knowledge transfer. To address this, we introduce the Learning-Aided Artificial Bee Colony (LA-ABC), a novel framework guided by a Neural Knowledge Transfer mechanism for global optimization. Our framework establishes a co-evolutionary mechanism between the search process of the ABC algorithm and an online neural knowledge learning engine. LA-ABC operates on a dual-pathway architecture, probabilistically arbitrating between foundational swarm exploration and a knowledge-transfer pathway. In this second pathway, an Artificial Neural Network (ANN) learns a predictive, non-linear model from a dynamic archive of historically successful solutions. This approach enables the model to interpret the complex context of successful moves, thereby preventing the negative knowledge transfer where a beneficial pattern in one region of the search space could be detrimental in another. This learned intelligence is then operationalized through a generative operator that transfers validated positive knowledge to create high-quality candidate solutions. The process transforms the ABC from a memoryless explorer into an intelligent agent that learns to navigate the fitness landscape with high efficacy. The superiority of the LA-ABC framework is demonstrated through comprehensive benchmarking on 23 standard test functions, the competitive IEEE CEC 2019 suite, and a real-world photovoltaic parameter extraction problem. Our proposed neural knowledge transfer approach significantly outperforms 12 state-of-the-art algorithms, including ABC, L-SHADE, JSO, L-DE, L-PSO, KL-variants, and RL variants with the significance of these improvements validated by rigorous statistical tests (Wilcoxon, Bonferroni-Dunn, Friedman, and ANOVA). Ultimately, LA-ABC provides a robust new paradigm for integrating reinforcement learning and knowledge transfer within evolutionary computation.

Keywords: Evolutionary Computation, Swarm Intelligence, Artificial Bee Colony Algorithm, Transfer Learning, Artificial Neural Network (ANN), Learning-Aided Evolution, Evolutionary Transfer Optimization,

1. Introduction

Evolutionary Computation (EC) is a broad paradigm of bio-inspired optimization methodologies that has gained substantial recognition for addressing complex, nonlinear, and large-scale real-world problems where classical optimization methods are often intractable [1, 2]. Within this paradigm, *Swarm Intelligence* (SI) algorithms, inspired by the collective behavior of decentralized and self-organized systems, have emerged as highly effective search techniques [3].

EC encompasses both *Evolutionary Algorithms* (EAs), such as Genetic Algorithms (GA) [4] and Differential Evolution (DE) [5], and SI-based approaches, including Particle Swarm Optimization (PSO) [6] and Artificial Bee Colony (ABC) [7]. These algorithms are characterized by their robustness, adaptability, and capability to handle high-dimensional and multimodal optimization landscapes without the need for gradient information. As a result, EC methods have found extensive applications in diverse domains, achieving notable success in biomedical optimization [8], aerospace system design and control [9], logistics and supply chain optimization [10], and general engineering problems [11].

Among these methods, the ABC algorithm [12] stands out for its structural simplicity and strong exploration ability. However, similar to many metaheuristic algorithms, canonical ABC relies primarily on stochastic operators. The standard neighbor search equation is inherently memoryless and stochastic; it treats every direction as equally probable, effectively discarding the historical data of successful moves. This design results in two critical structural deficiencies: (1) 'Stochastic Blindness,' leading to slow convergence due to aimless exploration, and (2) a high susceptibility to stagnation in local optima due to a lack of guided exploitation. Consequently, a significant "knowledge gap" exists between the algorithm's accumulated experience and its future search trajectory.

The effectiveness of EC algorithms is largely determined by their search operators, such as mutation and crossover in DE, or velocity and position updates in PSO. While these operators are effective in many scenarios, their efficiency is highly problem-dependent, and their stochastic nature often results in inefficient traversal of the search space. To overcome these limitations, researchers have explored the integration of external mechanisms to make EC more knowledge-driven. One promising direction is surrogate-assisted EC (SAEC) [13], where expensive fitness evaluations are approximated using surrogate models such as Gaussian processes or radial basis function networks [14].

Beyond surrogate models, another emerging line of research emphasizes the role of knowledge discovery and reuse in EC. Using data mining techniques, valuable "evolutionary knowledge" can be extracted from past search experiences to significantly improve the efficiency and effectiveness of optimization. This idea underpins the recent paradigm of *Learning-Aided Evolution for Optimization* (LEO), which systematically captures, models, and reuses knowledge from Successful Evolutionary Patterns (SEPs) to guide the search [15]. More generally, the Knowledge Learning Evolutionary Computation (KLEC) framework builds on this principle by leveraging large-scale evolutionary data to develop adaptive search strategies. The KLEC paradigm has been successfully incorporated into several advanced DE variants, including JADE [16], Adaptive Distributed DE (ADDE) [17], jSO [18], and HyDE-DF [19], as well as numerous PSO extensions such as SaDPSO [20], HPSO-TVAC [21], TAPSO [22], and AWPSO [23, 24].

In parallel, evolutionary transfer optimization has also been explored as a means of leveraging knowledge across related problems [25, 26, 27]. While effective, these approaches often treat knowledge as a temporary proxy rather than as a mechanism to fundamentally reshape and guide the search process. At this juncture, the unparalleled learning capacity of ANNs [28] offers a compelling opportunity. ANNs have demonstrated exceptional performance in pattern recognition and predictive modeling, making them strong candidates for capturing and reusing knowledge within EC. Although existing integrations such as neuro evolution [29], hyper-heuristics [30], and surrogate-assisted neuro evolution [31, 32, 33, 34, 35] have explored aspects of this synergy, a continuous, online symbiosis where the evolutionary process is refined by ANN-driven knowledge learning remains underdeveloped [15].

Motivated by this research frontier, we propose the LA-ABC framework, which augments the canonical ABC with a dedicated neural knowledge learning engine. The primary design philosophy of this study is 'Optimization via Generative Learning.' Unlike traditional hybrids that use surrogates merely for approximation, we introduce a mechanism that transforms the optimization process from stochastic guessing to intelligent prediction. The objective of LA-ABC is to address three critical limitations of traditional ABC: (1) its inability to retain and exploit historical knowledge, (2) susceptibility to negative knowledge transfer, and (3) relatively slow convergence in complex multimodal optimization landscapes. In LA-ABC, successful evolutionary patterns are continuously collected into a knowledge archive and modeled using ANNs, which learn predictive mappings between candidate solutions and their improved variants.

To operationalize this knowledge, we introduce a novel **Learning-aided Candidate Generation (LCG)** operator that reintegrates learned evolution knowledge back into the ABC search cycle. This transforms ABC from a memoryless, purely stochastic explorer into an intelligent, adaptive optimizer that incrementally learns from its own experiences. In doing so, LA-ABC represents a new class of ANN-assisted EC frameworks that establish knowledge-driven search strategies, thereby enhancing both convergence speed and solution quality [36, 37].

Furthermore, it is essential to distinguish the specific mode of transfer learning addressed in this work to clarify the theoretical contribution. While traditional Transfer Optimization often refers to "Spatial Transfer" across different tasks (Multitask Optimization), this study focuses on "Temporal Transfer Learning" within a single optimization process. In this context, the "Source Domain" is defined as the algorithm's accumulated search history (stored in the Archive), and the "Target Domain" is the active population at the current generation. The proposed LA-ABC addresses this concept by establishing a neural feedback loop that transfers validated search patterns from the past to optimize future convergence. This justifies the comparison with knowledge-driven algorithms like KLDE and KLPSO, which similarly leverage historical data to enhance single-task performance.

Figure 1 illustrates the progression of search strategies in evolutionary computation. Subfigure (a) shows Random Search, where agents move aimlessly without feedback, yielding poor efficiency. Subfigure (b) depicts canonical evolutionary algorithms, where operators such as selection, crossover, and mutation guide trajectories, yet risks of premature convergence remain. Subfigure (c) presents the proposed learning-aided framework, where traditional operators are augmented by a neural learning model that captures and transfers successful evolutionary patterns. This knowledge-driven guidance enables smarter trajectories, balanced exploration–exploitation, and faster convergence toward the global optimum, reflecting the growing intelligence of modern optimization frameworks.

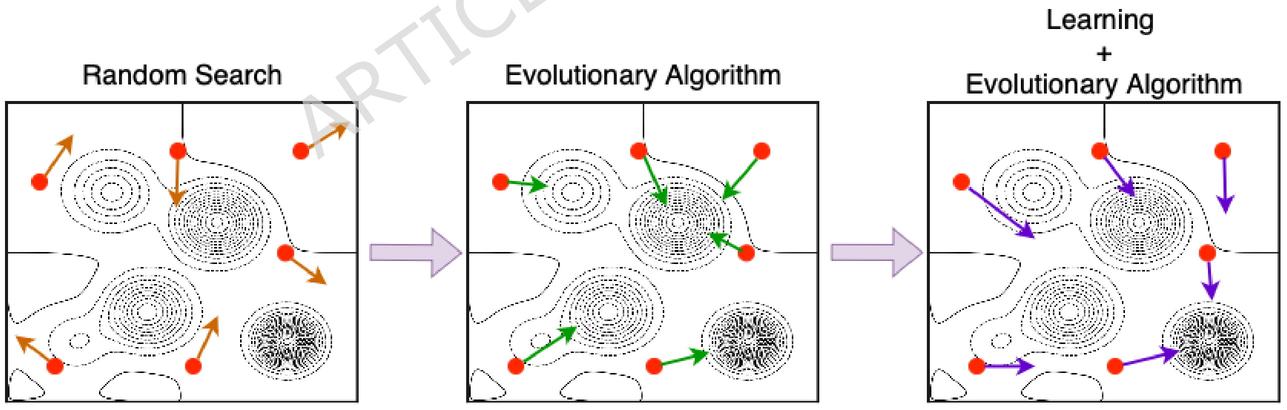


Figure 1: Progression of optimization strategies: (a) Random Search, through the population-based mechanisms of (b) Evolutionary Search, to the advanced paradigm of (c) Learning-Aided Evolutionary Search, where the search process is intelligently informed by neural knowledge models.

Contributions

The primary contributions of this study are summarized as follows:

- We propose the novel LA-ABC framework, a hybrid paradigm that tightly integrates an online ANN-based knowledge learning system within the canonical ABC algorithm.

- We design a **Knowledge Learning and Transfer Engine**, consisting of a SEPs archive for real-time knowledge acquisition and an ANN-based predictive model to guide future search.
- We formulate a new LCG operator that operationalizes the learned knowledge, reinjecting it into the search process to enhance convergence and solution quality.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 details the architecture and mechanics of the proposed LA-ABC framework. Experimental evaluations and comparative analyses on the IEEE CEC 2019 benchmark suite are presented in Section 4. Section 5 demonstrates the practical applicability of LA-ABC on real-world optimization tasks, specifically photovoltaic (PV) parameter extraction. Section 6 discusses the observed limitations and outlines potential directions for future enhancements. Finally, Section 7 concludes the study and summarizes key contributions.

2. Related Study

A significant research frontier in EC involves moving beyond traditional stochastic operators toward more intelligent, data-driven search strategies [15]. The existing body of research can be broadly classified based on the type of knowledge the learning component is designed to acquire: **problem knowledge** (e.g., approximating the fitness landscape) and **solution knowledge** (e.g., reusing or transferring good solutions). This section reviews these dominant paradigms to situate our proposed framework, which distinguishes itself by focusing on learning the **evolutionary knowledge** that is, the underlying process of "how to evolve well" itself.

2.1. The Artificial Bee Colony

The ABC algorithm, introduced by Karaboga [12], is a prominent swarm intelligence algorithm that models the foraging behavior of honeybees. The algorithm's procedure is divided into three main phases, driven by a population structured into three roles: employed bees, onlooker bees, and scout bees.

Employed Bee Phase

The operational cycle of ABC begins with the Employed Bee Phase. Each employed bee is assigned to a specific food source (a solution in the search space). It explores the vicinity of its source by generating a single candidate solution using the following equation:

$$v_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j}) \tag{1}$$

where $\mathbf{x}_i$ is the bee's current food source, $\mathbf{x}_k$ is a randomly chosen source from the population ($k \neq i$), j is a random dimension index, and $\phi_{i,j}$ is a random number in $[-1, 1]$. A greedy selection is then performed between $\mathbf{x}_i$ and the new candidate $\mathbf{v}_i$.

Onlooker Bee Phase

In the Onlooker Bee Phase, onlooker bees are recruited to exploit promising food sources. They observe the "waggle dance" of the employed bees and select a source via a probabilistic selection mechanism. Once a source is selected, the onlooker applies the same search strategy Eq. (1).

Scout Bee Phase

Finally, the **Scout Bee Phase** introduces the mechanism for abandoning depleted food sources. If a source's position cannot be improved for a predefined number of cycles, it is abandoned, and the associated bee becomes a scout, initiating a random search for a new source.

The primary strengths of ABC lie in its simplicity and strong global exploration capabilities [38]. However, a significant body of research has pointed out its tendency for slow convergence and weak exploitation. Our work diverges from traditional approaches that modify the search equation with another heuristic instead, we enhance it with an intelligent operator derived from online learning.

2.2. Literature Survey

This section provides a comprehensive review of prior research on the ABC algorithm and knowledge learning transfer frameworks, emphasizing their evolution, key contributions, and impact on advancing evolutionary optimization.

2.2.1. Literature of ABC

Since its introduction by Karaboga [7], the ABC algorithm has inspired an extensive body of research focused on mitigating its original shortcomings and improving its adaptability across diverse problem domains. One prominent direction has been hybridization with deterministic optimizers and adaptive strategies. For example, ABC-LM [39] embeds the Levenberg–Marquardt optimizer into ABC, significantly accelerating neural network training and reducing the risk of local stagnation. Similarly, the modified ABC proposed in [40] employs selective probabilities, chaotic perturbations, and opposition-based learning to better balance exploration and exploitation. The ABC Multi-strategy Ensemble (MEABC) [41] addresses the same challenge by orchestrating an ensemble of diverse competing search strategies. In [42], adaptability is reinforced through dynamic switching among multiple update rules, while MuABC [43] integrates three search strategies with Gaussian-based candidate generation to intensify local exploitation. Convergence speed is further improved in the Gaussian Bare-Bones ABC [44], which samples solutions around the global best and leverages opposition-based scouting. A notable hybrid, HABCDE [45], incorporates DE operators into the onlooker phase and redesigns the scout mechanism, thereby enhancing both exploitation and global search efficiency.

Beyond hybridization, several ABC variants have been tailored for improved exploration–exploitation balance and domain-specific applications. The Gbest-guided ABC (GABC) algorithm [46, 47] has demonstrated robust performance in solving load flow problems for power systems, providing a competitive alternative to the Newton–Raphson method and the basic ABC. NS-ABC [48] enhances local refinement by embedding a neighborhood search operator alongside an efficient ABC/best/1 strategy. Similarly, ABCADE [49] integrates DE operators with adaptive parameter control to improve robustness across complex search landscapes. The M-CABC variant [50] uses covariance-based modeling to address portfolio optimization problems, while ABCGLN [51] combines DE concepts with index-graph neighborhood structures to preserve search diversity. Other adaptive variants include SPABC [52], which employs self-adaptive parameter strategies, and ABCG [53], which introduces a gravity-inspired neighbor selection mechanism for improved guidance.

Additional extensions have further diversified ABC applications across emerging problem domains. CABC [54] accelerates convergence through covariance matrix learning, while MEL-ABC [55] introduces brain-inspired memory mechanisms for efficient helicopter path planning. In energy forecasting, M-ABC [56] improves accuracy for prediction of energy demand. DMABC-elite [57] enhances ANN training by integrating dimensional memory models with elite-based learning, whereas R-ABC [58] incorporates reinforcement learning to further refine the optimization of the

ANN. For binary optimization, oBABC [59] provides an efficient framework that ensures robust binary search behavior.

Hybrid approaches have also been proven to be highly effective, underscoring the synergy between ABC and other metaheuristics. For example, ABC-DE [60] integrates DE's mutation and crossover mechanisms into ABC's exploitation phase, striking a stronger balance between diversification and intensification. A comprehensive survey of hybrid ABCs [61] highlights their competitive advantage in addressing real-world optimization challenges.

A consolidated year-by-year summary of these notable ABC variants is provided in Table 1, highlighting the progressive evolution of the algorithm from its canonical form to advanced hybrid and domain-specific variants.

Table 1: Summary of ABC Variants and Knowledge Learning Frameworks.

SN	Algorithm / Framework	Year	ML Adaptation	Knowledge Adaptation	Tested on Real-life Application	Ref.
1	ABC with Levenberg–Marquardt (ABC-LM)	2011	×	×	Neural network training	[39]
2	Modified ABC (MABC)	2012	×	×	–	[40]
3	Gbest-guided ABC (GABC)	2012	×	×	Load flow in power systems	[46, 47]
4	ABC-DE (Hybrid with DE)	2013	×	×	Antenna arrays, amplitude modulation	[60]
5	MEABC (Multi-strategy Ensemble ABC)	2014	×	×	–	[41]
6	Adaptive ABC	2015	×	×	–	[42]
7	MuABC (Multi-strategy ABC)	2015	×	×	–	[43]
8	Gaussian Bare-bones ABC	2016	×	×	–	[44]
9	HABCDE (Hybrid with DE)	2017	×	×	–	[45]
10	NS-ABC (Neighborhood Search ABC)	2017	×	×	–	[48]
11	ABCADE (Enhanced ABC-DE)	2017	×	×	–	[49]
12	M-CABC (Multi-objective Co-variance ABC)	2017	×	×	Portfolio optimization	[50]
13	ABCGLN	2018	×	×	–	[51]
14	SPABC (Self-adaptive ABC)	2018	×	×	–	[52]
15	ABCG (Gravity-inspired ABC)	2018	×	×	–	[53]
16	CABC (Co-variance matrix ABC)	2018	×	×	–	[54]
17	MEL-ABC (Memory-Enhanced ABC)	2022	×	×	Helicopter path planning	[55]
18	M-ABC	2022	×	×	Energy demand forecasting	[56]
19	DMABC-Elite	2023	×	✓	–	[57]
20	R-ABC (Reinforcement-guided ABC)	2024	✓	✓	–	[58]
21	oBABC (Optimized Binary ABC)	2024	×	×	Binary search problems	[59]
22	Multi-factorial GA	2015	×	✓	Multitask optimization	[62]
23	Surrogate-assisted MTEA	2021	✓	✓	Scheduling tasks	[63]
24	Meta-Knowledge Transfer DE	2021	×	✓	Heterogeneous multitask problems	[64]
25	Task-Selective ABC (TSABC)	2019	×	✓	Car structural design	[65]
26	MTPSO (Multitask PSO)	2025	×	✓	–	[66]
*	**Proposed LA-ABC**	2025	✓	✓	PV extraction model	**This study**

2.2.2. Literature of Knowledge Learning Transfer Framework

Knowledge-learning-transfer frameworks have witnessed rapid growth in recent years, with a notable surge in contributions over the past five years. Early studies in this direction include Genetic algorithms, which implicitly addressed multitask optimization through the sharing of genetic material between tasks [62]. Zhang et al. [63] further advanced this idea with a surrogate-assisted evolutionary multitask algorithm using Gaussian processes to enable efficient knowledge transfer between tasks. Similarly, Li et al. [64] proposed a DE framework based on meta-knowledge transfer, which transfers generalizable "knowledge of knowledge" between heterogeneous tasks using a unified search strategy and an elite transfer mechanism.

Swarm-based approaches have also incorporated knowledge learning. Bian et al. [66] introduced a multitask PSO framework (MTPSO) with variable divvying, local meta-knowledge transfer, and adaptive mechanisms, improving convergence while reducing negative transfer. Yokoya et al. [65] developed Task Selective ABC (TSABC), enhanced with Firefly Algorithm-inspired operators, showing strong performance on real-world design problems.

3. The Proposed LA-ABC Framework

The central argument of this study is that an optimization algorithm can be considered truly intelligent only when it complements its search capability with the ability to continuously learn and adapt from its own evolutionary experience. Traditional metaheuristics, including the ABC, operate on predefined stochastic operators that remain static throughout the optimization run. While effective, these operators are knowledge blind they lack the mechanism to learn from the history of successful moves, leading to potentially inefficient exploration. To overcome this fundamental limitation, we propose the LA-ABC, a framework that introduces a new paradigm for the relationship between evolution and learning.

Our framework is conceptualized as a symbiotic system comprising two interconnected parts: an **evolution engine** and a **learning engine**. The novelty of LA-ABC lies not merely in their combination, but in their coevolutionary interaction. The evolution engine, driven by the foundational ABC operators, explores the search space and generates the raw data of the optimization process. Currently, the learning engine, powered by an ANN, acquires and models the SEPs from this data stream. The critical innovation and our primary departure from prior art is how this learned knowledge is utilized. Unlike surrogate models that passively approximate fitness, the LA-ABC learning engine actively ope-rationalizes its knowledge to create a new LCG. This operator does not just assist the search; it provides a more direct and efficient pathway to promising solutions by replacing stochastic guessing with intelligent prediction. This section provides a rigorous exposition of this symbiotic architecture, detailing the dual path design, the mechanics of the knowledge-learning and transfer engine, the formulation of the novel generative operator, and the complete algorithmic procedure.

3.1. Architectural Design

The LA-ABC framework is engineered with a dual-path adaptive metaheuristic architecture. It dynamically alternates between two complementary operational modes to achieve a superior balance between the explorative diversification and exploitative intensification of the search process.

- **Pathway A: Foundational Swarm Exploration.** This pathway executes the traditional operators of the ABC algorithm. It serves as the primary engine for global exploration and diversity preservation, representing the foundational search from which experiential knowledge is derived.

- **Pathway B: Knowledge-Guided Exploitation.** This pathway activates the framework's core intelligence: a novel LCG operator. This is the knowledge transfer pathway, where the intelligence captured by the predictive model is directly injected into the search process.

The arbitration between these pathways is governed by a **learning probability**, $l_p \in [0, 1]$, a crucial hyperparameter that dictates the balance of the search. At the commencement of each generation, a uniformly distributed random number $r \in [0, 1]$ is generated. If $r < l_p$, Pathway B is selected; otherwise, Pathway A is executed. The optimal setting for l_p is determined via empirical parameter sensitivity analysis, as is standard for metaheuristic design.

