Applied Intelligence
https://doi.org/10.1007/s10489-023-05202-2

A hierarchical learning based artificial bee colony algorithm for numerical global optimization and its applications

Qingke Zhang[1] · Xianglong Bu[1] · Hao Gao[1] · Tianqi Li[1] · Huaxiang Zhang[1]

Accepted: 25 November 2023

Abstract
The Artificial Bee Colony algorithm (ABC) is a swarm intelligence algorithm inspired by honey bee harvesting behavior. It boasts the benefits of minimal parameters and strong exploration capabilities. However, the ABC algorithm is still susceptible to local optima entrapment and lacks consideration of selection probability in the onlooker bee phase, leading to reduced convergence accuracy in later search stages. To address these issues, this paper introduces an enhanced ABC algorithm called Hierarchical Learning-based Artificial Bee Colony (HLABC). Initially, a hierarchical learning approach is devised, dividing the entire population into distinct layers based on solution quality. In this hierarchical approach, bees at lower layers can access much better advantageous information from higher layers. Secondly, the exploitation ability of onlooker bees is enhanced through novel strategies designed based on hierarchical learning. Thirdly, the exploration ability of scout bees is strengthened by implementing an opposition-based learning method. To evaluate the performance of the proposed algorithm, 69 benchmark functions from four benchmark suites (CEC2005, CEC2010, CEC2013 and CEC2022) are used to test the performance of HLABC, along with five variants of the ABC algorithm, The experimental statistical results show that the HLABC algorithm outperforms the ABC algorithm on all test problems with an average winning rate of 89%. Furthermore, to validate the performance of the HLABC algorithm in real-world optimization problems, this paper applies the HLABC algorithm to two practical applications: the deployment of wireless sensor networks (WSNs), the power scheduling problem in a smart home (PSPSH) and the multi-thresholding image segmentation (MIS). The experimental and statistical results demonstrate that HLABC is an efficient and stable optimizer. It shows better or comparable performance compared to other ABC variants when considering the quality of solutions for a suite of benchmark problems and real-world optimization problems. These findings affirm the effectiveness and versatility of the HLABC algorithm in addressing both theoretical and practical optimization challenges.

Keywords Artificial bee colony algorithm · Hierarchical learning · Numerical global optimization · Wireless sensor networks deployment · Power scheduling problem · Multi-thresholding image segmentation

1 Introduction

Optimization problems aim to discover the best possible global solution within feasible solution spaces. As industrial development progresses, numerous complex engineering optimization problems have emerged across various domains. These problems are often characterized by high dimensionality, non-linearity, and an abundance of local optimal solutions. Traditional analytical or numerical optimization methods struggle to obtain the global optimal solution for such problems, and in some cases, they may even be unable to solve them.

In recent years, modern bio-inspired stochastic optimization algorithms, such as Particle Swarm Optimization (PSO) [1], Differential Evolution (DE) [2], Artificial Bee Colony (ABC) [3–5], and Ant Colony Optimization (ACO) [6, 7], have been proposed to tackle these complex problems. These meta-heuristic techniques do not rely on any assumptions about the problem being solved and do not use the function's gradient. As a result, bionic-based intelligent optimization methods are well-suited for solving a wide range of optimization problems and have become a popular area of research.

✉ Qingke Zhang
tsingke@sdnu.edu.cn

[1] School of Information Science and Engineering, Shandong Normal University, Ji'nan 250358, China

Published online: 06 December 2023

Their advantage lies in their adaptability and robustness when dealing with challenging optimization scenarios.

Among these bio-inspired algorithms, the Artificial Bee Colony (ABC) algorithm, proposed by Karaboga in 2005 and successfully applied to numerical optimization [3], draws inspiration from the cyclic process of honey bee colonies in nature, where bees find and abandon honey sources. It has emerged as one of the most effective swarm intelligence algorithms for solving continuous numerical optimization problems. Compared to other swarm intelligence algorithms, ABC boasts two distinct advantages: it requires fewer control parameters and exhibits strong exploration capabilities. As a result, it has found wide application in diverse domains, including distribution network layout optimization, shortest path problems, feature selection, multi-cast path selection, single machine scheduling, pipeline balancing, and image processing [8–19].

However, the original ABC algorithm still faces certain challenges, including its poor exploitation ability and the difficulty of obtaining improved solutions in the later stages of the search process. Additionally, the ABC tends to converge prematurely, and the probability selection in the onlooker bee phase may not continue to effectively guide the population towards the global optimum in the late search stages. These limitations can hinder the algorithm's overall performance and convergence accuracy, particularly in complex optimization scenarios.

To address these challenges, researchers have proposed various variants of the ABC algorithm. Inspired by the Particle Swarm Optimization (PSO) algorithm, Zhu and Kwong incorporated the global best solution into the search equation [20], where the modification of the search equation utilizes information from the global best solution. P. Bhambu et al. improved the global search strategy [21] by adjusting the step size of the solution [22]. Other scholars have also introduced new strategies by modifying the position update equation of the ABC algorithm [23–32].

While the original ABC employs only one search strategy to update solutions, recent approaches have incorporated two or more search strategies to enhance search performance [25, 33, 34]. Wang et al. use three strategies to iterate the population [35], with each solution assigned a strategy at random and dynamically updated to balance exploitability and exploreability. Furthermore, to achieve a better balance between exploration and exploitation, the ABC is often combined with other heuristic algorithms [36–42]. For instance, Jadon et al. combined the DE algorithm with the ABC algorithm [43], employing an onlooker bee phase search strategy that combines direction-guided search and differential evolutionary cross-search. Additionally, Zhang et al. enhanced the exploitation ability of the ABC by combining it with the Teaching-Learning-Based Optimization (TLBO) algorithm [44].

In the ABC algorithm, the bee hive consists of three types of bees: employed bees, onlooker bees, and scout bees, which work collaboratively to evolve the swarm towards the honey source location. However, the search phase of different types of bees is performed independently, lacking effective communication among them. As a consequence, for high-dimensional optimization problems, the ABC algorithm has a higher probability of getting trapped in local optima. Furthermore, in the late search stage of the ABC algorithm, the differences among individuals in the population become less evident, making it difficult for the default roulette selection probability mechanism to filter out dominant individuals effectively. Consequently, the population may experience slow convergence and become stuck in local optima.

To address this issue, this paper proposes a novel variant of ABC by incorporating a hierarchical learning mechanism into the ABC algorithm to enhance the exploration ability of ABC. The inspiration for this framework comes from the characteristic of hierarchical learning in human social behavior. Similar to how human beings acquire knowledge through various channels, such as learning from elite professionals and engaging in peer-to-peer exchanges, our hierarchical learning approach aims to enable the algorithm to acquire more useful and comprehensive knowledge with less computational cost.

Comprehensive experimental results have demonstrated that our proposed method effectively enhances the exploration and exploitation abilities of the original ABC algorithm. To comprehensively evaluate the performance of our proposed algorithm, we conducted experiments on more than fifty classic benchmark problems, including the CEC2005 [45], CEC2010 [46], and CEC2013 [47] benchmark functions. Additionally, to analyze the performance of the proposed algorithm on the latest optimization problems, we utilized the CEC2022 benchmark functions [48] to test the HLABC algorithm and included efficient variants of PSO and DE as competitors. The experimental results consistently show that HLABC outperforms the compared ABC variants on these benchmark problems.

In addition, to verify the performance of HLABC on real-world optimization problems, three sets of simulation experiments are designed in this paper. The experiments apply the HLABC algorithm and five other ABC variants to the deployment of WSNs, PSPSH and multi-thresholding image segmentation.

With the increasing depth of research in computer science, electronic communication, automation, sensing, and other fields, the application of wireless sensor networks (WSN) has also been comprehensively expanded [49]. WSN is mainly composed of a large number of miniature sensors with built-in sensor, processing, and communication units. These sensors can collect data in the monitoring area, fuse

the data, perform communication and collaborative computation through single-hop or multi-hop transmission, and thus achieve perception and monitoring of the area. WSN also has the advantages of low cost, low power consumption, fast deployment, and self-organization. Therefore, it is widely used in industrial monitoring, national security, forest fire detection, agriculture, and other fields. The key technologies of WSN mainly include network security, node localization, network coverage, topology control, data fusion, etc. [50]. These technologies are the foundation for various applications to be implemented. Each technology has its own performance requirements and will also influence each other. In the deployment stage of WSN applications, sensors are often randomly deployed by throwing them into the target area. This results in unknown node location information and monitoring area coverage status beforehand, which makes many applications unable to be implemented. Based on this, in order to better improve the performance of wireless sensor networks, this paper uses the proposed HLABC algorithm to simulate experiments on the coverage of sensor networks in target areas.

In the past 10 years, the intelligence of household appliances has been constantly improved, and most electrical appliances can accomplish tasks automatically, greatly improving users' quality of life [51]. At the same time, this also further increases the demand for electricity. It is estimated that by 2040, the demand for electricity will increase by 56% [52]. However, traditional power grids cannot meet these transmission requirements. Therefore, in order to meet the demand for electricity and further optimize power supply, the concept of smart grids (SGs) has been proposed [53]. SGs serve two primary functions: PSCs deliver electricity to users, and PSCs gather users' electricity usage data to forecast future power consumption patterns and generate sufficient power. Additionally, SGs enhance the dependability and steadiness of PSCs' power systems by enabling users to optimize their power demand during peak hours. This optimization is achieved by managing the operation of household appliances within a time frame based on a dynamic pricing plan and several limits. The dynamic pricing plan, generated by PSCs, varies over time, offering high prices during peak hours and lower prices during off-peak hours. This scheduling process is commonly known as PSPSH [54]. This paper will use the HLABC algorithm to optimize the PSPSH problem.

Image segmentation is a crucial step in image analysis and research. Many techniques have been proposed for image segmentation while preserving image quality, such as region-based, threshold-based, edge-based, and feature-based. However, the thresholding image segmentation is considered the optimal technique because it is easy to use, requires less storage space, is highly accurate, and fast [55]. Thresholding segments pixels into multiple different categories. The main thresholding methods include Otsu's method [56] and Kaptur's entropy method [57]. Otsu's method is suitable for separating regions with pixels of similar intensity, by maximizing the variance between regions. On the other hand, the Kaptur's entropy method tries to find homogeneous regions by maximizing the variance within the image. While all the aforementioned methods effectively propose methods to find the optimal threshold, they cannot obtain the optimal threshold for multilevel thresholding, and their time complexity increases exponentially when the number of thresholds increases significantly. Therefore, new and efficient methods are needed to address the problem of multilevel thresholding, especially for images with a large number of thresholds. Therefore, this paper uses HLABC to optimize multilevel image segmentation problems.

The structure of this paper is organized as follows. Section 2 introduces the principle of ABC and its variants. Section 3 describes the proposed hierarchical learning artificial bee colony algorithm (HLABC). Section 4 shows the test results and related analysis of the proposed algorithm compared with other algorithms. The HLABC algorithm is applied to the deployment of wireless sensor networks, power scheduling problem in a smart home and e multi-thresholding image segmentation in Sections 5, 6 and 7. The conclusion is given in Section 8.

2 Artificial bee colony algorithm

Artificial Bee Colony Algorithm (called ABC) is a new swarm intelligence optimization algorithm proposed by Turkish scholar Karaboga in 2005 [3]. It mainly simulates the intelligent honey-harvesting behavior of bee colonies. There are three types of bee colonies: employed bees, onlooker bees, and scout bees. In the employed bee phase, the employed bee tries to find the best solution in each solution, while maintaining the current best solution. In the onlooker bee phase, the onlooker bee selects a better solution for further search and improves the convergence speed. In the scout bee phase, the scout bees abandon the solutions that have not been updated to avoid getting into local optima. The location of each nectar source represents a solution to the optimization problem, and the nectar content required by the nectar source corresponds to the quality or adaptability of the corresponding solution. The process of ABC algorithm is described into the following parts.

2.1 Population initialization

The initial population is randomly generated by (1) and it consists of SN solutions with D dimension. Among them, the solution is $\mathbf{x}_i = \left[x_{i,1}, x_{i,2}, ..., x_{i,D}\right]$. The upper bound

of the solution is $\boldsymbol{UB} = [UB_1, UB_2, ..., UB_D]$. The lower bound of each solution is $\boldsymbol{LB} = [LB_1, LB_2, ..., LB_D]$. The initial solution is generated by the following formula:

$$x_{i,j} = LB_j + rand \cdot \left(UB_j - LB_j\right) \tag{1}$$

where $i = 1, 2, ..., SN$, $j = 1, 2, ..., D$. $rand$ is a random number between [0,1]. $x_{i,j}$ is the solution of the ith population in the jth dimension.

2.2 Employed bee phase

During the employed bee phase, the employed bee randomly selects an individual based on its position $\mathbf{x}_i$ to produce a new candidate solution.

$$v_{i,j} = x_{i,j} + \varphi_{i,j} \cdot \left(x_{i,j} - x_{k,j}\right) \tag{2}$$

where $v_{i,j}$ is the new solution generated by the ith individual in the jth dimension. $\varphi_{i,j}$ is a random number generated in the range [-1,1]. $x_{k,j}$ is a randomly selected solution in the whole population, and $i \neq k, k = 1, 2, ..., SN$.

According to the greedy selection strategy as shown in (3), when the fitness of $\mathbf{v}_i$ is better than $\mathbf{x}_i$, the solution $\mathbf{x}_i$ will be replaced by $\mathbf{v}_i$, otherwise, $\mathbf{x}_i$ will be reserved in the next iteration.

$$\mathbf{x}_i = \begin{cases} \mathbf{v}_i & if\ f(\mathbf{v}_i) < f(\mathbf{x}_i) \\ \mathbf{x}_i & otherwise \end{cases} \tag{3}$$

According to (4), when the update succeeds, the value $trail_i$ is initialized to 0. Otherwise, it will be increase by 1.

$$trail_i = \begin{cases} 0 & if\ f(\mathbf{v}_i) < f(\mathbf{x}_i) \\ trail_i + 1 & otherwise \end{cases} \tag{4}$$

2.3 Onlooker bee phase

After receiving nectar information from the employed bees, the onlooker bee will adopt the roulette selection method based on the following selection probability in (5) to choose a target employed been for its location updates.

$$p_i = \frac{fit_i}{\sum_1^{SN} fit_i} \tag{5}$$

where, the $fit(x_i)$ denotes the fitness value of the ith solution, which is generated according to (6).

$$fit(x_i) = \begin{cases} \frac{1}{1+f(\mathbf{x}_i)} & if\ f(\mathbf{x}_i) \geq 0 \\ 1 + |f(\mathbf{x}_i)| & ohterwise \end{cases} \tag{6}$$

Where $f(\mathbf{x}_i)$ is the optimized objective function. Each onlooker bee selects a better solution based on the selection probability p_i. Then, a new solution is generated based on (2). The new offspring generated are retained or abandoned according to the fitness by (3).

2.4 Scout bee phase

If a solution is not updated after several iterations, then the employed bee or the onlooker bee will abandon the current solution. A new solution $\mathbf{v}_i$ is randomly generated to replace the original solution $\mathbf{x}_i$ by (1).

$$\mathbf{v}_i = \begin{cases} \boldsymbol{LB} + rand \cdot (\boldsymbol{UB} - \boldsymbol{LB}) & trial_i \geq L_i \\ \mathbf{x}_i & otherwise \end{cases} \tag{7}$$

Where, $rand$ is a randomly generated real number between [0,1]. $[\boldsymbol{LB}, \boldsymbol{UB}]$ is the boundary limit of the value. $trail_i$ is the number of repeated iterations of the ith nectar source. L_i is a pre-defined control parameter that records the number of failed solution iterations.

3 Hierarchical learning artificial bee colony

In this section, a novel ABC algorithm using a special designed hierarchical learning (HLABC) strategy is proposed. The main contribution of HLABC can be summarized as the follows: Firstly, a hierarchical learning topology is proposed. It divides the entire population into three layers according to the fitness of each individual in the population. Secondly, two novel search strategies in the employed bee phase and onlooker bee phase are constructed to balance the search performance of exploration and exploitation. Finally, the hierarchical learning and opposition-based learning [58, 59] is adopted in the scout bee phase to avoid search stagnation and enhance the diversity of the population. The details of the algorithm are described below.

3.1 Principle of Hierarchical Learning

In original ABC algorithm, the roulette selection strategy is used to select one well-adapted individual from current population for onlooker bee learning. However, in the late stage iteration, the value of the objective function obtained is so small that it is often estimated to be zero by the computer system for the minimum optimization problem. For example, there are two very small objective function values $f(\mathbf{x}_1)$=1.0E-30, $f(\mathbf{x}_2)$=1.0E-40 in the late iteration.Thus, the fitness results calculated by formula (6) are almost identical due to the small values of the function of individuals in the population. When applying (6), the result of the fitness calculation is infinitely close to 1. Therefore, the traditional probabilistic selection method based on fitness (roulette method) is difficult to find the excellent individuals. However,

in fact, those two potential solutions are clearly in different orders of magnitude. Thus, the roulette strategy is not effectively distinguish them according to such fitness values. Therefore, the algorithm suffers from the problem of search stagnation or getting stuck in a local optimum. To tackle this problem, [32] and [60] proposed a new neighborhood selection search strategy instead of the original probabilistic selection method. In [61], a new hybrid search strategy is used to replace the original roulette selection strategy. In [62], the onlooker bee phase of the ABC method was replaced by the CMA-ES method. These experimental results also demonstrate the feasibility of such idea.

In this paper, a new hierarchical learning framework based method is proposed to replace the original probability selection mechanism [63]. Briefly, in the first step, the whole population are divided into three layers according to the fitness of each individual. The number of individuals contained in each layer are represented as S_1, S_2, and S_3 respectively. The size of each level should satisfy the basic condition that $S_1 + S_2 + S_3 = SN, S_1 \geq 1, S_2 \geq 1, S_3 \geq 1$, where SN denotes the total size of the search population. Specially, the number of individuals contained in each layer can be determined or varied by the problem being solved. Then, the bees in the outer layer will learn from the bees in the inner layer during the search process and continuously improve the quality of their own solutions. In addition, the method does not require additional calculation of selection probabilities for each individual, and it also effectively reduces the risk of search stagnation and falling into local optima. To provide a more intuitive understanding of hierarchical learning, Fig. 1 shows a conceptual diagram of the hierarchical model.

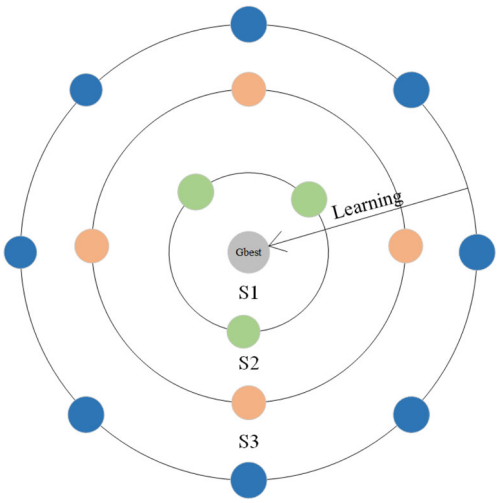


Fig. 1 Hierarchical learning model (outer layer learning from inner layer,S1, S2, S3 are the elite, middle and inferior layers respectively)

3.2 Modified onlooker bee phase

Based on hierarchical learning, a modified search strategy in the employed bee phase is proposed by replacing the randomly unique selected individual x_k in the original ABC with three individuals as x_{s_1}, x_{s_2}, x_{s_3} from different layer. In this way, the employed bees no longer search aimlessly in the global scope, but progressively learn from each other according to the differences between different hierarchies. The detailed search equations are modified as shown below.

