IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 May 2024, accepted 16 June 2024, date of publication 21 June 2024, date of current version 10 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3417530

RESEARCH ARTICLE

Grouping and Reflection of the Artificial Bee Colony Algorithm for High-Dimensional Numerical Optimization Problems

SONGYUT PHOEMPHON
Institute of Digital Arts and Science (DIGITECH), Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand
e-mail: songyut@sut.ac.th

This work was supported in part by the Institute of Digital Arts and Science (DIGITECH), Suranaree University of Technology (SUT); in part by Thailand Science Research and Innovation (TSRI); and in part by the National Science, Research and Innovation Fund (NSRF) under Project 179287.

ABSTRACT The artificial bee colony (ABC) algorithm, inspired by the cooperative foraging behaviors observed in bees, is a prominent example of a swarm intelligence algorithm that offers significant advantages in optimization problems. However, the efficacy of the ABC algorithm is limited in high-dimensional scenarios or when handling multimodal functions, which contain many local optima because of the random nature of the one-dimensional search process for improving the position in the employed and onlooker bee phases. As a result, ABC has a limited ability to obtain the optimum result (slow convergence rate). To address this limitation, this research introduces Grouping and Reflection of the Artificial Bee Colony (GRABC), a distinctive adaptation of the traditional ABC algorithm meticulously tailored to meet the specific demands of high-dimensional numerical optimization problems by balancing exploration and exploitation processes. GRABC strategically incorporates vector reflection and inertial weighting to formulate equations *vleft* and *vright*, which enhance both the employed and onlooker bee phases, substantially improving the convergence speed and improving the exploitation process. Moreover, the integration of grouping bees facilitates the exploration of food sources by promoting diversification and improving the exploration process. Additionally, an equation is derived to accurately compute the new positions of scout bees (exploration process), accounting for the possibility of becoming stuck in local optima and considering the proper limit values. The effectiveness of GRABC is thoroughly evaluated using 32 numerical benchmark functions, mostly including CEC 2017, which encompasses 100 dimensions. The empirical findings compellingly demonstrate that the GRABC algorithm outperforms alternative methodologies in terms of solution quality and convergence characteristics, as substantiated by comprehensive assessments that include metrics such as the worst, best, and average results and standard deviations.

INDEX TERMS Optimization problem, swarm intelligence, artificial bee colony, grouping, vector reflection.

I. INTRODUCTION

Attaining optimal solutions in the realm of optimization problems is often considerably challenging due to their considerable growth and intricate nature, which necessitate substantial computational resources. Nevertheless, the application of the swarm intelligence (SI) technique presents a promising avenue for addressing high-dimensional optimization problems or handling multimodal functions. SI comprises a collective of individual agents, each dynamically updating its position based on the positions of other agents. SI arises from new behaviors in which these agents interact with one another to produce a solution [1]. The fundamental ideas of collective intelligence are inspired by a variety of natural systems, with an emphasis on ecological perspectives. Notably, various SI techniques, including the genetic algorithm (GA) [2], differential evolution (DE) [3], ant colony optimization (ACO) [4], particle swarm optimization (PSO) [5], cuckoo search (CS) [6], firefly algorithm (FA) [7], bacterial foraging algorithm (BFA) [8], biogeography-based optimization (BBO) [9], and

The associate editor coordinating the review of this manuscript and approving it for publication was Guolong Cui.

artificial bee colony algorithm (ABC) [10], have exhibited impressive efficacy in effectively tackling optimization challenges.

The genetic process of selection, crossover, and mutation is employed within the GA framework of the search algorithm known as GA [2]. DE [3] adopts a mutation process to generate novel agents by merging the positions of two preexisting agents from the population. If an agent's new position exhibits improvement, it is deemed favorable and subsequently integrated into the population; conversely, it is disregarded. ACO [4] embodies an algorithm inspired by the foraging behavior of ants and demonstrates remarkable proficiency in locating the shortest path between food sources and nests. During traversal, ants communicate by depositing pheromones on the ground, thereby establishing a chemical signaling system. Consequently, routes with a higher concentration of pheromones are preferred. PSO [5] is a search algorithm that hinges upon interactions among particles. Moreover, the underlying principle entails the exchange of information about both personal best and global best positions, thereby optimizing the placement of each individual particle.

The obligatory brood parasitism observed in specific species of cuckoo, wherein eggs are deposited in the nests of host birds of different species, serves as the fundamental inspiration behind CS [6]. Within this algorithm, each egg within a nest represents a potential solution, and each cuckoo egg signifies a new solution. The primary objective entails harnessing the potential of cuckoos to supplant existing solutions within nests with novel and superior alternatives. The firefly algorithm, introduced by Yang [7], is a swarm algorithm that simulates the communication among fireflies through the emission of light flashes. Assuming that all fireflies are unisex, any firefly can be attracted to another based on its attractiveness, which is contingent upon brightness, an attribute governed by the objective function. Fireflies tend to gravitate toward more visible counterparts. Furthermore, in adherence to the inverse square law, brightness diminishes with increasing distance. BFA [8] is a swarm intelligence algorithm inspired by the foraging behavior of *Escherichia coli*, an organism that inhabits the human intestine and is renowned for its food-seeking and toxin-avoidance strategies. Additionally, it is able to release chemicals to attract other bacteria.

The ABC algorithm is an optimization technique that emulates the foraging behavior of honeybees. Three distinct groups of bees are included: employed bees, onlookers, and scouts [10]. The population of employed bees in the colony corresponds to the number of available food sources, with each food source assigned a single bee at any given time. The employed bees venture to their designated food sources, diligently gather nectar, and return to the hive. Upon their return, they engage in a unique dance, effectively communicating valuable information regarding the quality of the food source to their fellow colony members.

The dances performed by the employed bees serve as signals for the onlooker bees, who decide to follow the employed bees with high fitness solutions randomly, highlighting the exploitation characteristic. When an employed bee's food source is depleted, it transforms into a scout bee, signifying the presence of a local optimum and the subsequent need to seek out a new food source.

Numerous investigational results have shown that ABC enhancement is superior to or at least on par with other optimization approaches, such as GA, DE, and PSO [10], [11], [12]. ABC has also been successfully used to address a variety of issues, including multiobjective optimization [13], [14], clustering [15], [16], imaging satellite mission planning [17], and UAV path planning [18]. The basic ABC algorithm requires fewer control parameters than other optimization algorithms. In addition, this approach can be easily implemented in practical applications [10].

Similar to other SI algorithms, ABC updates a bee's position by selecting another particle at random to guide its search for new food sources during the employed bee phase, as depicted in (2). Consequently, ABC's random search strategy offers advantages in terms of exploring new regions within the search space, promoting diversity in the exploration process, and aiding in escaping local optima. Nevertheless, the absence of exploitation can result in a slow convergence rate due to the completely random search strategy, as previously reported [10], [22], [27], [31], [33], [34]. This limitation is especially pronounced in high-dimensional spaces, where it affects only a single parameter of the parent solution, essentially performing a one-dimensional random search. As a result, several strategies [11], [12], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44] have been suggested to enhance ABC performance in this regard.