3.2. The Knowledge Learning and Transfer Engine

The defining characteristic of LA-ABC is its capacity for online learning and the subsequent transfer of this knowledge. This is facilitated by an integrated mechanism comprising a knowledge repository, a predictive neural network model, and a transfer operator. The entire process, from acquiring raw data to applying the learned knowledge, is detailed in Algorithm 1.

3.2.1. The Knowledge Repository: SEP Archive

The raw material for learning is a history of successful moves. We formalize this concept as a SEP, defined as an ordered pair $(\mathbf{x}_{\text{parent}}, \mathbf{x}_{\text{offspring}})$ where the offspring solution demonstrates a superior fitness value. These SEPs are captured in real-time and stored in a fixed-size, first-in-first-out **archive**, denoted as $\mathcal{A}$. To ensure the learning process is agnostic to the specific problem's scale and bounds, all solutions within an SEP are normalized to the hypercube $[0,1]^D$ via the transformation:

$$\mathbf{x}' = \frac{\mathbf{x} - \mathbf{lb}}{\mathbf{ub} - \mathbf{lb}} \tag{2}$$

3.2.2. The Predictive Engine: ANN for Knowledge Modeling

An ANN serves as the predictive engine, tasked with learning the intricate, non-linear mapping $\mathcal{L} : [0,1]^D \to [0,1]^D$ from a parent solution to a predicted high-quality offspring.

Network Architecture

The proposed framework utilizes a fully connected Multi-Layer Perceptron (MLP) architecture designed to map the evolutionary trajectory from a parent solution to a superior offspring. As illustrated in Figure 3, the network topology consists of four layers: an **input layer** comprising D neurons (where D represents the problem dimension) with linear activation, followed by **two hidden layers**, each containing 16 neurons activated by the **Sigmoid** function to capture non-linear dependencies. The structure concludes with a **linear output layer** of D neurons, enabling the network to generate a complete high-quality candidate solution vector (w_j) based on the input features (x_i).

Forward Propagation

For a given normalized input vector $\mathbf{x}'$, the network prediction, $\hat{\mathbf{y}}'$, is computed as follows:

$$\mathbf{H} = \sigma(\mathbf{x}'\mathbf{W}_1 + \mathbf{b}_1) \tag{3}$$

$$\hat{\mathbf{y}}' = \mathcal{L}(\mathbf{x}') = \sigma(\mathbf{H}\mathbf{W}_2 + \mathbf{b}_2) \tag{4}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the weight matrices, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the bias vectors, and $\sigma(\cdot)$ is the sigmoid activation function. Eq. (5) defines the sigmoid activation function, which is graphically depicted in Fig. 2.

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

In-situ Model Adaptation

The ANN undergoes in-situ model adaptation at the end of each generation using all SEPs in the archive $\mathcal{A}$. The objective is to minimize the Mean Squared Error (MSE) loss function, $\mathcal{J}$:

$$\mathcal{J}(\mathbf{W}, \mathbf{b}) = \frac{1}{|\mathcal{A}|} \sum_{k=1}^{|\mathcal{A}|} \|\mathcal{L}(\mathbf{x}'_{\text{parent},k}) - \mathbf{x}'_{\text{offspring},k}\|^2 \tag{6}$$

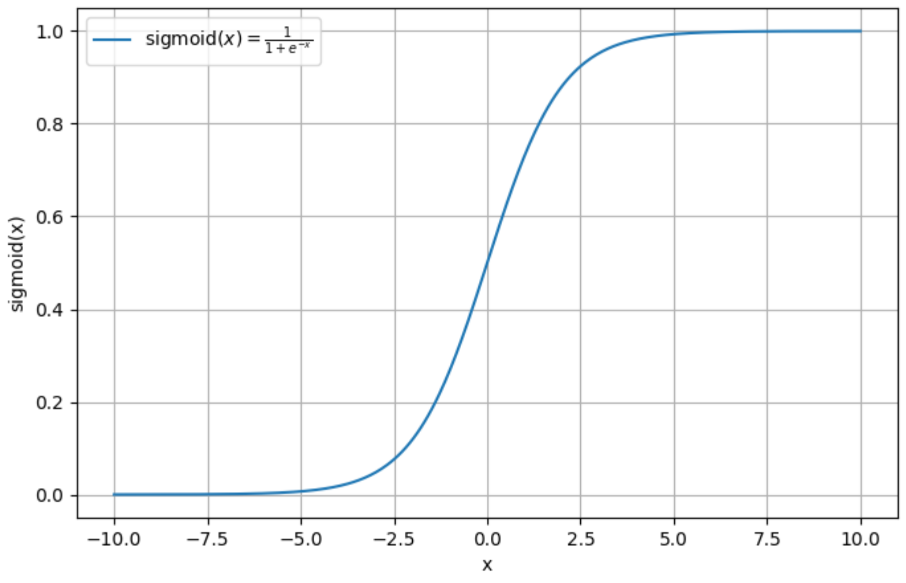


Figure 2: Sigmoid activation function used in the architecture of the ANN-based KLM

The network parameters are updated for one epoch ep using backpropagation with the general update rule for a weight w:

$$w \leftarrow w - \eta \frac{\partial \mathcal{J}}{\partial w} \tag{7}$$

where η is the learning rate. This continuous adaptation allows the ANN to progressively refine its internal model.

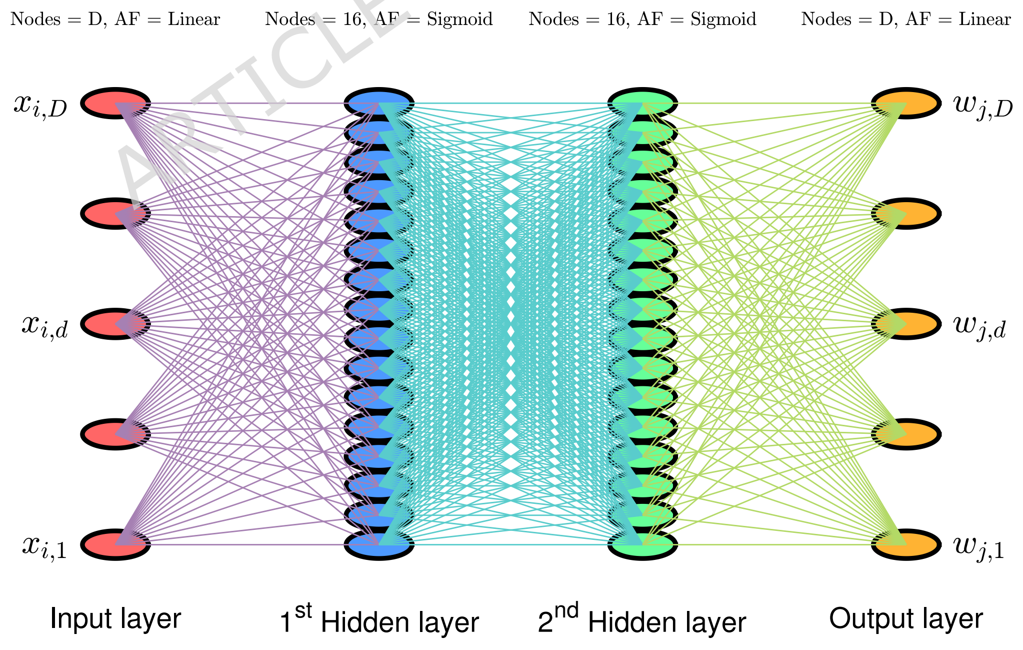


Figure 3: Architecture of the ANN, featuring two hidden layers with 16 nodes each using a *sigmoid* activation function, an output layer with D nodes using a linear activation function, and an input layer with D-dimensional input.

3.2.3. The LCG Operator: Operationalizing Knowledge Transfer

The LCG operator is the mechanism through which the abstract knowledge encoded in the ANN is transferred and operationalized. The specific steps of this cycle are encapsulated in Algorithm 1. The operator synthesizes the ANN's deterministic guidance with a stochastic differential perturbation term. The initial candidate vector $\tilde{\mathbf{v}}_i$ is formulated as:

$$\tilde{\mathbf{v}}_i = \text{denorm}(\mathcal{L}(\mathbf{x}'_i)) + \alpha(\mathbf{x}_j - \mathbf{x}_k) \tag{8}$$

Subsequently, a binomial crossover is performed between the parent $\mathbf{x}_i$ and the generated vector $\tilde{\mathbf{v}}_i$ to produce the final candidate solution $\mathbf{v}_i$:

$$v_{i,d} = \begin{cases} \tilde{v}_{i,d} & \text{if } \text{rand}_d(0,1) \leq \beta \text{ or } d = d_{\text{rand}} \\ x_{i,d} & \text{otherwise} \end{cases} \tag{9}$$

Algorithm 1 The Knowledge Learning and Transfer Engine Cycle

```
    Input: Current Population $\mathcal{P}$, SEP Archive $\mathcal{A}$, ANN parameters.
 1: $\mathcal{P}_{\text{new}} \leftarrow \emptyset$
 2: for each individual $\mathbf{x}_i \in \mathcal{P}$ do
 3:     Normalize parent: $\mathbf{x}'_i \leftarrow \text{norm}(\mathbf{x}_i)$.          ▷ using Eq. 2
 4:     Predict offspring: $\hat{\mathbf{y}}'_i \leftarrow \mathcal{L}(\mathbf{x}'_i)$.      ▷ using ANN Eq. 4
 5:     Generate candidate $\mathbf{v}_i$ via LCG operator                ▷ using Eqs. 8 & 9
 6:     $\mathcal{P}_{\text{new}} \leftarrow \mathcal{P}_{\text{new}} \cup \{\mathbf{v}_i\}$.
 7: end for
 8: Evaluate fitness of all $\mathbf{v}_i \in \mathcal{P}_{\text{new}}$.
 9: for $i = 1$ to $|\mathcal{P}|$ do
10:     if $f(\mathbf{v}_i) < f(\mathbf{x}_i)$ then
11:         Create SEP: $(\mathbf{x}'_i, \text{norm}(\mathbf{v}_i))$.
12:         Add SEP to Archive $\mathcal{A}$.
13:         if $|\mathcal{A}| >$ arch_size then
14:             Remove the oldest SEP from $\mathcal{A}$.
15:         end if
16:         $\mathbf{x}_i \leftarrow \mathbf{v}_i$.                  ▷ Update population with better solution
17:     end if
18: end for
19: if $|\mathcal{A}| > 0$ then
20:     Perform in-situ model adaptation on ANN.                     ▷ using Eq. 6
21: end if
22: return $\mathcal{P}$, $\mathcal{A}$, ANN_params
```

The complete LA-ABC procedure, which integrates the foundational swarm search with the knowledge engine, is detailed in Algorithm 2. For conceptual clarity, a flowchart of the framework is presented in Figure 4.

Mitigation of Overfitting and Stagnation:

To specifically address the challenges of early-stage model bias and overfitting, the framework implements a **Dynamic Regularization Mechanism**. The stochastic nature of the LCG operator injects necessary variance into the neural predictions, preventing the search from becoming purely deterministic even if the model fits to a local basin. Furthermore, the reliance on a fixed-size, First-In-First-Out (FIFO) archive creates a rolling horizon effect; this ensures that the neural model continuously unlearns early, suboptimal information and adapts to the evolving search trajectory. Together with the probabilistic control parameter (l_p), these mechanisms guarantee that global exploration is preserved, safeguarding the system against potential model inaccuracies in the initial generations.

Algorithm 2 The LA-ABC Framework

```
 1: Input: objective function $f(\mathbf{x})$, Dimension $D$, Pop Size $N$, $[\mathbf{lb}, \mathbf{ub}]$, Bounds Rate $\eta$, Probability $l_p$, $\alpha$, $\beta$.
 2: Initialize Population $\mathcal{P}$, ANN, and empty Archive $\mathcal{A}$.
 3: Evaluate $f(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathcal{P}$; set FE $\leftarrow N$.
 4: $\mathbf{x}_{\text{gbest}} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{P}} f(\mathbf{x})$.
 5: while FE < maxFE do
 6:     Generate a random number $r \in [0, 1]$.
 7:     if $r < l_p$ then                                   ▷ Activate knowledge-guided pathway: B
 8:         $(\mathcal{P}, \mathcal{A}, \text{ANN\_params}) \leftarrow$ Knowledge Cycle $(\mathcal{P}, \mathcal{A}, \text{ANN\_params})$.   ▷ Algorithm 1
 9:         FE $\leftarrow$ FE $+ N$.
10:     else                                                ▷ Activate exploration pathway: A
11:         Execute standard ABC.
12:         Update $\mathcal{P}$ and FE accordingly.
13:     end if
14:     Update $\mathbf{x}_{\text{gbest}}$ with the best solution in the current population $\mathcal{P}$.
15: end while
16: return $\mathbf{x}_{\text{gbest}}$
```

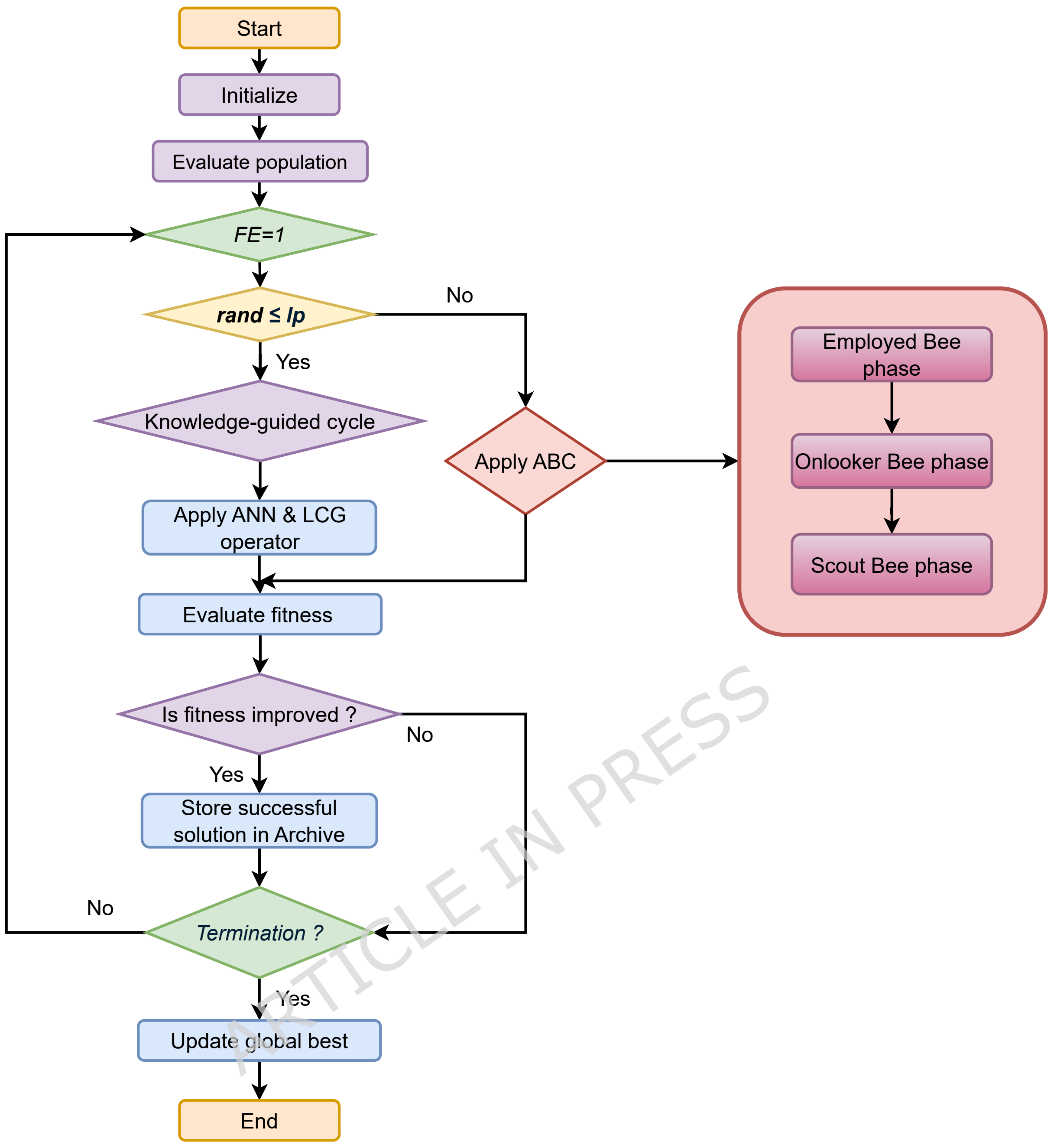


Figure 4: Framework of LA-ABC.

4. Experimental Results and Analysis

This section presents a comprehensive empirical validation of the proposed model. We begin by justifying the key architectural choices underlying the framework, followed by a rigorous comparative evaluation against state-of-the-art algorithms. Finally, a detailed analysis of the results is provided to demonstrate the effectiveness, robustness, and superiority of the proposed approach.

4.1. Experimental Settings

The experimental study was carried out using ***MATLAB R2023b*** on an HP 15 laptop equipped with an Intel Core i5 (12th Gen, 1235U) processor, 16 GB RAM, and a 512 GB SSD running Windows 11 Home. To evaluate the effectiveness of the proposed LA-ABC framework, we compared it against the canonical ABC algorithm and a set of recently developed knowledge learning-based algorithms, namely L-PSO [15], L-DE [15], KLPSO [24], KLDE [24], KLJADE, and KL-TAPSO. These algorithms represent the current frontier in knowledge-assisted evolutionary computation.

It is important to note that L-PSO, L-DE, KLPSO, and KLDE have consistently demonstrated significant performance gains over their standard counterparts, PSO and DE [15, 24]. For this reason, the standard PSO and DE algorithms were excluded from our comparisons, allowing the focus to remain on more competitive and knowledge-driven approaches.

To ensure fairness and reproducibility, all algorithmic parameters were retained as reported in their original publications. The control parameters were initialized within the standard effective operating zones of $F \in [0.4, 0.9]$ and $CR \in [0.5, 0.9]$, adhering to the empirical guidelines established in widespread Differential Evolution literature [67], to ensure a robust balance between exploration and exploitation. A detailed summary of these parameter settings is provided in Table 2, serving as the basis for subsequent performance evaluations.

4.2. Benchmark Problems

To provide a comprehensive and rigorous evaluation of the proposed LA-ABC framework, two complementary sets of benchmark functions are employed. **Set 1** consists of 23 widely used classical test functions in **Supplementary Table S1**, carefully selected to represent diverse optimization characteristics. **Set 2** is derived from the IEEE CEC 2019 benchmark suite **Supplementary Table S2**, which comprises 10 challenging test functions categorized into unimodal, multimodal, hybrid, and composition classes.

Together, these benchmark sets span a wide spectrum of optimization challenges, encompassing variations in separability, modality, landscape ruggedness, and global structural complexity. In particular, the IEEE CEC 2019 suite has become a de facto standard in the optimization community, renowned for its difficulty and practical relevance, with detailed specifications and reference implementations available in [68].

For consistency and fairness across experiments, the dimensionality of the decision space is set to $D = 50$ for the classical Set 1 functions and $D = 10$ for the CEC 2019 suite, following established conventions in the literature. Furthermore, the maximum number of iterations is uniformly fixed at 450 for both sets, ensuring a balanced and reproducible basis for comparative performance analysis.

4.3. Performance Analysis

To ensure statistical reliability and reproducibility, each algorithm is executed independently 30 times on every benchmark function. For each run, the best fitness value is recorded, and the overall performance is summarized using the mean and standard deviation (STD) across runs. These statistical measures provide a rigorous basis for fair comparison, capturing both the central tendency and the robustness of the algorithms under evaluation.

The parameter settings for all algorithms are presented in Table 2. For LA-ABC, a parameter sensitivity analysis was performed to determine the optimal setting for the key hyperparameter, particularly the learning probability l_p. Based on preliminary experiments, a value of $l_p = 0.5$ was found to provide a robust balance between the two pathways and was used for all reported results.

Table 2: Parameter values for algorithms.

Algorithm	**Parameter**	**Values**
LA-ABC	$\phi_{i,j}$	Uniform random in [-1,1]
	limit	$\frac{\text{colony size}}{2} \times D$
	Learning Probability (l_p)	0.5
	Q-Learning Rate (α)	0.01 – 0.3
	Discount Factor (β)	0.5 – 0.9
	η	0.01 – 0.3
	Archive Size ($\mathcal{A}$)	100
	Population size ($\mathcal{P}$)	100
ABC	$\phi_{i,j}$	Uniform random in [-1,1]
	limit	$\frac{\text{colony size}}{2} \times D$
L-DE	Scaling Factor (F)	0.4 – 0.9
	Crossover Rate (CR)	0.5 – 0.9
	Learning Probability (l_p)	0.5
	Archive Size ($\mathcal{A}$)	100
L-PSO	Inertia Weight (w)	0.9 – 0.4
	Cognitive Coefficient (c_1)	1.5 – 2.0
	Social Coefficient (c_2)	1.5 – 2.0
	Learning Probability (l_p)	0.5
	Archive Size ($\mathcal{A}$)	100
	LCG Scaling Factor (α)	0.5
	LCG Crossover Rate (β)	0.9
KLDE	Mutation Factor (F)	0.4 – 0.9
	Crossover Rate (CR)	0.5 – 0.9
	Learning Rate (lr)	0.2
KLPSO	Inertia Weight (w)	0.7 – 1.2
	Cognitive Coefficient (c_1)	1.5 – 2.0
	Social Coefficient (c_2)	1.5 – 2.0
	Learning Rate (lr)	0.2
KLJADE	p-best Selection Parameter (p)	0.1 – 0.5
	Adaptation Rate (c)	0.1 – 0.5
	Initial Mean Scaling Factor (mu_F)	0.5 – 0.9
	Initial Mean Crossover Rate (mu_{CR})	0.5 – 0.9
	Learning Rate (lr)	0.5
KL-TAPSO	Crossover probability (p_c)	0.1 – 0.9
	Mutation Probability (p_m)	0.1 – 0.5
	Initial inertia weight (w_{max})	0.5 – 0.9
	Final Inertia Weight (w_{min})	0.5 – 0.9
	Learning Rate (lr)	0.2
L-SHADE	Historical Memory (H)	6
	Archive Rate (r^{arc})	2.6
	Greediness (p)	0.11
JSO	Historical Memory (H)	5
	Max p-best (p_{max})	0.25
	Min p-best (p_{min})	0.125
ABC-RL	$\phi_{i,j}$	Uniform random in [-1,1]
	limit	$\frac{\text{colony size}}{2} \times D$
	Q-Learning Rate (α)	0.01
	Discount Factor (γ)	0.5
	ε	0.3
MRL-ABC	$\phi_{i,j}$	Uniform random in [-1,1]
	limit	$\frac{\text{colony size}}{2} \times D$
	Q-Learning Rate (α)	0.01
	Discount Factor (γ)	0.5
	ε	0.3
RLGA	Mutation Factor (P_c)	0.6
	Crossover Rate (P_m)	0.001
	Q-Learning Rate (α)	0.01
	Discount Factor (γ)	0.5
	ε	0.3

4.3.1. Performance Analysis on Set 1: State-of-the-Art Benchmark Functions

For set 1 in the 50-dimensional space, the performance of the proposed LA-ABC algorithm is summarized in Table 3. The results clearly show that LA-ABC achieves outstanding superiority in a majority of benchmark functions, particularly in F1 - F4, F6 - F12, F14, F16 - F19, and F21 - F23, where it demonstrates the best mean values or demonstrates highly stable performance with minimal standard deviation. In F2, LA-ABC performs on par with L-DE and L-PSO, indicating a comparable convergence towards the global optimum. The bold entries represent the best mean values across the compared algorithms.