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot \left(x_{s_2,j} - x_{s_3,j}\right) + \varphi_{i,j} \cdot \left(x_{gbest,j} - x_{s_1,j}\right) \quad (8)$$

where, x_{s_1}, x_{s_2}, x_{s_3} are represented three different individuals randomly selected from the first layer s_1, the second layer s_2 and the third layer s_3. The weighting factor $\phi_{i,j}$ is a random number generated in the range [-1,1]. $\varphi_{i,j}$ is a random number between [0,C]. According to [20], the value of this parameter C is suggested to be set to 1.5. $x_{gbest,j}$ denotes the jth dimension of a global best solution. Figure 2 shows the modification of the search equation for the employed bee phase. Combining with (8), a new solution $\mathbf{v}_i$ is generated. The better solution in $\mathbf{v}_i$ and $\mathbf{x}_i$ is selected according to (3) to replace the original $\mathbf{x}_i$.

The onlooker bees provide better solutions for the employed bees based on the probability selection. However, in the late iteration of the algorithm, the differences in fitness values become smaller and smaller, and the probability of onlooker bee selection is almost the same among the popu-

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot (x_{s2,j} - x_{s3,j}) + \varphi_{i,j} \cdot (x_{gbest,j} - x_{s1,j})$$
$$v_{i,j} = r_1 \cdot x_{i,jr} + r_2 \cdot (x_{s2,j} - x_{s3,j}) + r_3 \cdot (x_{gbest,j} - x_{s1,r})$$

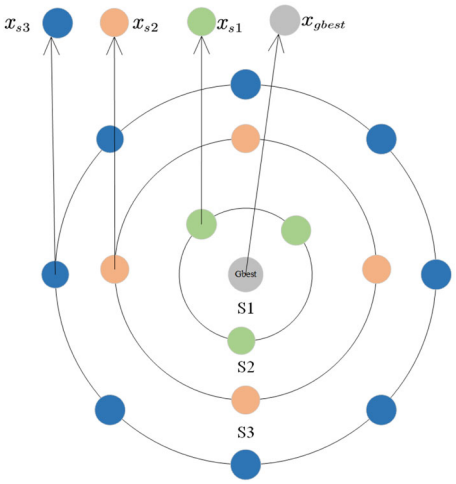


Fig. 2 The novel onlooker bee search strategy based on the hierarchical learning.(S1, S2, S3 are the elite, middle and inferior layers respectively)

lation. Therefore, the same selection probability causes the algorithm to lose its original function and leads to a slow convergence rate. To overcome this drawback, this paper proposes a new search strategy by combining hierarchical learning with the improved operator in MGABC [64] and abandoning the original probability selection mechanism. The mathematical formulation of the updated strategy is shown in (9) :

$$v_{i,j} = r_1 \cdot x_{i,j} + r_2 \cdot \left(x_{s_2,j} - x_{s_3,j}\right) + r_3 \cdot \left(x_{gbest,j} - x_{s_1,j}\right) \quad (9)$$

where, the parameters r_1, r_2, and r_3 are generated randomly in the range [0,1] and $r_1 + r_2 + r_3 = 1$. Besides, the $\mathbf{x}_{s_1}$, $\mathbf{x}_{s_2}$, $\mathbf{x}_{s_3}$ are three mutual different individuals chosen randomly from different layers, and $x_{gbest,j}$ denotes the jth dimension of the global best solution in current iteration.

3.3 Modified scout bee phase

During the search of the solution space, after all the employed and onlooker bees complete the search, the ABC algorithm will check if there is a poor quality nectar source in the population that needs to be abandoned in the scout bee phase. In order to decide whether a certain solution (honey source) needs to be abandoned, a hyperparameter named $limit$ is set in the algorithm, which is mainly used to count the number of failed position updates by the bees during the historical search. When the number of consecutive failures exceeds the $limit$, the scout bee will randomly re-generate a new nectar source by (1) to replace the current secret nectar source $\mathbf{x}_i$. However, as the number of iterations increases, the population is converging and the search range become smaller. If the global search continues to generate a new solution randomly in the population according to (1), then its convergence rate will decrease.

To tackle this problem, inspired by [32], three candidate solutions m_1, m_2 and m_3 are used to enhance the global convergence during the scout bee phase, and the new solution is chosen from m_1, m_2 and m_3 to replace the abandoned solution $\mathbf{x}_i$.

For the solution m_1, it is randomly generated according to (1) as same as the original ABC [3].

For the solution m_2, it is generated based on solutions $\mathbf{x}_{s_1}$, $\mathbf{x}_{s_2}$, $\mathbf{x}_{s_3}$ randomly selected in three different levels, and the detailed formula is provided in the following equation:

$$m_{2,j} = x_{s_1,j} + \phi_{i,j} \cdot \left(x_{s_2,j} - x_{s_3,j}\right) \quad (10)$$

where $j = 1, 2, 3, ..., D$, $\mathbf{x}_{s_1}$,$\mathbf{x}_{s_2}$,$\mathbf{x}_{s_3}$ denote three randomly selected individuals in the first layer, second layer and third layer respectively. $\phi_{i,j}$ is a random number generated in the interval [-1,1].

For the solution m_3, it is generated by an opposition-based learning approach. As reported in literatures [32, 58] that the opposition-based learning approach in some degree can help to find a better candidate solutions and prevent the population from trapping into local optima. In the iterative process of the algorithm, if the historical best solution $\mathbf{x}_{gbest}$ searched by the current population is not the global optimal solution of the problem to be solved, then the opposition-based method for the current optimal solution $\mathbf{x}_{gbest}$ can help the population to jump out of the local optimum and improve the global convergence. Thus, in the HLABC algorithm, the opposition-based learning approach are used to generate a new solution based on $\mathbf{x}_{gbest}$. The details of the formula are shown below in (11):

$$m_{3,j} = LB_j + UB_j - x_{gbest,j} \quad (11)$$

where, $j = 1, 2, 3, ..., D$, UB_j and LB_j is the upper and lower boundaries of the problem to be optimized, which are defined according to the problem. Figure 3 illustrates the process of opposition-based learning, where $x_{gbest,j}$ and $m_{3,j}$ are different solution on the jth dimension. As shown in Fig. 3, the space distance of $\left|x_{gbest,j} - LB_j\right|$ and $\left|UB_j - m_{3,j}\right|$ are equal. In the search spaces, the $x_{gbest,j}$ and $m_{3,j}$ are in symmetric and opposite positions. When the generation of the best solutions stagnates, the method can help population to find better solutions.

When all the candidate solutions m_1, m_2 and m_3 have been generated, then the greedy selection method is used to choose one of the best solutions among m_1, m_2 and m_3 to replace the abandoned solution $\mathbf{x}_i$.

3.4 The Framework of HLABC

As mentioned above, the proposed HLABC algorithm contains three modifications: 1) A novel hierarchical learning framework is designed for the better individual selection instead of the traditional roulette wheel selection; 2) Two novel learning strategies in employed bee phase and onlooker bee phase are introduced based on the proposed hierarchical learning topology,and help the population to search further in the neighborhood; 3) A greedy selection method based on more candidate solutions is adopted in the scout bee phase and in some degree it can prevent the population from trapping in local optima. The pseudocode and

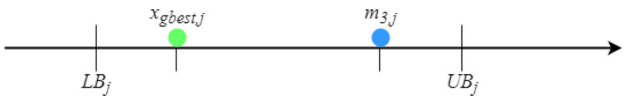


Fig. 3 Opposition-based learning: the global best solution $x_{gbest,j}$ filled in green color and its opposing solution $m_{3,j}$ filled in blue color in the jth dimension

Algorithm 1 HLABC Algorithm.

Input: Swarm size SN, maximum of failure times $limit$, serch space $[LB, UB]$, maximum of fitness evluations $MaxFEs$
Output: $x_{gbest}, f(x_{gbest})$
/* Initialization */
Produce SN initial solutions randomly by (1);
Compute their objective function values;
while $FEs < MaxFEs$ **do**
  Sort existing populations;
  /* Employed bee phase */
  for $i = 1$ to SN **do**
    Choose the global best solution x_{gbest};
    Selected three different individuals x_{s_1}, x_{s_2} and x_{s_3} randomly in s_1, s_2 and s_3;
    Generate $\mathbf{V}_i$ by (8);
    Calculate the objective function value of $\mathbf{V}_i$, and $FEs = FEs + 1$;
    Update $\mathbf{x}_i$ by (3);
    Update $trail_i$ by (4);
  end
  /* Onlooker bee phase */
  for $i = 1$ to SN **do**
    Choose the global best solution x_{gbest};
    Selected three different individuals x_{s_1}, x_{s_2} and x_{s_3} randomly in s_1, s_2 and s_3;
    Generate $\mathbf{V}_i$ by (9);
    Calculate the objective function value of $\mathbf{V}_i$,and $FEs = FEs + 1$;
    Update $\mathbf{x}_i$ by (3);
    Update $trail_i$ by (4);
  end
  /* Scout bee phase */
  for $i = 1$ to SN **do**
    if $\max trial_i > limit$ **then**
      Generate M1 by (1);
      Generate M2 by (10);
      Generate M3 by (11);
      Calculate the objective function values of M1, M2 and M3;
      Choose the best value between M1, M2, M3 to replace the abandoned $\mathbf{x}_i$
    end
  end
end

flowchart of the HLABC algorithm is shown in Algorithm 1 and Fig. 4, respectively, where the symbol FEs denotes the number of fitness evaluations of current objective function and $MaxFEs$ is the maximum value of FEs.

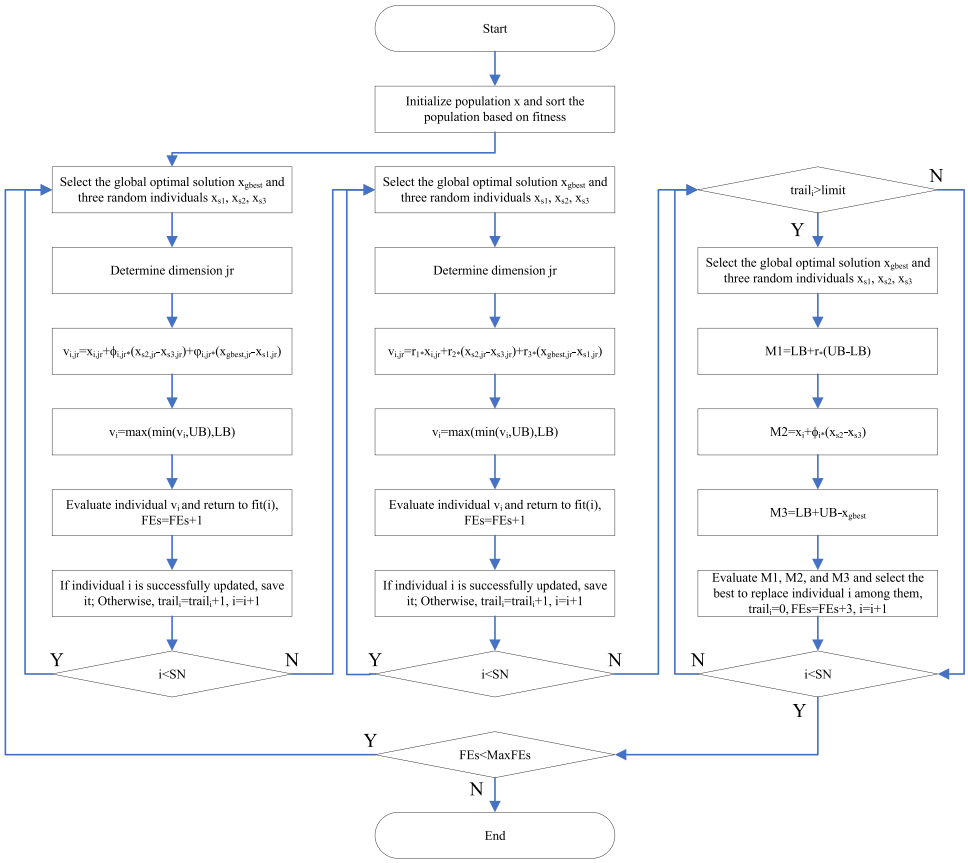
4 Experimental results and discussion

In this section, in order to evaluate the performance of the proposed HLABC algorithm, comprehensive experiments are conducted on a series of international benchmark optimization suites including CEC2005, CEC2010, and CEC2013 on different dimensionality. In addition, the experimental results of HLABC are also compared with the state-of-the-art ABC variants. The details are described below.

4.1 Benchmark functions

To verify the performance and efficiency of HLABC algorithm, a total of 57 benchmark functions from different benchmark suites are tested. At the beginning, 22 well-known benchmark functions including 8 unimodal functions, and 14 multimodal functions collected from the original paper in [45] are used to evaluate the performance of HLABC itself with different parameter settings. Among these functions f_1-f_8 are unimodal problems with only one global optimal, and the types of such functions can be used to study the local search ability of the algorithm. The functions f_9-f_{22} are the multimodal functions with more than two optimum, and they can be used to evaluate the global exploration ability of an optimizer. An overview of those benchmark functions is shown in Table 1.

Fig. 4 The flowchart of HLABC



Then, those functions together with other two CEC benchmark functions include CEC2010 and CEC2013 which are proposed in the IEEE CEC2010 and CEC2013 special sessions on large-scale global optimization respectively are used to evaluate the global search ability of HLABC and its peer compared algorithms. The CEC2010 test suites contains 20 functions. The characteristics and specific form of these functions can be referred to [46]. The CEC2013 functions contains 15 functions and are classified into five categories according to the relationship between decision variables [47]. The CEC2013 test suites are complex and difficult to solve than CEC2010, thus, it in some degree can better reflect the real performance of the compared ABC variants on optimizing the high-dimensional optimization problems.

4.2 Analysis of the parameter settings in HLABC

According to the previous contents, it's known that in HLABC algorithm the individuals in the population are divided into three layers according to the individual's fitness, and these individuals showed a hierarchical distribution. The number of individuals contained in each layer varies. In general, the majority of individuals are mainly distributed in the second layer s_2. While the first layer s_1 stores the better-adapted individuals than in the layer s_2. In contrast, the third layer s_3 contains the less-adapted individuals than in s_2. All the individuals distributed in different layers together constitute the population.

In this subsection, we will analyze the impact of different numbers of individuals in different layers on the performance of the proposed algorithm. Without loss of generality, the size of the population is uniformly set to 100 in the experimental tests. For clarity, the following eight typical proportion settings of the number of individuals between different layers is shown in Table 2. Then, we analyze the effect of different individual distributions in each layer on CEC2005 test suites. The other parameter settings are set the same as SN=50, D=30, $MaxFEs$=15000, $limit$=100. For a fair comparison, the HLABCs with different proportion settings are independently run 30 times respectively.

The experimental results for different combinations of proportion settings are shown in Table 3, where the performance metric $mean$ and std acquired by computing the average global best fitness in 30 runs are used to measure the performance of different proportion settings. The best results for each test function are highlighted in boldface. The smaller means and standard deviations imply that the algorithm can achieve much better overall quality with more stably. Besides, to reflect the experimental results more intu-

Table 1 The properties of benchmark functions

Probelm name	Range	Minimum
Sphere (f_1)	[-100,100]	0
Elliptic (f_2)	[-100,100]	0
SumSquare (f_3)	[-10,10]	0
Schwefel 1.2 (f_4)	[-100,100]	0
Schwefel 1.2 with noise (f_5)	[-10,10]	0
Schwefel 2.21 (f_6)	[-100,100]	0
Schwefel 2.22 (f_7)	[-10,10]	0
Step (f_8)	[-100,100]	0
Quartic (f_9)	[-1.28,1.28]	0
Rosenbrock (f_{10})	[-2,2]	0
Griewank (f_{11})	[-600,600]	0
Schwefel 2.26 (f_{12})	[-500,500]	-418.98289·D
Rastrigin (f_{13})	[-5,5]	0
NCRastrigin (f_{14})	[-5.12,5.12]	0
Ackley (f_{15})	[-50,50]	0
Penalized1 (f_{16})	[-100,100]	0
Penalized2 (f_{17})	[-100,100]	0
Bohachevsky2 (f_{18})	[-100,100]	0
Alphine (f_{19})	[-10,10]	0
Levy (f_{20})	[-10,10]	0
Weierstrass (f_{21})	[-5,5]	-D
Michalewicz (f_{22})	[0,π]	-9.66015

itively, we counted the average ranking of the compared HLABCs and listed it in the last row of Table 4. According to the experimental statistics, it is obvious that the convergence accuracy of the HLABC algorithm differs from each other when the population are distributed in different proportions for each layer. In addition, when the number of outer layers s_3 is significantly more or less than the number of inner layers s_1, the average convergence accuracy of the algorithm is poor, and none of them can reach the optimal result. However, when the ratio of individuals within the population between s_1, s_2 and s_3 is close to 1:7:2, the algorithm can achieve relatively better results for all tested functions in general. Therefore,

Table 2 The proportion of the number of individuals among different layers (%)

Algorithm	s_1	s_2	s_3
HLABC-case-1	5%	60%	35%
HLABC-case-2	5%	70%	25%
HLABC-case-3	10%	60%	30%
HLABC-case-4	10%	70%	20%
HLABC-case-5	15%	60%	25%
HLABC-case-6	15%	70%	15%
HLABC-case-7	20%	60%	20%
HLABC-case-8	20%	70%	10%

the HLABC with a proportion of 1:7:2 among the layers is as default set in this paper.

4.3 Parameter settings for compared algorithms

In this paper, we compared HLABC with the basic ABC algorithm and other efficient ABC variants to further evaluate the performance of HLABC. The other ABCs are listed as follows:

- The basic ABC(ABC) [3]
- The global best guided ABC(GABC) [20]
- Multi-strategy ensemble of ABC (MEABC) [35]
- Hybrid ABC-TLBO(ABC-TLBO) [61]
- Multi-elite guidance ABC(MGABC) [64]
- Hierarchical learning ABC (the proposed HLABC)

To have a fair comparison, the parameters for the compared algorithms are set according to their original papers. The population size SN is set to 50 and the solution quality supervision threshold $limit$ is set to 100. The parameter C associated with ϕ_i in (8) is set to 1.5 according to [20]. The proportion of the number of individuals in s_1, s_2, and s_3 is set to 1:7:2. Besides, the hyperparameters in GABC, MEABC, ABC-TLBO, MGABC are set according to the corresponding literatures in [20, 35, 61], and [64], respectively. The detailed hyperparameter settings are shown in Table 5. In the table, φ represents the coefficient of the individual update formula in the Employed Bee Phase of the ABC algorithm; C is the upper limit of the random coefficient in the GABC and MEABC algorithms; In the ABC-TLBO algorithm, r and ϕ are the coefficients of the individual update formula and MR and CR are the probabilities of selecting different search strategies; In MGABC, q is used to control the number of elite individuals, p is introduced to control the probability of using the modified neighborhood search operator, and MR is used to control the frequency of perturbation. For each tested problem, all ABCs need to run 30 times independently.