Similarly, ABC exhibits a lower convergence rate in multimodal functions than in unimodal functions due to the presence of a greater number of local optima, primarily because ABC has limited exploitation ability (one-dimensional random search). Consequently, addressing high-dimensional and multimodal problems is significantly challenging for researchers aiming to enhance the performance of ABC. In summary, ABC has been explored effectively. However, its lack of exploitation capability leads to slow convergence, particularly for high-dimensional and multimodal functions.

In addition, during the scout bee phase, new positions are randomly assigned to bees that are stuck in local optima to facilitate their escape [10]. However, the new position of the scout bee particle, as depicted in (1), is random and does not consider information from other potentially useful bee particles that could accelerate convergence.

Due to the lack of exploitation of employed bees (one-dimensional random search) and the randomness of the new position of the scout bee, in this paper, several significant contributions are introduced with the aim of enhancing the

performance of the ABC algorithm in effectively addressing the abovementioned challenges (slow convergence rate, especially in high-dimensional or multimodal functions). These contributions are organized into three distinct sections. First, we propose a novel grouping strategy for bees and introduce an equation that utilizes the best position within each group (x_{lbest}) to obtain more diverse search positions for individual bees. Second, during the employed bee and onlooker bee stages, we employ an equation based on x_{lbest}, the inertial weight, and vector reflection to determine improved positions, ensuring that new positions surpass the current positions and leading to rapid exploitation (a fast convergence rate). Finally, we find a suitable limit value for our improvement algorithm in the employed and onlooker bee phases. We designed an equation that calculates the locations of scout bees based on the limit value by averaging the positions of bees (x_t) with a $trial_t$ count of 0, considering x_{lbest}. The key contributions of this study are succinctly summarized below.

- In this paper, we propose dividing the bees and utilizing the bee with the best fitness value from each group to determine the direction for enhancing the position of the others within the same group for more diverse searching.
- We propose using vector reflection to enhance the search position of bees in each round, aiming for faster convergence, particularly in scenarios with a high number of dimensions or when handling multimodal functions.
- We propose a method for determining a bee's position when it becomes stuck in a local optimum by averaging the positions of bees (x_t) with a trial count ($trial_t$) equal to 0 and x_{lbest} within the group.

The paper is organized as follows. In Section II, we explain the fundamental concepts of the ABC algorithm. In Section III, we present a comprehensive overview of related works on ABCs, with a specific focus on the evolution of ABCs. In Section IV, we present a detailed account of our proposed scheme, encompassing the grouping of bees, improvements to the employed and onlooker bee phases, and an enhanced scout bee approach. We present the experimental results that validate the performance of our proposed method in Section V. In Section VI, we discuss the study in detail. Finally, in Section VII, we provide the conclusions drawn from our research and outline potential avenues for future work.

II. ARTIFICIAL BEE COLONY

Numerical tasks serve as a means to evaluate the efficacy of the ABC method, which was originally introduced in 2005 [10]. The fundamental objective of the ABC algorithm is to address optimization challenges by discerning an optimal or near-optimal solution on a global scale. This feat is accomplished through the emulation of foraging patterns observed within bee colonies. The bee populace is divided into two distinct factions: employed bees and onlooker bees. Employed bees undertake expeditions to procure nectar from food sources while simultaneously accumulating pertinent positional intelligence regarding these sources. Onlooker bees, in turn, rely upon the information disseminated by employed bees to conduct their own quest for nourishment. Once a food source has been fully depleted, employed bees promptly relinquish it and transform into scout bees to unearth fresh provisions. In accordance with the ABC algorithm, the quantity of food sources corresponds precisely to the number of employed bees, while reciprocally, the number of employed bees mirrors that of the onlookers. The intricacies of the foraging process in ABC warrant further exploration.

- The bees are initially positioned using (1).

$$x_{i,j} = x_j^L + rand\left(x_j^U - x_j^L\right) \tag{1}$$

where $x_{i,j}$ is the i^{th} employed bee of the i^{th} food source at the j^{th} dimension; x_j^U and x_j^L are the upper and lower bounds of the j^{th} dimension, respectively; and *rand* is a random number between 0 and 1.

Let D be the number of dimensions of the optimization problem, where $j = 1, 2, 3, \ldots, D$. The number of employed bees is NF, where $i = 1, 2, 3, \ldots, NF$.

- Then, employed bees search for nearby food sources by creating candidate food sources ($v_{i,j}$ is the i^{th} candidate food source of the j^{th} dimension) surrounding x_{ij}. v_{ij} can be calculated by another random food source (r^{th}) in the j^{th} dimension ($x_{r,j}$) according to (2). Moreover, the value of $v_{i,j}$ must be in the range $[x_j^L \le v_{i,j} \le x_j^U]$.

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

where $\varphi_{i,j}$ is a random number in $[-1,1]$ and $i \neq r$.

- The fitness value for the minimization problem of $x_{i,j}$ is calculated as shown in (3).

$$fit_i = \begin{cases} \dfrac{1}{1+f_i}, & f_i \ge 0 \\ 1 + |f_i|, & f_i < 0 \end{cases} \tag{3}$$

where f_i is the value of the objective function of the i^{th} food source.

- The i^{th} food source is replaced with the best position between $x_{i,j}$ and $v_{i,j}$ by comparing the fitness values.
- Moreover, similar to employed bees, onlookers forage for food sources, especially food sources with good fitness values. The probability (p_i) is used to determine the food source an onlooker selects, as stipulated by (4).

$$p_i = \frac{fit_i}{\sum_{m=1}^{NF} fit_m} \tag{4}$$

If p_i is greater than a random number from 0 to 1, onlookers (x_i) search for nearby food sources.

- Finally, as the optimization problem becomes more challenging as the number of rounds progresses (commonly referred to as approaching the limit), employed bees

are unable to alter their positions. Thus, these steadfast bees transform into scout bees, signifying their pivotal role in the subsequent phase. Consequently, the previously occupied positions of the employed bees are relinquished, allowing new positions to be determined using (1) to escape local optima [10].