Figures 5 and 6 further illustrate the convergence behavior of the algorithms for representative functions (F1, F2, F3, F4, F5, F6, F7, F10, F15, and F20). LA-ABC demonstrates faster and smoother convergence compared to competing methods, particularly by avoiding the premature stagnation often observed in ABC, KLPSO, and KLDE. Complementary box plots for these functions, shown in Figures 7 and 8, reinforce these findings by highlighting the tighter distribution of the LA-ABC results and its narrower maximum-minimum span compared to other algorithms.

4.3.2. Performance Analysis on Set 2: IEEE CEC 2019 Benchmark Functions

For the 10-dimensional IEEE CEC 2019 benchmark suite, Table 4 summarizes the comparative performance of LA-ABC against ABC, L-DE, L-PSO, KLDE, KLPSO, KLJADE, and KL-TAPSO. The results reveal that LA-ABC achieves clear superiority on 8 out of the 10 functions, including F1, F2, F3, F4, F5, F7, F8, and F10, where it consistently attains the lowest mean values along with competitive or lowest standard deviations. These results highlight the ability of the algorithm to deliver high accuracy and stable convergence across a diverse set of problem landscapes. However, in F6 and F9, other algorithms exhibit stronger performance, with KLPSO, KLJADE, and KL-TAPSO showing notable competitiveness. Importantly, despite these exceptions, LA-ABC consistently shows better or comparable results in the majority functions, maintaining a clear advantage over classical ABC and other learning-based baselines.

Figures 9 and 10 further illustrate the superior convergence behavior of LA-ABC, which achieves rapid and stable progress toward global optima. The box plots in Figures 11 and 12 reinforce these findings, showing that LA-ABC delivers a narrower spread of fitness values and consistently better median performance compared to its competitors, which highlights its robustness and reliability in the CEC 2019 benchmark suite.

4.4. Statistical Analysis

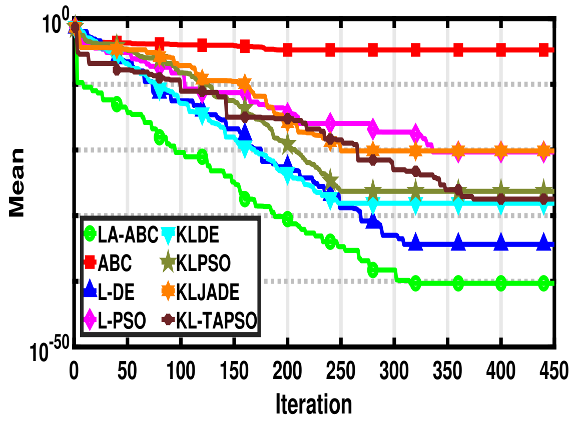
To rigorously validate the performance of the proposed LA-ABC algorithm, both non-parametric and parametric statistical tests were employed on the two benchmark sets. Specifically, the **Wilcoxon signed-rank**, **Friedman**, and **Bonferroni-Dunn** tests were conducted, with results summarized in Tables 5 and 6. Additionally, one-way **ANOVA** based on metrics such as p-score, z-score, and F-score further validates its superiority with high F-values and low p-values. These tests collectively confirm the statistical superiority of LA-ABC over its competitors across diverse optimization scenarios.

4.4.1. Statistical Analysis Using Wilcoxon Signed-Rank Test, Bonferroni-Dunn Test and Friedman Test

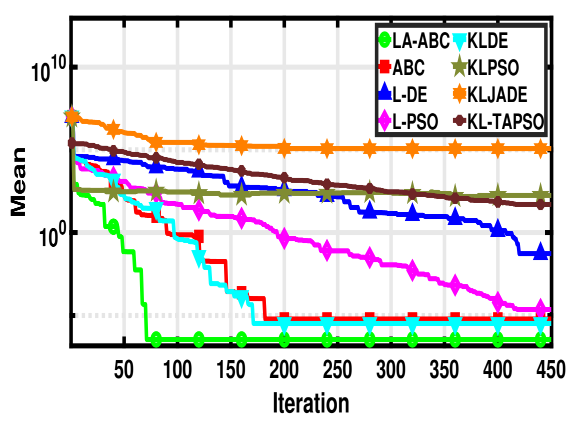
The statistical analysis across both benchmark suites confirms the decisive superiority of the LA-ABC framework. In the first benchmark suite (Set 1), LA-ABC establishes a commanding lead over all peer algorithms, as detailed in Table 5. The Friedman test confirms LA-ABC as the top-performing method, achieving the best **average rank of 1.782609**. This is significantly ahead of the nearest competitors, L-DE (rank 2, 2.913043) and L-PSO (rank 3, 3.521739). The

Table 3: Results of state-of-the-art benchmark functions of set 1, with 50 dimensions.

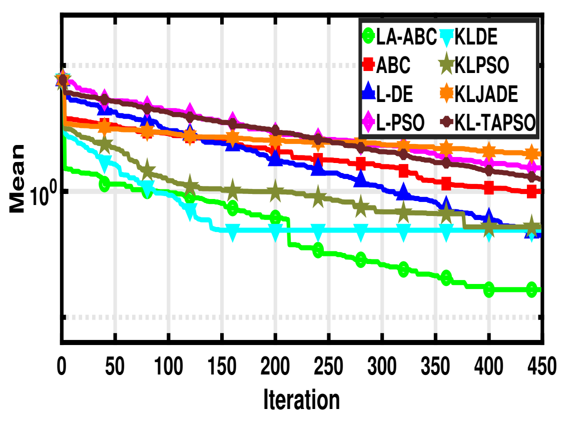
Func	Metric	LA-ABC	ABC	L-DE	L-PSO	KLDE	KLPSO	KLJADE	KL-TAPSO
F1	**Mean**	**2.798816E-60**	3.388490E-04	2.080933E-19	3.348808E-53	8.775354E-09	1.592167E-13	3.339266E-10	4.240137E-06
	STD	8.962026E-60	5.412292E-04	0.000000E+00	1.468310E-52	8.757746E-09	6.871092E-14	5.608268E-12	4.188375E-06
F2	**Mean**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**	**-1.031628E+00**
	STD	6.648565E-16	2.258510E-14	2.278130E-16	1.906020E-16	3.562220E-16	1.836687E-16	5.059159E-14	5.059159E-14
F3	**Mean**	**1.288196E-19**	2.092285E+02	2.759541E+00	1.371484E-06	2.587395E-14	2.373189E-13	3.791643E-01	1.596616E+03
	STD	9.404333E-20	4.261762E+02	4.534757E-23	1.787537E-06	4.498658E-13	4.395106E-13	7.733001E-02	4.270860E+02
F4	**Mean**	**2.065008E-15**	4.335555E-01	6.633802E-13	1.056375E-04	8.754875E-07	7.499992E+00	5.435879E-01	2.226764E+01
	STD	7.589746E-18	1.851113E-01	5.336089E-13	1.356947E-04	8.795420E-07	1.164156E+01	5.903062E-02	5.511588E+00
F5	**Mean**	1.139265E+04	3.175265E+04	2.714718E+00	2.651926E+03	9.077686E+03	4.000000E+03	**5.363476E-01**	4.740711E+03
	STD	2.062793E+03	5.130867E+03	1.122204E+00	9.462500E+02	1.318847E+03	8.941002E+03	4.397751E-01	1.761458E+03
F6	**Mean**	**1.041545E-03**	7.018965E+01	6.773011E-02	8.562989E-01	9.875346E-01	1.498728E+01	2.533782E+00	1.687359E+01
	STD	2.253095E-02	4.211851E+00	2.144421E-01	4.838094E-01	2.223344E-01	1.123641E+01	5.716874E-01	2.484741E+00
F7	**Mean**	**1.659342E-03**	8.861466E-01	5.197568E-01	4.119518E-02	8.753519E-01	1.500000E+01	6.223490E-02	2.513940E+02
	STD	4.559539E-02	3.576385E-01	1.151530E-03	1.456405E-02	3.456236E-01	6.708204E+01	1.453648E-02	2.166589E+02
F8	**Mean**	**1.371696E-33**	2.175718E-01	2.973711E-27	4.288054E-10	5.638700E-09	9.828969E-13	6.546245E-05	2.359589E-01
	STD	7.618093E-34	1.160743E-01	2.363664E-27	9.363516E-10	5.646740E-09	1.480222E-13	1.110350E-05	8.309329E-02
F9	**Mean**	**2.497809E-20**	4.762313E+01	1.490239E+01	2.521580E+01	2.353219E-14	2.094529E-15	2.738585E+01	4.170895E+01
	STD	1.311211E-21	1.639172E+01	1.242935E+00	5.801453E-01	4.547641E-14	5.773871E-15	3.322587E-01	5.265679E+00
F10	**Mean**	**2.840715E-28**	4.079087E+04	9.334149E-15	1.553698E+00	2.876445E-10	1.212982E+07	2.034802E+03	4.330925E+07
	STD	3.398758E-30	5.761699E+04	1.004032E-14	3.640415E+00	1.324458E-09	3.296253E+07	5.480590E+02	2.533203E+07
F11	**Mean**	**4.372787E-14**	2.163495E-01	1.841209E-12	1.001570E-07	2.832994E-01	5.000000E-02	7.896233E-05	1.948485E+00
	STD	1.624460E-14	1.998526E-01	1.404608E-12	2.243906E-07	3.284357E-01	2.236068E-01	2.667730E-05	7.289015E-01
F12	**Mean**	**2.799222E-02**	3.037371E+00	1.942444E-01	7.804099E-01	2.544865E-01	5.487473E-02	1.758477E-01	4.166248E+00
	STD	3.663445E-02	1.661454E+00	3.772556E-01	1.442149E-01	9.233859E-01	6.674115E-03	1.692250E-02	5.124847E-01
F13	**Mean**	9.496743E+00	8.387734E+01	**1.624485E+00**	1.480692E+01	4.567423E+00	1.348234E+02	1.539045E+02	1.774012E+02
	STD	7.388173E+00	1.499341E+01	3.054707E+00	4.370267E+00	7.923516E+00	3.397927E+01	1.182845E+01	2.009455E+01
F14	**Mean**	**1.063416E-12**	1.302641E+01	7.309552E-06	7.531218E-02	5.456126E-07	4.076801E+00	3.355925E-01	9.983343E+00
	STD	4.332296E-13	9.735505E-01	1.933648E-06	3.357521E-01	2.344640E-06	1.175192E+00	5.002051E-02	1.409206E+00
F15	**Mean**	2.716300E-05	3.961784E+00	5.291476E-05	3.275232E-03	**5.328627E-07**	1.279192E-02	6.257887E-01	1.956495E+01
	STD	4.986937E-05	4.503234E+00	8.325156E-05	7.521231E-03	3.563387E-07	1.332856E-02	8.708852E-02	6.052163E+00
F16	**Mean**	**1.822325E-04**	4.997379E-01	3.722408E-02	1.220918E-01	7.635536E-02	3.894821E-01	4.009684E-01	4.855003E-01
	STD	2.133812E-04	6.316405E-02	2.796212E-03	2.211325E-02	6.352240E-02	5.838569E-02	1.722883E-02	7.881885E-03
F17	**Mean**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**	**-7.833233E+01**
	STD	1.445379E-14	1.458003E-14	1.458003E-14	1.458003E-14	1.458003E-14	1.458003E-14	1.458003E-14	1.458003E-14
F18	**Mean**	**3.236353E-18**	5.789656E+00	1.241758E-04	3.410017E-04	9.875542E-02	2.672136E-10	1.749605E+00	1.246656E+01
	STD	3.569926E-18	1.557715E+00	4.368167E-04	1.114633E-03	7.643430E-02	1.163060E-09	1.168284E+00	2.150593E+00
F19	**Mean**	**1.989300E-12**	2.435994E-01	1.345481E-05	3.730319E-09	1.398248E+01	1.530000E+02	5.874285E-04	3.044227E+00
	STD	6.849661E-13	3.987357E-01	7.600962E-06	6.798769E-09	8.734660E-01	3.003857E+01	1.744656E-04	2.986855E+00
F20	**Mean**	2.713155E-02	1.277066E+00	4.451166E-02	**7.402915E-04**	2.873467E-01	1.780989E+00	4.087066E+01	8.694064E+03
	STD	1.059620E-01	1.649839E+00	5.503283E-02	1.501827E-03	1.178270E-01	2.574781E+00	1.365088E+01	1.024507E+04
F21	**Mean**	**5.046708E+02**	4.002181E+03	3.030041E+03	1.917724E+03	7.244000E+03	4.357330E+03	7.796093E+03	7.662968E+03
	STD	2.949716E+02	2.864638E+02	3.129914E+02	2.068363E+02	2.345436E+03	6.177798E+02	3.358392E+02	7.320544E+02
F22	**Mean**	**1.634700E-01**	9.095967E+01	1.026094E+00	1.469766E+01	7.865740E+00	2.904874E+02	1.508489E+02	1.376861E+02
	STD	2.049300E-01	1.031492E+01	2.126252E+00	4.385316E+00	5.356345E+00	5.377264E+01	1.248313E+01	9.714540E+00
F23	**Mean**	**5.095486E-10**	5.067395E-02	6.655454E-08	7.618893E-08	2.935600E+00	8.466467E+01	2.181369E-01	4.908159E+05
	STD	4.646891E-10	3.986639E-02	7.598950E-08	9.640253E-08	2.754876E-02	8.272246E+01	1.949072E-01	7.982377E+05
LA-ABC outperforms on: F1, F3, F4, F6, F7, F8, F9, F10, F11, F12, F14, F16, F18, F19, F21, F22, F23									
LA-ABC underperforms on: F5, F13, F15, F20									
All algorithms match on: F2, F17									



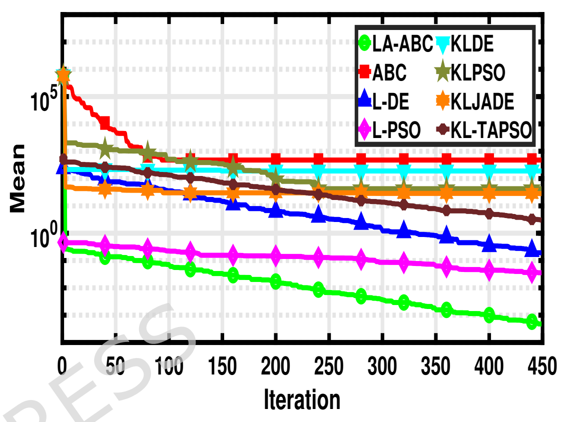
(a) F_1



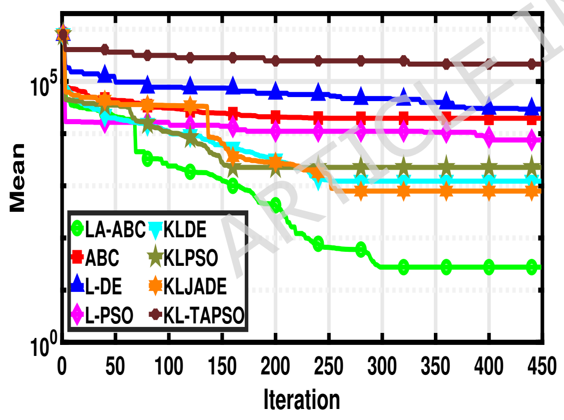
(b) F_2



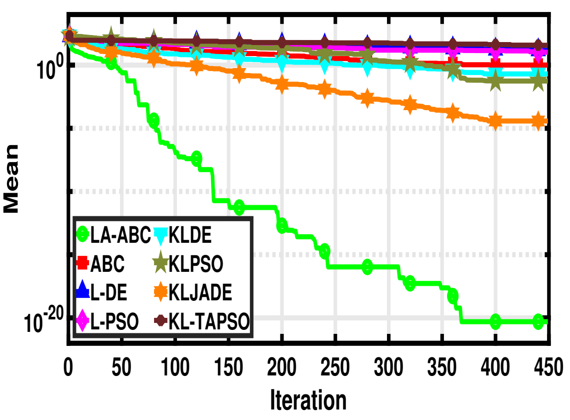
(c) F_3



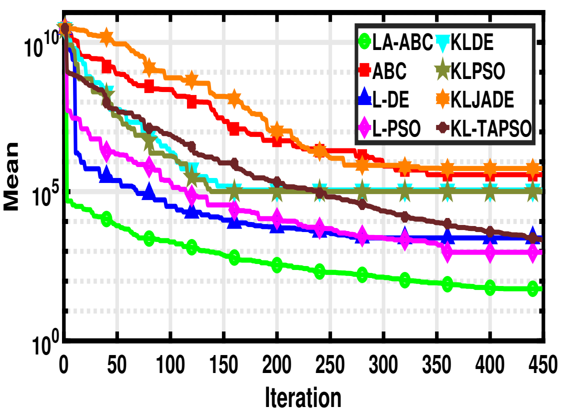
(d) F_4



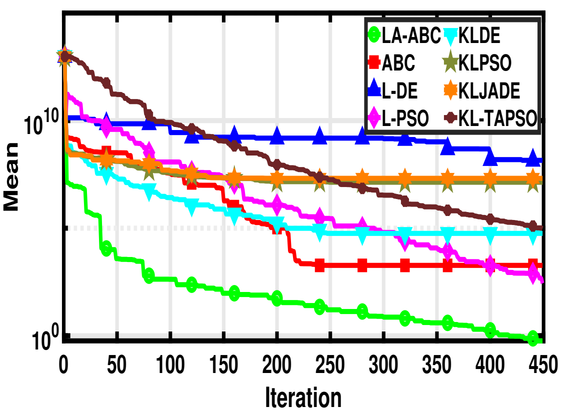
(e) F_5



(f) F_6

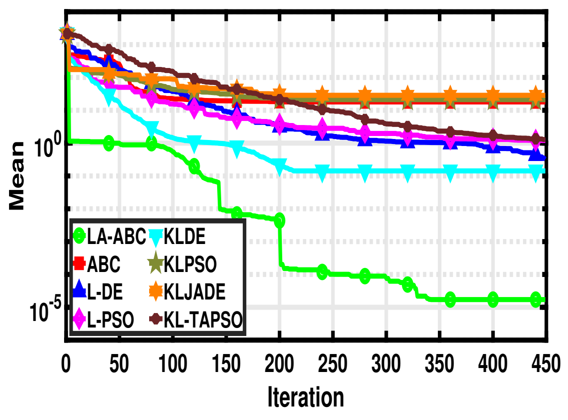


(g) F_7

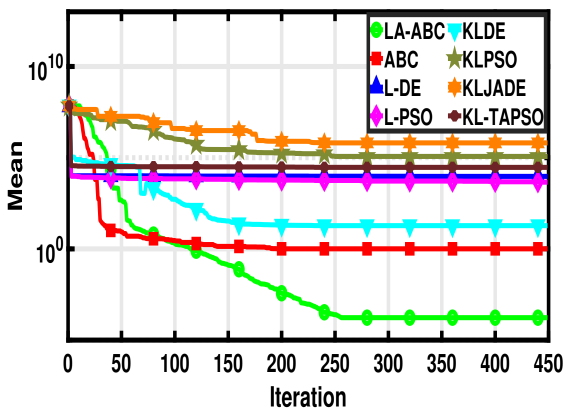


(h) F_{10}

Figure 5: Convergence graphs for F_1—F_7 and F_{10} on state-of-the-art benchmark functions of set 1.



(a) F_{15}

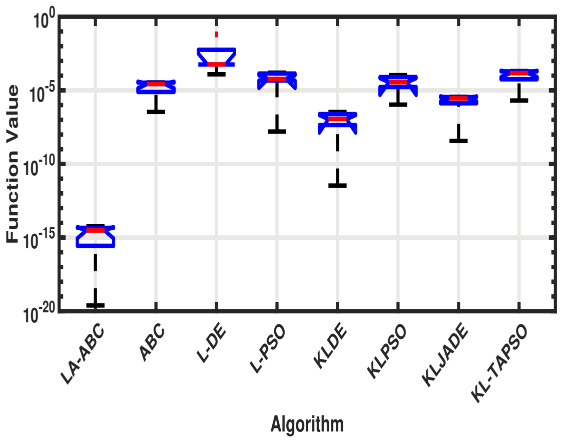


(b) F_{20}

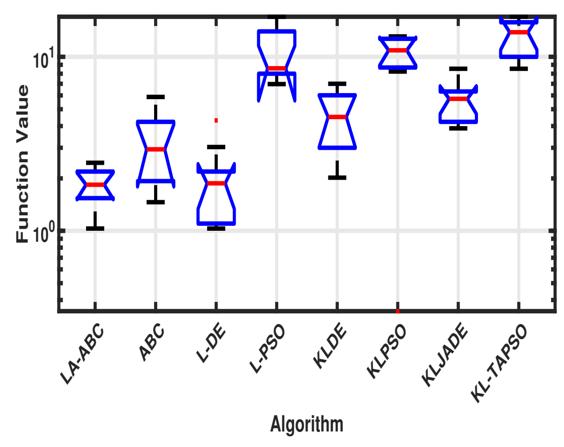
Figure 6: Convergence graphs for F_{15} and F_{20} on state-of-the-art benchmark functions of set 2.

Table 4: Results of benchmark functions of IEEE CEC 2019 with 10 dimensions.