4.4 Comparison and analysis of optimization results

In this section, we compared the proposed algorithm on a set of 57 benchmark functions, including 22 commonly used unimodal, multimodal functions, and 35 additional functions from the CEC2010 and CEC2013 test suites. Both quantitative and qualitative validation metrics are utilized for the comparison. The quantitative metrics include the mean and standard deviation(std) values of the different test functions, and the qualitative measures include convergence curves and statistical box diagrams. Figures 5, 6, 7 show the convergence curves of the average fitness values of different algorithms. Specifically, the optimal solutions found by each genera-

Table 3 Performance of HLABC algorithms using different proportions settings among the hierarchical layers

Probelm name	HLABC-case-1 Mean	HLABC-case-2 Mean	HLABC-case-3 Mean	HLABC-case-4 Mean	HLABC-case-5 Mean	HLABC-case-6 Mean	HLABC-case-7 Mean	HLABC-case-8 Mean
Sphere (f_1)	4.31E-83	4.35E-77	**1.14E-86**	2.99E-82	3.20E-81	7.09E-71	4.28E-79	6.20E-70
Elliptic (f_2)	1.72E-67	1.05E-76	1.88E-68	1.67E-73	3.15E-64	1.39E-75	**2.19E-77**	4.25E-69
SumSquare (f_3)	3.19E-83	1.26E-82	2.37E-89	2.58E-83	**7.64E-90**	2.28E-78	8.36E-84	1.82E-77
Schwefel 1.2 (f_4)	**2.74E-87**	2.43E-83	7.20E-80	1.73E-78	3.88E-85	6.09E-69	1.20E-83	2.53E-63
Schwefel 1.2 with noise (f_5)	2.22E-39	1.40E-42	3.04E-34	2.45E-41	**3.63E-44**	5.85E-33	1.38E-32	3.53E-42
Schwefel 2.21 (f_6)	1.73E+00	1.17E+00	1.32E+00	**1.04E+00**	1.06E+00	1.82E+00	1.43E+00	1.06E+00
Schwefel 2.22 (f_7)	**2.66E-53**	1.23E-51	3.62E-52	2.23E-47	3.20E-51	1.01E-46	3.20E-48	1.27E-44
Step (f_8)	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Quartic (f_9)	2.67E-46	8.51E-41	**6.85E-58**	2.52E-47	2.14E-37	1.03E-31	2.31E-43	5.02E-32
Rosenbrock (f_{10})	**1.42E+00**	1.88E+00	1.96E+00	1.49E+00	1.63E+00	1.56E+00	1.50E+00	1.51E+00
Griewank (f_{11})	**0.00E+00**	**0.00E+00**	0.004109334	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Schwefel 2.26 (f_{12})	**-12569.5**	**-12569.5**	**-12569.5**	**-12569.5**	**-12569.5**	**-12569.5**	**-12569.5**	**-12569.5**
Rastrigin (f_{13})	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
NCRastrigin (f_{14})	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Ackley (f_{15})	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Penalized1 (f_{16})	3.11E-35	3.46E-36	1.22E-35	1.30E-34	2.36E-35	2.36E-34	1.69E-34	**2.35E-36**
Penalized2 (f_{17})	2.36E-34	1.13E-34	2.33E-35	1.26E-34	4.89E-35	1.06E-33	1.02E-34	**1.03E-35**
Bohachevsky2 (f_{18})	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Alphine (f_{19})	8.07E-04	2.97E-05	3.30E-05	**1.80E-06**	2.23E-04	1.66E-05	1.89E-06	2.36E-06
Levy (f_{20})	2.70E-32	2.02E-32	2.23E-32	**1.12E-32**	2.41E-32	2.33E-32	1.22E-32	1.13E-32
Weierstrass (f_{21})	**2.49E-02**	5.27E-02	2.62E-02	5.58E-02	8.10E-02	3.90E-02	4.50E-02	4.80E-02
Michalewicz (f_{22})	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**	**-2.97E+01**
Friedman mean rank	3.09	3.09	2.82	**2.77**	3.45	4.14	2.95	3.27
Rank	4	4	2	1	7	8	3	6

Table 4 The Friedman mean Rank of HLABC with different proportions

Algorithm	Mean rank
HLABC-case-1	3.09
HLABC-case-2	3.09
HLABC-case-3	2.82
HLABC-case-4	**2.77**
HLABC-case-5	3.45
HLABC-case-6	4.14
HLABC-case-7	2.95
HLABC-case-8	3.27

tion of the algorithm are recorded and plotted as curves. This image can help readers more intuitively understand the optimization capabilities and convergence speed of different algorithms. Figure 8 shows the average ranking results of the algorithm on different problems using a histogram, which better demonstrates the comprehensive performance gap betweeen algorithms. In addition, Figs. 9, 10 and 11 show statistical box plots of the optimal solutions output by the algorithm multiple times, which can help readers understand the stability exhibited by the algorithm when optimizing problems.

4.4.1 Experimental results and discussions on CEC2005 test suite

First,the performance of these ABC variants is compared on the 22 benchmark functions on CEC2005 test suites. The dimension D is set to 30 and 100, respectively. The maximum number of fitness evaluations for each algorithm are all set to $MaxFEs = 50 * 10^4$.

Table 6 shows the average global best fitness obtained by HLABC and the other five ABC variants on solving the 30-dimensional problems. For clarity, the best solutions are highlighted in bold. The comparison results show that HLABC outperforms the original ABC and GABC on all problems. For function f_8, the performance of HLABC and MEABC is comparable. However, for the remaining problems, HLABC outperforms MEABC. HLABC achieves the same results as MEABC on 4 problems. On the remaining 18 problems, HLABC outperforms MEABC. Among the five ABC algorithms other than HLABC, ABC-TLBO is the better performing algorithm. ABC-TLBO achieves the same results as HLABC on 4 problems and achieves better results than HLABC on f_{10} and f_{16}. On the remaining problems, HLABC outperforms ABC-TLBO. MGABC performs similarly to ABC-TLBO, achieving the same results as HLABC on 2 problems and achieves better results than HLABC on f_{12} and f_{17}. For the remaining problems, HLABC is better than MGABC. The experimental results clearly show that HLABC has a strong global search capability on unimodal and multimodal functions.

Table 5 Hyperparameter settings for different ABC variants

Algorithm	Hyperparameters	Value
ABC	φ	[-1,1]
GABC	C	1.5
MEABC	C	1.5
ABC-TLBO	r	[0,1]
	ϕ	[0,1]
	MR	[0,1]
	CR	[0,1]
MGABC	q	0.1
	p	0.1
	MR	0.5
HLABC	s_1:s_2:s_3	1:7:2

Table 7 shows the CEC2005 comparison results of the six ABC variants on 100 dimensions. When the search dimension of the problem is increased from 30 to 100, HLABC achieves better results in general than the basic ABC on all problems except f_8. For function f_8, all algorithms get the similar best results. For function f_{10}, HLABC is the second-best algorithm after GABC. On f_8, f_{13}, and f_{14}, all five ABCs achieve the same global best solutions. On other problems, HLABC achieves better solutions than GABC. For f_{11} and f_{12}, MEABC, ABC-TLBO, MGABC, and HLABC all achieve globally the best solutions. ABC-TLBO achieves better solutions than HLABC on f_{17} and f_{20}. HLABC outperforms ABC-TLBO on 9 problems. MGABC achieves better results than HLABC on f_{17}. For f_{16}, f_{18} and f_{22}, ABC-TLBO, MGABC, and HLABC totally achieve the best solutions. From the experimental results, it can be seen that the convergence accuracy of the algorithm decreases as the dimension increases. However, HLABC still finds the best solution on most of these problems.

Figure 5 gives the convergence curves of the six types of ABCs on all problems in 30 dimensions. In f_1, f_3, f_4, f_6, f_7, f_9, and f_21, HLABC converges the fastest and the best solutions are achieved. In f_{19}, the convergence speed of ABC-TLBO in the early phase is comparable to that of HLABC. As the number of iterations increases, the HLABC algorithm is faster than the other ABCs in the final search phase. For f_{21}, the convergence speed of HLABC starts to increase explosively in the late iterations. As can be seen from the figure, HLABC has good convergence speed and performance.

4.4.2 Experimental results and discussions on CEC2013 test suite

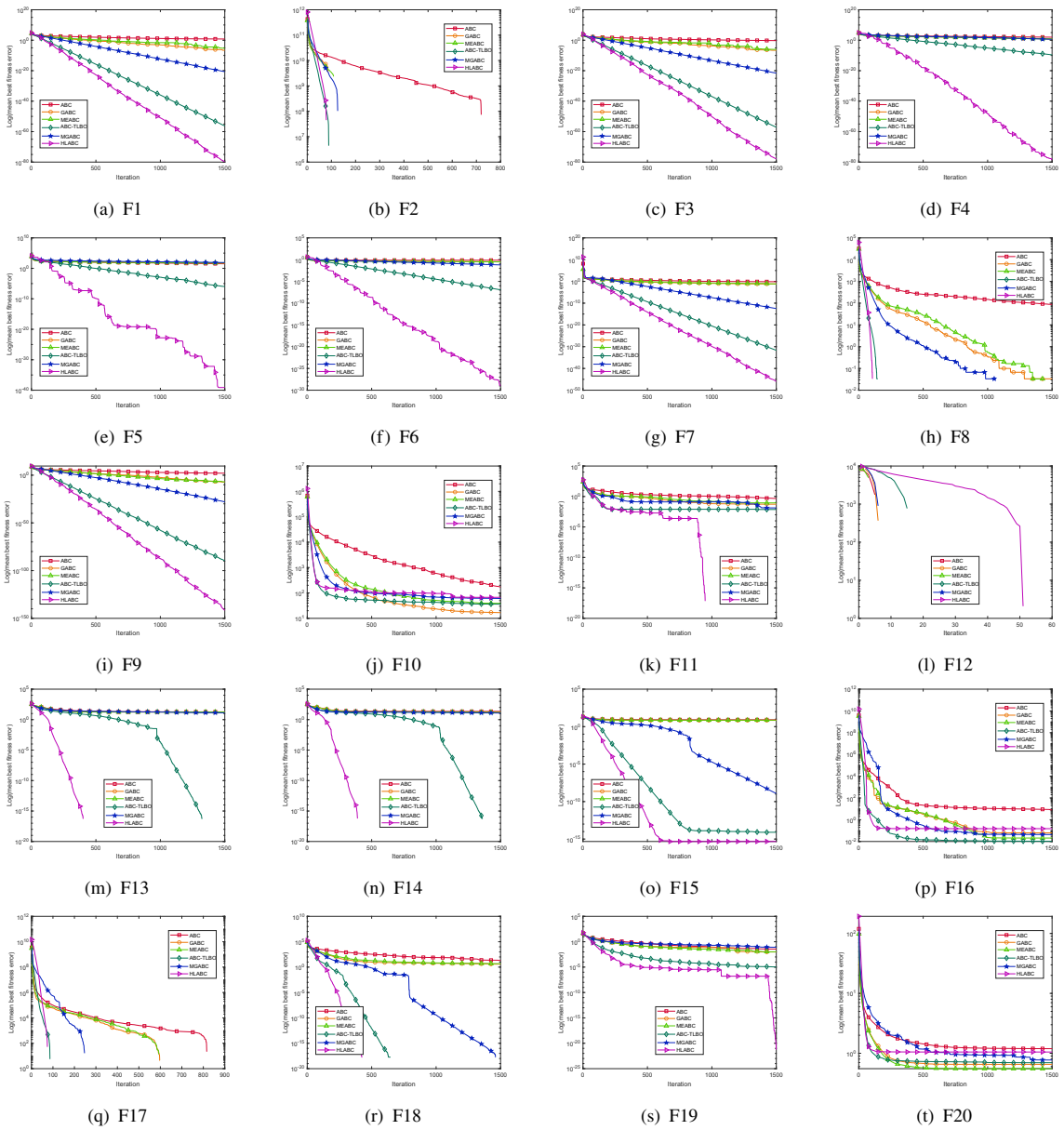
To further verify the efficiency and effectiveness of HLABC, we then compare it with five efficient ABC variants on the IEEE CEC2013 test suite with 20 functions. Among

Table 6 The comparison results of HLABC with other algorithms on CEC2005 with 30 Dimensional problems

Problem name	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Sphere (f_1)	5.22E+00±1.05E+00	4.78E-06±2.00E-05	2.31E-05±3.67E-05	3.88E-57±4.06E-56	8.69E-22±3.44E-21	**2.99E-80±3.10E-80**
Elliptic (f_2)	-1.54E+302±6.55E+04	6.55E+04±0.00E+00	6.55E+04±0.00E+00	6.55E+04±0.00E+00	6.55E+04±0.00E+00	**-2.83E+172±6.55E+04**
SumSquare (f_3)	1.49E-01±6.23E-01	1.80E-07±2.12E-07	7.84E-07±1.34E-04	3.98E-58±1.14E-57	1.23E-22±4.80E-22	**3.33E-83±4.02E-84**
Schwefel 1.2 (f_4)	9.98E+01±6.26E+01	7.14E+00±5.78E+00	7.69E+00±4.92E+00	2.61E-10±5.10E-10	4.04E+00±2.79E+00	**3.22E-79±1.76E-78**
Schwefel 1.2 with noise (f_5)	3.18E+01±1.40E+01	5.10E+01±2.72E+01	6.00E+01±2.84E+01	1.05E-06±2.68E-06	7.96E+01±4.53E+01	**7.52E-40±4.08E-39**
Schwefel 2.21 (f_6)	7.32E-01±1.02E-01	3.11E-01±9.05E-02	3.08E-01±8.89E-02	1.18E-07±7.74E-08	6.18E-02±2.87E-02	**1.13E-29±6.11E-29**
Schwefel 2.22 (f_7)	4.26E-01±9.20E-01	5.26E-02±1.40E-01	6.13E-02±8.00E-02	1.38E-32±2.33E-32	3.62E-13±3.58E-13	**1.81E-47±1.46E-46**
Step (f_8)	8.77E+01±6.19E+01	3.33E-02±1.83E-01	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Quartic (f_9)	1.10E+02±4.23E+02	6.51E-08±2.76E-07	8.96E-08±3.43E-07	2.45E-90±1.34E-89	9.51E-29±2.36E-28	**2.77E-142±1.17E-141**
Rosenbrock (f_{10})	1.71E+02±1.44E+02	1.68E+01±2.88E+01	3.88E+01±4.71E+01	**3.62E+01±2.42E+01**	6.06E+01±3.90E+01	6.73E+01±7.73E+01
Griewank (f_{11})	4.64E-01±2.50E-01	5.36E-02±5.12E-02	8.96E-02±6.94E-02	7.38E-03±8.44E-03	1.35E-02±2.05E-02	**0.00E+00±0.00E+00**
Schwefel 2.26 (f_{12})	-2.25E+109±7.14E+109	6.55E+04±6.55E+04	6.55E+04±6.55E+04	-1.14E+90±4.14E+90	**-2.81E+111±1.54E+112**	-4.57E+03±1.38E+04
Rastrigin (f_{13})	1.88E+01±7.43E+00	1.74E+01±5.53E+00	1.79E+01±5.34E+00	**0.00E+00±0.00E+00**	1.31E+01±3.25E+00	**0.00E+00±0.00E+00**
NCRastrigin (f_{14})	2.24E+01±7.05E+00	1.66E+01±4.19E+00	1.52E+01±2.98E+00	**0.00E+00±0.00E+00**	1.23E+01±2.63E+00	**0.00E+00±0.00E+00**
Ackley (f_{15})	8.36E+00±1.81E+00	6.90E+00±1.81E+00	7.36E+00±1.38E+00	8.05E-15±2.46E-15	1.21E-09±1.94E-09	**4.74E-16±1.80E-15**
Penalized1 (f_{16})	8.05E+00±6.40E+00	6.22E-02±1.31E-01	2.07E-02±6.56E-02	**1.04E-02±3.28E-02**	4.37E-02±7.30E-02	1.51E-01±2.10E-01
Penalized2 (f_{17})	-7.90E+02±3.86E+02	-5.21E+02±1.16E+02	-5.87E+02±1.33E+02	-1.38E+03±1.55E+02	**-1.40E+03±2.10E+02**	-7.58E+02±1.85E+02
Bohachevsky2 (f_{18})	2.01E+01±1.55E+01	3.84E+00±1.61E+00	4.75E+00±1.73E+00	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Alphine (f_{19})	3.62E-02±5.07E-02	9.60E-03±2.69E-02	1.04E-02±4.24E-02	1.08E-05±2.32E-05	8.08E-02±1.05E-01	**5.80E-22±3.18E-21**
Levy (f_{20})	1.19E+00±7.27E-01	6.53E-01±5.96E-01	**5.54E-01±4.18E-01**	6.96E-01±2.65E-01	7.72E-01±1.83E-01	1.05E+00±3.21E-01
Weierstrass (f_{21})	1.25E+01±3.28E+00	1.03E+01±2.73E+00	1.07E+01±2.95E+00	2.97E-01±1.09E-01	2.70E+00±1.29E+00	**5.98E-02±4.82E-02**
Michalewicz (f_{22})	-2.72E+01±1.12E+00	-2.75E+01±7.81E-01	-2.74E+01±1.03E+00	-2.64E+01±1.14E+00	-2.70E+01±1.04E+00	**-2.15E+01±2.29E+00**

Table 7 The comparison results of HLABC with other algorithms on CEC2005 with 100 Dimensions