Algorithm 1 ABC

1. The parameters (*trial*, *maxiter*, *iter* = 0) are assigned, and populations x_i are created, where $i = 1, 2, 3, \ldots, NF$.
2. The fitness of each population is evaluated.
3. Determine $limit = NF \times ND$
4. **DO**
5. **FOR** $i \in NF$
6. Update a new candidate food source v_i by using (2) for employed bees.
7. Evaluate the fitness value of the candidate v_i.
8. Select the better candidate between v_i and x_i.
9. If solution x_i does not improve, $trial_i = trial_i + 1$; otherwise, $trial_i = 0$.
10. **END FOR**
11. Calculate the probability p_i by using (4) for the solutions x_i using fitness values
12. **FOR** $i \in NF$
13. **IF** rand(0, 1) $< p_i$
14. Update a new candidate food source v_i by using (2) for the employed bees.
15. Evaluate the fitness value of the candidate v_i.
16. Apply a greedy selection process between v_i and x_i.
17. If solution x_i does not improve, $trial_i = trial_i + 1$; otherwise, $trial_i = 0$.
18. **END IF**
19. **END FOR**
20. **IF** max($trial_i$) > limit then
21. Replace x_i with a new randomly produced candidate solution via (1).
22. Evaluate the fitness value of the new randomly.
23. **END IF**
24. $iter = iter + 1$
25. **UNTIL** $iter \leq maxiter$

Algorithm 1 presents the pseudocode of the ABC algorithm, which is meticulously designed to facilitate the pursuit of an optimal value. The algorithm begins with the diligent foraging conduct exhibited by the employed bees, who are wholeheartedly dedicated to systematically exploring potential food sources (Lines 5-10). Subsequently, onlooker bees gracefully enter the realm, capitalizing on positions artfully determined by their industrious counterparts. This enables onlookers to actively and meticulously harvest nourishing sustenance from discerned food sources (Lines 12-19). In the culminating phase, aptly referred to as the scout bees step (Lines 20-23), any bees trapped within the confinements of local optima undergo a judicious repositioning process, thus rekindling their continuous exploration while carefully evading the perilous local optima trap.

III. ABC LITERATURE REVIEW

The ABC algorithm was initially introduced in 2005 [10]. In 2007, Karaboga and Basturk [11] thoroughly evaluated its performance using numerical benchmark functions. Subsequently, in 2009, Karaboga and Basturk [12] performed a comprehensive comparative analysis of the ABC, GA and PSO algorithms. These findings unequivocally demonstrated that ABC performed better than the other competing algorithms.

Since the ABC algorithm was introduced, significant research efforts have been dedicated to enhancing the algorithm to improve the optimal solutions and expedite convergence. For instance, in 2010, Alatas [19] employed chaotic maps to adapt parameters and mitigate the issue of becoming ensnared in local optima encountered in the original ABC algorithm. In 2012, AKay and Karaboga [20] updated the ABC algorithm and applied this updated version to real-parameter optimization. They introduced two novel control parameters: the modification rate (*MR*) for accelerating the convergence of ABC and the scaling factor (φ) for regulating the magnitude of ABC perturbations. The scaling factor was appropriately adjusted from the range of [−1, 1] to [$-\varphi$, φ], as exemplified in (2). Similarly, Gao and Liu [21] proposed an equation for facilitating the foraging of nearby food sources by utilizing the best solutions within the current population, denoted as the ABC/best/1 algorithm for the employed bee step. Moreover, they presented an equation for enhancing the foraging capabilities of onlooker bees in the onlooker bee step, as depicted in (5). In addition, they conducted comprehensive experiments employing numerical benchmark functions to meticulously evaluate the performances of their proposed methodologies.

$$v_{i,j} = x_{pbest,j} + \varphi_{i,j}\left(x_{i,j} - x_{k,j}\right) \tag{5}$$

In 2015, Kiran and Findik [22] presented a direct foraging food source because the search process for ABCs is undirected (φ in the range [−1, 1]). They used (6) to improve the position of $v_{i,j}$ using direction information for each dimension of each food source position ($d_{i,j}$).

$$v_{i,j} = \begin{cases} x_{i,j} + \varphi_{i,j}\left(x_{i,j} - x_{r,j}\right), & d_{i,j} == 0 \\ x_{i,j} + r_{i,j}\left|x_{i,j} - x_{r,j}\right|, & d_{i,j} == 1 \\ x_{i,j} - r_{i,j}\left|x_{i,j} - x_{r,j}\right|, & d_{i,j} == -1 \end{cases} \tag{6}$$

In 2015, Maeda and Tsuda [23] attempted to remove bees until they reached the decision criteria. The bee removed in each iteration was the bee with the worst fitness value.

In 2015, Li et al. [24] presented a novel hybrid methodology that synergistically combined the local search phase derived from the PSO algorithm, specifically tailored for the exploitation process (commonly known as the PSO phase), with a refined search approach integrated into the onlooker bee phase. This integration exploited (7), enabling the efficient identification and acquisition of new food sources. Additionally, the researchers proposed an innovative adaptation to the scout bee phase, incorporating (8), which was notably conducive to the exploration process. In each iterative cycle, all individual bees experienced either an exploitation process or an exploration process, engendering a comprehensive exploration–exploitation dynamic.

$$v_{i,j} = x_{i,j} + \varphi_{i,j}\left(x_{i,j} - x_{pbest,j}\right) \tag{7}$$

$$x_{i,j} = x_{i,j} + \varphi_{i,j}\left(x_{pbest_{k1},j} - x_{pbest_{k2},j}\right) \tag{8}$$

In 2015, Yan et al. [25] applied a crossover operator from a GA to select the parent and create eight children from them (four from one-point crossover and four from arithmetic crossover).

$$v_{i,j} = \begin{cases} NF(\frac{x_{i,j}^G + x_{best,j}}{2}, \left|x_{i,j}^G - x_{best,j}\right|), & rand_j < CR \\ x_{i,j}, & otherwise \end{cases} \tag{9}$$

where x_{best} is the global best solution of the current population and CR is a control parameter.

In 2016, Zhou et al. [26] created a new food source using Gaussian bare bones and used the global best solution according to (9). They also proposed a generalized opposition-based learning (GOBL) strategy to create new food sources in the scout bee step.[da_j, db_j] is the dynamic boundary constraint for the j^{th} dimensional variable, which is defined as follows.

$$x_{i,j} = \varphi_{i,j}\left(da_j - db_j\right) - x_{i,j} \tag{10}$$

In the same year, Gao et al. [27] applied the population initialization technique based on chaotic opposition to the initial position of each bee particle. In the employed bee phase, they used the global best value to improve the exploitation process (GABC). Consequently, this method has poor exploration ability. Thus, they used a probabilistic value, which is computed from the number of successful candidate solutions in the next generation and the number of discarded candidate solutions in the next generation, to choose between employing the DE or GABC algorithm.

Cui et al. [28] attempted to improve the performance of the ABC exploitation process via depth-first search (DFS) because DFS emphasizes exploitation. A random particle is used to update the location. Then, the new position is computed and compared to the current position. If the new position is better than the current position, the current position is updated. Afterward, the next position is computed in the same manner until the new position is no better than the current position according to the DFS.