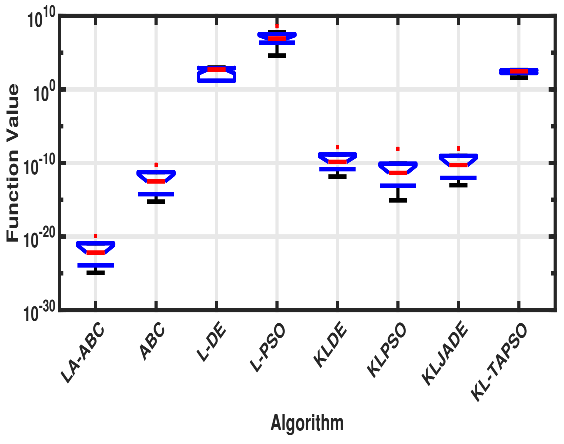
Func.	Metric	LA-ABC	ABC	L-DE	L-PSO	KLDE	KLPSO	KLJADE	KL-TAPSO
F1	Mean	**2.538301E-02**	6.238442E+02	1.206667E-01	1.596879E+01	1.346827E+02	7.586827E+02	4.310676E+01	1.206438E+03
	STD	3.425697E-02	4.400556E+02	2.857434E-01	4.063294E+01	4.881430E-01	8.712903E+02	1.012080E+01	1.482688E+03
F2	Mean	**1.078124E+00**	1.075168E+02	6.862992E+00	2.416900E+02	2.968909E+02	4.129139E+00	2.599055E+00	6.332764E+00
	STD	1.076765E+00	1.242636E-01	7.296316E+00	4.336531E+02	9.728000E-01	3.953579E-01	1.853858E-01	1.611479E+00
F3	Mean	**1.025240E+01**	2.360011E+02	4.480461E+01	2.805779E+01	3.877698E+02	1.554474E+01	1.529926E+01	1.306954E+01
	STD	3.826203E+00	4.341449E-03	1.507529E+01	1.098587E+01	7.830390E-01	4.089629E-09	1.441908E-01	9.866194E-01
F4	Mean	**4.059090E-02**	3.804898E+02	8.930982E-02	6.031802E-02	4.066711E+02	4.199066E+01	9.648012E+01	2.072666E+01
	STD	3.457065E-02	4.027762E+00	4.506116E-02	2.352311E-02	4.383961E-03	2.816799E+01	2.511202E+00	6.214540E+00
F5	Mean	**1.965927E+00**	4.203530E+02	2.000047E+01	1.999999E+01	5.636502E+02	3.445106E+00	3.928329E+01	2.175403E+00
	STD	1.163750E-08	7.341100E-03	2.339650E-03	6.512170E-05	4.922059E+00	3.281482E+00	1.262290E+00	1.057756E-01
F6	Mean	3.094440E+01	5.048705E+02	3.529787E+01	3.422035E+01	6.150757E+02	4.458809E+00	5.201340E+00	**2.777581E+00**
	STD	8.838445E+00	3.177033E-01	1.397820E+01	2.636571E+01	4.696610E-01	9.153335E-01	1.468935E+00	8.207070E-01
F7	Mean	**7.960797E+00**	1.484046E+02	1.919608E+02	7.919608E+01	7.878030E+02	1.534324E+03	1.907202E+03	8.733134E+02
	STD	4.709332E+00	1.094498E+02	2.287093E+02	4.270933E+01	6.601601E+00	2.175545E+02	1.955480E+02	3.260919E+02
F8	Mean	**9.586893E-02**	7.312271E+02	2.057909E+00	1.499760E+00	8.049536E+02	4.396297E+00	3.641617E+00	3.126012E+00
	STD	6.892852E-02	8.844665E-02	2.045963E+00	1.759783E+00	2.810028E+00	3.886986E-01	2.686866E-01	3.255200E-01
F9	Mean	3.969878E+00	8.907996E+02	**1.601155E+00**	7.400817E+00	9.050603E+02	8.143723E+00	5.174904E+00	9.375402E+00
	STD	3.834349E-01	4.577462E-02	3.268417E-01	9.719836E-01	3.598930E-01	2.267114E-01	7.585834E-02	6.224794E-02
F10	Mean	**6.018991E+00**	9.090945E+02	9.492576E+00	8.172502E+00	1.803352E+03	3.895604E+01	2.870978E+01	2.159867E+01
	STD	3.050204E+00	6.384449E-03	6.069374E+00	4.273328E+00	1.776052E+02	4.638640E-02	9.552794E-02	6.430672E-02
LA-ABC outperforms on: F1, F2, F3, F4, F5, F7, F8, F10									
LA-ABC underperforms on: F6, F9									



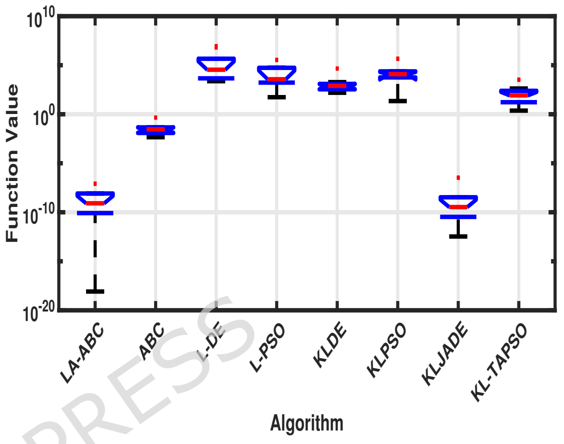
(a) F_1



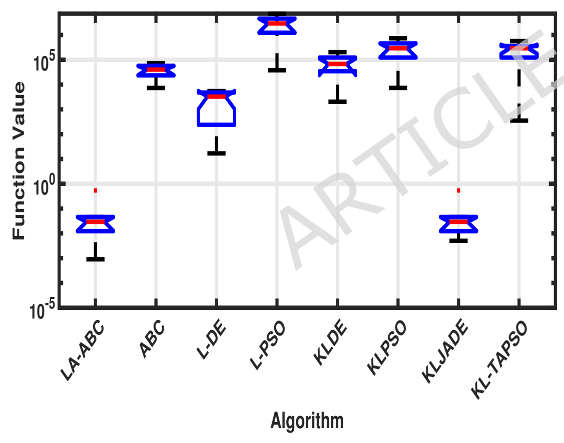
(b) F_2



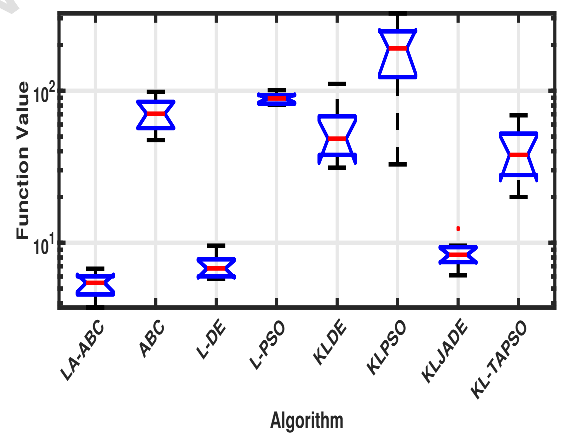
(c) F_3



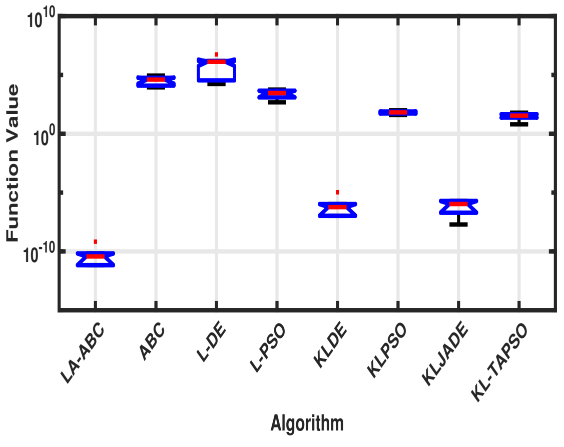
(d) F_4



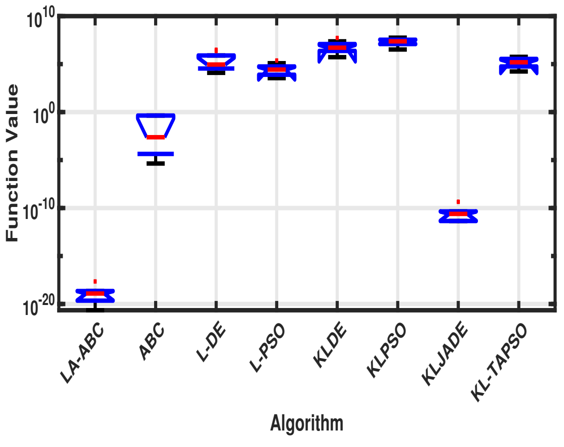
(e) F_5



(f) F_6

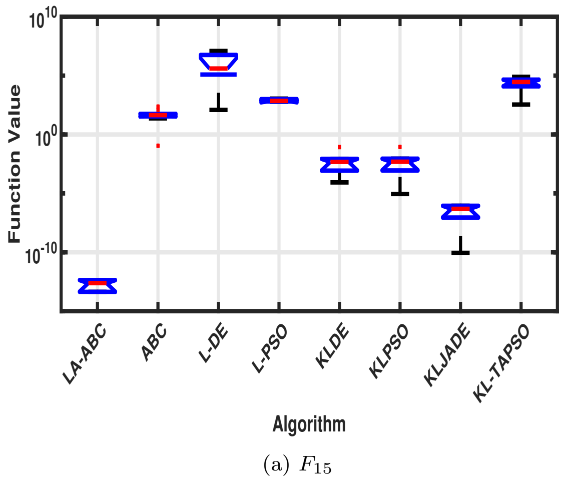


(g) F_7

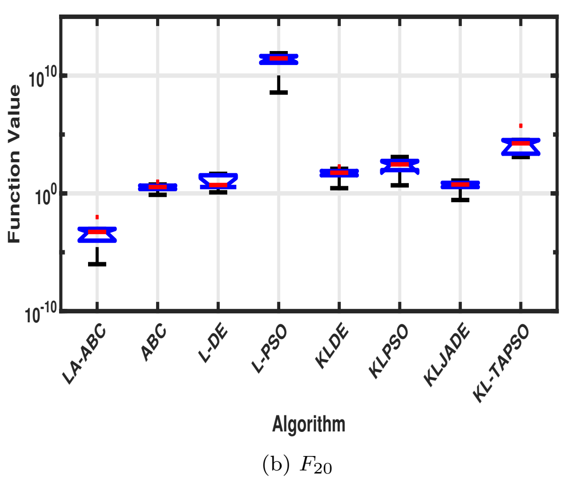


(h) F_{10}

Figure 7: Variations in global optimization of F_1—F_7 and F_{10} on state-of-the-art benchmark functions of set 1.



(a) F_{15}



(b) F_{20}

Figure 8: Variations in global optimization of F_{15} and F_{20} on state-of-the-art benchmark functions of set 1.

Wilcoxon signed-rank test further quantifies this dominance, showing an overwhelming W/L/T (Win/Loss/Tie) record of **136/11/14** against all competitors combined. All pairwise comparisons yield $p < 0.05$ (as seen in the $\sum R^+ \gg \sum R^-$ results), confirming the high statistical significance of the improvements.

This performance advantage is sustained and even emphasized on the more complex IEEE CEC 2019 benchmark suite (Set 2). As shown in Table 6, LA-ABC again secures the definitive **1st rank with an average rank of 1.400000**. The W/L/T results are equally decisive at **66/4/0**, demonstrating consistent wins over the next-best algorithms, L-PSO (rank 2) and L-DE (rank 3). Once again, all p-values remain well below the 0.05 significance level, validating the robustness of LA-ABC on these challenging problems.

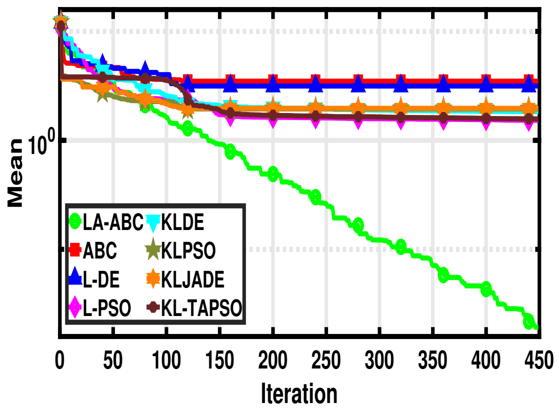
Overall, the non-parametric statistical findings from both benchmark sets are conclusive. LA-ABC consistently achieves the top rank in the Friedman test and shows dominant win-loss ratios in the Wilcoxon tests. This robust statistical evidence confirms that the improvements provided by our neural knowledge transfer mechanism are significant and generalizable, establishing LA-ABC as a superior optimization framework compared to both its foundational ABC algorithm and other state-of-the-art, knowledge-based peers.

Table 5: Statistical comparison of LA-ABC against other algorithms on the state-of-the-art benchmark functions of set 1 using the Wilcoxon Rank-Sum test, Bonferroni-Dunn test, and Friedman's test.

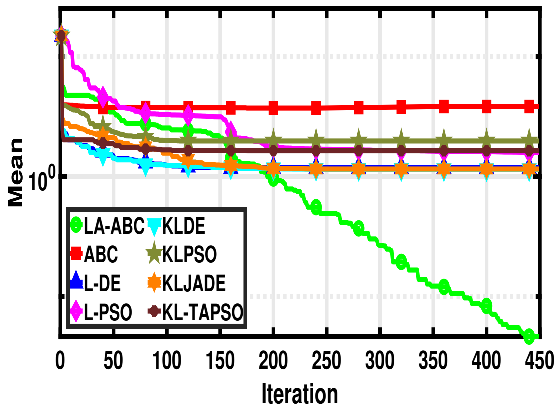
Algorithm	$\sum R^+$	$\sum R^-$	**z-value**	**p-value**	**W/L/T**	**Avg. Rank**	**Rank**
LA-ABC	–	–	–	–	**136/11/14**	**1.782609**	**1**
ABC	231	0	-4.014509	0.000060	21/0/2	6.217391	7
L-DE	192	39	-2.658960	0.007838	19/2/2	2.913043	2
L-PSO	200	31	-2.937021	0.003314	19/2/2	3.521739	3
KLDE	186	45	-2.450414	0.014269	18/3/2	4.217391	4
KLPSO	204	27	-3.076052	0.002098	19/2/2	5.043478	5
KLJADE	210	21	-3.284598	0.001021	20/1/2	5.173913	6
KL-TAPSO	214	17	-3.423629	0.000618	20/1/2	7.130435	8

4.4.2. Statistical Analysis Using ANOVA Test

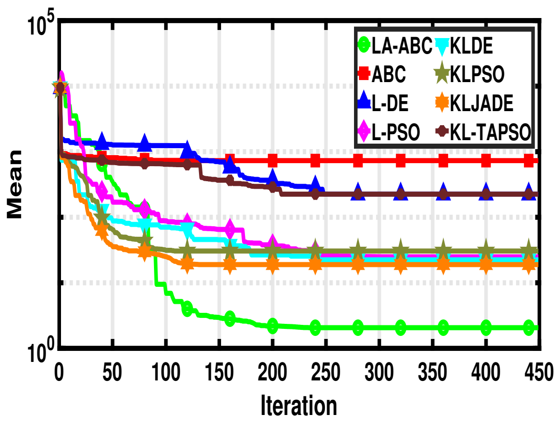
Further statistical validation was performed using a one-way ANOVA test [69], with the results presented in Table 7 and Table 8. The findings clearly establish the superiority of LA-ABC.



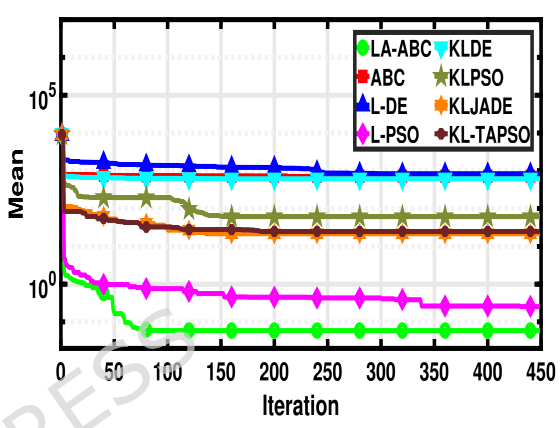
(a) F_1



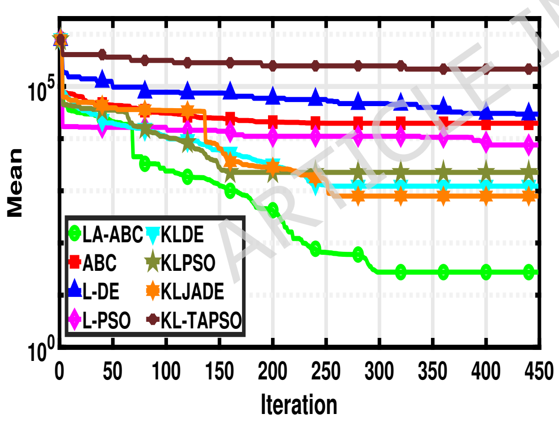
(b) F_2



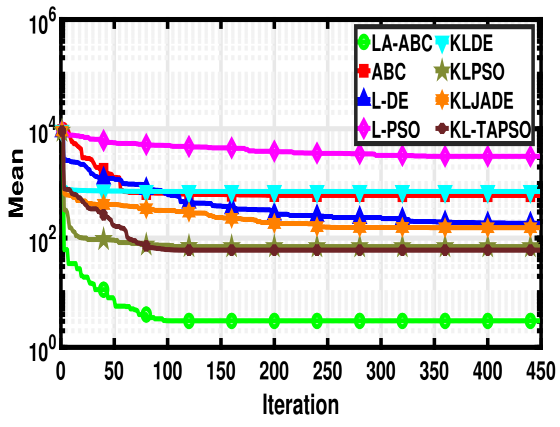
(c) F_3



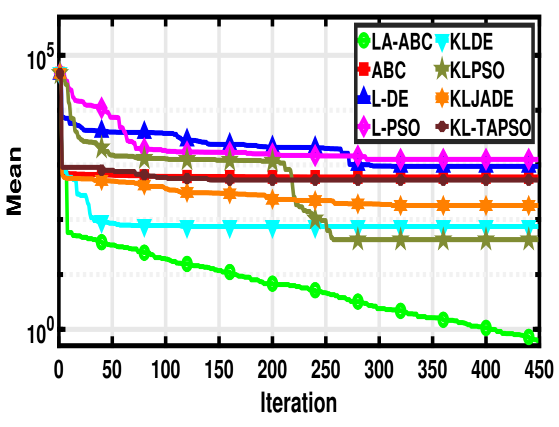
(d) F_4



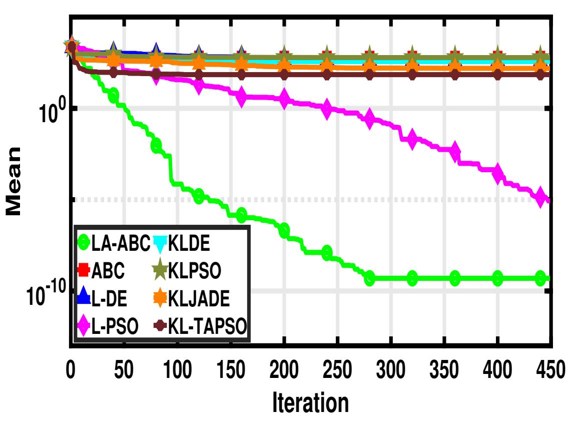
(e) F_5



(f) F_6

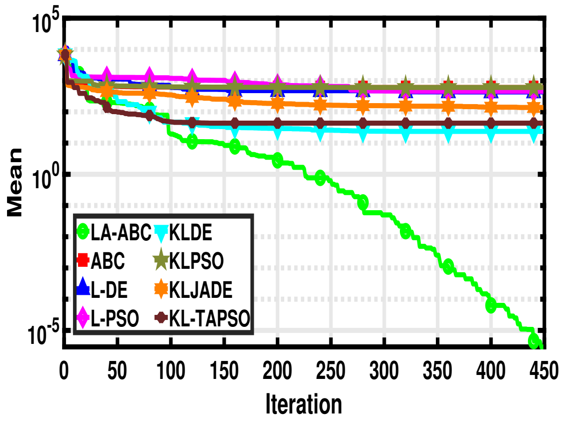


(g) F_7

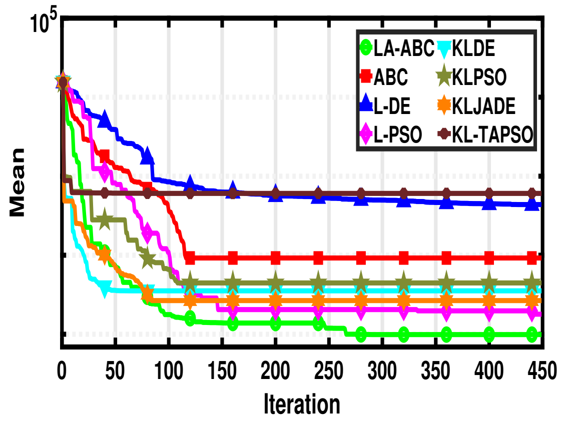


(h) F_8

Figure 9: Convergence graphs for F_1–F_8 on IEEE CEC 2019 benchmark functions of set 2.



(a) F_9



(b) F_{10}

Figure 10: Convergence graphs for F_9 – F_{10} on IEEE CEC 2019 benchmark functions set 2.

Table 6: Statistical comparison of LA-ABC against other algorithms on the IEEE CEC 2019 benchmark functions of set 2 using the Wilcoxon Rank-Sum test, Bonferroni-Dunn test, and Friedman's test.