Problem name	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Sphere (f_1)	1.77E+01±7.59E+02	6.17E-04±1.61E-04	5.99E-04±9.37E-04	4.50E-52±1.34E-53	1.28E-19±3.66E-20	**7.70E-80±6.87E-81**
Elliptic (f_2)	5.53E+00±5.16E-01	6.55E-04±4.66E-04	2.54E-08±7.23E-09	4.52E-59±5.67E-59	2.66E-45±3.23E-44	**1.39E-72±1.36E-71**
SumSquare (f_3)	8.64E+00±7.79E+01	1.65E-07±7.04E-07	2.02E-10±7.99E-10	6.94E-53±9.36E-55	6.52E-20±3.48E-19	**2.18E-82±3.16E-82**
Schwefel 1.2 (f_4)	5.44E-01±1.97E-01	1.65E-06±7.72E-06	1.91E+03±7.70E+03	2.63E-39±6.11E-38	2.82E-21±1.01E-22	**7.17E-66±6.16E-67**
Schwefel 1.2 with noise (f_5)	2.38E+00±3.97E-01	8.12E-07±8.03E-07	7.81E-12±2.88E-13	1.15E-31±6.81E-31	1.88E-25±2.68E-26	**1.75E-40±4.57E-39**
Schwefel 2.21 (f_6)	1.12E+02±8.34E+00	6.82E+01±4.23E-01	6.80E+01±1.60E-01	1.42E+01±1.41E+00	1.53E+01±3.39E-01	**1.33E+01±4.62E+00**
Schwefel 2.22 (f_7)	6.28E+00±2.58E+00	1.33E-01±7.68E-02	8.20E-02±4.52E-01	1.25E-31±5.66E-32	8.45E-13±1.55E-13	**1.56E-45±1.36E-45**
Step (f_8)	**0.00E+00±3.54E+00**	**0.00E+00±7.07E-01**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Quartic (f_9)	3.39E+01±6.58E+00	5.61E-04±7.93E-04	3.44E-06±4.86E-05	1.86E-91±3.67E-90	1.55E-27±2.14E-28	**2.11E-131±5.13E-130**
Rosenbrock (f_{10})	1.79E+03±3.18E+02	**4.65E+00±2.01E+03**	9.43E+02±2.60E+02	1.22E+02±3.69E+01	2.57E+02±3.73E+01	9.82E+01±2.31E-02
Griewank (f_{11})	8.63E-09±3.06E-10	2.41E-20±3.57E-22	**0.00E+00±5.63E-02**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Schwefel 2.26 (f_{12})	-65492±7.31E+05	-65529±6.55E+04	**-65535±6.55E+04**	**-65535±6.55E+04**	**-65535±6.12E+04**	**-65535±2.51E+03**
Rastrigin (f_{13})	6.98E-07±1.78E-07	**0.00E+00±3.53E+00**	**0.00E+00±6.29E-01**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
NCRastrigin (f_{14})	1.35E+00±1.93E+00	**0.00E+00±1.41E+00**	**0.00E+00±1.41E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Ackley (f_{15})	1.21E-03±2.36E-03	8.78E-07±8.50E-07	1.13E-08±4.51E-08	**2.13E-14±3.42E-15**	3.20E-08±3.14E-06	**2.13E-14±3.61E-15**
Penalized1 (f_{16})	1.68E-03±3.75E-04	1.18E-11±1.67E-11	3.65E-21±4.14E-21	**1.04E-32±1.74E-32**	**1.04E-32±4.24E-33**	**1.04E-32±1.13E-32**
Penalized2 (f_{17})	1.87E-03±3.12E-03	2.29E-11±2.64E-12	1.96E-20±4.56E-18	**1.26E-33±3.79E-33**	**1.26E-33±1.96E-32**	1.44E-33±7.78E-33
Bohachevsky2 (f_{18})	1.23E-01±6.47E+00	2.10E-07±2.35E-08	2.35E-07±6.12E-07	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**	**0.00E+00±0.00E+00**
Alphine (f_{19})	5.05E-01±1.56E-01	3.58E-02±4.34E-02	1.21E-02±3.27E-03	9.41E-06±1.33E-09	1.96E-01±2.73E-01	**4.25E-06±2.25E-06**
Levy (f_{20})	4.93E-05±1.03E-05	2.02E-11±1.34E-13	1.80E-14±9.08E-16	**2.23E-32±2.51E-32**	1.79E-21±2.53E-21	6.28E-31±9.69E-31
Weierstrass (f_{21})	8.30E+02±1.27E+01	5.45E+02±2.05E-01	1.82E+01±6.16E-01	9.64E-01±2.46E-02	2.06E+00±2.87E-02	**1.12E-01±1.58E-02**
Michalewicz (f_{22})	-9.34E+01±5.17E-01	-9.64E+01±1.28E+00	-9.96E+01±2.06E+00	**-9.96E+01±1.17E+00**	**-9.96E+01±9.88E-01**	**-9.96E+01±3.93E-01**



(a) F1 (b) F2 (c) F3 (d) F4

(e) F5 (f) F6 (g) F7 (h) F8

(i) F9 (j) F10 (k) F11 (l) F12

(m) F13 (n) F14 (o) F15 (p) F16

(q) F17 (r) F18 (s) F19 (t) F20

Fig. 5 The convergence curves of the comparison of HLABC with other algorithms on CEC2005 benchmark

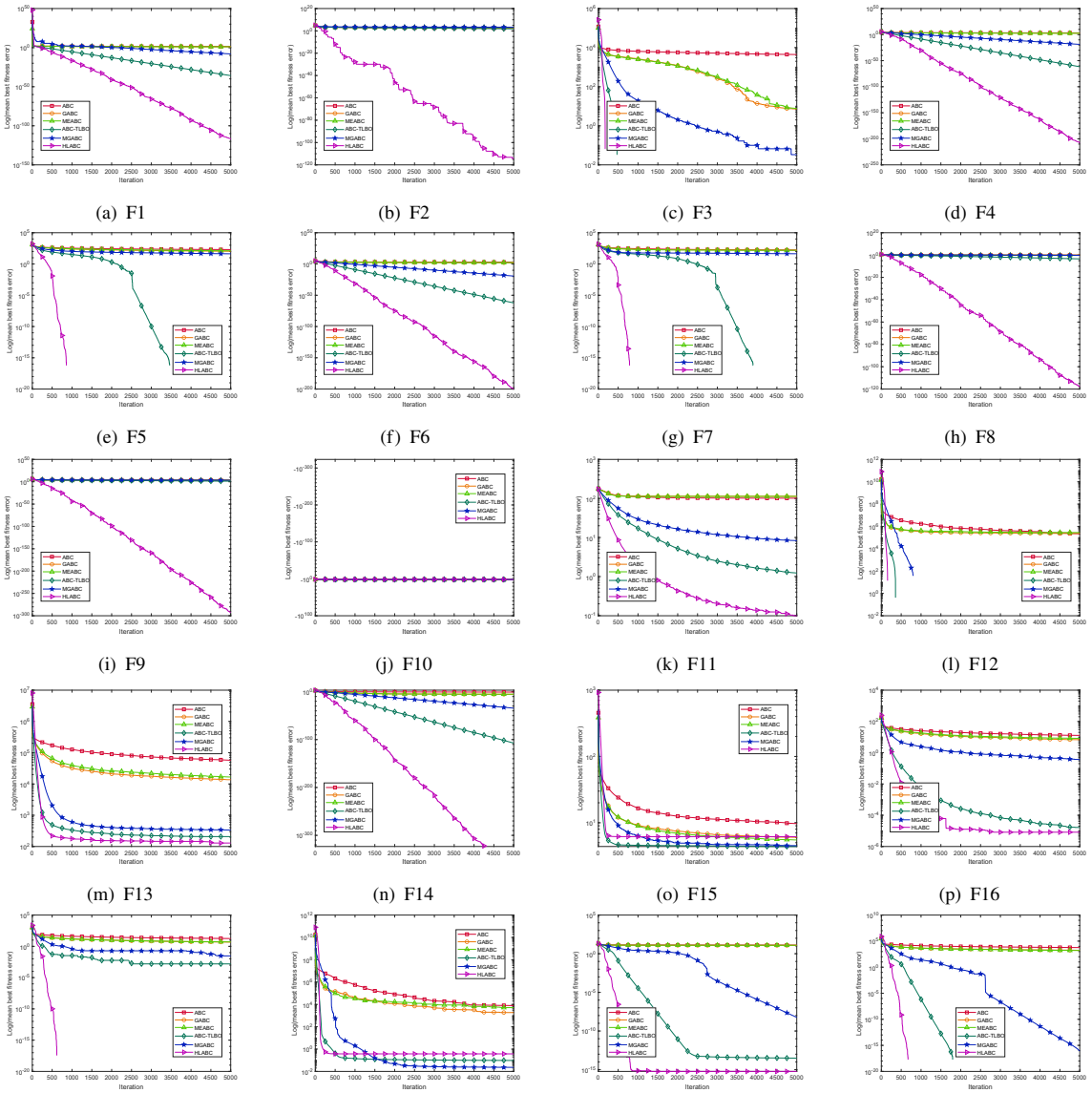
them, f_1-f_5 are unimodal functions. f_6-f_{20} are Basic Multimodal Functions. They were tested on 30 and 100 dimension, respectively.

For 30-dimensional problems, the experimental results are shown in Table 8. It can be observed that HLABC still shows good global convergence performance on the complex and high-dimensional problems. Compared with CEC2005, the experimental results obtained on CEC2013 are more similar. For f_1-f_5, the differences among the algorithms are obvious. On f_1 f_2, f_3 f_4 and f_5, HLABC algorithms all achieve the best solution. For f_10, MGABC finds the best solution. For f_6-f_{20}, the advantages of HLABC start to emerge, and the best solution is obtained on 11 problems. The convergence curves of the six classes of algorithms on $D = 30$ are shown in Fig. 6. It can be seen from the figure that HLABC converges faster. Especially, in the early iterations, the convergence results of HLABC is substantially ahead.

For the 100-dimensional problems, the performance of these algorithms rapidly diminishes as the dimensionality increases. However, the performance change of HLABC is not particularly obvious. For f_1-f_5, HLABC also achieves a good result. However, ABC-TLBO is a better solution than HLABC for two problems. For f_6-f_{20}, all algorithms achieve the best solution at f_{20}. HLABC and ABC-TLBO have more similar performance. Both reach the same solution on f_7. However, HLABC achieves the better solutions than ABC-

Table 8 The comparison results of HLABC with other algorithms on CEC2013 with 30 Dimensions

Functions	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Sphere Function(f_1)	2.91E+01±3.47E+00	1.28E+01±2.96E+00	1.39E+01±3.68E+00	1.60E-36±1.23E-36	9.01E-09±1.72E-08	**1.21E-117±6.54E-117**
Rotated High Conditioned Elliptic Function(f_2)	2.34E+02±4.31E+01	7.59E+02±1.94E+02	8.13E+02±2.24E+02	7.02E+01±4.30E+01	1.98E+03±6.06E+02	**3.27E-115±1.79E-114**
Rotated Bent Cigar Function(f_3)	4.43E+03±9.28E+02	7.27E+00±2.97E+00	7.53E+00±3.70E+00	**0.00E+00±0.00E+00**	3.33E-02±1.83E-01	**0.00E+00±0.00E+00**
Rotated Discus Function(f_4)	1.96E+03±5.70E+02	4.70E+02±3.66E+02	3.91E+02±1.81E+02	2.27E-62±5.53E-62	1.66E-19±1.27E-19	**1.30E-208±0.00E+00**
Different Powers Function(f_5)	2.04E+02±3.77E+01	1.09E+02±2.07E+01	1.30E+02±3.09E+01	**0.00E+00±0.00E+00**	4.21E+01±7.29E+00	**0.00E+00±0.00E+00**
Rotated Rosenbrock's Function(f_6)	8.31E+02±2.79E+02	1.25E+02±9.32E+01	1.98E+02±1.00E+02	5.99E-63±7.72E-63	5.31E-20±3.94E-20	**2.81E-200±0.00E+00**
Rotated Schaffers F7 Function(f_7)	1.67E+02±4.16E+01	1.66E+02±4.88E+01	1.56E+02±4.08E+01	**0.00E+00±0.00E+00**	4.31E+01±5.93E+00	**0.00E+00±0.00E+00**
Rotated Ackley's Function(f_8)	1.09E+00±9.72E-02	7.00E-01±9.66E-02	7.21E-01±1.05E-01	3.85E-04±2.02E-04	5.39E-01±9.30E-02	**1.42E-118±6.84E-118**
Rotated Weierstrass Function(f_9)	5.66E+03±1.77E+03	1.65E+03±5.10E+02	1.86E+03±5.84E+02	1.65E+01±1.05E+01	3.00E+03±1.09E+03	**4.46E-293±0.00E+00**
Rotated Griewank's Function(f_{10})	-7.63E+01±7.63E+00	-7.38E+01±4.05E+00	-7.26E+01±4.72E+00	-8.93E+01±2.90E+00	**-9.22E+01±2.42E+00**	-6.05E+01±3.55E+00
Rastrigin's Function(f_{11})	1.01E+02±1.76E+01	1.14E+02±9.86E+00	1.13E+02±8.49E+00	1.22E+00±2.70E-01	8.22E+00±2.29E+00	**9.89E-02±7.97E-02**
Rotated Rastrigin's Function(f_{12})	2.21E+05±3.30E+05	2.38E+05±7.18E+04	2.81E+05±6.71E+04	**-1.09E+03±9.10E+01**	-1.04E+03±1.64E+02	-1.66E+02±8.24E+01
Non-Continuous Rotated Rastrigin's Function(f_{13})	5.73E+04±1.30E+04	1.37E+04±9.65E+03	1.67E+04±1.07E+04	2.08E+02±7.28E+01	3.36E+02±5.65E+01	**1.29E+02±4.40E+01**
Schwefel's Function(f_{14})	1.81E-01±1.60E-01	4.09E-06±1.13E-05	6.43E-06±9.54E-06	5.27E-108±1.50E-107	4.66E-35±6.10E-35	**0.00E+00±0.00E+00**
Rotated Schwefel's Function(f_{15})	9.71E+00±3.44E+00	6.03E+00±1.70E+00	5.51E+00±2.18E+00	**4.38E+00±4.70E-01**	4.52E+00±4.86E-01	6.17E+00±7.36E-01
Rotated Katsuura Function(f_{16})	1.28E+01±3.50E+00	6.78E+00±2.69E+00	8.20E+00±3.12E+00	1.66E-05±9.06E-06	3.50E-01±2.12E-01	**8.23E-06±4.38E-05**
Lunacek Bi-Rastrigin Function(f_{17})	1.78E+01±4.57E+00	4.71E+00±2.12E+00	5.15E+00±2.60E+00	1.56E-03±4.90E-03	2.96E-02±1.07E-01	**0.00E+00±0.00E+00**
Rotated Lunacek Bi-Rastrigin Function(f_{18})	7.86E+03±2.70E+04	1.85E+03±3.85E+03	5.17E+03±7.57E+03	9.52E-02±6.25E-02	**2.31E-02±1.91E-02**	3.73E-01±1.74E-01
Expanded Griewank's plus Rosenbrock's Function(f_{19})	1.31E+01±7.00E-01	1.22E+01±6.86E-01	1.24E+01±6.49E-01	3.10E-14±3.72E-15	5.91E-09±5.51E-09	**5.92E-16±1.89E-15**
Expanded Scaffer's F6 Function(f_{20})	5.81E+03±1.76E+03	1.74E+03±1.04E+03	1.60E+03±7.66E+02	**0.00E+00±0.00E+00**	8.88E-17±1.82E-16	**0.00E+00±0.00E+00**



(a) F1 (b) F2 (c) F3 (d) F4

(e) F5 (f) F6 (g) F7 (h) F8

(i) F9 (j) F10 (k) F11 (l) F12

(m) F13 (n) F14 (o) F15 (p) F16

(q) F17 (r) F18 (s) F19 (t) F20

Fig. 6 The convergence curves of the comparison of HLABC with other algorithms on CEC2013 benchmark

TLBO with a total of 12 for all problems. MGABC have a better solution than HLABC also for two problems. Overall, HLABC achieves the best solution in 13 problems (Table 9).

4.4.3 Experimental results and discussions on CEC2010 test suite

To further verify the search ability of HLABC on solving the high-dimensional especially the large-scale optimization problems, the CEC2010 benchmark suite is selected to test the performance of the proposed HLABC and its peer compared ABC variants. The CEC2010 benchmark can be divided into three groups: 3 separable functions f_1-f_3, 15 partially separable functions f_4-f_{18} and 2 non-separable functions f_{19} and f_{20}. All these functions were shifted and rotated, which makes it difficulty for optimizers to find the global optimal solution. Thus, such benchmark functions are able to measure the real search performance of those compared algorithms.

The detailed experimental results on 1000-dimensional problems of HLABC with other five ABC variants on the CEC2010 test suite are listed in Table 10. It can be seen that for the separable functions f_1-f_3, HLABC performs better than these competitors on f_1. However, in the convergence results of functions f_2 and f_3, the HLABC algorithm has a slightly worse performance than ABC-TLBO and MGABE. In particular, the HLABC algorithm shows better results on

Table 9 The comparison results of HLABC with other algorithms on CEC2013 with 100 Dimensions