In 2017, Song et al. [29] improved the equation for finding a new food source in the employed bee phase. The authors proposed the midpoint of two ABCs (M2ABCs) by two random food sources (x_{r_1} and x_{r_2}) and the global best solution (x_{best}). In addition, the fitness of the two selected global best solutions was used to increase the exploitation process shown in (11).

$$v_{i,j} = \frac{(x_{r_1,j} + x_{r_2,j})}{2} + \varphi_{i,j}(x_{r_1,j} - x_{r_2,j})\frac{\left(f\left(x_{r_2}\right) - f(x_{r_1})\right)}{\left(f\left(x_{r_1}\right) + f(x_{r_2})\right)} + \varphi_{i,j}(x_{best,j} + x_{r_1,j}) \tag{11}$$

where r_1, r_2, and r_3 are random numbers between 1 and NF.

In the onlooker bee phase, they adopted a trigonometric mutation operation and ABC/best (TMABC) to search for food sources in the exploration process shown in (12).

$$v_{i,j} = \frac{\left(x_{r_1,j} + x_{r_2,j} + x_{r_3,j}\right)}{3} + (p_2 - p_1)\left(x_{r_1,j} - x_{r_2,j}\right) + (p_3 - p_2)\left(x_{r_2,j} - x_{r_3,j}\right) + (p_1 - p_3)\left(x_{r_3,j} - x_{r_1,j}\right) \tag{12}$$

where $p_i = \left|f(x_{r_i})\right| / P$ and $P = \sum_i^3 \left|f(x_{r_i})\right|$

In 2017, Jadon et al. [30] used the best solution of the current swarm to search for food sources in the employed bee phase according to (13). In addition, a differential evolution operation (mutation and crossover) was applied in the onlooker bee phase instead of in (2).

$$v_{i,j} = x_{i,j} + \varphi_{i,j}\left(x_{i,j} - x_{k,j}\right) + \varphi_{i,j}\left(y_d - x_{i,j}\right) \tag{13}$$

where y_d is the d^{th} dimension of the best solution of the current swarm and is a random number between (0, C) and C is a positive constant defined by the user.

In 2017, Liang et al. [31] applied a differential operator (mutation and crossover) in the employed bee phase to search for food sources. Equation (14) represents the selection of two food sources (x_{r_1} and x_{r_2}) for creating the new food source ($v_{i,j}$).

$$v_{i,j} = x_{r_1,j} + \varphi_{i,j}\left(x_{r_1,j} - x_{r_2,j}\right) \tag{14}$$

They also designed a new probability equation. Then, three random food sources (x_{r_1}, x_{r_2}, and x_{r_3}) were used in the onlooker bee phase according to (15).

$$v_{i,j} = x_{r_1,j} + \varphi_{i,j}\left(x_{r_2,j} - x_{r_3,j}\right) \tag{15}$$

In the same year, Li et al. [32] applied gene recombination (crossover operator) after three phases of ABC to generate the new position of the bee particle (v_i) as a candidate solution of the current position (x_i).

In 2018, Yu et al. [33] proposed a novel probability equation that calculates the ratio between the best fitness and the worst fitness of particles, presenting it as a compelling alternative to the conventional method (4). In the onlooker bee phase, greedy methodology was employed to carefully select the best t optimal fitness values, thus diverging from the stochastic nature outlined in (16).

$$v_{i,j} = x_{best,j}^t + \varphi_{i,j}\left(x_{best,j}^t - x_{k,j}\right) \tag{16}$$

Afterward, the food source was adjusted. If the onlooker bee outperformed the employed bee, the number of t decreased, or vice versa, as shown in (17).

$$t\,(i+1) = \begin{cases} t\,(i) + \Delta t, & U_e > U_o \\ t\,(i) - \Delta t, & U_e < U_o \\ t\,(i), & U_e = U_o \end{cases} \tag{17}$$

where Δt is the adjustment step, which is an integer within the range [1, NF], and U_e and U_o denote the recorded numbers

of successful position updates for the employed bees and onlooker bees, respectively.

In 2018, Lin et al. [34] incorporated a local search component within the employed bee phase. They introduced a visual scope, as defined by (18), to facilitate the selection of the optimal food source by the employed bee. Furthermore, during the onlooker bee phase, which is depicted by (19), a global search strategy was employed.

$$v_{i,j} = x_{pbest,j} + \varphi_{i,j}\left(x_{i,j} - x_{k,j}\right) \tag{18}$$

$$v_{i,j} = x_{best,j} + \varphi_{i,j}\left(x_{pbest,j} - x_{i,j}\right) \tag{19}$$

where $x_{pbest,j}$ is a randomly selected high-quality food source position that differs from x_i; x_k, which is distinct from $x_{pbest,j}$ and x_i, is randomly selected from the entire population; $x_{best,j}$ is the best food source position among the current population; and j is a randomly selected dimension.

In 2019, Xiao et al. [37] improved the use of forage food sources in the employed bee phase using an elite set (E) that stores the best $\rho \times N$ solutions in the current population, as shown in (20).

$$v_{i,j} = \frac{1}{2}\left(E_{l,j} + Gbest_j\right) + \varphi^1_{i,j}\left(x_{i,j} - E_{l,j}\right) + \varphi^2_{i,j}\left(x_{i,j} - Gbest_j\right), \tag{20}$$

where $\rho \in (0, 1)$, E_l denotes a solution randomly chosen from the Elite set E, $\varphi^1_{i,j}$ is a random value between -0.5 and 0.5, and $\varphi^2_{i,j}$ is a random value between 0 and 1.

They created a search function for the onlooker bee to address the exploitation process shown in (21).

$$v_{i,j} = \frac{1}{2}\left(E_{m,j} + Gbest_j\right) + \varphi_{i,j}\left(x_{i,j} - E_{l,j}\right) + \varphi_{i,j}\left(x_{i,j} - Gbest_j\right) \tag{21}$$

where $m = 1, 2, \ldots, M$, and M is the elite set size.

In the same year, Xiang et al. [38] presented a search equation by considering the switch between a random vector and the global best vector for the employed bee phase. For the onlooker bee phase, they used an *MR* value such as [14] as a condition for adjusting the onlooker bee phase equation.

Similarly, Gao et al. [39] presented three new equations to find a new food source in the employed bee phase (v_{i_1}, v_{i_2}, and v_{i_3}). Then, the best position (v_i) was selected from among the three candidate positions, with the highest density as a criterion. The density was computed by the Parzen window method. x_i was replaced with v_i if v_i had a better fitness value than x_i.

In 2021, Zhou et al. [40] conducted a research study with artificial bees as the subject of investigation. The bees were systematically divided into two equally sized groups, designated Group A and Group B. The group assignments were based on the bees' respective fitness values, with those demonstrating higher fitness values being allocated to Group A, while those with comparatively lower fitness values were assigned to Group B. To further differentiate the members within Group A, the researchers established two distinct subgroups: A1 and A2. Notably, A1 exhibited superior fitness values to those of A2.