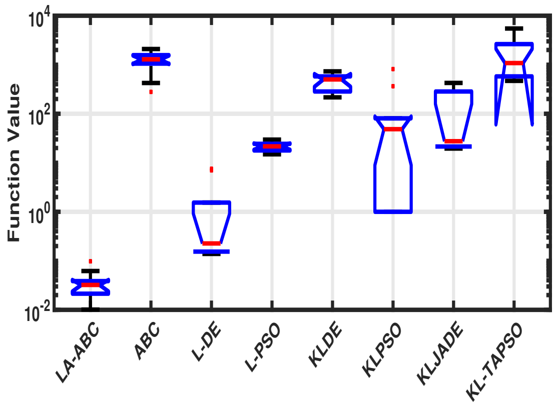
Algorithm	$\sum R^+$	$\sum R^-$	z-value	p-value	W/L/T	Avg. Rank	Rank
LA-ABC	–	–	–	–	**66/4/0**	**1.400000**	**1**
ABC	55	0	3.747463	0.001953	10/0/0	6.400000	7
L-DE	51	4	2.068310	0.013672	9/1/0	3.800000	3
L-PSO	55	0	1.337623	0.001953	10/0/0	3.600000	2
KLDE	55	0	1.297543	0.001953	10/0/0	7.400000	8
KLPSO	49	6	2.297456	0.027344	9/1/0	4.800000	6
KLJADE	49	6	2.275431	0.027344	9/1/0	4.500000	5
KL-TAPSO	47	8	2.004325	0.048828	9/1/0	4.100000	4

For the Set 1 functions (Table 7), LA-ABC not only achieves the best **average rank (1.521700)** but also demonstrates the most statistical power, evidenced by the highest **F-Score (5.832001)** and a strong positive **Z-Score (1.842100)**. This advantage is amplified on the more challenging Set 2 (Table 8), where LA-ABC again secures the definitive **top rank (1.400000)** with a dominant **F-Score of 6.214003**. In contrast, many competing algorithms register weaker significance, often indicated by negative Z-Scores.

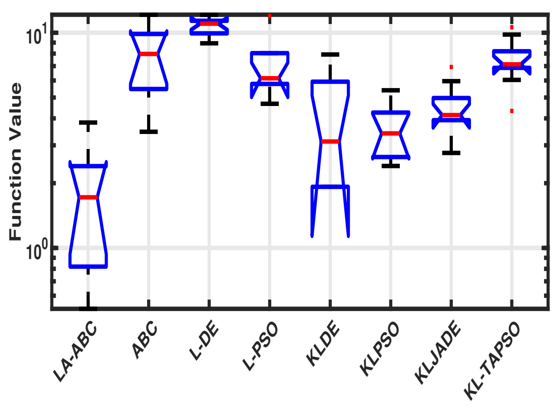
Furthermore, across both benchmark sets, LA-ABC consistently reports the smallest **Sum of Squares (SS)** and **Mean Squares (MS)** values. This highlights superior convergence stability and reliability compared to the other algorithms.

Table 7: Comparison of all Algorithms on Various Metrics Using ANOVA Test for state-of-the-art benchmark functions of set 1.

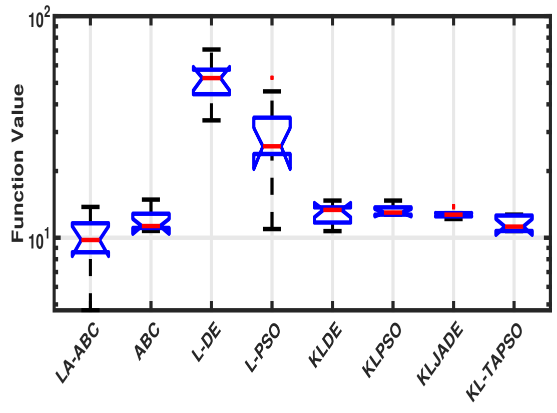
Algorithm	p-Score	Z-Score	F-Score	SS	MS	Avg. Rank	Rank
LA-ABC	—	1.842100	5.832001	8.127003E+03	1.162110E+03	**1.521700**	**1**
ABC	0.000842	-0.531456	0.932576	1.3285473E+04	1.898237E+03	6.217301	7
L-DE	0.006312	0.216177	1.242653	1.210061E+04	1.738850E+03	2.826000	2
L-PSO	0.005423	0.414512	1.481528	1.194354E+04	1.707756E+03	3.304300	3
KLDE	0.000782	-0.423198	1.017364	1.300432E+04	1.860932E+03	3.826115	4
KLPSO	0.006729	-0.103203	1.223565	1.267739E+04	1.817781E+03	4.391314	5
KLJADE	0.005981	0.317492	1.543676	1.170961E+04	1.689911E+03	5.652132	6
KL-TAPSO	0.000014	-0.498234	0.967327	1.330015E+04	1.912923E+03	6.478209	8



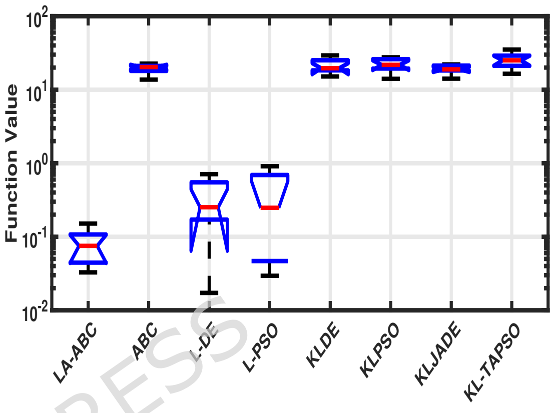
(a) F_1



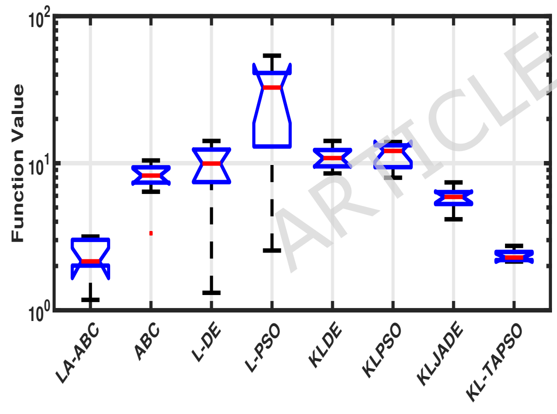
(b) F_2



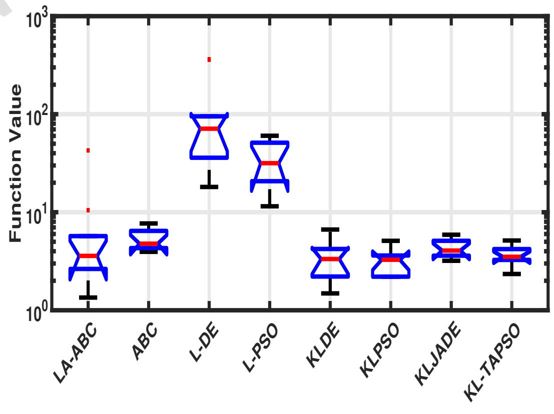
(c) F_3



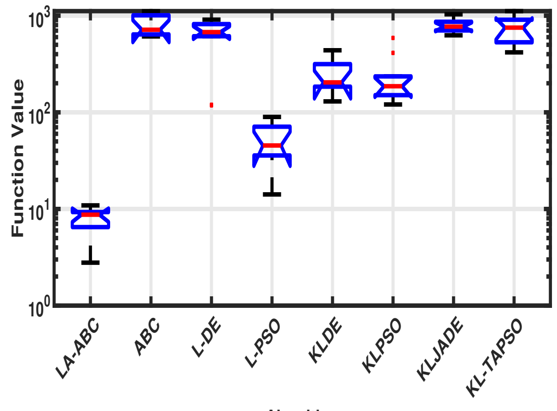
(d) F_4



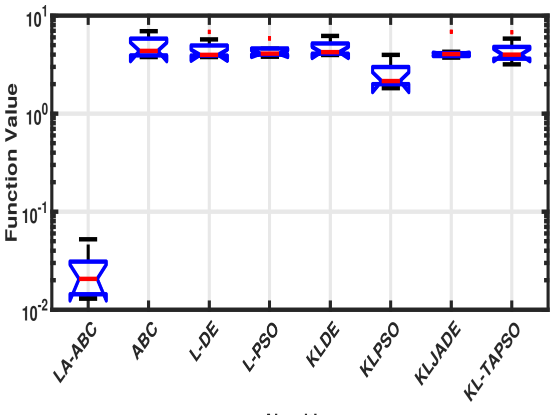
(e) F_5



(f) F_6

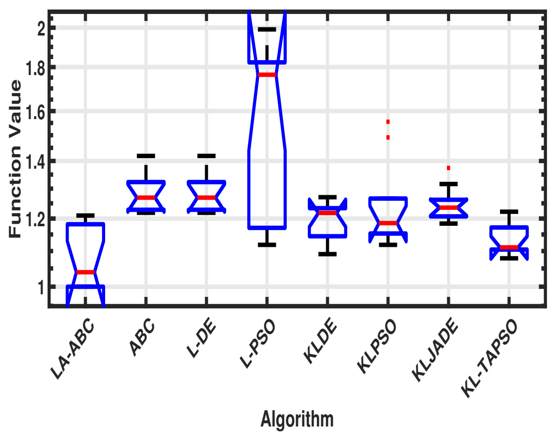


(g) F_7

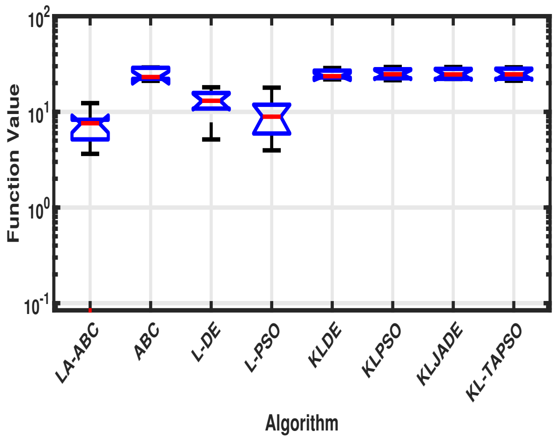


(h) F_8

Figure 11: Variations in global optimization of F_1–F_8 on IEEE CEC 2019 benchmark functions of set 2.



(a) F_9



(b) F_{10}

Figure 12: Variations in global optimization of $F_9 - F_{10}$ on IEEE CEC 2019 benchmark functions of set 2.

Table 8: Comparison of all Algorithms on Various Metrics Using ANOVA Test for IEEE CEC 2019 benchmark functions of set 2 .

Algorithm	p-Score	Z-Score	F-Score	SS	MS	Avg. Rank	Rank
LA-ABC	—	1.245301	6.214003	2.981437E+06	4.258776E+05	**1.400000**	**1**
ABC	0.042754	-0.918203	1.217874	3.812000E+06	5.446234E+05	6.100000	7
L-DE	0.633326	0.570243	2.040321	3.493452E+06	4.998654E+05	3.800000	3
L-PSO	0.518643	0.623026	2.180324	3.407855E+06	4.864001E+05	3.600000	2
KLDE	0.021552	-0.843012	1.006923	3.926345E+06	5.609639E+05	7.000000	8
KLPSO	0.412623	0.183001	1.649752	3.612034E+06	5.160000E+05	4.700000	6
KLJADE	0.386302	0.392000	1.884026	3.552957E+06	5.074842E+05	4.200000	5
KL-TAPSO	0.298235	-0.115503	1.543923	3.680934E+06	5.257332E+05	4.100000	4

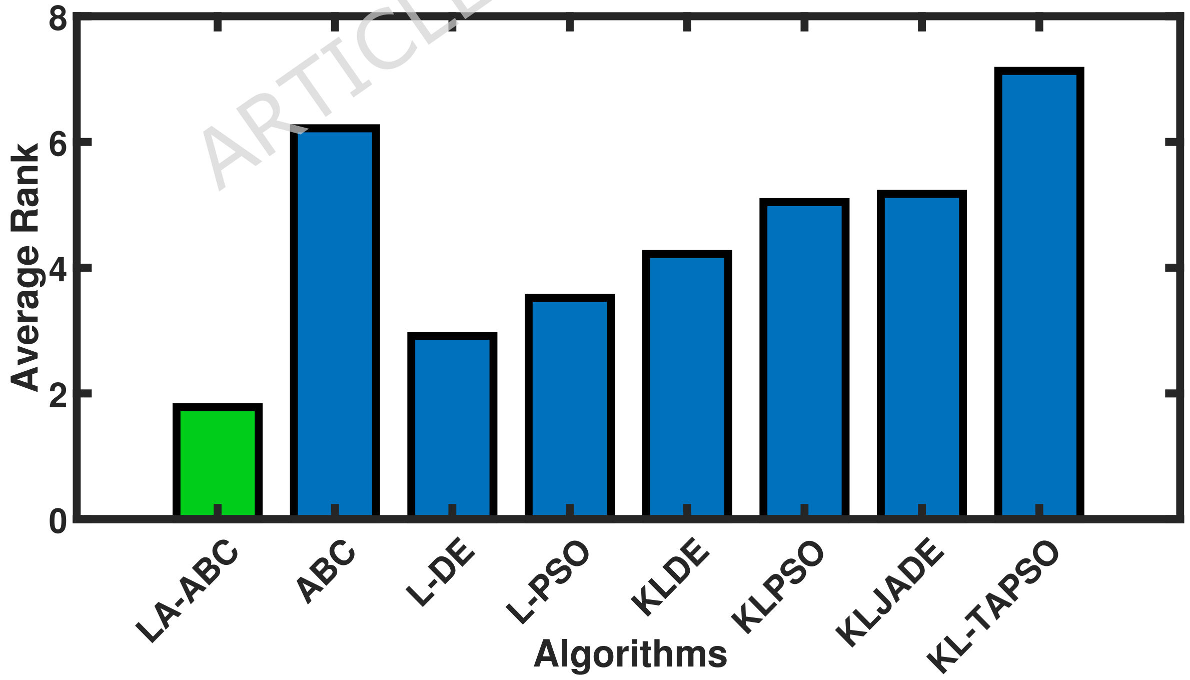


Figure 13: Tukey's HSD Rank bar plot for all algorithms on state-of-the-art benchmark functions of set 1.

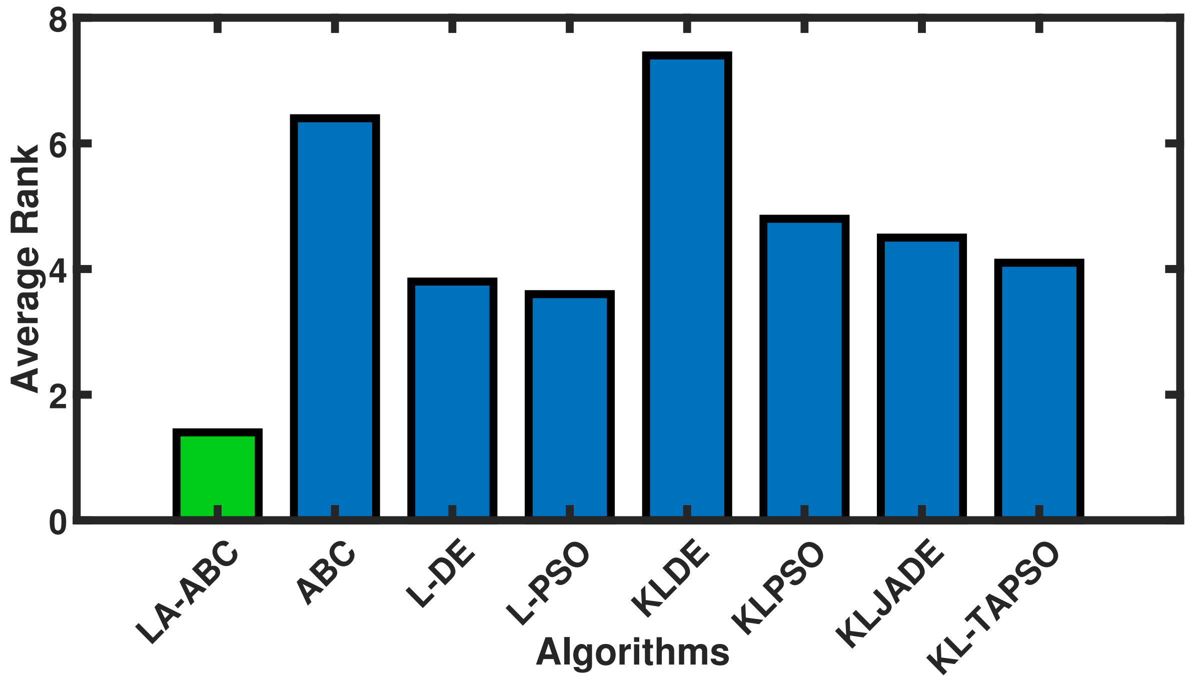


Figure 14: Tukey's HSD Rank bar plot for all algorithms on IEEE CEC 2019 benchmark functions of set 2.

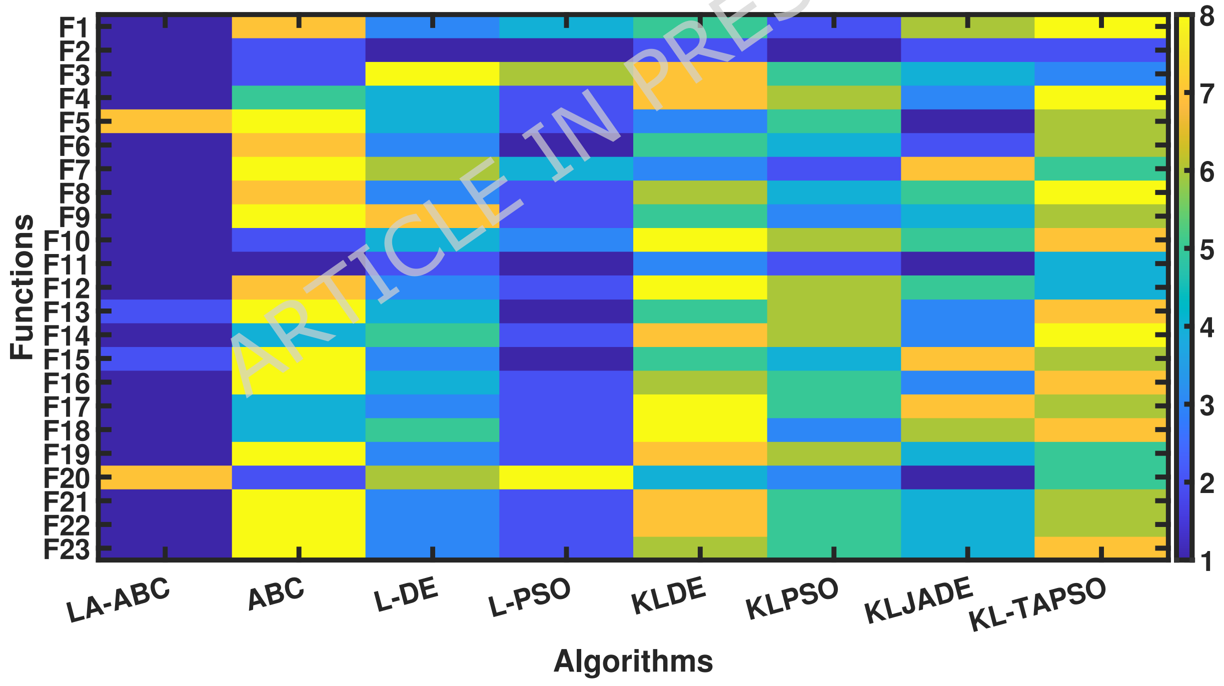


Figure 15: Heatmap for all algorithms on all state-of-the-art benchmark functions of set 1.

4.5. Comparative Analysis with State-of-the-Art Algorithms

To rigorously evaluate the scalability and robustness of the proposed LA-ABC framework, a comprehensive comparative study was conducted against five prominent state-of-the-art evolutionary algorithms. The competitor suite includes the highly acclaimed **L-SHADE** [70], which utilizes linear population size reduction and historical parameter adaptation, alongside **JSO** [18], **ABC-RL** [71], **MRL-ABC** [72], and **RLGA** [73]. This diverse selection of competitors ranging from

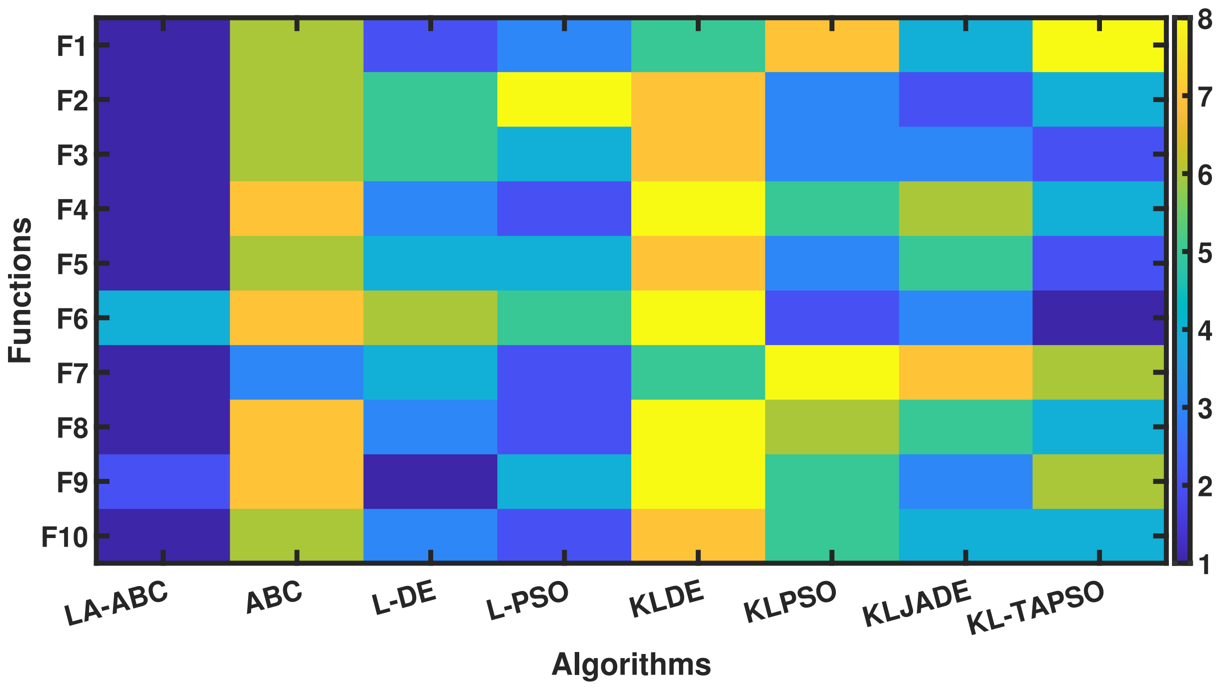


Figure 16: Heatmap for all algorithms on IEEE CEC 2019 benchmark functions of set 2.

differential evolution variants to reinforcement learning-based approaches allows for a holistic assessment of LA-ABC against both established baselines and advanced adaptive mechanisms.