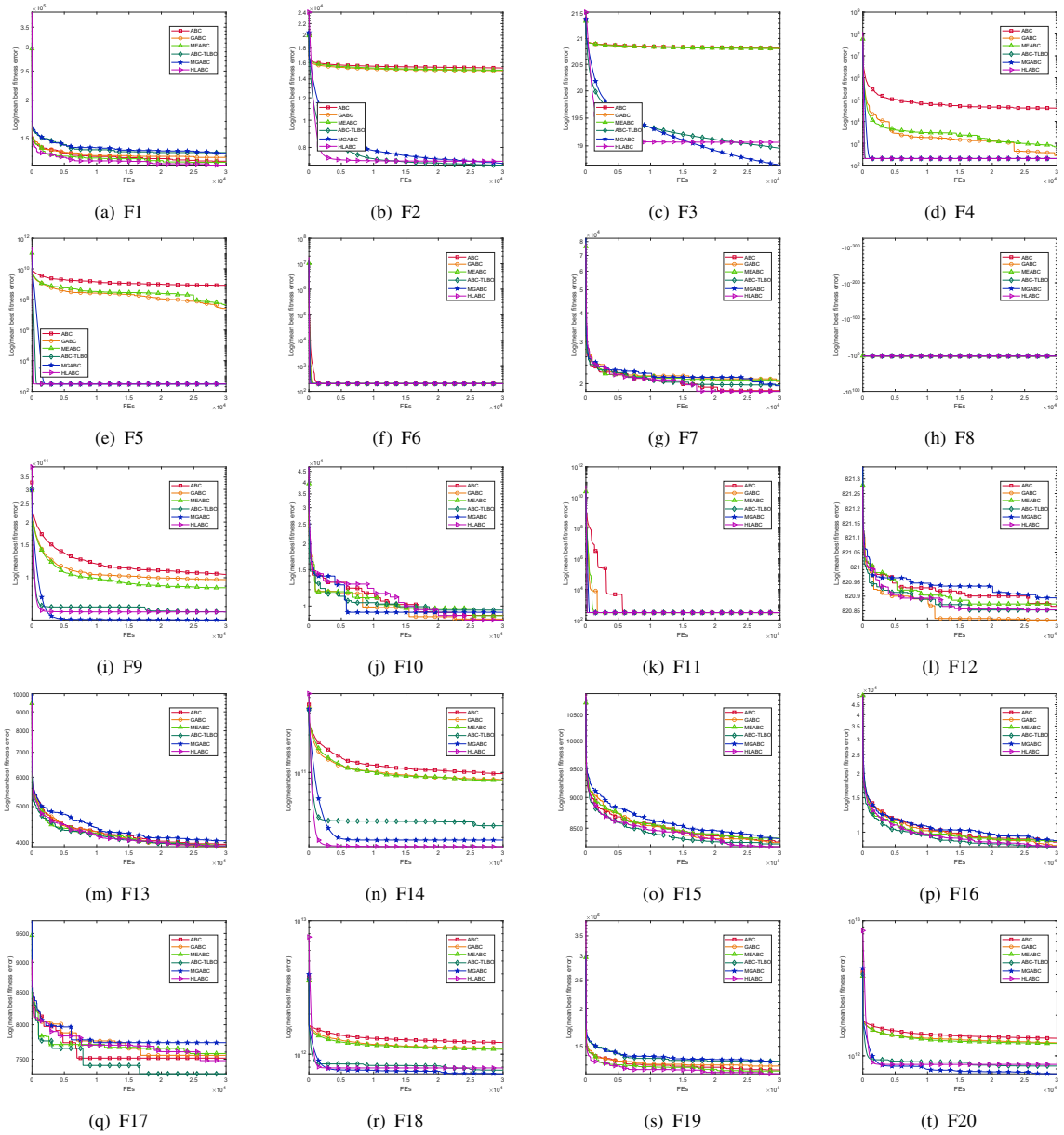
Functions	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Sphere Function(f_1)	1.35E+05±4.42E+03	1.28E+05±3.99E+03	1.28E+05±3.76E+03	**1.24E+05±6.70E+03**	1.32E+05±4.19E+03	1.25E+05±5.37E+03
Rotated High Conditioned Elliptic Function(f_2)	2.61E+04±4.24E+04	2.34E+02±6.45E+01	4.62E+02±4.53E+02	2.00E+02±3.03E-14	2.03E+03±6.11E-14	**2.00E-95±0.00E+00**
Rotated Bent Cigar Function(f_3)	6.32E+04±5.93E+04	3.06E+04±3.85E+05	2.96E+04±4.62E+04	3.01E+04±6.31E+04	2.62E+04±5.21E+04	**1.63E+04±4.62E+04**
Rotated Discus Function(f_4)	6.36E+04±1.23E+00	4.32E+04±2.13E+01	4.13E+04±1.30E-01	3.66E-57±4.26E+00	4.48E+-17±2.33E-01	**4.21E-177±1.33E+00**
Different Powers Function(f_5)	8.40E+04±6.69E+03	8.65E+04±5.96E+03	7.97E+04±9.30E+03	**7.89E+04±1.06E+04**	8.69E+04±8.03E+03	8.29E+04±8.16E+03
Rotated Rosenbrock's Function(f_6)	1.80E+04±4.84E+03	1.69E+04±6.86E+03	1.47E+04±4.33E+03	1.04E-59±3.29E+03	1.39E-16±4.26E+03	**1.45E-186±1.26E+03**
Rotated Schaffers F7 Function(f_7)	1.22E+12±1.48E+12	6.33E-02±7.68E-02	3.20E-01±4.52E-01	**0.00E+00±0.00E+00**	8.45E+1±1.55E+1	**0.00E+00±0.00E+00**
Rotated Ackley's Function(f_8)	8.21E+02±9.27E-03	5.00E-01±7.07E-01	6.12E+00±5.87E+00	6.53E-03±4.52E-02	9.23E+00±6.21E+00	**2.51E-77±8.35E-76**
Rotated Weierstrass Function(f_9)	-1.62E+03±1.71E+01	5.61E-04±7.93E-04	3.44E-15±4.86E-15	6.52E+01±9.22E+02	6.32E+00±7.21E+00	**3.56E-88±5.43E-87**
Function(f_{10})	7.67E+03±1.53E+02	7.62E+03±1.64E+02	7.67E+03±1.36E+02	7.59E+03±1.45E+02	7.68E+03±1.54E+02	**7.58E+03±2.05E+02**
Rastrigin's Function(f_{11})	8.67E+08±9.46E+07	8.56E+07±1.69E+06	4.95E+07±4.32E+06	**3.95E+06±2.66E+06**	6.31E+06±6.33E+05	4.93E+06±4.66E+05
Rotated Rastrigin's Function(f_{12})	7.32E+08±2.64E+05	5.21E+07±6.05E+06	4.32E+07±1.22E+06	6.13E+07±2.53E+06	4.19E+07±3.61E+05	**3.62E+07±2.66E+05**
Non-Continuous Rotated Rastrigin's Function(f_{13})	8.56E+04±3.66E+03	7.22E+04±8.46E+04	5.95E+04±6.25E+04	5.66E+04±5.84E+03	4.51E+04±2.35E+03	**4.16E+04±5.31E+04**
Schwefel's Function(f_{14})	5.96E+05±2.34E+03	2.33E+03±1.61E+03	1.94E+03±3.22E+02	8.83E+02±1.62E+01	6.22E+02±3.33E+02	**3.62E+02±2.23E+01**
Rotated Schwefel's Function(f_{15})	8.31E+03±9.43E+03	8.30E+03±8.16E+03	8.39E+03±4.22E+02	8.26E+03±5.14E+03	8.39E+03±5.14E+04	**8.00E+03±3.84E+04**
Rotated Katsuura Function(f_{16})	5.62E+07±9.22E+07	3.14E+06±4.62E+06	1.32E+06±4.86E+07	8.54E+06±9.16E+05	**4.32E+05±2.63E+05**	7.46E+05±9.36E+05
Lunacek Bi-Rastrigin Function(f_{17})	4.29E+04±8.16E+01	4.27E+04±8.31E+01	4.27E+04±7.66E+01	4.23E+04±8.15E+01	4.31E+04±7.16E+01	**4.22E+04±7.66E+01**
Rotated Lunacek Bi-Rastrigin Function(f_{18})	2.66E+04±3.21E+02	4.22E+02±1.66E+01	3.62E+02±2.33E+00	**1.02E-15±2.32E+00**	3.66E-13±2.33E+00	3.22E-14±1.32E+01
Expanded Griewank's plus Rosenbrock's Function(f_{19})	6.13E+07±3.46E+06	4.68E+06±6.15E+05	3.44E+06±2.66E+06	6.33E+05±8.74E+04	**1.52E+05±6.31E+06**	3.29E+05±4.85E+05
Expanded Scaffer's F6 Function(f_{20})	**2.01E+03±1.92E-01**	**2.01E+03±1.47E-01**	**2.01E+03±1.69E-01**	**2.01E+03±9.82E-02**	**2.01E+03±1.19E-01**	**2.01E+03±1.68E-01**

Table 10 The comparison results of HLABC with other algorithms on CEC2010 with 1000 dimensions

Functions	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Shifted Elliptic Function(f_1)	1.25E+05±3.46E+03	1.29E+05±5.06E+03	1.24E+05±4.82E+03	1.34E+05±6.97E+03	1.33E+05±4.19E+03	**1.22E+05±5.26E+03**
Shifted Rastrigin Function(f_2)	1.53E+04±1.64E+02	1.50E+04±2.13E+02	1.50E+04±1.33E+02	**6.92E+03±3.60E+02**	7.02E+03±2.86E+02	7.14E+03±3.03E+02
Shifted Ackley Function (f_3)	2.08E+01±3.31E-02	2.08E+01±2.04E-02	2.08E+01±2.73E-02	1.90E+01±3.46E-02	**1.87E+01±6.24E-02**	1.91E+01±2.88E-02
Single-group Shifted and m-rotated Elliptic Function(f_4)	4.07E+04±6.45E+04	2.73E+02±9.96E+01	6.85E+02±6.63E+02	**2.00E+02±3.00E-14**	**2.00E+02±3.28E-14**	**2.00E+02±9.47E-15**
Single-group Shifted and m-rotated Rastrigin's Function(f_5)	8.39E+08±3.44E+08	2.53E+07±2.92E+07	4.41E+07±6.13E+07	**3.00E+02±1.89E-14**	**3.00E+02±1.00E-13**	**3.00E+02±1.89E-14**
Single-group Shifted and m-rotated Ackley's Function(f_6)	**2.00E+02±0.00E+00**	**2.00E+02±0.00E+00**	**2.00E+02±0.00E+00**	**2.00E+02±1.42E-14**	**2.00E+02±0.00E+00**	**2.00E+02±0.00E+00**
Single-group Shifted m-dimensional Schwefel's Problem 1.2(f_7)	1.87E+04±1.39E+03	2.04E+04±1.35E+03	2.08E+04±1.26E+03	1.98E+04±1.01E+03	1.96E+04±1.70E+03	**1.86E+04±2.36E+03**
Single-group Shifted m-dimensional Rosenbrock's Function(f_8)	-3.30E+02±1.15E+01	-3.23E+02±1.68E+01	-3.19E+02±6.16E+00	-3.47E+02±9.97E+00	-3.15E+02±9.46E+00	**-3.47E+02±1.34E+01**
$\frac{D}{2m}$-group Shifted and m-rotated Elliptic Function(f_9)	1.04E+11±8.53E+09	9.76E+10±8.03E+09	8.82E+10±5.87E+09	6.53E+10±1.09E+10	**5.91E+10±7.04E+09**	6.53E+10±7.90E+09
$\frac{D}{2m}$-group Shifted and m-rotated Rastrigin's Function(f_{10})	8.99E+03±4.06E+02	8.64E+03±1.74E+03	8.96E+03±1.42E+03	9.55E+03±1.78E+03	9.32E+03±6.40E+02	**8.55E+03±2.59E+03**
$\frac{D}{2m}$-group Shifted and m-rotated Ackley's Function(f_{11})	**3.00E+02±0.00E+00**	**3.00E+02±2.84E-14**	**3.00E+02±5.68E-14**	**3.00E+02±0.00E+00**	**3.00E+02±2.84E-14**	**3.00E+02±0.00E+00**
$\frac{D}{2m}$-group Shifted m-dimensional Schwefel's Problem 1.2(f_{12})	**8.21E+02±5.60E-02**	**8.21E+02±8.38E-02**	**8.21E+02±4.17E-02**	**8.21E+02±4.06E-02**	**8.21E+02±3.61E-02**	**8.21E+02±3.43E-02**
$\frac{D}{2m}$-group Shifted m-dimensional Rosenbrock's Function(f_{13})	3.96E+03±3.91E+02	3.90E+03±3.19E+02	3.92E+03±2.73E+02	3.91E+03±2.51E+02	4.04E+03±3.34E+02	**3.92E+03±2.73E+02**
$\frac{D}{m}$-group Shifted and m-rotated Elliptic Function f_{14}	9.79E+10±5.68E+09	8.77E+10±6.40E+09	8.61E+10±7.38E+09	3.73E+10±9.59E+09	2.86E+10±3.07E+09	**2.54E+10±4.93E+09**
$\frac{D}{m}$-group Shifted and m-rotated Rastrigin's Function(f_{15})	8.29E+03±3.10E+02	8.27E+03±3.27E+02	8.34E+03±2.14E+02	8.26E+03±1.92E+02	8.34E+03±1.83E+02	**8.21E+03±2.01E+02**
$\frac{D}{m}$-group Shifted and m-rotated Ackley's Function f_{16}	8.88E+03±1.19E+03	8.54E+03±1.49E+03	8.98E+03±1.08E+03	8.44E+03±1.39E+03	9.05E+03±1.62E+03	**8.43E+03±1.25E+03**
$\frac{D}{m}$-group Shifted m-dimensional Schwefel's Problem 1.2(f_{17})	7.51E+03±8.83E+01	7.55E+03±6.43E+01	7.58E+03±3.12E+02	**7.30E+03±3.54E+02**	7.74E+03±1.13E+02	7.48E+03±1.41E+02
$\frac{D}{m}$-group Shifted m-dimensional Rosenbrock's Function(f_{18})	1.22E+12±4.58E+10	1.11E+12±6.08E+10	1.09E+12±5.82E+10	7.59E+11±7.32E+10	**7.14E+11±5.88E+10**	7.89E+11±7.59E+10
Shifted Schwefel Problem 1.2(f_{19})	1.25E+05±3.46E+03	1.29E+05±5.06E+03	1.24E+05±4.82E+03	1.34E+05±6.97E+03	1.33E+05±4.19E+03	**1.22E+05±5.26E+03**
Shifted Rosenbrock Function(f_{20})	1.35E+12±4.47E+10	1.25E+12±4.95E+10	1.23E+12±5.17E+10	8.41E+11±1.32E+11	**7.36E+11±3.96E+10**	8.64E+11±2.06E+10

most of the non-separable functions. For f_4, f_5 and f_{17}, the convergence results of ABC-TLBO, MGABC, HLABC are comparable. They all achieve the best solution. ABC-TLBO performs better than HLABC on f_{17}. For this problem, the search performance of HLABC is only inferior to ABC-TLBO and has a comparable search performance on f_{19} and f_{20}. On the problems f_4-f_{18}, HLABC achieves the best solution on 12 problems. These results demonstrate that HLABC is a competitive global search optimizer with good performance in terms of fitness values on solving high-dimensional problems.

In order to investigate the search behavior of the compared algorithms on the CEC2010 test suite, the average fitness values obtained in each generation are calculated, and then the convergence curves showed in Fig. 7 are plotted based on those values. From the figures, it can be seen that each comparison algorithm converges gradually during the process of iteration, but the final convergence accuracy results vary significantly. Some algorithms converged prematurely or failed to search for a more optimal solution on the complex functions such as f_{13}, f_{14}, and f_{17} during the search process. However, HLABC still can converge to the relatively good solution quickly and converges significantly faster than the other algorithms. Overall, HLABC generally performs better than ABC, GABC, MEABC, ABC-TLBO, MGABC on the high-dimensional problem.



(a) F1 (b) F2 (c) F3 (d) F4

(e) F5 (f) F6 (g) F7 (h) F8

(i) F9 (j) F10 (k) F11 (l) F12

(m) F13 (n) F14 (o) F15 (p) F16

(q) F17 (r) F18 (s) F19 (t) F20

Fig. 7 The convergence curves of the comparison of HLABC with other algorithms on CEC2010 benchmark

Table 11 The winning rate of HLABC on different test suites

Test suites	ABC	GABC	MEABC	ABC-TLBO	MGABC
CEC2005-30D	82%	82%	77%	55%	64%
CEC2005-100D	95%	82%	73%	50%	59%
CEC2013-30D	95%	90%	90%	60%	80%
CEC2013-100D	90%	95%	90%	70%	85%
CEC2010-1000D	85%	80%	80%	35%	50%
Average winning rate	89%	86%	82%	54%	68%

4.4.4 The statistical results and analysis

Firstly, to evaluate the overall performance of HLABC on different test suites, Table 11 shows the rate of Mean that HLABC outperforms the competitors. From Table 11, it can be observed that HLABC outperforms ABC and GABC algorithms on all five test suites with a rate of more than 80%, while outperforming MEABC and MGABC algorithms at 70% and 50% rates, respectively. Although HLABC only outperforms ABC-TLBO on the 1000-dimensional CEC2010 at a rate of 35%, it performs with a high rate of more than 50% on all other suites. Therefore, HLABC shows significant advantages on all five test suites.

Moreover, in order to verify whether the results acquired by the compared algorithms are statistical different from results obtained by other methods, the non-parametric Wilcoxon rank sum test are also conducted to further analyze the performance of HLABC and the ABC variants. The p-value is generated from the Wilcoxn sum test, and it reflects whether the results of two sets follow the same distribution with a preset significance level α, such as $\alpha = 0.05$. In general, a smaller p-value than the predetermined significance level implies there is a significant difference between the two sets. Conversely, a larger p-value than the significance level indicated that there is no significant difference.

Table 12 shows the p-values between HLABC and the other five ABCs. If the p-value is less than 0.05, then there is a significant difference between HLABC and its compared algorithm. From the experimental results, it can be seen that there is a significant difference between HLABC and other ABCs on 30-dimensional and 100-dimensional problems. For CEC2013, D=30, HLABC is not significantly better than ABC-TLBO and MGABC. Overall, HLABC outperforms other ABC algorithms significantly in terms of the average fitness value.

In addition, another non-parameter Friedman test was also utilized to measure the average performance of compared algorithms as further statistical analysis. The average ranking values generated from the Friedman test is shown in the Fig. 8. The smaller average ranking values mean that the algorithms have better overall performance. For HLABC, the smallest ranking value was achieved in all comparisons. This indicates that HLABC has the best overall performance among the six ABCs. The performance of ABC-TLBO, MGABC is slightly inferior to HLABC. ABC and GABC achieve the worst results.

Figures 5, 6 and 7 show the convergence curves of algorithms on different optimization problems. These figures show that the HLABC algorithm exhibits the fastest convergence rate on 39 of the 60 functions, indicating that the learning pattern from the outer layer to the inner layer in the hierarchical learning model provides a more reasonable evolutionary direction for population evolution, allowing the population to quickly evolve towards the dominant region.

In order to observe the performance of the compared algorithms more intuitively, the Boxplot figures are used to show the potential of the compared methods. From the Boxplot, one can be seen how consistently the optimal solutions are obtained on the problem-solving. Figures 9, 10, and 11 showed the Boxplots of compared algorithms on test functions selected from the CEC2005,CEC2010 and CEC2013 suites.

Table 12 The Wilcoxon rank sum test results on CEC2005,CEC2010, and CEC2013

Algorithm	CEC2005		CEC2013		CEC2010
	D=30	D=100	D=30	D=100	D=1000
ABC	4.15E-05	2.02E-05	1.95E-03	7.65E-04	3.46E-04
GABC	2.13E-03	2.39E-03	1.80E-02	3.54E-04	4.27E-03
MEABC	5.62E-03	9.33E-03	3.60E-02	1.44E-03	1.69E-03
ABC-TLBO	2.64E-02	3.43E-02	1.12E-01	2.01E-02	3.10E-02
MGABC	3.51E-02	4.57E-02	2.48E-01	3.80E-02	5.28E-03

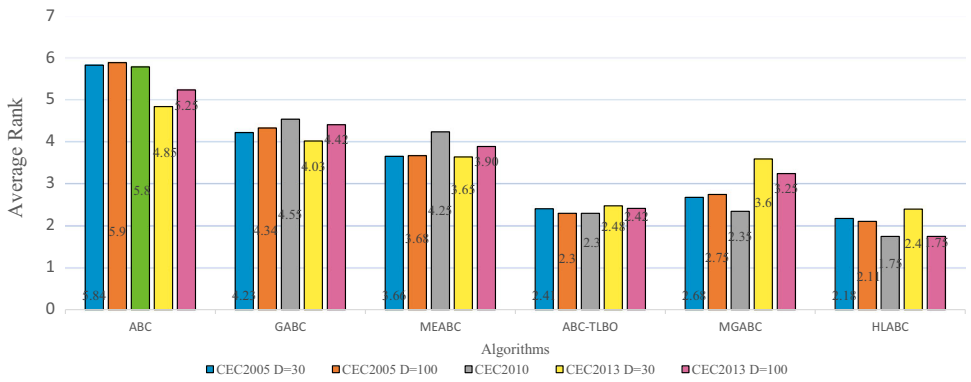
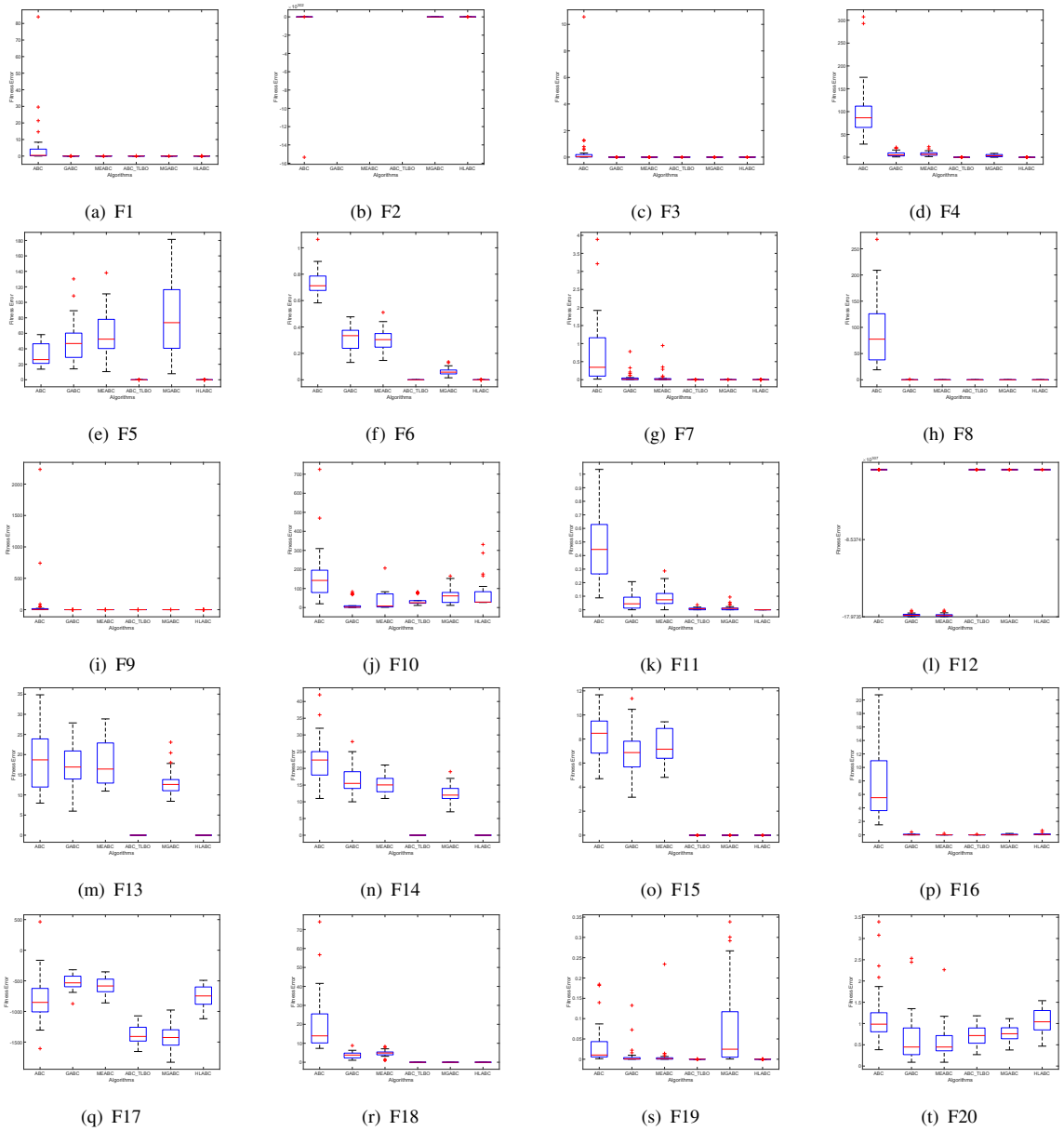