To facilitate efficient exploration and optimization, researchers have developed specific search equations for the employed bee and onlooker bee phases within each group. Group A1 employed the Gaussian bare-bones search equation [26], while Group A2 implemented the best neighbor-guided search strategy [41]. Conversely, Group B employed the search strategy detailed in [42] and [43]. Moreover, in the pursuit of accurate positioning for scout bees, opposition-based learning techniques or global neighborhood searches [43] were embraced in the methodology.

In 2022, an Eigen coordinate system was constructed through the assimilation of information derived from the cumulative population distribution by Jiang et al. [44]. The process of adaptively acquiring the encoding of search equations involved learning from both the Eigen coordinate system and the original coordinate system. Moreover, a dynamic amalgamation occurred, where the search equations were effectively merged with a multivariable perturbation strategy while carefully considering their respective success rates. This intricate integration safeguards bees against perilous entrapment within local optima during the scout bee phase.

In 2022, Wang et al. [45] presented a novel Bayesian estimator with the explicit objective of enhancing the probability equation related to (4). Central to their approach was a directional guidance mechanism specifically designed to augment the search capabilities of bees during the onlooker bee and scout bee phases. By doing so, they aimed to achieve an optimal equilibrium between exploration and exploitation abilities. Moreover, the researchers devised a mathematical equation to enable the value of the control parameter *MR* to be dynamically adapted.

In 2023, Li et al. [46] proposed a neighborhood search strategy with *MR* to create a search equation for employed and onlooker bee phases. In addition, the scout bee utilized the *MR* and global best bee position to determine the new bee position.

Recently, Ye et al. [47] improved two-archive ABC for many objective optimization problems. The convergence archive and diversity archive [48] were presented to improve the exploitation and exploration processes in the employed and onlooker bee phases, respectively. Moreover, a convergence archive was used to update the scout bee phase.

IV. GROUP REFLECTION ABC (GRABC)

ABC has undergone notable advancements through diverse methodologies, as highlighted by the aforementioned body of related research. These advancements include the optimization of control parameters [20], [26], the application of GA or DE operators [24], [25], [31], and augmentations to the forage food equation during both the employed and onlooker bee phases [21], [22], [29], [30], [33], [34]. These enhancements

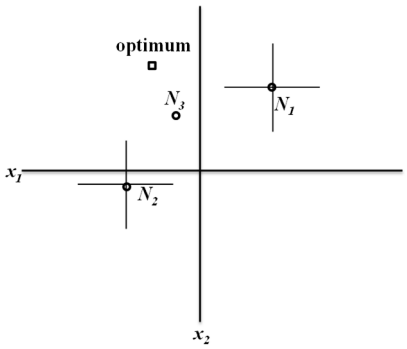


FIGURE 1. Preview of ABC's new food source search.

substantially contribute to the attainment of improved global optimum positions. Furthermore, the search equation utilized for employed and onlooker bees has been meticulously optimized to improve the efficiency of the exploration and exploitation process [37], [38], [39], [40], [41], [42], [43], [44].

In the ABC algorithm, the search for the food source is performed stochastically, with the direction of movement along the selected dimension (whether ascending or descending in the j^{th} dimension) determined randomly. Consequently, the positional coordinates pertaining to the other dimensions remain unchanged. This variable, denoted as φ and defined in (2), represents a uniformly distributed random number ranging between -1 and 1. The search pattern for the food source is graphically depicted in Fig. 1. N_2, operating at dimension x_2, successfully identifies a new food source by leveraging the information obtained from N_1. Consequently, when the value of $\varphi_{i,j}$ is less than zero, N_2's position undergoes a displacement that brings it closer to the optimal location. In contrast, a positive value of $\varphi_{i,j}$ results in a more substantial deviation from the optimal position, thereby increasing the distance between N_2 and the optimum position. The search methodology employed in the x_1 dimension mirrors that of x_2 for identifying new food sources. Importantly, the search process for each individual bee involves iterative refinement within a single dimension, as simultaneously exploring multiple dimensions would slow convergence.

As mentioned previously, ABC has a low convergence rate due to a one-dimensional random search (as shown in Fig. 1) in the employed and onlooker bee phases. In this paper, a novel algorithm is proposed to address this problem. This algorithm consists of 3 steps, as shown in Fig. 2 below.

- Grouping bees: In this step, bee particles are separated into groups that enhance the exploration process (enabling a more diverse search), with the best bee particle of each group (x_{lbest}) representing the employed and onlooker bee phases (as shown in section IV-A).
- Improving employed and onlooker bees using vector reflection: In this step, the equation for employed and onlooker bees is created by considering x_{lbest} from the previous step, inertial weights, and vector reflection to obtain a better position from each iteration, which strengthens the exploitation process (as shown in section IV-B). If the new position does not improve (bee particles are trapped in local optima), the trail value of that particle is increased for consideration in the scout bee phase.
- Improving the scout bee: In this step, the bee particle selects the best new position based on vector reflection (two new possible positions), obtaining a new position that is better than the current position. Thus, the limit value should be recalculated to be more compatible with our algorithm. In addition, the positions of bees (x_t) with a $trial_t$ count of 0 and x_{lbest} are used to determine the new position of the scout bee (as shown in section IV-C) instead of determining the new position randomly.

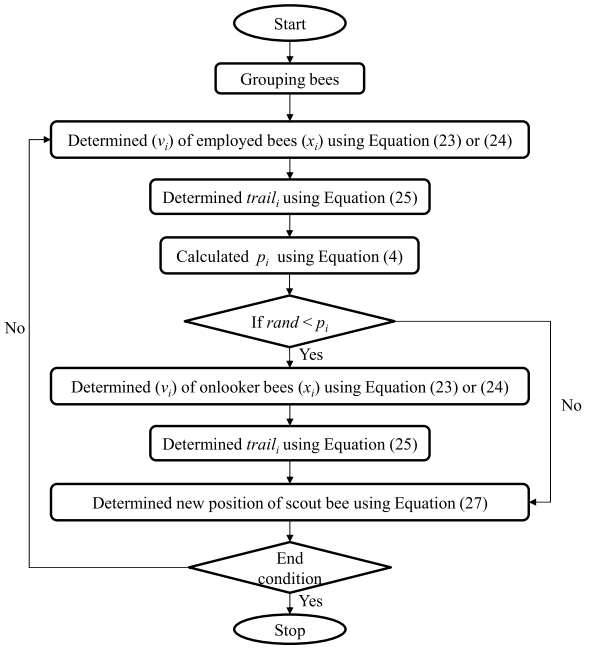
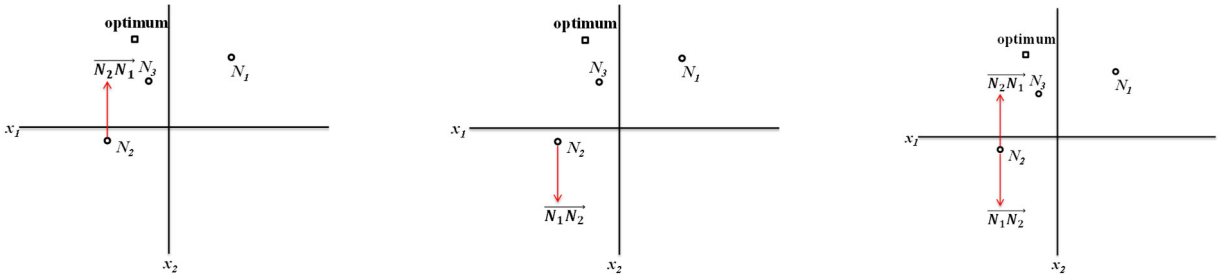