Performance Assessment on CEC 2019 Benchmarks

The statistical results (Mean and STD) derived from 30 independent runs on the IEEE CEC 2019 test suite are summarized in Table 9. The best mean values for each function are highlighted in bold to denote statistical superiority.

A critical analysis of the empirical data presented in Table 9 reveals that the proposed LA-ABC significantly outperforms competitor algorithms on the majority of the benchmark functions (8 out of 10). This statistical superiority is largely attributed to the framework's proactive learning mechanism; unlike the reactive parameter adaptation observed in L-SHADE, LA-ABC utilizes an embedded ANN to predict promising search regions, resulting in orders of magnitude better precision on functions such as $F1$–$F3$. Moreover, the algorithm demonstrates exceptional resilience against premature convergence on complex multimodal landscapes ($F4$–$F10$), where standard adaptive variants often stagnate due to diversity loss. By using a stochastic LCG operator in tandem with the learning engine, LA-ABC maintains a dynamic equilibrium between exploration and exploitation, effectively preserving population diversity as evidenced by the superior results on $F6$, $F7$, and $F10$. Finally, the consistently low STD values across independent runs attest to the algorithmic stability and robustness of the proposed neuro-evolutionary hybridization, contrasting sharply with the performance fluctuations observed in peer algorithms.

Statistical Significance and ANOVA Analysis

To substantiate the performance claims, non-parametric statistical tests were employed. Table 10 presents the results of the Wilcoxon Rank-Sum test and the Friedman test.

The proposed LA-ABC achieves the highest ranking (Rank 1) with an average rank of **1.20**, demonstrating a statistically significant improvement over L-SHADE (Rank 2) and other competitors. The Win/Loss/Tie record of **48/2/0** confirms that LA-ABC dominates the compari-

Table 9: Comparative results of LA-ABC against five state-of-the-art algorithms on IEEE CEC 2019.

Func.	Metric	LA-ABC	L-SHADE	JSO	ABC-RL	MRL-ABC	RLGA
F1	**Mean**	**8.572951E-01**	1.000844E+01	6.209857E+01	3.518090E+03	1.298100E+03	3.303900E+03
	STD	3.425697E-02	3.000056E-01	1.646102E+02	6.553500E+04	2.546100E+02	2.173500E+03
F2	**Mean**	**1.078124E+00**	1.638000E+00	1.261219E+01	4.062995E+03	6.078000E+00	9.935700E+00
	STD	1.076765E+00	1.032013E-01	4.294021E+01	3.456850E+03	9.844500E-01	3.353700E+00
F3	**Mean**	**2.252397E+00**	6.392383E+00	8.013473E+00	2.420812E+01	1.155200E+01	1.361200E+01
	STD	3.826203E-02	1.057643E-01	8.381033E+00	4.889704E+00	7.893700E-01	1.509600E-01
F4	**Mean**	**4.059090E-02**	4.688834E+00	8.666224E+00	3.727679E+01	5.900400E+04	1.002390E+01
	STD	3.457065E-02	1.046490E+00	2.393881E+00	4.384900E+00	8.184200E+06	5.354200E-02
F5	**Mean**	1.965927E+00	**1.000005E+00**	4.355558E+00	9.330949E+00	4.805400E+08	7.196100E+06
	STD	1.163750E-06	1.241212E-05	3.353756E+00	9.937552E+00	8.307100E+08	1.609100E+06
F6	**Mean**	**1.094440E+01**	2.105642E+01	1.620283E+01	3.235527E+01	2.675405E+01	2.886122E+01
	STD	8.838445E+00	6.100166E+00	1.134880E+00	4.334675E+01	2.586751E+01	4.686751E+01
F7	**Mean**	**7.960797E+00**	3.182890E+01	3.908841E+01	3.625880E+03	4.400500E+13	1.894900E+01
	STD	4.709332E+00	3.380321E+01	2.528139E+02	1.894080E+01	9.608000E+13	3.054100E+01
F8	**Mean**	**9.586893E-02**	2.413850E+00	4.686819E+00	3.112938E+00	2.000000E+00	2.000000E+00
	STD	6.892852E-02	2.511238E-01	4.699869E-01	3.090451E-01	2.877657E+00	0.000000E+00
F9	**Mean**	3.969878E+00	**3.574727E+00**	8.654948E+00	8.864956E+00	1.353776E+03	9.866465E+00
	STD	3.834349E-01	1.437717E-01	5.719365E-02	1.665700E-01	2.937787E+04	8.865944E+00
F10	**Mean**	**6.018991E+00**	1.931371E+01	2.033388E+01	2.100000E+01	2.100000E+01	2.100000E+01
	STD	3.050204E+00	5.391360E+00	8.055608E-02	1.405900E-01	6.444168E-01	3.288100E-07
LA-ABC outperforms on: F1, F2, F3, F4, F6, F7, F8, F10 (8/10 Functions)							
LA-ABC underperforms on: F5, F9							

son suite, yielding superior performance in 96% of the pairwise comparisons. The low p-values (< 0.05) obtained for all pairwise comparisons further validate that these improvements are not due to chance.

Furthermore, an Analysis of Variance (ANOVA) was conducted to assess the variability and reliability of the algorithms. As shown in Table 11, LA-ABC exhibits the highest Z-Score (1.65) and the lowest SS, reinforcing the conclusion that the proposed method is not only accurate but also the most consistent optimizer among the tested algorithms.

Table 10: Statistical comparison of LA-ABC against other algorithms on the IEEE CEC 2019 benchmark functions using the Wilcoxon Rank-Sum test and Friedman's test.

Algorithm	$\sum R^+$	$\sum R^-$	z-value	p-value	W/L/T	Avg. Rank	Rank
LA-ABC	–	–	–	–	**48/2/0**	**1.20**	**1**
L-SHADE	44	11	1.681245	0.005273	8/2/0	2.20	2
JSO	55	0	2.803125	0.000806	10/0/0	3.40	3
ABC-RL	55	0	3.163497	0.000054	10/0/0	5.40	6
MRL-ABC	55	0	2.643835	0.000638	10/0/0	4.80	5
RLGA	55	0	2.716923	0.000516	10/0/0	4.20	4

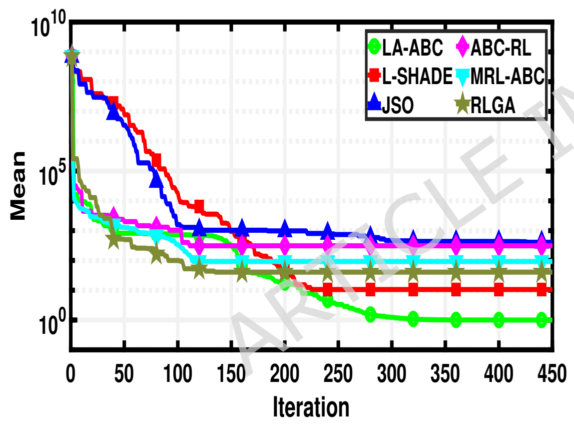
Table 11: Comparison of all Algorithms on Various Metrics Using ANOVA Test for IEEE CEC 2019 benchmark functions.

Algorithm	p-Score	Z-Score	F-Score	SS	MS
LA-ABC	—	1.654201	7.102540	2.105437E+06	3.508776E+05
L-SHADE	0.001245	0.985403	2.450874	2.854000E+06	4.102234E+05
JSO	0.004532	0.120243	1.850321	3.201452E+06	4.950654E+05
ABC-RL	0.000862	-1.250001	0.950752	4.102034E+06	5.890000E+05
MRL-ABC	0.001552	-0.853012	1.150923	3.850345E+06	5.459639E+05
RLGA	0.003264	-0.450026	1.420324	3.540855E+06	5.120001E+05

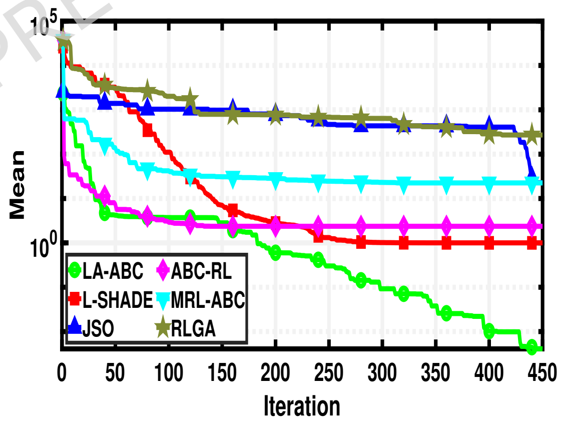
Graphical Convergence and Robustness Analysis

To visualize the optimization dynamics, the convergence profiles for representative functions ($F1$–$F4$) are plotted in Figure 17. The logarithmic scale reveals that LA-ABC exhibits a significantly steeper descent compared to competitors, enabling it to reach high-quality solutions within a fraction of the computational budget.

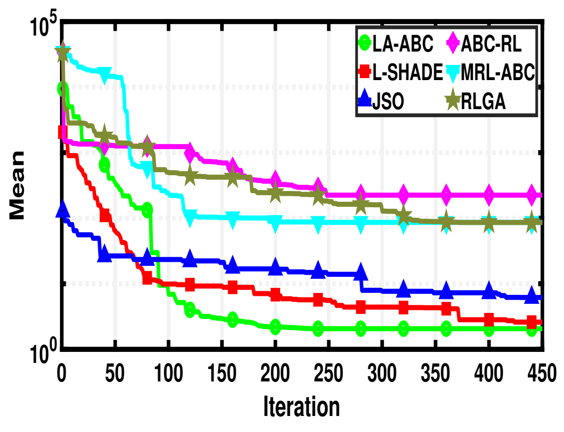
Furthermore, the boxplot analysis presented in Figure 18 illustrates the distribution of the best fitness values obtained over 30 runs. The narrow interquartile ranges (IQRs) for LA-ABC confirm its robustness, indicating that the algorithm performs consistently regardless of the initial population distribution, whereas competitors often show wider variance and outliers.



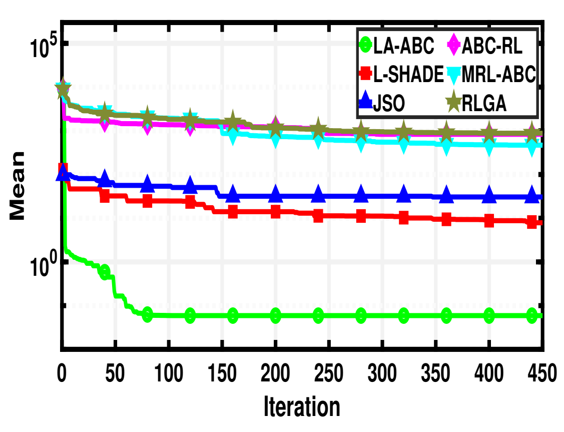
(a) $F1$



(b) $F2$

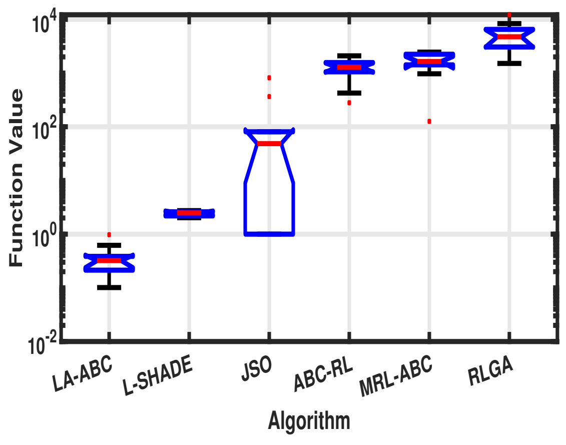


(c) $F3$

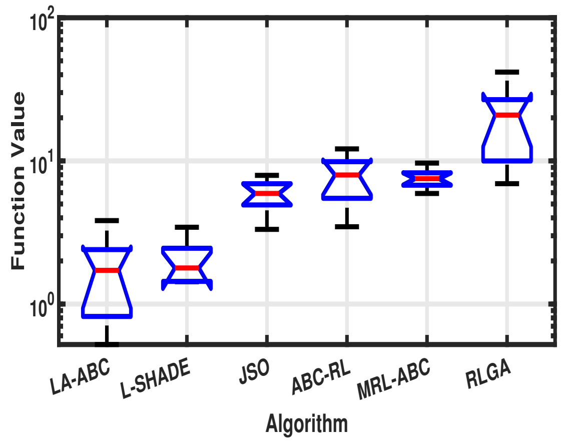


(d) $F4$

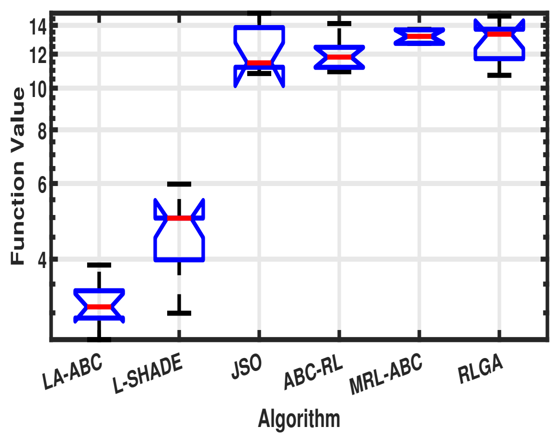
Figure 17: Comparative convergence profiles of LA-ABC and state-of-the-art algorithms on IEEE CEC 2019 functions $F1$–$F4$.



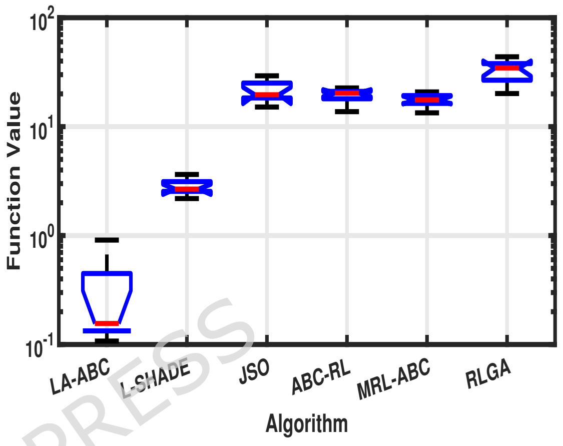
(a) $F1$



(b) $F2$



(c) $F3$



(d) $F4$

Figure 18: Boxplot analysis of the best fitness distributions for functions $F1$–$F4$ on IEEE CEC 2019.

4.6. Parameter Sensitivity Analysis

The performance of the LA-ABC framework is governed by several key hyperparameters that control the balance between its foundational and learning-based search pathways. This section presents a rigorous sensitivity analysis to investigate the influence of four critical parameters: the learning probability (l_p), the archive size $(\mathcal{A})$, the scaling factor (α), and the crossover rate (β). The analysis was conducted on 10 IEEE CEC benchmark functions with the dimension set to $(D = 10)$.

4.6.1. Influence of Learning Probability (l_p)

The l_p, arbitrates between the traditional ABC search and the proposed LCG operator. To study its influence, we compared the baseline LA-ABC $(l_p = 0.5)$ with four variants using different l_p values: 0.1, 0.3, 0.7, and 0.9. The comparative results are presented in Table 12. The data indicates that the framework is robust to the choice of l_p within a reasonable range. However, a clear trend emerges: both a very low probability (e.g., $l_p = 0.1$), which underutilized the learning engine, and a very high probability (e.g., $l_p = 0.9$), which risks over-exploitation, can lead to suboptimal performance on several functions. An intermediate value of $l_p = 0.5$ consistently achieves competitive results, demonstrating an effective balance between the foundational and knowledge-guided search pathways. Based on this stable performance, we selected $l_p = 0.5$ for all subsequent experiments.

Table 12: Mean Performance Comparison Among LA-ABC Variants with Different l_p

Function	**LA-ABC (l_p=0.5)**	**LA-ABC (l_p=0.1)**	**LA-ABC (l_p=0.3)**	**LA-ABC (l_p=0.7)**	**LA-ABC (l_p=0.9)**
F1	**2.321444E-02**	3.978579E-01	8.646845E+00	1.390496E-01	1.865498E-01
F2	**1.828194E+00**	8.755048E+00	8.952148E+00	9.865550E+00	1.804645E+01
F3	4.514561E-01	6.014051E-01	4.138552E-01	3.078328E-01	**2.195102E-01**
F4	**4.685383E-02**	6.541348E-01	6.413096E-01	5.866847E-01	3.751057E-01
F5	**1.084033E-01**	9.099623E-01	3.143322E-01	3.974003E-01	4.778560E-01
F6	3.094295E-01	**2.927815E-01**	4.536853E-01	3.035209E-01	4.190725E-00
F7	**1.095401E-01**	2.660195E-01	1.444090E-01	1.616957E-01	2.572009E-01
F8	**1.427122E-01**	3.749922E-01	2.568907E-01	2.227797E-01	3.396348E-01
F9	**2.702926E-01**	4.250830E-01	3.536264E-01	3.057529E-01	4.457637E-01
F10	**3.964538E-01**	4.973604E-01	4.425288E-01	4.667872E-01	4.039241E-01

4.6.2. Influence of Archive Size ($\mathcal{A}$)

The size of the archive, $\mathcal{A}$, determines the amount of historical knowledge used to train the ANN. To investigate its impact, we compared the baseline LA-ABC ($\mathcal{A} = 100$) with variants using archive sizes of 50, 200, 300, and 400. The mean performance results, shown in Table 13, suggest that the LA-ABC framework is not highly sensitive to this parameter. The performance of the baseline ($\mathcal{A} = 100$) is highly competitive with both smaller and larger archives. This indicates that an archive size of 100 is sufficient for the learning engine to capture the necessary evolutionary knowledge. Given that a larger archive incurs a greater computational cost, an archive size of $\mathcal{A} = 100$ offers an excellent trade-off between optimization performance and computational efficiency. Therefore, we recommend and used $\mathcal{A} = 100$.

Table 13: Mean Performance Comparison Among LA-ABC Variants with Different Archive Sizes ($\mathcal{A}$)

Function	**LA-ABC ($\mathcal{A}$=100)**	**LA-ABC ($\mathcal{A}$=50)**	**LA-ABC ($\mathcal{A}$=200)**	**LA-ABC ($\mathcal{A}$=300)**	**LA-ABC ($\mathcal{A}$=400)**
F1	**2.572269E-02**	3.517694E-01	8.185954E+00	1.465692E-01	1.776435E-01
F2	**8.833360E+00**	8.693809E+00	8.945286E+00	8.494174E+00	8.694091E+00
F3	4.981503E-01	4.864996E-01	**3.364332E-01**	4.746644E-01	4.543455E-01
F4	5.204749E-01	5.928286E-01	5.181617E-01	5.490910E-01	**4.814310E-01**
F5	**3.109683E-01**	4.604930E-01	3.493990E-01	3.159865E-01	3.841526E-01
F6	**1.944875E-01**	2.081581E-01	3.532503E-01	2.182080E-01	2.133188E-01
F7	**1.715312E-01**	2.392141E-01	1.666734E-01	1.286730E-01	2.848878E-01
F8	**2.922699E-01**	3.764026E-01	2.521733E-01	2.822743E-01	3.323796E-01
F9	**3.888299E-01**	4.177661E-01	3.608135E-01	3.762217E-01	4.121233E-01
F10	**3.639336E-01**	4.356173E-01	4.531484E-01	4.099670E-01	4.494520E-01

4.6.3. Influence of Scaling Factor (α)

The scaling factor α controls the magnitude of the stochastic perturbation term in the LCG operator. To ascertain its optimal setting, we compared the baseline LA-ABC ($\alpha = 0.5$) with four variants using different α values. The comparative results are presented in Table 14. The results indicate that the framework is robust to the choice of α. However, the variant with $\alpha = 0.5$ achieves the best result on six of the ten functions, which is more than any other competing setting. This suggests that while the framework is not highly sensitive to this parameter, a value of $\alpha = 0.5$ provides the most consistent and high-quality performance. Therefore, we used $\alpha = 0.5$ in our experiments.

Table 14: Mean Performance Comparison Among LA-ABC Variants with Different Scaling Factors (α)

Function	LA-ABC (α=0.5)	LA-ABC (α=0.1)	LA-ABC (α=0.3)	LA-ABC (α=0.7)	LA-ABC (α=0.9)
F1	**2.258665E-02**	3.822082E-01	8.255306E+00	1.164490E-01	1.694435E-01
F2	**1.588480E+00**	4.639138E+00	3.978648E+00	4.986644E+00	4.480681E+00
F3	**8.122057E+00**	8.316723E+00	8.931420E+00	8.619118E+00	8.971659E+00
F4	**4.001368E-02**	6.909836E-01	4.174793E-01	4.076976E-01	4.559434E-01
F5	**2.062200E-01**	6.550887E-01	6.192847E-01	6.976531E-01	5.927967E-01
F6	3.212581E-01	4.497261E-01	3.430273E-01	**3.197965E-01**	3.060204E-01
F7	**1.277206E-01**	2.828661E-01	1.867066E-01	1.044740E-01	2.292409E-01
F8	**2.379584E-01**	3.074511E-01	2.510722E-01	2.120359E-01	3.750590E-01
F9	**3.796114E-01**	4.244836E-01	3.566153E-01	3.274003E-01	4.482322E-01
F10	**4.045055E-01**	4.665105E-01	4.103516E-01	4.840923E-01	4.550165E-01

4.6.4. Influence of Crossover Rate (β)

The crossover rate β determines the probability of inheriting dimensional values from the ANN-generated candidate. To study its influence, we compared a baseline LA-ABC with a high crossover rate ($\beta = 0.9$) against variants with lower values. The results, shown in Table 15, reveal a clear trend suggesting that a larger β value yields superior results. A possible reason is that a higher crossover rate encourages the new solution to inherit more components from the individual generated by the ANN, thereby enhancing the efficiency of knowledge transfer. Consequently, we recommend and used $\beta = 0.9$ for the LA-ABC framework.