Fig. 8 The average ranking results of the compared algorithms by Friedman tests



(a) F1 (b) F2 (c) F3 (d) F4

(e) F5 (f) F6 (g) F7 (h) F8

(i) F9 (j) F10 (k) F11 (l) F12

(m) F13 (n) F14 (o) F15 (p) F16

(q) F17 (r) F18 (s) F19 (t) F20

Fig. 9 The boxplot of the comparison of HLAB with other algorithms on CEC2005 benchmark functions

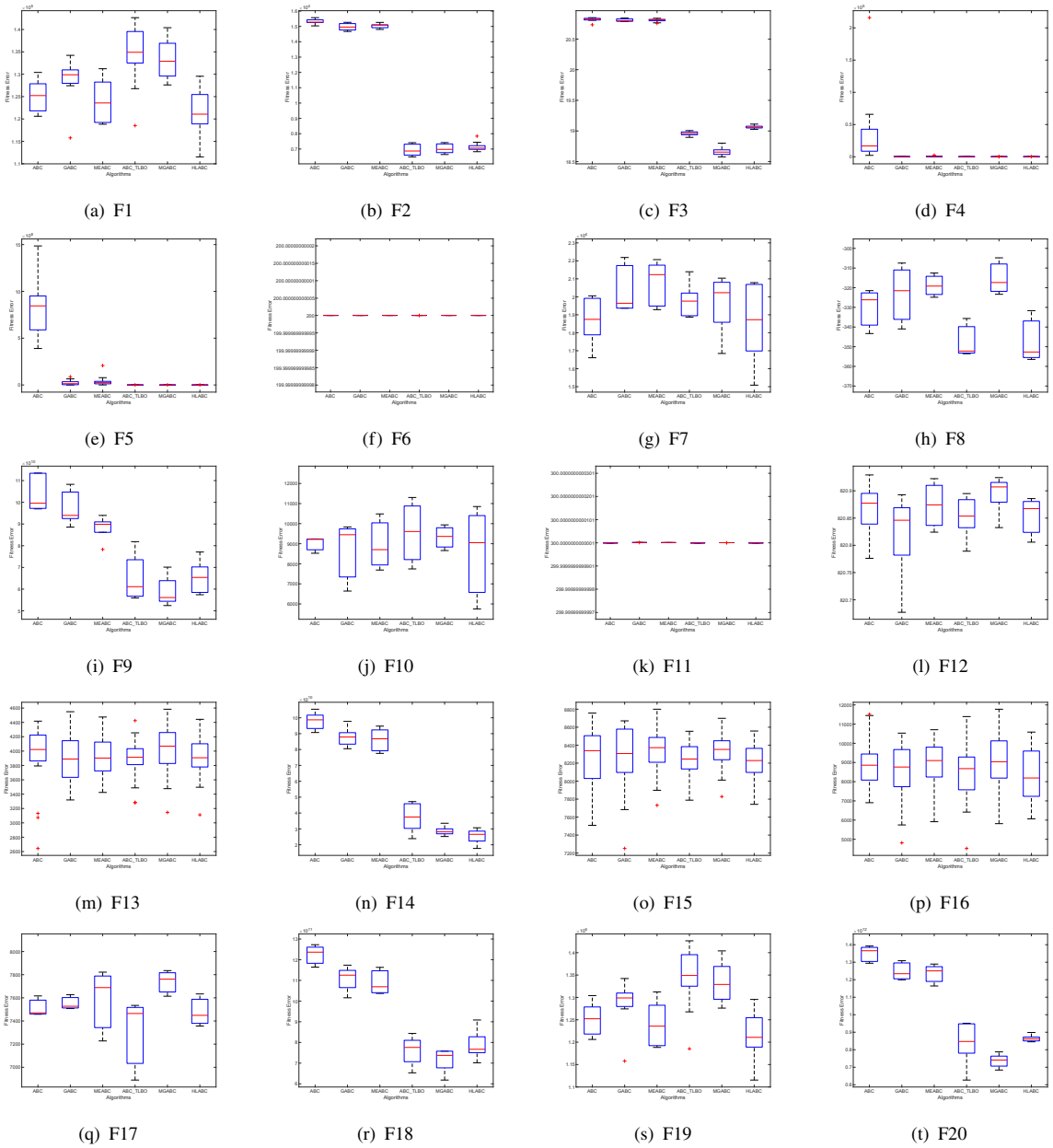
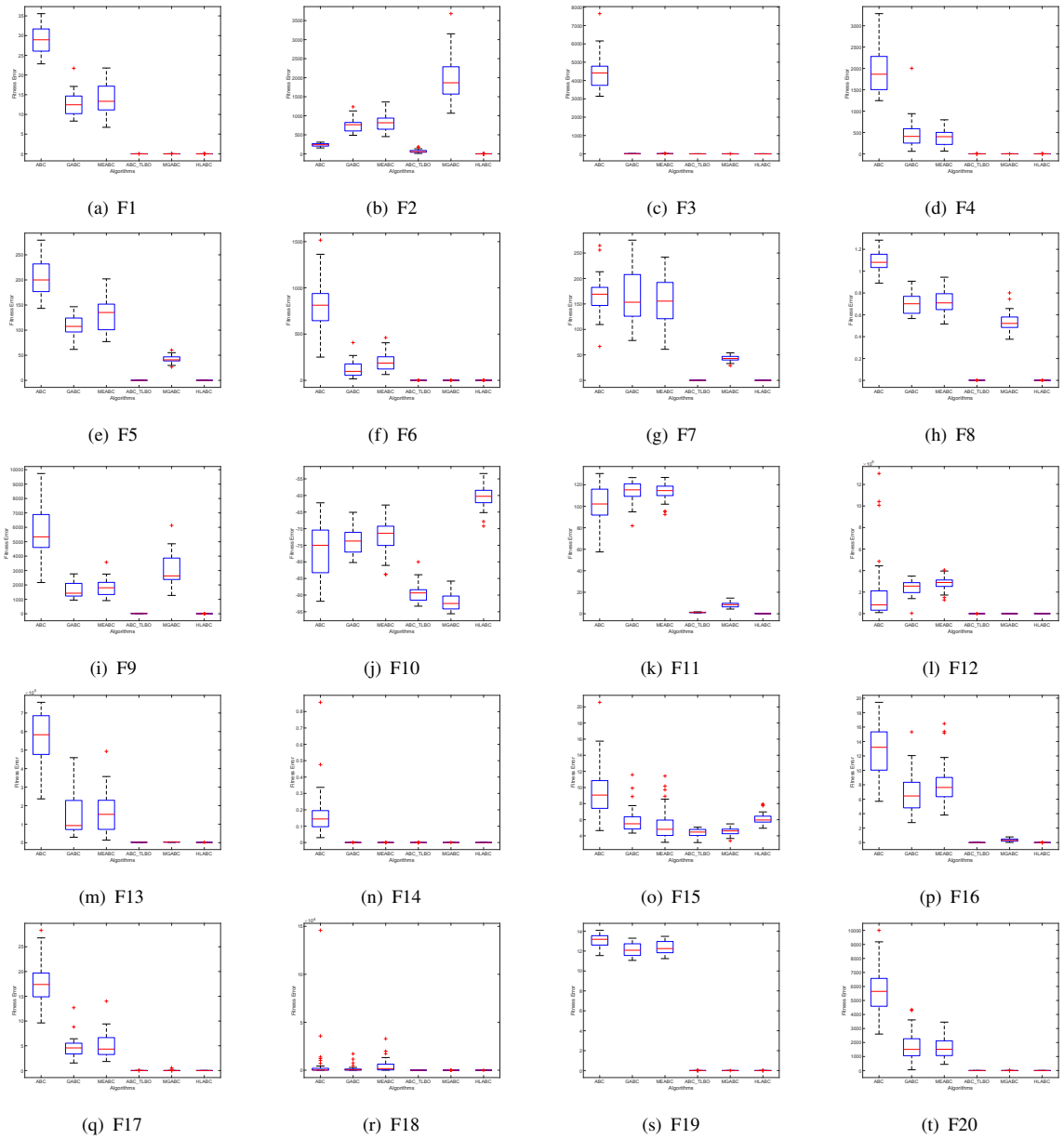


Fig. 10 The boxplot of the comparison of HLAB with other algorithms on CEC2010 benchmark functions

It can be seen that the performance of HLABC on unimodal, multimodal, and large scale functions is significantly improved compared to other algorithms.

This indicates that the HLABC has stronger stability, further proving that the learning at different levels in the hierarchical learning model provides the population with more evolutionary directions to avoid the population quickly falling into local optima and causing the algorithm to prematurely mature, which greatly improves the robustness of the ABC algorithm. Finally, we display the 3D images of some functions and the 2D search history of HLABC on these functions in Fig. 12.

It can be seen that individuals exhibit two distribution states, namely scattered throughout the space and clustered in some regions. Firstly, scattered individuals demonstrate the exploration process of the population throughout the search space, which can help the algorithm locate the advantageous regions and avoid the population falling into local optima. Secondly, the clustering of individuals demonstrates the development process of the population, which is the process of continuous search in the advantageous regions. This can help the algorithm find more accurate solutions. From the 2D search image of HLABC algorithm, it can be seen that the hierarchical learning model well balances the exploration



(a) F1 (b) F2 (c) F3 (d) F4

(e) F5 (f) F6 (g) F7 (h) F8

(i) F9 (j) F10 (k) F11 (l) F12

(m) F13 (n) F14 (o) F15 (p) F16

(q) F17 (r) F18 (s) F19 (t) F20

Fig. 11 The boxplot of the comparison of HLAB with other algorithms on CEC2013 benchmark functions

and development process of the algorithm, which ensures the population continues to evolve towards the advantageous regions.

4.5 The performance of HLABC on the latest problems

In order to test the performance of the proposed HLABC algorithm on the latest problems, this section conducted comparative experiments on CEC2022 between the HLABC algorithm and advanced PSO and DE variants, including SAPSO [65], AIWPSO [66], EHDE [67], and APSO-DEE [68]. The experimental data are shown in Table 13

From the experimental results, it can be seen that the HLABC algorithm obtained the best test results in a total of 9 test functions, including F2, F3, F4, F5, F7, F8, F10, F11, and F12. HLABC has demonstrated excellent comprehensive testing results at CEC2022.

5 Deployment of wireless sensor networks

As an intelligent information acquisition system, wireless sensor networks(WSNs) relies on a large number of wireless sensors to realize data dissemination and interaction through

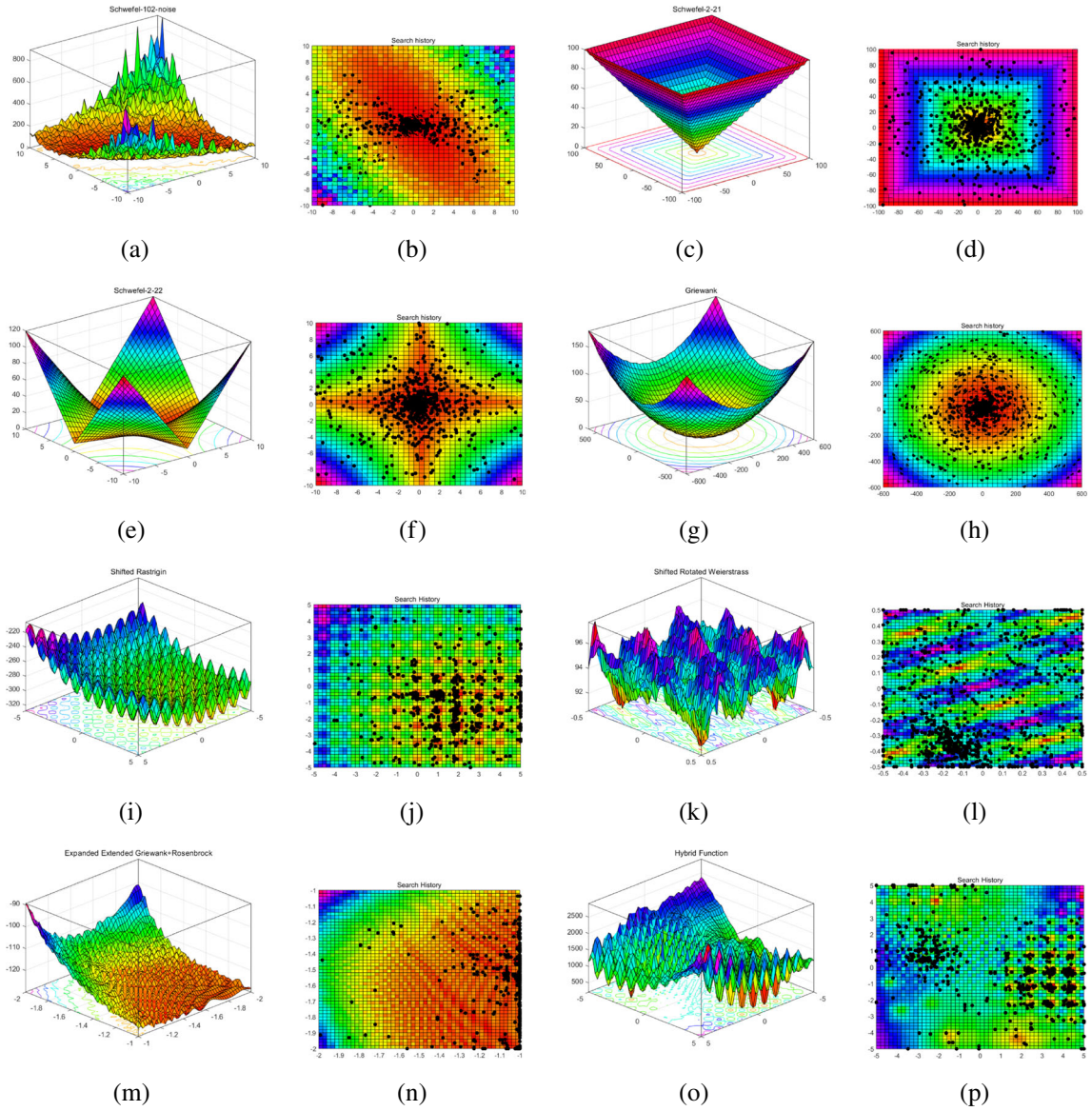


Fig. 12 The search history of individuals in HLABC algorithm on different types of benchmark functions

electromagnetic wave signals, while monitoring external objects to achieve information collection and transmission.

In order to ensure that WSNs can efficiently complete the monitoring tasks in the designated area, it is necessary to optimize the deployment of WSNs [69]. Therefore, the WSNs coverage optimization problem has long been a hot topic of research as an important element of the deployment of WSNs [70].

5.1 Mathematical model of WSNs coverage rate

As the end structure of WSNs, wireless sensors are the basic units for monitoring the external objects, and the quality of service of WSNs is directly affected by the area coverage rate. Therefore, in this paper, the area coverage rate is used as evaluation index for the deployment of WSNs, and the problem of maximizing the coverage rat is optimized.

Assume that v isomorphic wireless sensors are randomly scattered in the two-dimensional plane P, the set of these sensors is denoted $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_v\}$, and the ith sensor coordinate is $\boldsymbol{c}_i = (cx_i, cy_i)$. Each sensor has the same sensing radius R_s and communication radius R_c satisfying $R_c = 2R_s$.

The Euclidean distance from sensor $\boldsymbol{c}_i$ to any point in the monitoring area is:

$$d(\boldsymbol{c}_i, z) = \sqrt{(x - cx_i)^2 + (y - cy_i)^2} \tag{12}$$

Table 13 The comparison results of HLABC with other algorithms on CEC2022

Functions	SAPSO Mean±Std	AIWPSO Mean±Std	EHDE Mean±Std	APSO-DEE Mean±Std	HLABC Mean±Std
F1	1.17E+03±1.24E+03	**0.00E+00±5.17E-29**	6.14E-07±4.68E-07	1.14E+05±8.43E+04	1.18E+02±5.68E+01
F2	9.65E+01±5.81E+01	8.99E+00±2.19E+01	4.69E+00±4.17E+00	3.15E+03±1.00E+03	**4.43E+00±3.55E+00**
F3	2.59E+01±1.02E+01	1.14E-01±2.49E-01	3.09E-01±6.67E-01	9.30E+01±1.87E+01	**4.17E-02±4.09E-02**
F4	3.40E+01±1.95E+01	2.25E+01±9.07E+00	1.64E+01±8.63E+00	1.30E+02±1.95E+01	**1.63E+01±5.90E+00**
F5	1.10E+02±1.92E+02	1.32E+01±3.86E+01	1.10E+01±2.13E+01	3.47E+03±8.21E+02	**1.35E+00±1.14E+00**
F6	5.10E+05±7.62E+05	2.01E+03±1.74E+03	**3.30E+01±2.60E+01**	2.16E+09±1.59E+09	2.58E+03±1.43E+03
F7	8.86E+01±4.74E+01	1.70E+01±8.76E+00	1.91E+01±8.01E+00	2.75E+02±8.10E+01	**1.69E+01±3.73E+00**
F8	5.01E+01±3.96E+01	2.15E+01±6.94E-01	1.84E+01±6.44E+00	5.37E+02±2.13E+02	**1.38E+01±5.41E+00**
F9	3.43E+02±1.09E+02	**2.05E+02±1.23E+01**	2.29E+02±3.64E-12	8.62E+02±2.84E+02	2.29E+02±2.82E-03
F10	2.71E+02±4.17E+02	1.48E+02±6.19E+01	1.13E+02±3.78E+01	1.24E+03±8.92E+02	**9.82E+01±6.98E+00**
F11	3.08E+02±2.13E+02	4.50E+01±1.01E+02	4.00E+01±1.26E+02	2.03E+03±1.08E+03	**2.47E+01±5.01E+00**
F12	1.92E+02±2.38E+01	1.63E+02±2.07E+01	1.65E+02±1.91E+00	5.27E+02±1.77E+02	**1.60E+02±1.19E+00**

Thus the detection probability of sensor $\boldsymbol{c}_i$ for any point $\boldsymbol{z}$ can be expressed in (13).

$$P(\boldsymbol{c}_i, \boldsymbol{z}) = \begin{cases} 1 & d(\boldsymbol{c}_i, \boldsymbol{z}) \le R_s - R_e \\ \exp\left(\frac{-\lambda_1 \alpha_1{}^{\beta_1}}{\alpha_2{}^{\beta_2} + \lambda_2}\right) & otherwise \\ 0 & d(\boldsymbol{c}_i, \boldsymbol{z}) \ge R_s + R_e \end{cases} \tag{13}$$

where $R_e (0 < R_e < R_s)$ is the reliability parameter;$\alpha_1 = R_e - R_s + d(\boldsymbol{c}_i, \boldsymbol{z})$;$\alpha_2 = R_e + R_s - d(c_i, z)$; $\beta_1, \beta_2, \lambda_1$ and λ_2 are the measurement parameter related to the sensor, and in this paper they are set to $\beta_1 = 1, \beta_2 = 1.5, \lambda_1 = 1$ and $\lambda_2 = 0$.