FIGURE 2. Overview of our proposed method.

As mentioned earlier, ABCs have poor exploration ability. Thus, this paper aims to enhance this problem by grouping bees, where the bee particles are separated into distinct groups to promote search diversity. The employed bees are improved using vector reflection so that they can locate better food sources. Subsequently, onlooker bees are refined to expedite convergence to the optimal value through vector reflection, collecting food from the locations indicated by the employed bees. Finally, the new position of the scout bee is determined using either the global best position or a randomly generated new position. This approach aims to balance the exploitation and exploration processes, as illustrated in Fig. 2.

The proposed method (GRABC) possesses a better exploitation ability, achieved by applying the vector reflection algorithm to onlooker bees. Additionally, the processes of grouping and employing bees aim to improve the performance of the exploration process. Thus, the method maintains a better balance of exploitation and exploration than does the original ABC algorithm.



(a) Searching in the right direction (b) Searching in the wrong direction (d) Searching in two directions

FIGURE 3. The direction of searching for the new food source.

A. GROUPING BEES

To enhance the exploration of food sources, a systematic approach involving partitioning bees into distinct groups is proposed in this paper. This division enables individual bees to augment their searching and food harvesting processes during the employed and onlooker bee phases. This augmentation is achieved by leveraging the bee's own position, characterized by the optimal fitness value ($x_{lbest_g,j}$), and the positions of other bees assigned to the same group. The practical implementation of this concept is presented and explained in (22). This concept increases the diversity of the search for the next step for employed and onlooker bees.

$$v_{i_g,j} = x_{i_g,j} + \varphi_{i,j}\left(x_{k,j} - x_{i_g,j}\right) + r\left(x_{lbest_g,j} - x_{i_g,j}\right) \quad (22)$$

where r denotes a random number between [0, 1]; $x_{lbest_g,j}$ is the bee's position with the best fitness value of the g^{th} group; and $x_{i_g,j}$ is another bee's position in group g in the j^{th} dimension.

B. IMPROVED EMPLOYED AND ONLOOKER BEES USING VECTOR REFLECTION

In this section, the equation employed for identifying and procuring novel food sources through the implementation of vector reflection is refined. As a preliminary measure, bees are segregated into distinct groups. Vector reflection is utilized to expedite convergence, thereby bolstering the overall efficiency of the search process.

Fig. 3 illustrates that N_3 exhibits the highest fitness value within the bee population. Conversely, N_2 needs its food source position to be adjusted, while N_1 represents an alternative position for a bee. When the x_2 dimension is selected to locate the new food source, Equation (2) generates a new position in the directions depicted in Fig. 3(a) or 3(b). Consequently, the convergence rate to the optimal solution is low due to the random search of the original ABC algorithm. To surmount this obstacle, vector reflection is utilized to determine the direction of bee movement. If the fitness value of the new position fails to surpass that of the old position, the new position is established in the opposite direction. Fig. 3(c) visually shows the potential positions of N_2 in two directions: above N_2 and below N_2 (according to x_2) or to the left of N_2 and to the right of N_2 (according to x_1). The ultimate movement direction of N_2 is contingent upon the superior fitness value between the two plausible positions. Consequently, with each iteration, the new food source progressively improves, exploiting the optimal position and accelerating the convergence time.

Vector reflection is employed to augment the convergence time by systematically investigating two prospective locations for a novel food source. The existing position is then superseded by the superior alternative among the two options. To facilitate the generation of these potential positions, we introduce (23) and (24), denoted as *vleft* and *vright*, respectively. These positions assume pivotal roles within the employed bee and onlooker bee phases, contributing significantly to the overall efficacy of the system.

$$vleft_{i_g,j} = x_{i_g,j} - \varphi_{i,j}\left(x_{k,j} - x_{i_g,j}\right) w^{(t)} + r\left(x_{lbest_g,j} - x_{i_g,j}\right) \quad (23)$$

where $vleft_{i_g,j}$ and $vright_{i_g,j}$ are the new possible positions of bee (x_i) in the j^{th} dimension of the g^{th} group and φ is a random value between ranges -1 and 1, which is also applied for (24) (constraint values for both (23) and (24)). Thus, the positions of $vleft_i$ and $vright_i$ will be the left and right positions of the current bee position x_i, respectively.

$$vright_{i_g,j} = x_{i_g,j} + \varphi_{i,j}\left(x_{k,j} - x_{i_g,j}\right) w^{(t)} + r\left(x_{lbest_g,j} - x_{i_g,j}\right) \quad (24)$$

While other algorithms update the trial values ($trial_i$) by considering one direction with a random search, in accordance with φ (either $vleft_i$ or $vright_i$), a stochastic selection procedure is employed between $vleft_i$ and $vright_i$ to optimize the positioning of the food source in this study. When $vleft_i$ is chosen, it replaces x_i, and the value of $trial_i$ is subsequently reset to 0 due to the superior fitness value of $vleft_i$ to x_i. Similarly, if $vright_i$ is selected, x_i is substituted with $vright_i$, and $trial_i$ is updated to 0 due to the superior fitness value of $vright_i$ to x_i. Consequently, in each iteration, the foraging bees converge more quickly in their search for novel food sources. If the fitness value of x_i exceeds that of both $vleft_i$ and $vright_i$, $trial_i$ is incremented by 1 (as shown in (25)) because the bee particle is stuck in a local optimum for the considered dimension (described in Algorithm 2 from lines 18-28 and 38-48 for the employed and onlooker bee phases,