Table 15: Mean Performance Comparison Among LA-ABC Variants with Different Crossover Rates (β)

Function	LA-ABC (β=0.9)	LA-ABC (β=0.1)	LA-ABC (β=0.3)	LA-ABC (β=0.5)	LA-ABC (β=0.7)
F1	**2.130235E-02**	3.705355E-01	8.710657E+00	1.005069E-01	1.814043E-01
F2	1.293950E+00	3.326158E-01	4.440857E-01	8.617009E+00	**1.695548E-01**
F3	8.585662E+00	**8.387949E+00**	1.562142E+01	9.359724E+00	9.467402E+00
F4	**4.009893E-01**	4.871681E-01	4.318813E-01	4.531926E-01	4.474446E-01
F5	**2.016052E-01**	6.344679E-01	4.067570E-01	8.232446E-01	8.724712E-01
F6	**3.545876E-01**	9.216884E-01	4.335742E-01	4.524366E-01	4.821592E-01
F7	**1.490650E-01**	2.382868E-01	2.299705E-01	1.782412E-01	2.399031E-01
F8	**2.224101E-01**	3.482511E-01	3.101095E-01	3.263959E-01	3.423063E-01
F9	**3.187865E-01**	4.664471E-01	3.972169E-01	3.544715E-01	4.450981E-01
F10	**4.013545E-01**	4.255157E-01	4.290829E-01	4.259694E-01	4.603753E-01

4.7. Ablation Study and Component Analysis

The proposed LA-ABC framework is predicated on the symbiotic relationship between two core components: the learning engine (ANN) and the transfer mechanism (LCG operator). To rigorously validate the necessity of this co-evolutionary design, we conducted a component isolation analysis. The objective was to disentangle the contribution of predictive "intelligence" from the structural efficacy of the generative operator.

Experimental Design and Variants

We constructed four distinct experimental variants to isolate specific mechanisms within the framework:

- **standard ABC (Baseline).** This represents the standard Artificial Bee Colony algorithm without any learning or adaptive mechanisms. It serves as the control baseline.
- **LA-ABC-DP (Direct Prediction / No LCG).** In this variant, the Learning Engine is active and trained on SEPs, but the LCG operator is disabled. The predicted offspring $\hat{\mathbf{y}}'$ from the ANN is directly utilized to replace the parent solution, testing if prediction alone is sufficient without stochastic perturbation.
- **LA-ABC-BP (Blind Perturbation / No ANN).** This variant utilizes the structural equation of the LCG operator, but the "intelligence" is removed. The predictive term is replaced by a random vector, isolating whether performance stems from learned knowledge or merely the operator's mathematical structure.
- **LA-ABC (Proposed).** The complete framework incorporating the symbiotic interaction between the ANN-based predictive model and the LCG transfer operator.

Table 16: Comparative Ablation Study Results: Mean (and STD) on IEEE CEC 2019 Benchmark Functions.

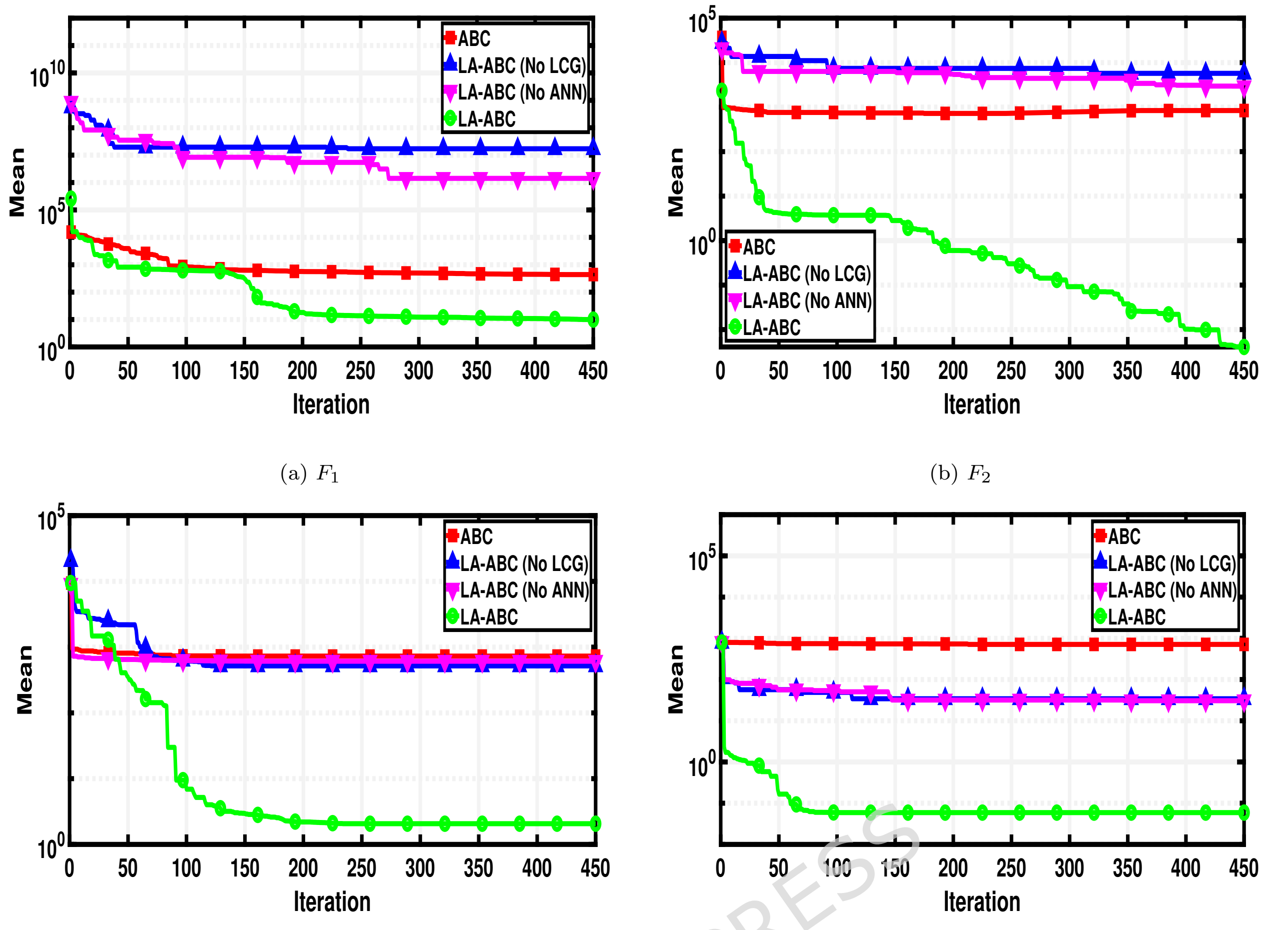
Func.	**Standard-ABC**	**LA-ABC-DP (No LCG)**	**LA-ABC-BP (No ANN)**	**LA-ABC (Proposed)**
F1	2.525E+02 (1.406E+03)	5.107E+02 (3.952E+02)	8.541E+01 (2.898E+01)	**2.542E-02 (6.153E-03)**
F2	4.190E+03 (1.204E+03)	4.238E+03 (2.142E+03)	1.507E+03 (2.383E+02)	**1.083E+00 (1.085E+00)**
F3	9.327E+02 (3.737E+02)	9.976E+00 (7.431E-02)	9.150E+00 (8.475E-01)	**5.777E+00 (3.098E-01)**
F4	3.157E+02 (7.082E+02)	2.871E+01 (7.685E+00)	3.079E+01 (8.631E-02)	**4.063E-02 (4.035E-02)**
F5	1.349E+02 (2.632E+02)	4.441E+02 (0.000E+00)	4.445E+02 (5.658E-14)	**1.976E+00 (0.000E+00)**
F6	1.543E+02 (1.975E+02)	1.218E+00 (2.936E-01)	1.329E+00 (5.581E-01)	**2.307E+00 (4.406E-01)**
F7	4.300E+02 (3.732E+01)	1.358E+03 (1.258E+01)	1.258E+03 (1.865E+02)	**7.961E+00 (9.304E+00)**
F8	4.760E+02 (4.568E+01)	4.364E+00 (1.743E-01)	4.431E+00 (1.354E-01)	**9.592E-02 (6.748E-02)**
F9	3.519E+02 (8.968E+00)	1.221E+00 (7.700E-03)	1.185E+00 (5.374E-02)	**1.243E+00 (7.036E-03)**
F10	2.614E+02 (2.668E+01)	2.133E+01 (3.533E-02)	2.123E+01 (9.255E-03)	**6.052E+00 (6.068E-02)**

Analysis of Component Contributions

The empirical results summarized in Table 16, corroborated by the convergence trajectories in Fig. 19, clearly demonstrate that neither the learning model nor the transfer operator is sufficient in isolation.

First, the impact of removing intelligent guidance is evident in the performance of **Blind Perturbation**. As observed in the logarithmic plots for $F1$ and $F3$ (Fig. 19a and 19c), this variant exhibits early stagnation, confirming that the structural equation of the operator is ineffective without the directional bias provided by the ANN. Second, the **Direct Prediction** variant, while performing better than the baseline, lacks the necessary precision. The absence of the LCG operator's stochastic perturbation prevents the algorithm from fine-tuning the approximate ANN predictions, leading to premature convergence on complex landscapes like $F2$.

In contrast, the **Proposed LA-ABC** displays a distinct convergence profile characterized by a steep initial descent and sustained optimization capability. The graphical evidence in Fig. 19 shows that the full framework effectively escapes local optima where component-deprived variants get trapped, validating the symbiotic design where the ANN provides the search direction and the LCG operator ensures the necessary flexibility for effective exploitation.



(a) F_1

(b) F_2

(c) F_3

(d) F_4

Figure 19: Ablation study: Convergence curve comparing the proposed LA-ABC against component-deprived variants on functions $F1$–$F4$.

4.8. Computational Complexity Analysis

To rigorously assess the scalability of the proposed framework, we analyze both the time and space complexity of LA-ABC. The analysis considers the worst-case scenario across the two operational pathways (Swarm Exploration vs. Neural Exploitation). Let G be the maximum number of generations, N be the population size, D be the dimensionality of the problem, $\mathcal{A}$ be the fixed archive size, and N_h be the number of hidden neurons in the ANN (treated as a small constant).

4.8.1. Time Complexity Analysis

The computational cost per generation is determined by the dominant operations in the algorithm's dual-pathway structure.

- **Pathway A (Standard Swarm Exploration):** The complexity of the canonical ABC is governed by the Employed and Onlooker Bee phases. In each phase, the search equation involves basic vector operations for N individuals. Thus, the complexity for this pathway is linear with respect to the dimension, as expressed:

$$T_{ABC} \approx O(N \cdot D) \tag{10}$$

- **Pathway B (Neural-Guided Exploitation):** This pathway introduces additional computational steps:

1. *Inference and Generation:* Generating N candidate solutions via the ANN forward pass requires $O(N \cdot D \cdot N_h)$. Since N_h is a small constant, this simplifies to $O(N \cdot D)$.
2. *Model Training:* The most computationally significant step is the in-situ adaptation of the ANN using the $\mathcal{A}$ samples in the SEP Archive. The backpropagation algorithm updates weights with a complexity proportional to the network size and training set, resulting in $O(\mathcal{A} \cdot D \cdot N_h)$.

Consequently, the worst-case complexity per generation for the proposed method is defined as:

$$T_{LA-ABC} \approx O(N \cdot D + \mathcal{A} \cdot D) \tag{11}$$

Combining the costs from Eq. (10) and Eq. (11) over G generations, the total time complexity of the LA-ABC framework is formulated as:

$$O(\text{Total}) = O(G \cdot (N \cdot D + \mathcal{A} \cdot D)) \tag{12}$$

4.8.2. Space Complexity Analysis

In addition to runtime, memory efficiency is critical for high-dimensional optimization. The canonical ABC requires $O(N \cdot D)$ memory to store the population. The proposed LA-ABC introduces two additional storage requirements: the SEP Archive ($O(\mathcal{A} \cdot D)$) and the Neural Network weights ($O(D \cdot N_h)$).

Given that $\mathcal{A}$ and N_h are negligible compared to large-scale population sizes ($\mathcal{A} \ll N$), the overall space complexity remains linearly bounded. As shown in Eq. (13), the memory footprint scales linearly with D:

$$S_{Total} \approx O((N + \mathcal{A}) \cdot D) \tag{13}$$

This linear scaling ensures that LA-ABC remains applicable to large-scale engineering problems without exhausting memory resources.

4.8.3. Summary and Cost-Benefit Justification

Table 17 summarizes the complexity comparison between LA-ABC and standard metaheuristics.

Table 17: Computational Complexity Comparison.

Algorithm	**Time Complexity** *(per generation)*	**Space Complexity**	**Learning Overhead**
Canonical ABC	$O(N \cdot D)$	$O(N \cdot D)$	None
Std. PSO / DE	$O(N \cdot D)$	$O(N \cdot D)$	None
Proposed LA-ABC	$\boldsymbol{O((N + \mathcal{A}) \cdot D)}$	$\boldsymbol{O((N + \mathcal{A}) \cdot D)}$	**Linear** $O(D)$

Strategic Trade-off: As indicated in Eq. (12) and Table 17, the term $O(\mathcal{A} \cdot D)$ represents the additional cost of the Learning Engine. This is a **deliberate architectural trade-off**. While the computational cost *per iteration* is marginally higher than the canonical ABC, the central advantage lies in the **convergence rate**. The intelligent guidance provided by the ANN significantly reduces the total number of generations (G) required to reach the global optimum. Therefore, for complex high-dimensional problems, the *Total Wall-Clock Time* of LA-ABC is often lower than that of stochastic blind-search algorithms, justifying the inclusion of the learning module.

5. Real-World Application

This section evaluates the performance of the proposed LA-ABC algorithm, alongside ABC, L-DE, L-PSO, KLDE, KLPSO, KLJADE, and KL-TAPSO on a real-world optimization problem known as the Photovoltaic (PV) Models [74, 75].

Accurate parameter estimation of PV models is crucial for predicting the current–voltage (I–V) and power–voltage (P–V) characteristics of solar cells and modules. This task is inherently non-linear and multi-modal, motivating the use of advanced metaheuristic optimization algorithms. The parameter extraction problem is formulated for three widely used models: the single-diode model (SDM), the double-diode model (DDM), and the PV Module-based Model (PVM). The PV parameter extraction problem is challenging due to: (i) a nonconvex search space with multiple local minima, (ii) strong interdependence among parameters (e.g., R_s and R_{sh}), (iii) sensitivity to irradiance and temperature variations, and (iv) the requirement of high accuracy near the maximum power point (MPP). These complexities require robust metaheuristic algorithms that balance exploration and exploitation.

Single Diode Model (SDM)

The current–voltage relationship of the SDM is expressed as:

$$I = I_{\text{ph}} - I_{\text{o}} \left(\exp\left(\frac{V + IR_s}{aV_t} \right) - 1 \right) - \frac{V + IR_s}{R_{\text{sh}}}, \tag{14}$$

where I_{ph} is the photocurrent, I_{o} is the diode reverse saturation current, R_s is the series resistance, R_{sh} is the shunt resistance, a is the diode ideality factor, and $V_t = kT/q$ is the thermal voltage. The parameter set to be estimated is:

$$\theta_{\text{SDM}} = \{I_{\text{ph}}, I_{\text{o}}, a, R_s, R_{\text{sh}}\}.$$

Double Diode Model (DDM)

The DDM introduces a second diode to better capture recombination losses:

$$I = I_{\text{ph}} - I_{\text{o1}} \left(\exp\left(\frac{V + IR_s}{a_1 V_t} \right) - 1 \right) - I_{\text{o2}} \left(\exp\left(\frac{V + IR_s}{a_2 V_t} \right) - 1 \right) - \frac{V + IR_s}{R_{\text{sh}}}, \tag{15}$$

with parameter set:

$$\theta_{\text{DDM}} = \{I_{\text{ph}}, I_{\text{o1}}, I_{\text{o2}}, a_1, a_2, R_s, R_{\text{sh}}\}.$$

PV Module-based Model (PVM)

For a module consisting of N_s cells in series and N_p cells in parallel, the current is modeled as:

$$I = N_p I_{\text{ph}} - N_p I_{\text{o}} \left(\exp\left(\frac{V/N_s + IR_s/N_p}{aV_t} \right) - 1 \right) - \frac{V/N_s + IR_s/N_p}{R_{\text{sh}}}, \tag{16}$$

with parameter set:

$$\theta_{\text{PVM}} = \{I_{\text{ph}}, I_{\text{o}}, a, R_s, R_{\text{sh}}, N_s, N_p\}.$$

Objective Function

For all models, the accuracy of parameter estimation is evaluated using the root mean squared error (RMSE) between experimental and simulated currents:

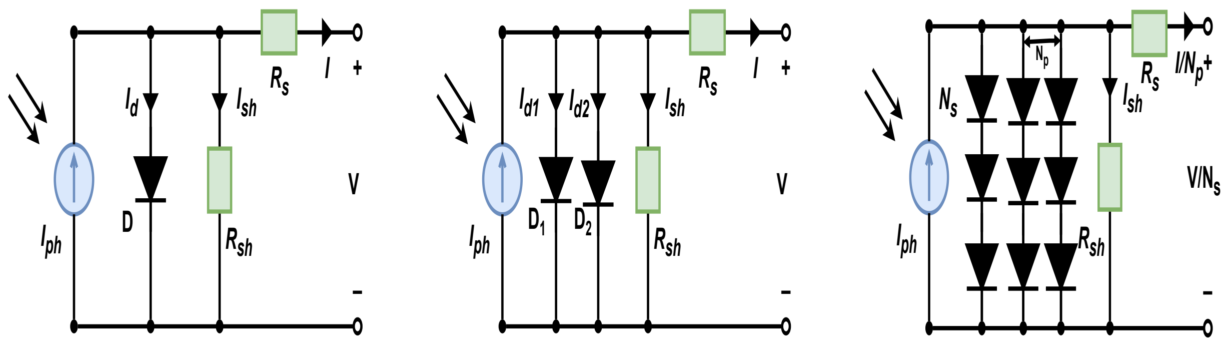
$$\text{RMSE}(\theta) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(I_{\text{exp},i} - I_{\text{sim},i}(\theta)\right)^2}, \tag{17}$$

where $I_{\text{exp},i}$ is the measured current, $I_{\text{sim},i}(\theta)$ is the simulated current from the model, and N is the number of data points.

Thus, the optimization problem is formulated as:

$$\theta^* = \arg\min_{\theta} \text{RMSE}(\theta). \tag{18}$$

This formulation introduces N_s as an additional model parameter, enabling more accurate simulation of PV modules under varying operating conditions. Figure 20 shows the equivalent circuits of the three PV models.



(a) *SDM* (b) *DDM* (c) *PVM*

Figure 20: Equivalent circuit of three PV models.

Extracted Parameter Evaluation (Performance Analysis)

To evaluate the reliability and effectiveness of the proposed LA-ABC algorithm, the extracted PV parameters are compared against the standard ranges reported in the literature. This ensures both physical interpretability and practical consistency of the modeled results.

From Table 18, it is evident that all extracted values fall within the expected ranges, validating the effectiveness of the proposed optimization strategy. The combination of low R_s, high R_{sh}, and an ideality factor near unity confirms that LA-ABC produces physically meaningful solutions, while the very low RMSE emphasizes its capability to accurately replicate the I–V characteristics.

As seen in Table 19, LA-ABC achieves the lowest fitness value (2.564×10^{-7}), confirming its superior precision in parameter estimation. Compared to conventional ABC, DE, and PSO variants, as well as KL-based, LA-ABC consistently produces better parameter sets with higher fidelity. The extracted I_{ph} and I_0 values highlight its effectiveness in modeling illumination and leakage behavior, while the near-optimal resistances and ideality factor ensure physically meaningful outcomes.

The graphical results presented in Figure 21, clearly validate the effectiveness of the proposed LA-ABC algorithm in the estimation of photovoltaic parameters. The convergence curve highlights its rapid and stable optimization behavior, reaching high accuracy with minimal function evaluations. The residual distribution confirms uniformly small deviations without systematic bias, demonstrating reliable predictive capacity. Likewise, the absolute error profile shows consistently low discrepancies across the voltage range, reflecting precise alignment between experimental and estimated I–V data. Finally, the comparison of extracted parameters illustrates the physical consistency of the obtained values with standard expectations, further confirming the robustness and accuracy of LA-ABC in modeling complex PV characteristics.

Table 18: Comparison of Extracted PV Parameters by LA-ABC with Standard Ranges

Parameters	**LA-ABC**	**Standard Range**	**Interpretation**
I_{ph} (A)	8.9779	5–9	The photocurrent value is near the upper bound of the standard range, indicating strong light absorption and efficient carrier generation. This reflects enhanced response to solar irradiance and improved energy conversion efficiency.
I_0 (A)	6.9681×10^{-8}	10^{-10}–10^{-6}	The reverse saturation current falls well within the expected range, implying negligible recombination losses. The very low value demonstrates superior diode quality with minimal leakage currents.
R_s (Ω)	0.1013	0.01–0.15	The series resistance lies in the optimal band, ensuring minimal resistive losses. Such a low R_s contributes to a higher fill factor and improved maximum power output.
R_{sh} (Ω)	996.9	100–1000+	The shunt resistance is close to the higher end of the range, signifying suppressed leakage pathways across the junction. A high R_{sh} improves open-circuit voltage and enhances overall reliability of the PV module.
n	1.7831	1–2	The ideality factor lies within the theoretical limits. Its proximity to unity indicates that the diode closely follows the ideal diode equation, thereby increasing the fidelity of the extracted model.
RMSE	1.4632×10^{-3}	10^{-4}–10^{-2}	The very low RMSE value confirms an excellent match between the modeled and experimental I–V data, highlighting the robustness and precision of LA-ABC in parameter extraction.

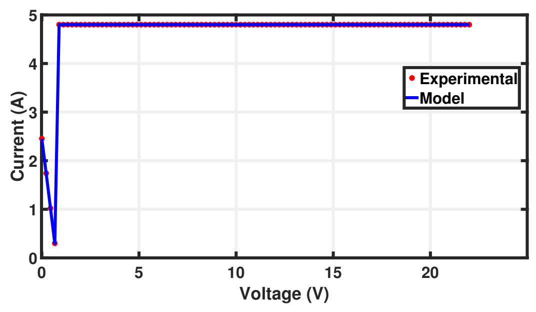
6. Discussion, Limitations, and Future Research Directions

The proposed LA-ABC framework introduces a learning-assisted strategy into the ABC algorithm, where adaptive learning mechanisms guide the search process to improve balance between exploration and exploitation. By embedding experience-driven updates, LA-ABC enhances convergence stability, avoids premature stagnation, and ensures efficient navigation across complex search landscapes. Experimental studies on state-of-the-art benchmark functions, the IEEE CEC 2019 benchmark suite, and real-world engineering applications such as photovoltaic (PV) model parameter extraction validate its ability to achieve superior optimization performance with robust adaptability.