After obtaining the detection probability for any point based on (13), the joint detection probability of all sensors for a target point z is calculated as follows.

$$P(C, \boldsymbol{z}) = 1 - \prod_{i=1}^{v} (1 - P(\boldsymbol{c}_i, \boldsymbol{z})) \tag{14}$$

In this paper, a grid-based approach is used to select the detection points $\boldsymbol{z}$. i.e. the detection area is discretized into $m \times n$ pixel points,, then the network coverage rate is calculated by the following equation.

$$P_{\text{cov}} = \frac{\sum_{x=1}^{m} \sum_{y=1}^{n} P(C, \boldsymbol{z})}{m \times n} \tag{15}$$

5.2 Experiments and analysis

In order to verify the effectiveness of the HLABC algorithm on real-world optimization problems, this section compares the HLABC algorithm with the other five excellent intelligent optimization algorithms mentioned earlier on the deployment of WSNs.

The specific optimization principle for WSNs is to treat the individuals within the algorithm population as a node deployment scheme, similar to (16), and then use the algorithm to continuously optimize the deployment scheme until the output coverage is optimal.

$$\begin{gathered} [c_1, c_2, \cdots, c_v] \\ \Downarrow \\ [(cx_1, xy_1), (cx_2, cy_2), \cdots, (cx_v, cy_v)] \end{gathered} \tag{16}$$

The simulation experiment environment is the same as that used for the performance test in Section 4, and in the

Table 14 The comparison results of HLABC with other algorithms on deployment of WSNs

Scenarios	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
scenario 1	0.8974±0.0061	0.9122±0.0034	0.9042 ± 0.0021	**0.9167±0.0064**	0.9084 ±0.0088	0.9112±0.0083
scenario 2	0.9614±0.0041	0.9624 ±0.0043	0.9641 ±0.0011	0.9665 ±0.0054	0.9555 ± 0.0083	**0.9693±0.0018**
scenario 3	0.8078 ± 0.0101	0.8273 ± 0.0077	0.8152 ± 0.0087	0.8316 ± 0.0141	0.8369 ± 0.0123	**0.8518 ±0.0063**
scenario 4	0.8922 ± 0.0157	0.8979 ± 0.0094	0.9034 ± 0.0079	0.9250 ± 0.0043	0.9250 ± 0.0178	**0.9413±0.0049**

experiment the $MaxFEs$ of each algorithm are set to 25000, the population size SN set to 50, and each group of simulation experiments is run 30 times independently, and the better once sensor distribution map is recorded.

This experiment designs four scenarios for deployment of WSNs. The scenario 1 and 2 are to deploy 15 and 20 sensors whose Rs is 5 in the 30x30 area respectively. And the scenario 3 and 4 are to deploy 30 and 40 sensors whose Rs is 10 in the 100x100 area respectively. The experimental results of the six algorithms on the four scenarios are shown in Table 14, and Fig. 13 shows four distributions of sensor, which are the best results of HLABC optimization.

It can be seen from Table 14 that HLABC algorithm is not as effective as ABC-TLBO algorithm on scenario 1, but is superior to other algorithms in the other three scenarios. In general, HLABC algorithm has better performance on deployment of WSNs.

6 Power scheduling problem in a smart home

This section introduces the Power Scheduling Problem in a Smart Home (PSPSH) [71] , in order to achieve the best effect of PSPSH, it is necessary to consider the influence of various factors, including electricity bill(EB), peak-to-average ratio(PAR) and user comfort level(UCL).

6.1 Types of appliances

In this paper, appliances are divided into two categories Shiftable Appliances(SAs) and Non-shiftable Appliances (NSAs) according to whether they can be operated automatically or not.

6.1.1 Shiftable appliances

SAs are able to operation automatically according to user's pre-defined time, such as air conditioner and washing machine. The vector $\boldsymbol{S}$ is used to represent SAs of a smart home.

$$\boldsymbol{S} = [s_1, s_2, \ldots, s_m] \tag{17}$$

Where s_i represents the ith SA.

Appliances can be scheduled to operate within a timeframe $\boldsymbol{T}$. The $\boldsymbol{T}$ is divided into several time slots as follows.

$$\boldsymbol{T} = \left[t^1, t^2, \ldots, t^n\right] \tag{18}$$

Where t^j denotes the jth time slot.

Thus the power demand of SAs in a smart home can be expressed using matrix (19).

$$\boldsymbol{PS} = \begin{bmatrix} ps_1^1 & ps_2^1 & \cdots & ps_m^1 \\ ps_1^2 & ps_2^2 & \cdots & ps_m^2 \\ \vdots & \vdots & \ldots & \vdots \\ ps_1^n & ps_2^n & \cdots & ps_m^n \end{bmatrix} \tag{19}$$

where ${ps_i}^j$ represents the power required by the ith SA in the jth time slot. The vectors $\boldsymbol{OTP}_s$ (20) and $\boldsymbol{OTP}_e$ (21) represent start time and end time of the allowed operations of SAs, respectively.

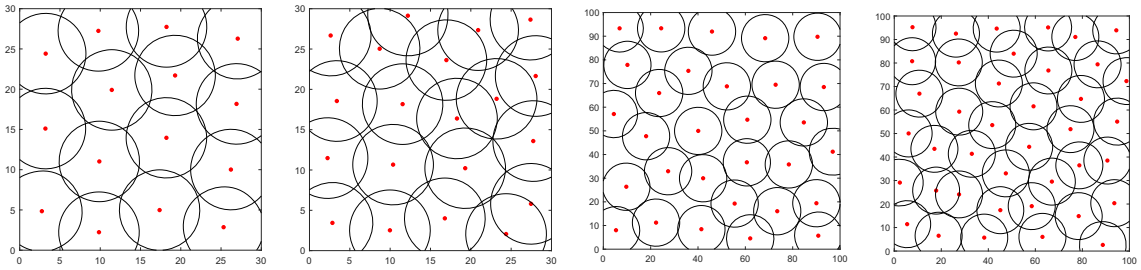
$$\boldsymbol{OTP}_s = [OTP_{s1}, OTP_{s2}, \ldots, OTP_{sm}] \tag{20}$$

$$\boldsymbol{OTP}_e = [OTP_{e1}, OTP_{e2}, \ldots, OTP_{em}] \tag{21}$$

where OTP_{si} and OTP_{ei} are the start time and end time of the allowed operation for the ith SA, respectively. And the vector $\boldsymbol{L}$ (22) is used to represent the time it takes for SAs to complete operation.

$$\boldsymbol{L} = [l_1, l_2, \ldots, l_m] \tag{22}$$

where l_i denotes the time required by the ith SA to finish operation. SAs should start and end operations at specific



(a) scenario 1(coverage rate %92 .4)

(b) scenario 2(coverage rate %97 .1)

(c) scenario 3(coverage rate %85 .8)

(d) scenario 4(coverage rate %94 .6)

Fig. 13 The sensor distribution after HLABC optimization

time slots, using the vectors $\boldsymbol{St}$ and $\boldsymbol{Et}$ to represent them.

$$\boldsymbol{St} = [st_1, st_2, \ldots, st_m] \tag{23}$$

$$\boldsymbol{Et} = [et_1, et_2, \ldots, et_m] \tag{24}$$

where st_i and et_i are the actual start and end operation time slots of the ith SA, respectively.

6.1.2 Non-shiftable appliances

The NSAs do not have pre-set operating times, such as TVs and lights, they run in real time based on user demand. The NSAs in a smart home are represented using the vector $\boldsymbol{NS}$.

$$\boldsymbol{NS} = \left[ns_1, ns_2, \ldots, ns_q\right] \tag{25}$$

where ns_i is the ith NSA. Meanwhile the power demand of NSAs is expressed using the vector $\boldsymbol{PNS}$ as follows.

$$\boldsymbol{PNS} = \left[pns_1, pns_2, \ldots, pns_q\right] \tag{26}$$

6.2 Electricity bill

Reducing electricity bill(EB) is the main objective of PSPSH optimization. EB is calculated using the following equation.

$$EB = \sum_{j=1}^{n} \sum_{i=1}^{m} ps_i^j \times pc^j \tag{27}$$

Where pc^j represents the electricity price in the jth time slot, and the pc^j is obtained according to (28).

$$pc^j = \begin{cases} a^j \ if \ 0 \leq ps^j \leq C \\ h^j \ if \ ps^j > C \end{cases} \tag{28}$$

$$h^j = \lambda \times a^j \tag{29}$$

where C denotes a threshold of electricity price change, a^j and h^j are the normal and higher electricity prices, respectively, and λ is the percentage increase of electricity price from a^j to h^j.

6.3 Peak-to-average ratio

PAR represents the ratio of maximum power consumption to average power consumption. for power systems, optimizing PAR can balance the system pressure distribution during the time of use and improve the life of the system. the formula for calculating PAR is as follows.

$$PAR = \frac{PS_{max}}{PS_{Avg}} \tag{30}$$

$$PS_{Avg} = \frac{\sum_{j=1}^{n} ps^j}{n} \tag{31}$$

where PS_{max} and PS_{avg} represent the maximum and average power demand within T, respectively.

Table 15 The main characteristics of the shiftable appliances

NO.	SAs	L	$OTP_s - OTP_e$	Power	No.	SAs	L	$OTP_s - OTP_e$	Power
1	Dishwasher	105	540- 780	1.2	19	Dehumidifier	30	1-120	0.25
2	Dishwasher	105	840-1080.	1.2	20	Dehumidifier	30	120- 240	0.25
3	Dishwrasher	105	1200-1440	1.2	21	Dehumidifier	30	240- 360	0.25
4	Air Conditioner	30	1-120	1.6	22	Dehumidifier	30	360-480	0.25
5	Air Conditioner	30	120- 240	1.6	23	Dehumidifier	30	480- 600	0.25
6	Air Conditioner	30	240-360	1.6	24	Dehumidifier	30	600 - 720	0.25
7	Air Conditioner	30	360- 480	1.6	25	Dehumidifier	30	720 - 840	0.25
8	Air Conditioner	30	480- 600	1.6	26	Dehumidifier	30	840 - 960	0.25
9	Air Conditioner	30	600- 720	1.6	27	Dehumidifier	30	960-1080	0.25
10	Air Conditioner	30	720- 840	1.6	28	Dehumidifier	30	1080-1200	0.25
11	Air Conditioner	30	840- 960	1.6	29	Dehumidifier	30	1200-1320	0.25
12	Air Conditioner	30	960 - 108o	1.6	30	Dehumidifier	30	1320-1440	0.25
13	Air Conditioner	30	1080-1200	1.6	31	Electric Water Heater	35	300-420	1.8
14	Air Conditioner	30	1200-1320	1.6	32	Electric Water Heater	35	1100-1440	1.8
15	Air Conditioner	30	1320- 1440	1.6	33	Coffee Maker	10	300-150	0.1
16	Washing Machine	55	60- 300	0.3	34	Coffee Maker	10	1020-1140	0.1
17	Clothes Dryer	60	300- 480	0.1	35	Robotic Pool Filter	180	1-540	1.5
18	Refrigerator	1440	1-1440	0.2	36	Robotic Pool Filter	180	900-1440	1.5

Table 16 Non-shiftable appliances

No.	NSAs	Power
1	Light	0.6
2	Attic Fan	0.3
3	Table Fan	0.8
4	Iron	1.5
5	Toaster	1
6	Computer Charger	1.5
7	Cleaner	1.5
8	TV	0.3
9	Hair Dryer	1.2
10	Hand Drill	0.6
11	Water Pump	2.5
12	Blender	0.3
13	Microwave	1.18
14	Electric Vehicle	1

6.4 User comfort level

In this paper, we improve the UCL by reducing the average waiting time WTR_{avg} for running SAs and decreasing the average capacity power limiting rate CPR_{avg} for running more NSAs with ps not exceeding C. First, the equation for calculating WTR_{avg} is as follows.

$$WTR_{avg} = \frac{\sum_{i=1}^{m}(st_i - OTP_{s_i})}{\sum_{i=1}^{m}(OTP_{e_i} - OTP_{s_i} - l_i)} \tag{32}$$

Then, the CPR_{avg} for all time slots is calculated using the following equation.

$$CPR_{avg} = \frac{\sum_{j=1}^{n}\sum_{k=1}^{q} ONA_k^j}{q \times n} \tag{33}$$

$$ONA_k^j = \begin{cases} 0 \; if \;\; pns_k < AP^j \\ 1 \quad otherwise \end{cases} \tag{34}$$

$$AP^j = C - ps^j \tag{35}$$

where C and ps^j have the same meaning as in (28).

6.5 Objective function

Since the PSPSH needs to consider EB, PAR and UCL simultaneously, it is a multi-objective optimization problem. In this paper, the multi-objective problem is transformed into a single objective function by assigning weights to different objective functions as follows.

$$\min F(X) = w_1 \times \frac{EB}{EB + A} + w_2 \times \frac{PAR}{PAR + B} + w_3 \times WTR_{avg} + w_4 \times CPR_{avg} \tag{36}$$

where w_1, w_2, w_3 and w_4 are the weight parameters which represent importance of objective functions, A and B are two positive numbers.

6.6 Experiments and analysis

This section compares the proposed HLABC algorithm with five other ABC variants on the PSPSH. Appliances used in the experiment are from paper [54] and characteristics of appliances are shown in Tables 15 and 16.

Seven scenarios were designed for the PSPSH simulation experiments. The specific scenarios are shown in Table 17. In this simulation experiment, environment is the same as in Section 4, and the $MaxFEs$ and SN are set to 5000 and 50, respectively. Each group of simulation experiments is run 30 times independently. For the parameters, this paper refers to the settings in paper [54]. The weights w_1, w_2, w_3 and w_4 are set to 0.4,0.2,0.2 and 0.2 respectively, the threshold C and ratio λ are 0.0333 and 1.543 respectively.

Specifically, the optimization process of PSPSH is to use the start time St of different appliances as the dimension of each individual in the population, and then find the optimal start time for each appliance between OTP_s and OTP_e.

The experimental results are shown in Table 18, it can be seen that the HLABC algorithm achieves the best results in scenarios 1,2 and 6.

Table 17 Scenarios of shiftable appliances

Scenarios	SAs	dimension
scenario 1	1,3,4,6,7,15,18,20,21,24,26,27,29,33,35	15
scenario 2	1,2,4,5,6,7,10,11,12,18,25,26,27,28,29,31,32,33,34,36	20
scenario 3	3,4,5,6,7,8,9,10,11,12,13,14,15,18,23,24,25,26,27,28,31,32,33,34,35	25
scenario 4	1,2,3,5,7,8,11,12,13,16,17,18,20,21,23,24,25,26,27,28,29,30,31,32,33,34,35,36	28
scenario 5	3,4,5,6,7,8,9,10,11,12,13,14,15,17,18,19,20,21,23,24,25,26,27,28,29,31,32,33,34,35	30
scenario 6	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,33,34,35	34
scenario 7	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36	36

Table 18 The comparison results of HLABC with other algorithms on PSPSH

Scenarios	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
scenario 1	0.3532 ± 0.0053	0.3371 ± 0.0038	0.3388 ± 0.0037	**0.3341±0.0000**	0.3378 ± 0.0084	**0.3341±0.0000**
scenario 2	0.3689 ± 0.0066	0.3604 ± 0.0053	0.3619 ± 0.0028	0.3540 ± 0.0004	0.3548 ± 0.0007	**0.3533±0.0036**
scenario 3	0.3756 ± 0.0030	0.3625 ± 0.0051	0.3657 ± 0.0038	**0.3479±0.0014**	0.3521 ± 0.0035	0.3500 ± 0.0032
scenario 4	0.3701 ± 0.0159	0.3612 ± 0.0090	0.3606 ± 0.0053	0.3410 ± 0.0014	**0.3407±0.0021**	0.3418 ± 0.0043
scenario 5	0.3789 ± 0.0077	0.3732 ± 0.0056	0.3743 ± 0.0051	**0.3538±0.0031**	0.3553 ± 0.0012	0.3551 ± 0.0026
scenario 6	0.3822 ± 0.0052	0.3708 ± 0.0027	0.3733 ± 0.0053	0.3475 ± 0.0026	0.3477 ± 0.0019	**0.3464±0.0028**
scenario 7	0.3676 ± 0.0070	0.3752 ± 0.0045	0.3662 ± 0.0047	0.3421 ±0.0034	**0.3386±0.0056**	0.3403 ± 0.0058

7 Multi-thresholding image segmentation

Image segmentation is a technique for simplifying images, which reduces the complexity of the image and is a critical step in image analysis. Kapur et al. proposed a simple and effective image segmentation technique called Kapur's entropy method, which yields excellent segmentation results [72].

7.1 Kapur's entropy method

For an image with a gray interval of $[0, L-1]$, the image is divided into M subclasses: $C_0 = \{0, 1, \cdots, t\}$, $C_1 = \{t_1+1, t_1+2, \cdots, t_2\}$, $C_{M-1} = \{t_{M-1}+1, t_{M-1}+2, \cdots, L-1\}$, and the entropy is defined by (37).

$$\phi(t_1, t_2, \cdots, t_{M-1}) = -\sum_{i=0}^{t_1} \frac{p_i}{P_0} \cdot \ln \frac{p_i}{P_0} - \cdots \sum_{i=t_1+1}^{t_2} \frac{p_i}{P_1} \cdot \ln \frac{p_i}{P_1} - \sum_{i=t_{M-1}+1}^{L-1} \frac{p_i}{P_{M-1}} \cdot \ln \frac{p_i}{P_{M-1}} \tag{37}$$

where $\{t_1, t_2, \cdots, t_{M-1}\}$ are the thresholds used in image segmentation, and $P_j = \sum_{i \in C_j} p_i$, $j = 0, 1, \cdots, M-1$.

Therefore, the optimal thresholds are determined by:

$$\{t_1^*, t_2^*, \cdots, t_{M-1}^*\} = \underset{0<t1<\cdots<t_{M-1}<L-1}{\arg\max} \phi(t_1, t_2, \cdots, t_{M-1}) \tag{38}$$

7.2 Experiment and analysis

In order to test the performance of HLABC in image segmentation, four commonly used test images, Cameraman, Sailboat, Monkey, and Tank, are used for testing. In the process of optimizing image segmentation, each threshold is a different dimension of the population, so an individual represents a threshold scheme for image segmentation. During the algorithm's run, the threshold continues to evolve towards the optimal direction until the final optimal solution is output.