Algorithm 2 GRABC

1. The parameters ($trial$, w_{min}, w_{max}, $maxiter$, $iter = 0$, $g = 1$, n_gp) are assigned, and populations x_i, $i = 1, 2, 3, \ldots, NF$, *are created.*
2. The fitness value of each population is evaluated.
3. Determine $limit = \sqrt{NF \times ND}$
4. **FOR** $i \in NF$
5. **IF** Fitness(x_i) < x_{lbest_g}
6. x_{lbest_g} = Fitness(x_i)
7. $xl_g = x_g$
8. **END IF**
9. **IF** mod(i, n_gp) is equal to 0.
10. $g = g+1$
11. **END IF**
12. **END FOR**
13. **DO**
14. Set $g = 1$
15. **FOR** $i \in NF$
16. Update a new candidate food source $vleft_i$ and $vright_i$ of v_i by using (23) and (24).
17. Evaluate the fitness values of the candidates $vleft_i$ and $vright_i$.
18. **IF** Fitness($vleft_i$) < x_{lbest_g}
19. x_{lbest_g} = Fitness($vleft_i$)
20. $xl_g = vleft_i$
21. $trial_i = 0$
22. **ELSE IF** Fitness($vright_i$) < x_{lbest_g}
23. x_{lbest_g} = Fitness($vright_i$)
24. $xl_g = vright_i$
25. $trial_i = 0$
26. **ELSE**
27. $trial_i = trial_i+1$
28. **END IF**
29. **IF** mod(i, n_gp) == 0
30. $g = g+1$
31. **END IF**
32. **END FOR**
33. Calculate the probability p_i by using (4) for the solutions x_i using fitness values.
34. **FOR** $i \in NF$
35. **IF** rand(0, 1) < p_i
36. Update a new candidate food source $vleft_i$ and $vright_i$ of v_i by using (23) and (24).
37. Evaluate the fitness values of the candidates $vleft_i$ and $vright_i$.
38. **IF** Fitness($vleft_i$) < x_{lbest_g}
39. x_{lbest_g} = Fitness($vleft_i$)
40. $xl_g = vleft_i$
41. $trial_i = 0$
42. **ELSE IF** Fitness($vright_i$) < x_{lbest_g}
43. x_{lbest_g} = Fitness($vright_i$)
44. $xl_g = vright_i$
45. $trial_i = 0$
46. **ELSE**
47. $trial_i = trial_i+1$
48. **END IF**
49. **IF** mod(i, n_gp) == 0
50. $g = g+1$
51. **END IF**
52. **END IF**
53. **END FOR**
54. **IF** max($trial_i$) > $limit$ then
55. Replace x_i with a new randomly produced candidate solution via (27).
56. Evaluate the fitness value of the new x_i.
57. **END IF**
58. $iter = iter + 1$
59. **UNTIL** $iter \leq maxiter$

respectively), thereby serving as a criterion for considering the scout bee phase. Thus, GRABC can indicate which bee is stuck in a local optimum in the considered dimension more quickly than other algorithms due to vector reflection.

Equation (25) explicitly illustrates the interrelationship among the fitness values of $vleft_i$, $vright_i$, and x_i, which dictates the updating of $trial_i$.

$$trial_i = \begin{cases} 0, & fitness\ of\ vleft_{i_g,j} < fitness\ of\ x_{i_g,j} \\ 0, & fitness\ of\ vright_{i_g,j} < fitness\ of\ x_{i_g,j} \\ trial_i + 1, & otherwise \end{cases} \tag{25}$$

TABLE 1. Unimodal and multimodal benchmark problems.

Function	Function range	*ND*	f^*_{min}	*C*	Formulae
f_1	[-100, 100]	100	0	U	$\sum_{i=1}^{N} x_i^2$
f_2	[-1, 1]	100	0	U	$\sum_{i=1}^{N} \lvert x_i \rvert^{i+1}$
f_3	[-100, 100]	100	0	U	$\sum_{i=1}^{N} \lvert x_i \rvert$
f_4	[-100, 100]	100	0	U	$\mathrm{Max}\{\lvert x_i \rvert, 1 \le i \le N\}$
f_5	[-100, 100]	100	0	U	$\sum_{i=1}^{N} (\lvert x_i + 0.5 \rvert)^2$
f_6	[-5.12, 5.12]	100	-575	U	$25 + \sum_{i=1}^{N} \lvert x_i \rvert$
f_7	[-100, 100]	100	0	U	$\sum_{i=1}^{N} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
f_8	[-10, 10]	100	0	U	$\sum_{i=1}^{N} x_i^{10}$
f_9	[-30, 30]	100	0	U	$\sum_{i=1}^{N} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$
f_{10}	[-1, 4]	100	0	U	$\sum_{i=1}^{N-1} x_i^{2(x_{i+1}^2+1)} x_{i+1}^{2\,(x_i^2+1)}$
f_{11}	[-10, 10]	100	0	U	$(x_1 - 1)^2 + \sum_{i=1}^{N} i(2x_i^2 - x_{i-1})^2$
f_{12}	[-4, 5]	100	0	U	$\sum_{i=1}^{N/4} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} + 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$
f_{13}	[-20, 20]	100	0	U	$exp\left(-\sum_{i=1}^{N} \left(\frac{x_i}{\beta}\right)^{2m}\right) - 2exp\left(\sum_{i=1}^{N} x_i^2\right)\left(\prod_{i=1}^{n} \cos x_i^2\right)$ $\beta = 15, m = 5$
f_{14}	[-5, 5]	5	0	U	$\sum_{i=1}^{N} \left(\sum_{j=1}^{N} (j + \beta)\left(x_j^i - \frac{1}{j^i}\right)\right)^2, \beta = 0.5$
f_{15}	[-10, 10]	100	0	U	$\sum_{i=1}^{N} i x_i^2$
f_{16}	[-500, 500]	100	0	M	$418.9829N - \sum_{i=1}^{N} -x_i^2 \, sin\left(\sqrt{\lvert x_i \rvert}\right)$
f_{17}	[-5.12, 5.12]	100	0	M	$\sum_{i=1}^{N} \left(x_i^2 - 10cos(2\pi x_i)\right) + 10N$
f_{18}	[-10, 10]	100	1	M	$1 + \sum_{i=1}^{N} \sin(2x_i) - 0.1exp\left(-\sum_{i=1}^{N} x_i^2\right)$
f_{19}	[-500, 500]	100	0	M	$\sum_{i=1}^{N} (x_i^2 - i)^2$
f_{20}	[-10, 10]	100	0	M	$\sum_{i=1}^{N} \lvert x_i \sin(x_i) + 0.1x_i \rvert$
f_{21}	[-5, 5]	100	0	M	$\sum_{i=1}^{N} rand[0,1] \lvert x_i \rvert^i$
f_{22}	[-32.768, 32.768]	100	0	M	$-20exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}\right) - exp\left(\frac{1}{N}\sum_{i=1}^{N} cos(2\pi x_i)\right) + 20 + e$
f_{23}	[-500, 500]	100	0	M	$\sum_{i=1}^{N} (8\sin(7(x_i - 0.9)^2)^2 + 6\sin(14(x_1 - 0.9)^2)^2 + (x_i - 0.9)^2)$
f_{24}	[-100, 100]	100	0	M	$1 - \cos\left(2\pi\sum_{i=1}^{N} x_i^2\right) + 0.1\sqrt{2\pi\sum_{i=1}^{N} x_i^2}$
f_{25}	[-5, 5]	100	-3916.6	M	$\frac{1}{2}\sum_{i=1}^{N} (x_i^4 - 16x_i^2 + 5x_i)$
f_{26}	[-100, 100]	100	0	M	$\frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
f_{27}	[-10, 10]	100	0	M	$\left(\sum_{i=1}^{N} \sin^2(x_i) - exp\left(-\sum_{i=1}^{N} x_i^2\right)\right) exp\left(-\sum_{i=1}^{N} \sin^2\left(\sqrt{\lvert x_i \rvert}\right)\right)$
f_{28}	[-2π, 2π]	100	0	M	$\left(\sum_{i=1}^{N} \lvert x_i \rvert\right) exp\left(-\sum_{i=1}^{N} sin(x_i^2)\right)$
f_{29}	[-50, 50]	100	0	M	$\frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{N-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{N} u(x_i, 10,100,4)$. $y_i = 1 + \frac{x_i+1}{4}$.