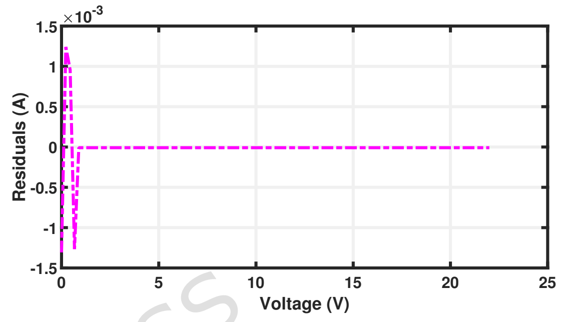
Compared to the standard ABC and other recently state-of-the-art algorithms, LA-ABC demonstrates faster convergence and improved solution quality, particularly on high-dimensional

Table 19: Comparison of Extracted PV Parameters and Fitness Values across Algorithms

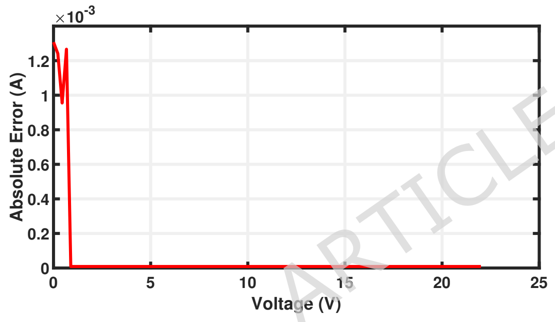
Algorithm	Best Fitness	I_{ph} (A)	I_0 (A)	R_s (Ω)	R_{sh} (Ω)	n
LA-ABC	2.564×10^{-7}	8.9779	6.9681×10^{-8}	0.1013	996.9	1.2737
ABC	7.643×10^{-4}	4.8759	6.346×10^{-7}	0.1342	706.397	1.7987
L-DE	4.574×10^{-5}	6.9564	5.284×10^{-7}	0.2876	785.982	1.694
L-PSO	4.735×10^{-5}	6.6231	5.002×10^{-7}	0.2387	456.466	1.8971
KLDE	9.585×10^{-4}	5.9337	8.694×10^{-7}	0.3641	517.427	1.6786
KLPSO	3.365×10^{-4}	4.8736	4.662×10^{-8}	0.1265	723.283	1.6331
KLJADE	1.763×10^{-4}	6.6728	1.689×10^{-7}	0.1589	505.165	1.6345
KL-TAPSO	4.128×10^{-4}	6.0582	4.768×10^{-6}	0.3257	579.668	1.7832



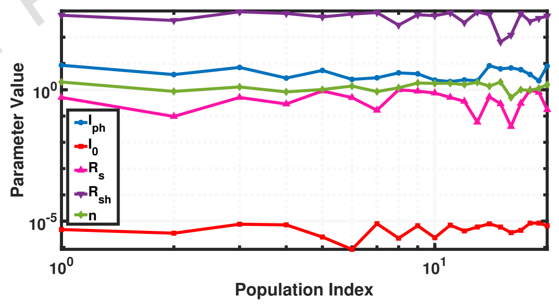
(a) Convergence curve of the LA-ABC algorithm in minimizing the objective function during PV parameter extraction.



(b) Residual plot for the LA-ABC algorithm, indicating uniformly distributed and minimal prediction errors.



(c) Absolute error plot between estimated and experimental I–V values using LA-ABC.



(d) Extracted PV model parameters using the LA-ABC algorithm.

Figure 21: Graphical evaluation of LA-ABC for PV parameter extraction.

and nonlinear problems. In the photovoltaic parameter extraction task, the framework produces highly accurate and physically consistent parameter estimates while maintaining low residual and error values, confirming its practical reliability. The integration of learning-assisted adaptation enables the algorithm to accumulate experience and dynamically guide the colony's search, thereby reinforcing successful search behaviors and reducing ineffective exploration.

Despite these strengths, LA-ABC also presents certain limitations. Its performance can be sensitive to fixed hyperparameters such as the learning rate and training epochs used for the learning module. These parameters, while effective in the present study, may not generalize across all problem domains and often require problem-specific tuning. Additionally, the use of a static learning structure may limit adaptability in highly irregular or hierarchical landscapes, where more flexible learning architectures could better capture problem complexity. The added computational cost introduced by the learning component also poses challenges for real-time or large-scale optimization scenarios.

To address these limitations, future research may explore dynamic or self-adaptive learning mechanisms that adjust hyperparameter automatically during the optimization process. Employing lightweight or incremental training models could reduce computational overhead while maintaining effective learning capacity. Further, hybridizing LA-ABC with complementary metaheuristics or embedding advanced knowledge transfer strategies could expand its applicability to constrained, multimodal, and large-scale optimization problems. Extending the framework toward multitask and time-varying problem domains, as well as leveraging parallel and distributed computing architectures, may further enhance its scalability and practical utility.

In summary, LA-ABC offers a promising advancement in evolutionary computation, combining the simplicity of ABC with adaptive learning-driven guidance. Its strong performance on both benchmark functions and real-world PV modeling tasks highlights its potential for broader adoption and motivates future work on enhancing adaptability, scalability, and domain-specific applications.

7. Conclusion

This study presents the LA-ABC framework, a learning-assisted variant of the ABC algorithm designed to enhance convergence efficiency, adaptability, and robustness across complex optimization landscapes. By incorporating adaptive learning into the ABC framework, LA-ABC systematically exploits past search experiences, balances exploration and exploitation more effectively, and maintains population diversity to mitigate stagnation. The proposed approach preserves the simplicity of ABC while significantly strengthening its capacity to handle nonlinear, high-dimensional, and real-world optimization challenges.

Comprehensive evaluations conducted on standard benchmark functions, the IEEE CEC 2019 benchmark suite, and the challenging task of photovoltaic (PV) model parameter extraction confirm the superiority of LA-ABC over conventional ABC and other recent 11 state-of-the-art metaheuristics (including L-SHADE, JSO, RL variants and KL variants). In the PV modeling problem, LA-ABC demonstrates its effectiveness by accurately estimating critical physical parameters I_{ph}, I_0, R_s, R_{sh}, and n with extracted values consistent with theoretical and practical expectations. The resulting low series resistance, high shunt resistance, and an ideality factor near unity validate the electrical soundness of the obtained model. Furthermore, the algorithm achieves minimal residual and absolute error values, as well as rapid and stable convergence, underscoring its predictive reliability and practical applicability.

Statistical analyses using the Wilcoxon Rank-Sum, Friedman post hoc, Bonferroni-Dunn, and ANOVA tests further confirm the robustness and significance of the performance gains achieved by LA-ABC. Compared to state-of-the-art approaches, LA-ABC consistently attains lower fitness errors and exhibits faster, more stable convergence behavior, making it a reliable and versatile tool for solving both synthetic and real-world optimization tasks.

Looking ahead, future research can extend the LA-ABC framework to address multimodal, multitask, many-objective, and large-scale optimization problems. Investigating adaptive learning structures or lightweight models will help reduce computational costs while maintaining high solution quality. Additionally, embedding self-adaptive parameter tuning strategies and hybridizing LA-ABC with complementary metaheuristics may further enhance its scalability and generalization.

Acknowledgment

The authors are thankful to the Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh, India, for the necessary support for this research.

Compliance with ethical standards

Conflict of interest: All authors declare that they have no conflict of interest.
Ethical approval: This article does not contain studies with human participants or animals performed by any of the authors.

Data Availability

All data genrated or analysed during this study are included in this published article.

Funding Information

The authors did not receive funding for this work.

References

[1] T. Back, Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms, Oxford university press, 1996.

[2] J.-Y. Li, Z.-H. Zhan, J. Zhang, Evolutionary computation for expensive optimization: A survey, Machine Intelligence Research 19 (1) (2022) 3–23.

[3] Z.-H. Zhan, L. Shi, K. C. Tan, J. Zhang, A survey on evolutionary computation for complex continuous optimization, Artificial Intelligence Review 55 (1) (2022) 59–110.

[4] J. H. Holland, Genetic algorithms, Scientific american 267 (1) (1992) 66–73.

[5] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (4) (1997) 341–359.

[6] R. Eberhart, J. Kennedy, Particle swarm optimization, Proceedings of ICNN'95 - International Conference on Neural Networks 2 (1995) 1942–1948.

[7] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, Journal of global optimization 39 (3) (2007) 459–471.

[8] M. Chawla, M. Duhan, Applications of recent metaheuristics optimisation algorithms in biomedical engineering: a review, International Journal of Biomedical Engineering and Technology 16 (3) (2014) 268–278.

[9] A. Arias-Montano, C. A. C. Coello, E. Mezura-Montes, Multiobjective evolutionary algorithms in aeronautical and aerospace engineering, IEEE transactions on evolutionary computation 16 (5) (2012) 662–694.

[10] A. M. Helmi, R. Carli, M. Dotoli, H. S. Ramadan, Efficient and sustainable reconfiguration of distribution networks via metaheuristic optimization, IEEE Transactions on Automation Science and Engineering 19 (1) (2021) 82–98.

[11] K. Rajwar, K. Deep, S. Das, An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges, Artificial Intelligence Review 56 (11) (2023) 13187–13257.

[12] D. Karaboga, et al., An idea based on honey bee swarm for numerical optimization (2005).

[13] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2) (2011) 61–70.

[14] Y. S. Ong, P. B. Nair, A. J. Keane, Evolutionary optimization of computationally expensive problems via surrogate modeling, AIAA journal 41 (4) (2003) 687–696.

[15] Z.-H. Zhan, J.-Y. Li, S. Kwong, J. Zhang, Learning-aided evolution for optimization, IEEE Transactions on Evolutionary Computation 27 (6) (2022) 1794–1808.

[16] J. Zhang, A. C. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Transactions on evolutionary computation 13 (5) (2009) 945–958.

[17] Z.-H. Zhan, Z.-J. Wang, H. Jin, J. Zhang, Adaptive distributed differential evolution, IEEE transactions on cybernetics 50 (11) (2019) 4633–4647.

[18] J. Brest, M. S. Maučec, B. Bošković, Single objective real-parameter optimization: Algorithm jso, in: 2017 IEEE congress on evolutionary computation (CEC), IEEE, 2017, pp. 1311–1318.

[19] F. Lezama, J. Soares, R. Faia, Z. Vale, Hybrid-adaptive differential evolution with decay function (hyde-df) applied to the 100-digit challenge competition on single objective numerical optimization, in: Proceedings of the genetic and evolutionary computation conference companion, 2019, pp. 7–8.

[20] J. J. Liang, L. Guo, R. Liu, B.-Y. Qu, A self-adaptive dynamic particle swarm optimizer, in: 2015 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2015, pp. 3206–3213.

[21] A. Ratnaweera, S. K. Halgamuge, H. C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Transactions on evolutionary computation 8 (3) (2004) 240–255.

[22] X. Xia, L. Gui, F. Yu, H. Wu, B. Wei, Y.-L. Zhang, Z.-H. Zhan, Triple archives particle swarm optimization, IEEE transactions on cybernetics 50 (12) (2019) 4862–4875.

[23] W. Liu, Z. Wang, Y. Yuan, N. Zeng, K. Hone, X. Liu, A novel sigmoid-function-based adaptive weighted particle swarm optimizer, IEEE transactions on cybernetics 51 (2) (2019) 1085–1093.

[24] Y. Jiang, Z.-H. Zhan, K. C. Tan, J. Zhang, Knowledge learning for evolutionary computation, IEEE transactions on evolutionary computation (2023).

[25] K. C. Tan, L. Feng, M. Jiang, Evolutionary transfer optimization-a new frontier in evolutionary computation research, IEEE Computational Intelligence Magazine 16 (1) (2021) 22–33.

[26] X. Xue, C. Yang, L. Feng, K. Zhang, L. Song, K. C. Tan, Solution transfer in evolutionary optimization: An empirical study on sequential transfer, IEEE Transactions on Evolutionary Computation 28 (6) (2023) 1776–1793.

[27] Y. Guo, G. Chen, M. Jiang, D. Gong, J. Liang, A knowledge guided transfer strategy for evolutionary dynamic multiobjective optimization, IEEE Transactions on Evolutionary Computation 27 (6) (2022) 1750–1764.

[28] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep learning, Vol. 1, MIT press Cambridge, 2016.

[29] K. O. Stanley, J. Clune, J. Lehman, R. Miikkulainen, Designing neural networks through neuroevolution, Nature Machine Intelligence 1 (1) (2019) 24–35.

[30] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, R. Qu, Hyper-heuristics: A survey of the state of the art, Journal of the Operational Research Society 64 (12) (2013) 1695–1724.

[31] J.-Y. Li, Z.-H. Zhan, J. Xu, S. Kwong, J. Zhang, Surrogate-assisted hybrid-model estimation of distribution algorithm for mixed-variable hyperparameters optimization in convolutional neural networks, IEEE Transactions on Neural Networks and Learning Systems 34 (5) (2021) 2338–2352.

[32] D. Xu, B. Cao, Adaptive multiobjective evolutionary generative adversarial network for metaverse network intrusion detection, Research 8 (2025) 0665.

[33] E. Galván, P. Mooney, Neuroevolution in deep neural networks: Current trends and future challenges, IEEE Transactions on Artificial Intelligence 2 (6) (2021) 476–493.

[34] Z.-H. Zhan, J.-Y. Li, J. Zhang, Evolutionary deep learning: A survey, Neurocomputing 483 (2022) 42–58.

[35] P. Larrañaga, C. Bielza, Estimation of distribution algorithms in machine learning: a survey, IEEE transactions on evolutionary computation (2023).

[36] V. Mishra, L. Kane, A survey of designing convolutional neural network using evolutionary algorithms, Artificial Intelligence Review 56 (6) (2023) 5095–5132.

[37] J. Zhang, Z.-h. Zhan, Y. Lin, N. Chen, Y.-j. Gong, J.-h. Zhong, H. S. Chung, Y. Li, Y.-h. Shi, Evolutionary computation meets machine learning: A survey, IEEE Computational Intelligence Magazine 6 (4) (2011) 68–75.

[38] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: artificial bee colony (abc) algorithm and applications, Artificial intelligence review 42 (1) (2014) 21–57.

[39] C. Ozturk, D. Karaboga, Hybrid artificial bee colony algorithm for neural network training, in: 2011 IEEE congress of evolutionary computation (CEC), IEEE, 2011, pp. 84–88.

[40] W.-f. Gao, S.-y. Liu, A modified artificial bee colony algorithm, Computers & Operations Research 39 (3) (2012) 687–697.

[41] H. Wang, Z. Wu, S. Rahnamayan, H. Sun, Y. Liu, J.-s. Pan, Multi-strategy ensemble artificial bee colony algorithm, Information Sciences 279 (2014) 587–603.

[42] M. S. Kiran, H. Hakli, M. Gunduz, H. Uguz, Artificial bee colony algorithm with variable search strategy for continuous optimization, Information Sciences 300 (2015) 140–157.

[43] W.-f. Gao, L.-l. Huang, S.-y. Liu, F. T. Chan, C. Dai, X. Shan, Artificial bee colony algorithm with multiple search strategies, Applied Mathematics and Computation 271 (2015) 269–287.

[44] X. Zhou, Z. Wu, H. Wang, S. Rahnamayan, Gaussian bare-bones artificial bee colony algorithm, Soft Computing 20 (2016) 907–924.

[45] S. S. Jadon, R. Tiwari, H. Sharma, J. C. Bansal, Hybrid artificial bee colony algorithm with differential evolution, Applied Soft Computing 58 (2017) 11–24.

[46] S. S. Jadon, H. Sharma, E. Kumar, J. C. Bansal, Application of binary particle swarm optimization in cryptanalysis of des, in: Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011) December 20-22, 2011: Volume 1, Springer, 2012, pp. 1061–1071.

[47] N. Garg, S. S. Jadon, H. Sharma, D. Palwalia, Gbest-artificial bee colony algorithm to solve load flow problem, in: Proceedings of the Third International Conference on Soft Computing for Problem Solving: SocProS 2013, Volume 2, Springer, 2014, pp. 529–538.

[48] X. Zhou, H. Wang, M. Wang, J. Wan, Enhancing the modified artificial bee colony algorithm with neighborhood search, Soft Computing 21 (10) (2017) 2733–2743.

[49] Z. Liang, K. Hu, Q. Zhu, Z. Zhu, An enhanced artificial bee colony algorithm with adaptive differential operators, Applied soft computing 58 (2017) 480–494.

[50] D. Kumar, K. Mishra, Portfolio optimization using novel co-variance guided artificial bee colony algorithm, Swarm and evolutionary computation 33 (2017) 119–130.

[51] S. S. Jadon, J. C. Bansal, R. Tiwari, H. Sharma, Artificial bee colony algorithm with global and local neighborhoods, International Journal of System Assurance Engineering and Management 9 (2018) 589–601.

[52] S. S. Jadon, H. Sharma, R. Tiwari, J. C. Bansal, Self-adaptive position update in artificial bee colony, International Journal of System Assurance Engineering and Management 9 (2018) 802–810.

[53] W.-l. Xiang, X.-l. Meng, Y.-z. Li, R.-c. He, M.-q. An, An improved artificial bee colony algorithm based on the gravity model, Information Sciences 429 (2018) 49–71.

[54] D. Kumar, K. Mishra, Co-variance guided artificial bee colony, Applied Soft Computing 70 (2018) 86–107.

[55] Z. Han, M. Chen, S. Shao, Q. Wu, Improved artificial bee colony algorithm-based path planning of unmanned autonomous helicopter using multi-strategy evolutionary learning, Aerospace Science and Technology 122 (2022) 107374.

[56] D. Özdemir, S. Dörterler, D. Aydın, A new modified artificial bee colony algorithm for energy demand forecasting problem, Neural Computing and Applications 34 (20) (2022) 17455–17471.

[57] Y. Zhang, B. Pang, Y. Song, Q. Xu, X. Yuan, Artificial bee colony algorithm based on dimensional memory mechanism and adaptive elite population for training artificial neural networks, IEEE Access 11 (2023) 107616–107637.

[58] T. Lamjiak, B. Sirinaovakul, S. Kornthongnimit, J. Polvichai, A. Sohail, Optimizing artificial neural network learning using improved reinforcement learning in artificial bee colony algorithm, Applied Computational Intelligence and Soft Computing 2024 (1) (2024) 6357270.

[59] F. Zhu, Z. Shuai, Y. Lu, H. Su, R. Yu, X. Li, Q. Zhao, J. Shuai, obabc: a one-dimensional binary artificial bee colony algorithm for binary optimization, Swarm and Evolutionary Computation 87 (2024) 101567.

[60] J. Yang, W.-T. Li, X.-W. Shi, L. Xin, J.-F. Yu, A hybrid abc-de algorithm and its application for time-modulated arrays pattern synthesis, IEEE Transactions on Antennas and Propagation 61 (11) (2013) 5485–5495.

[61] G. Saini, S. S. Jadon, A comprehensive review of hybrid variants of artificial bee colony algorithm, High-Performance Automation Methods for Computational Intelligent Systems: Challenges, Opportunities, and Applications (2025) 148.

[62] A. Gupta, Y.-S. Ong, L. Feng, Multifactorial evolution: Toward evolutionary multitasking, IEEE Transactions on Evolutionary Computation 20 (3) (2015) 343–357.

[63] F. Zhang, Y. Mei, S. Nguyen, M. Zhang, K. C. Tan, Surrogate-assisted evolutionary multitask genetic programming for dynamic flexible job shop scheduling, IEEE Transactions on Evolutionary Computation 25 (4) (2021) 651–665.

[64] J.-Y. Li, Z.-H. Zhan, K. C. Tan, J. Zhang, A meta-knowledge transfer-based differential evolution for multitask optimization, IEEE Transactions on Evolutionary Computation 26 (4) (2021) 719–734.

[65] G. Yokoya, H. Xiao, T. Hatanaka, Multifactorial optimization using artificial bee colony and its application to car structure design optimization, in: 2019 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 3404–3409.

[66] X. Bian, D. Chen, F. Zou, F. Ge, Y. Zheng, F. Liu, Multitask particle swarm optimization algorithm leveraging variable chunking and local meta-knowledge transfer, Swarm and Evolutionary Computation 92 (2025) 101823.

[67] R. Mallipeddi, P. N. Suganthan, Q.-K. Pan, M. F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Applied soft computing 11 (2) (2011) 1679–1696.

[68] K. Price, N. Awad, M. Ali, P. Suganthan, Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization, in: Technical report, Nanyang Technological University Singapore, 2018.

[69] G. Saini, S. S. Jadon, An improved artificial bee colony algorithm based on spider monkey optimization global search for complex benchmarks and engineering applications, Physica Scripta 100 (7) (2025) 075229.

[70] R. Tanabe, A. S. Fukunaga, Improving the search performance of shade using linear population size reduction, in: 2014 IEEE congress on evolutionary computation (CEC), IEEE, 2014, pp. 1658–1665.

[71] Y. Song, L. Wei, Q. Yang, J. Wu, L. Xing, Y. Chen, Rl-ga: A reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem, Swarm and Evolutionary Computation 77 (2023) 101236.

[72] F. Zhao, Z. Wang, L. Wang, T. Xu, N. Zhu, et al., A multi-agent reinforcement learning driven artificial bee colony algorithm with the central controller, Expert Systems with Applications 219 (2023) 119672.

[73] Y. Cui, W. Hu, A. Rahmani, A reinforcement learning based artificial bee colony algorithm with application in robot path planning, Expert Systems with Applications 203 (2022) 117389.

[74] Y. Li, W. Gong, S. Li, Multitasking optimization via an adaptive solver multitasking evolutionary framework, Information Sciences 630 (2023) 688–712.

[75] T. Zhang, W. Gong, Y. Li, Multitask differential evolution with adaptive dual knowledge transfer, Applied Soft Computing 165 (2024) 112040.