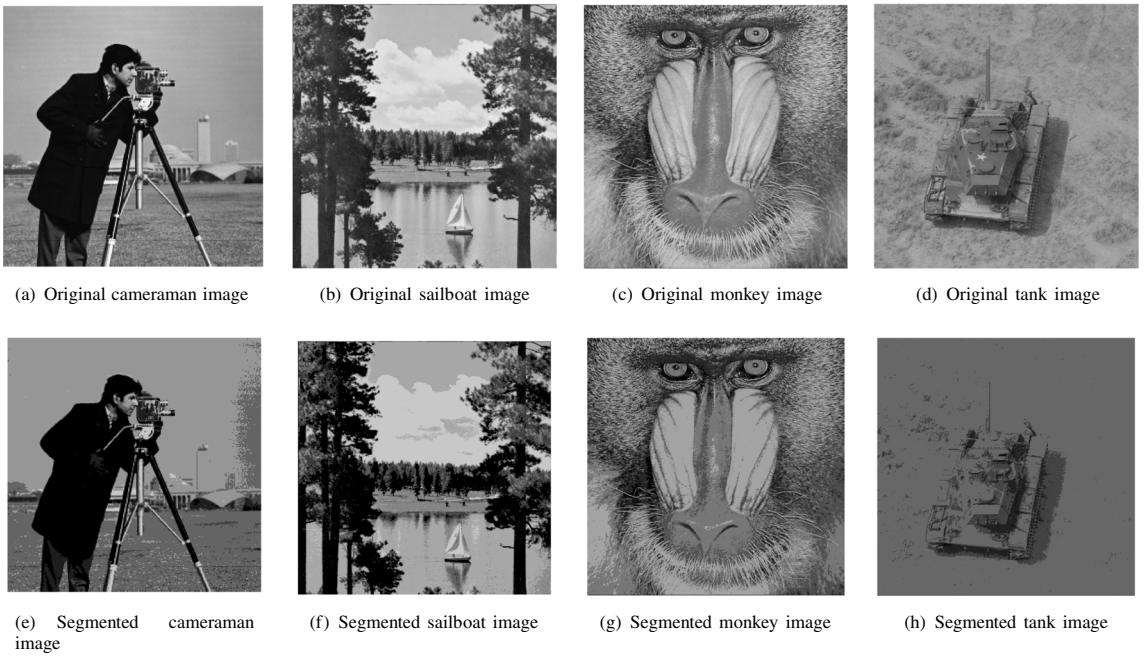
In this section, the images are segmented into 5 levels, and MaxFEs was set to 10000. The experimental results are shown in Table 19. From the table, it can be observed that the HLABC algorithm demonstrates the best performance on Cameraman and Tank images, followed by Sailboat and Monkey images where HLABC's performance is slightly below that of MEABC. Additionally, Fig. 14 displays the original images and the segmented images obtained using HLABC. From the figures, it is evident that the images segmented by HLABC exhibit excellent quality.

8 Conclusion

This paper introduces a novel ABC algorithm named HLABC, which is based on a hierarchical learning topology. In HLABC, three novel modification strategies are designed to enhance the global searching ability of the ABC algorithm. Firstly, a hierarchical ring topology is proposed, dividing the entire population into three layers based on individual fit-

Table 19 The comparison results of HLABC with other algorithms on image segmentation

Scenarios	ABC Mean±Std	GABC Mean±Std	MEABC Mean±Std	ABC-TLBO Mean±Std	MGABC Mean±Std	HLABC Mean±Std
Cameraman	21.170±0.008	21.167±0.018	21.173±0.011	21.174±0.026	21.163±0.043	**21.176±0.014**
Sailboat	21.034±0.030	21.028±0.015	**21.061±0.036**	21.042±0.038	21.031±0.019	21.043±0.048
Monkey	21.262±0.040	21.271±0.022	**21.293±0.007**	21.261±0.070	21.182±0.215	21.278±0.035
Tank	18.690±0.043	18.733±0.016	18.724±0.026	18.720±0.026	18.745±0.000	**18.753±0.041**



(a) Original cameraman image (b) Original sailboat image (c) Original monkey image (d) Original tank image

(e) Segmented cameraman image (f) Segmented sailboat image (g) Segmented monkey image (h) Segmented tank image

Fig. 14 The images segmented by HLABC

ness levels. Each individual is then updated using differential learning among the three layers. Secondly, two new search strategies are established based on the hierarchical ring topology. Finally, the scout bee phase is improved by incorporating hierarchical and opposition-based learning.

The performance of HLABC is compared with other ABC variants on three test suites that include the CEC2005, CEC2010, and CEC2013 benchmark functions. Comprehensive experimental results demonstrate that HLABC significantly outperforms other ABC variants on most testing problems. Moreover, the performance of HLABC remains consistent for problems with dimensions $D = 30$ and $D = 100$. Statistical tests, including the Friedman test and Wilcoxon rank sum test, further confirm that HLABC is an efficient and powerful optimizer compared to other algorithms. The statistical boxplots also provide intuitive evidence of HLABC's superior performance, making it a promising choice for tackling various optimization problems.

To verify the performance of HLABC in real-world optimization problems, this paper applies the algorithm to the deployment of WSNs, PSPSH and multi-thresholding image segmentation. The optimization results demonstrate that HLABC exhibits better optimization ability in solving real-world problems.

However, it is worth noting that the success of HLABC relies on the hierarchical learning topology with a fixed proportion of the population. Future research may explore adaptive proportions based on the specific problems being solved. Additionally, there is potential for applying HLABC to multi-objective optimization and other real-world applications, which opens up exciting avenues for future research.

Acknowledgements This work is supported by the National Natural Science Foundation of China (62006144, 61976127), the Major Fundamental Research Project of Shandong, China (No. ZR2019ZD03), the Natural Science Foundation of Shandong Province (ZR2022MF346), and the Taishan Scholar Project of Shandong, China (No.ts20190924).

Author Contributions All authors contributed to the study's conception and design. Material preparation, data collection, and analysis were performed by Qingke Zhang and Tianqi Li. The first draft of the manuscript was written by Qingke Zhang, Tianqi Li, Xianglong Bu, and Hao Gao. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript. In addition, we would like to thank two contributors outside of the current author list: Yanyan Tan and Lei Lv. Yanyan Tan completed the grammar check and grammar revision work on the revised manuscript. Lei Lv completed the additional experiments in the revised manuscript.

Data Availability The data and materials supporting the findings of this study have been explicitly described in the manuscript or are accessible through public repositories. Key algorithms used in this research have been presented in the form of pseudocode within this paper or can be provided upon request. We are committed to ensuring the transparency and accessibility of relevant data and resources, facilitating the reproducibility and validation of the research outcomes.

Declarations

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

Ethical and Informed Consent for Data Used The data utilized in this study were collected and processed in accordance with ethical guidelines and principles. All necessary informed consents were obtained from relevant parties or authorized sources prior to the use of the data. Proper protocols were followed to ensure the privacy, confidentiality, and proper handling of sensitive information.

References

1. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the sixth international symposium on micro machine and human science, pp 39–43. IEEE
2. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359
3. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical Report
4. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Glob Optim 39(3):459–471
5. Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132
6. Dorigo M, Di Caro G (1999) Ant colony optimization: a new metaheuristic. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406), vol 2, pp 1470–1477. IEEE
7. Kler R, Gangurde R, Elmirzaev S, Hossain MS, Vo NV, Nguyen TV, Kumar PN (2022) Optimization of meat and poultry farm inventory stock using data analytics for green supply chain network. Discret Dyn Nat Soc 2022:1–8
8. Forghany Z, Davarynejad M, Snaar-Jagalska BE (2012) Gene regulatory network model identification using artificial bee colony and swarm intelligence. In: 2012 IEEE Congress on evolutionary computation, pp 1–6
9. Shunmugapriya P, Kanmani S (2013) Optimization of stacking ensemble configurations through artificial bee colony algorithm. Swarm Evol Comput 12:24–32
10. Kojima M, Nakano H, Miyauchi A (2013) An artificial bee colony algorithm for solving dynamic optimization problems. In: 2013 IEEE Congress on evolutionary computation, pp 2398–2405
11. Yang L, Sun X, Peng L, Shao J, Chi T (2015) An improved artificial bee colony algorithm for optimal land-use allocation. Int J Geogr Inf Sci 29(8):1470–1489
12. Greenwood GW, Chopra S (2013) A modified artificial bee colony algorithm for solving large graph theory problems. In: 2013 IEEE Congress on evolutionary computation, pp 713–717
13. Vural RA, Yildirim T, Kadioglu T, Basargan A (2012) Performance evaluation of evolutionary algorithms for optimal filter design. IEEE Trans Evol Comput 16(1):135–147
14. Zangeneh MA, Ghazvini M (2017) An energy-based clustering method for wsns using artificial bee colony and genetic algorithm. In: 2017 2nd Conference on swarm intelligence and evolutionary computation (CSIEC), pp 35–41
15. Fong CW, Asmuni H, McCollum B (2015) A hybrid swarm-based approach to university timetabling. IEEE Trans Evol Comput 19(6):870–884
16. Dara S, Vaishnavai P, NageswaraRao B, Ravi KJ (2018) Artificial bee colony algorithm: A survey and recent applications. Int J Pure Appl Math 120(6):313–321
17. Agarwal SK, Yadav S (2019) A comprehensive survey on artificial bee colony algorithm as a frontier in swarm intelligence. In: Hu Y-C, Tiwari S, Mishra KK, Trivedi MC (eds) Ambient communications and computer systems. Springer, Singapore, pp 125–134
18. Ghambari S, Rahati A (2018) An improved artificial bee colony algorithm and its application to reliability optimization problems. Appl Soft Comput 62:736–767
19. Gao Y, Li X, Dong M, Li H-p (2018) An enhanced artificial bee colony optimizer and its application to multi-level threshold image segmentation. J Cent South Univ 25(1):107–120
20. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173
21. Lin A, Sun W, Yu H, Wu G, Tang H (2019) Global genetic learning particle swarm optimization with diversity enhancement by ring topology. Swarm Evol Comput 44:571–583
22. Bhambu P, Sharma S, Kumar S (2018) Modified gbest artificial bee colony algorithm. In: Pant M, Ray K, Sharma TK, Rawat S, Bandyopadhyay A (eds) Soft computing: theories and applications. Springer, Singapore, pp 665–677
23. Lynn N, Suganthan PN (2015) Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation. Swarm Evol Comput 24:11–24
24. Lin Q, Zhu M, Li G, Wang W, Cui L, Chen J, Lu J (2018) A novel artificial bee colony algorithm with local and global information interaction. Appl Soft Comput 62:702–735
25. Tsai H-C (2020) Artificial bee colony directive for continuous optimization. Appl Soft Comput 87:105982
26. Durgut R, Aydin ME (2021) Adaptive binary artificial bee colony algorithm. Appl Soft Comput 101:107054
27. Xiang W-l, Meng X-l, Li Y-z, He R-c, An M-q (2018) An improved artificial bee colony algorithm based on the gravity model. Inf Sci 429:49–71
28. Kong D, Chang T, Dai W, Wang Q, Sun H (2018) An improved artificial bee colony algorithm based on elite group guidance and combined breadth-depth search strategy. Inf Sci 442:54–71
29. Zhou X, Wu Y, Zhong M, Wang M (2021) Artificial bee colony algorithm based on multiple neighborhood topologies. Appl Soft Comput 111:107697
30. Pian J, Wang G, Li B (2018) An improved abc algorithm based on initial population and neighborhood search. IFAC-PapersOnLine 51(18):251–256
31. Borowska B (2020) Genetic learning particle swarm optimization with interlaced ring topology. In: International conference on computational science, pp 136–148. Springer
32. Wang H, Wang W, Xiao S, Cui Z, Xu M, Zhou X (2020) Improving artificial bee colony algorithm using a new neighborhood selection mechanism. Inf Sci 527:227–240
33. Song X, Zhao M, Yan Q, Xing S (2019) A high-efficiency adaptive artificial bee colony algorithm using two strategies for continuous optimization. Swarm Evol Comput 50:100549
34. Chen Y, Li L, Peng H, Xiao J, Wu Q (2018) Dynamic multi-swarm differential learning particle swarm optimizer. Swarm Evol Comput 39:209–221
35. Wang H, Wu Z, Rahnamayan S, Sun H, Liu Y, Pan J-s (2014) Multi-strategy ensemble artificial bee colony algorithm. Inf Sci 279:587–603
36. Barani F, Mirhosseini M (2018) Classification of binary problems with svm and a mixed artificial bee colony algorithm. In: 2018 3rd Conference on swarm intelligence and evolutionary computation (CSIEC), pp 1–9
37. Beed R, Roy A, Sarkar S, Bhattacharya D (2020) A hybrid multi-objective tour route optimization algorithm based on particle swarm optimization and artificial bee colony optimization. Comput Intell 36(3):884–909
38. Zhong Y, Lin J, Wang L, Zhang H (2017) Hybrid discrete artificial bee colony algorithm with threshold acceptance criterion for traveling salesman problem. Inf Sci 421:70–84

39. Liang JJ, Qin AK, Suganthan PN (2006) Baskar S Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295
40. Wang F, Li Y, Zhou A (2019) Tang K An estimation of distribution algorithm for mixed-variable newsvendor problems. IEEE Trans Evol Comput 24(3):479–493
41. Fong CW, Asmuni H, McCollum B (2015) A hybrid swarm-based approach to university timetabling. IEEE Trans Evol Comput 19(6):870–884
42. Attia A-F, Abd Elaziz M, Hassanien AE (2020) El-Sehiemy RA Prediction of solar activity using hybrid artificial bee colony with neighborhood rough sets. IEEE Transactions on Computational Social Systems 7(5):1123–1130
43. Jadon SS, Tiwari R, Sharma H (2017) Bansal JC Hybrid artificial bee colony algorithm with differential evolution. Appl Soft Comput 58:11–24
44. Wang C-N, Yang F-C, Nguyen VTT (2022) Vo NT Cfd analysis and optimum design for a centrifugal pump using an effectively artificial intelligent algorithm. Micromachines 13(8):1208
45. Liang J-J, Suganthan PN, Deb K Novel composition test functions for numerical global optimization. In:Proceedings 2005 IEEE swarm intelligence symposium, 2005. SIS 2005., pp 68–75 (2005). IEEE
46. Tang K, Li X, Suganthan P, Yang Z, Weise T (2010) Benchmark functions for the cec2010 special session and competition on large-scale global optimization
47. Liang J, Qu B, Suganthan P, Hernández-Díaz AG (2013) Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report 201212(34):281–295
48. Sun B, Sun Y, Li W (2022) Multiple topology shade with tolerance-based composite framework for cec2022 single objective bound constrained numerical optimization. In: 2022 IEEE congress on evolutionary computation (CEC), pp 1–8. IEEE
49. Maheshwari P, Sharma AK (2021) Verma K Energy efficient cluster based routing protocol for wsn using butterfly optimization algorithm and ant colony optimization. Ad Hoc Networks 110:102317
50. Borges LM, Velez FJ, Lebres AS (2014) Survey on the characterization and classification of wireless sensor network applications. IEEE Communications Surveys & Tutorials 16(4):1860–1890
51. Yılmaz Faruk, Pardalos PM (2017) Minimizing average lead time for the coordinated scheduling problem in a two-stage supply chain with multiple customers and multiple manufacturers. Comput Ind Eng 114:244–257
52. Khalid A, Javaid N, Guizani M, Alhussein M, Aurangzeb K (2018) Ilahi M Towards dynamic coordination among home appliances using multi-objective energy optimization for demand side management in smart buildings. IEEE Access 6:19509–19529
53. Dileep G (2020) A survey on smart grid technologies and applications. Renewable Energy 146:2589–2625
54. Makhadmeh SN, Khader AT, Al-Betar MA, Naim S, Abasi AK (2021) Alyasseri ZAA A novel hybrid grey wolf optimizer with min-conflict algorithm for power scheduling problem in a smart home. Swarm Evol Comput 60:100793
55. Kuruvilla J, Sukumaran D, Sankar A, Joy SP (2016) A review on image processing and image segmentation. In: 2016 International conference on data mining and advanced computing (SAPIENCE), pp 198–203
56. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9(1):62–66
57. Kapur JN, Sahoo PK (1985) Wong AK A new method for gray-level picture thresholding using the entropy of the histogram. Compt Vis Graph Image Process 29(3):273–285
58. Rahnamayan S, Tizhoosh HR (2008) Salama MM Opposition-based differential evolution. IEEE Trans Evol Comput 12(1):64–79
59. Rojas-Morales N, Riff Rojas M-C (2017) Montero Ureta E A survey and classification of opposition-based metaheuristics. Comput Ind Eng 110:424–435
60. Xiao S, Wang H, Wang W, Huang Z, Zhou X (2021) Xu M Artificial bee colony algorithm based on adaptive neighborhood search and gaussian perturbation. Appl Soft Comput 100:106955
61. Zhang M, Pan Y, Zhu J, Chen G (2018) Abc-tlbo: A hybrid algorithm based on artificial bee colony and teaching-learning-based optimization. In: 2018 37th Chinese Control Conference (CCC), pp 2410–2417. IEEE
62. Zhang X, Lou Y, Yuen SY, Wu Z, He Y, Zhang X (2019) Hybrid artificial bee colony with covariance matrix adaptation evolution strategy for economic load dispatch. In: 2019 IEEE Congress on evolutionary computation (CEC), pp 204–209
63. Chen L, Liu H-L, Tan KC, Li K (2021) Transfer learning based parallel evolutionary algorithm framework for bi-level optimization.IEEE Trans Evol Comput, pp 1–1
64. Zhou X, Lu J, Huang J, Zhong M (2021) Wang M Enhancing artificial bee colony algorithm with multi-elite guidance. Inf Sci 543:242–258
65. Santos R, Borges G, Santos A, Silva M, Sales C (2018) Costa JCWA A semi-autonomous particle swarm optimizer based on gradient information and diversity control for global optimization. Appl Soft Comput 69:330–343
66. Nickabadi A, Ebadzadeh MM (2011) Safabakhsh R A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11(4):3658–3670
67. Zhong X, Cheng P (2021) An elite-guided hierarchical differential evolution algorithm. Appl Intell 51:4962–4983
68. Li D, Guo W, Lerch A, Li Y, Wang L, Wu Q (2021) An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization. Swarm Evol Comput 60:100789
69. Khedr AM, Osamy W (2009) Agrawal DP Perimeter discovery in wireless sensor networks. J Parallel Distrib Comput 69(11):922–929
70. Himanshu, Khanna R, Kumar A (2022) Artificial intelligence applications for target node positions in wireless sensor networks using single mobile anchor node. Comput Ind Eng 167:107998
71. Akbari-Dibavar A, Nojavan S, Mohammadi-Ivatloo B (2020) Zare K Smart home energy management using hybrid robust-stochastic optimization. Comput Ind Eng 143:106425
72. Kapur JN, Sahoo PK (1985) Wong AKC A new method for gray-level picture thresholding using the entropy of the histogram. Computer Vision, Graphics, and Image Processing 29(3):273–285