TABLE 1. *(Continued.)* **Unimodal and multimodal benchmark problems.**

					$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m x_i>a\\0-a<x_i<a\\k(-x_i-a)^m x_i<a\end{cases}$
f_{30}	[-50, 50]	100	0	M	$0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{N-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1}+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\right\}+\sum_{i=1}^{N}u(x_i,5,100,4)$
f_{31}	$[0,\pi]$	100	-99.864	M	$-\sum_{i=1}^{N}\sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$
f_{32}	[-1.28, 1.28]	100	0	M	$\sum_{i=1}^{N}ix_i^4+rand[0,1]$

TABLE 2. **Algorithm parameters for the functions in Table 1 for unimodal and multimodal functions.**

Parameters	Algorithms							
	ABC	DABC	ABCADE	MTABC	AABCLGII	AABC	EMABC-NS	GRABC
NF	20	20	20	20	20	20	20	20
ND	100	100	100	100	100	100	100	100
Limit	*NF*×*D* [10]	*NF*×*D* [22]	*NF*×*D* [31]	*NF*×*D* [29]	*NF*×*D* [34]	*NF*×*D* [33]	*NF*×*D* [46]	$\sqrt[3]{NF\times ND}$
Maxiter	5000	5000	5000	5000	5000	5000	5000	5000
n_gp	-	-	-	-	-	-	-	5

The inertial weight (w) is a fundamental parameter linked to bee particles. It substantially controls the impact of the recent running speed of (23) and (24) for v_{left} and v_{right}, respectively. To circumvent the entrapment of bee particles within a local minimum, w must be assigned an initial value of significant magnitude. Inadequately initializing w may substantially decelerate the convergence process. Consequently, a strategy of initially assigning w a considerably elevated value and gradually decreasing its value in subsequent iterations is recommended. This systematic reduction in w manifests in a linear manner alongside the progressive increase in iteration count, as clearly exemplified by (26).

$$w^{(t)}=w_{max}-\frac{w_{max}-w_{min}}{iter_{max}}\times t, \tag{26}$$

where w_{max} is the initial inertial weight (1.0), w_{min} is the termination inertial weight $w_{max}/maxiter$, and $maxiter$ is the maximum number of iterations.

The fitness value of each individual bee particle (x_i) is determined by utilizing (3), which is similar to the conventional ABC approach. We employ (4) to calculate the probability (p_i) of selecting employed bees during the foraging of food sources in the onlooker bee phase.

C. IMPROVED SCOUT BEE

Despite the inherent capacities of employed and onlooker bees to explore and identify improved food sources, a bee becomes immovable when it encounters a local optimum, when neither $vleft_i$ nor $vright_i$ demonstrate superior fitness values compared to x_i. For ABC, a limit value is assigned. This is a high value and does not fit our improvement process (previous step). Thus, the limit value should be recalculated (as shown in section V).

In addition, in this study, we propose adopting a specific equation, labeled (27), to ascertain the position of the scout bee. This equation effectively calculates the average between the bee's position with a trial value equal to zero (x_t) and x_{lbest} within the g^{th} group. An intriguing aspect of this equation is its utilization of a $trial_t$ value of zero. The following equation can successfully improve the exploitation ability and better balance the exploration and exploitation abilities of bees by sharing information among i, t, and $lbest$.

$$x_{i_g}=\begin{cases}x_{i_g}, & trial_i\leq limit\\ \dfrac{x_t+x_{lest_g}}{2}, & trial_i> limit\end{cases} \tag{27}$$

where x_t is the random position of the bee when $trial_t=0$ and the $limit$ is equal to $\sqrt[3]{NF\times ND}$ (see section V).

Algorithm 2 presents a comprehensive overview of the GRABC procedure. Initially, we diligently calculate the local best (x_{lbest_g}) for each distinct group (Lines 4-12). We then employed targeted strategies to augment the efficiency of our search for viable food sources during the deployed bee phase (Lines 15-32). Furthermore, we meticulously updated the position of x_{lbest_g} for each individual group by comparing $vleft$, $vright$, and x_{lbest_g} (lines 18-31). Our diligent bees are then directed toward skillfully harvesting food sources in the subsequent onlooker bee phase (lines 34-53). Furthermore, we ensure that the position of x_{lbest_g} for each group is continually refined (lines 38-51). Ultimately, we apply (27) to precisely ascertain the position of any bee ensnared within a local optimum, thereby enabling us to accurately calculate the fitness value of the resulting position (Lines 54-57).

V. EXPERIMENTAL RESULTS

In this section, the efficiency of GRABC is evaluated, and a comparative analysis is conducted with other pertinent works by employing 32 widely acknowledged benchmark functions [49], [50], [51], [52], mostly including CEC 2017. The evaluation test bed is a standard configuration for

TABLE 3. Comparison of limit values.

Functions	Limit value			
	$NF \times ND$	$\sqrt{NF \times ND}$	$\sqrt[3]{NF \times ND}$	$\sqrt[4]{NF \times ND}$
f_1	1.12E-16	2.59E-17	**2.10E-17**	2.88E-17
f_2	4.25E-18	**6.91E-18**	1.10E-17	1.55E-17
f_3	2.63E-15	1.82E-16	**1.33E-17**	1.62E-16
f_4	33.575	3.55E-16	**1.53E-16**	1.93E-16
f_5	58125	2.27E-15	**2.10E-15**	98.106
f_6	**-575**	**-575**	**-575**	**-575**
f_7	0.12901	2.41E-05	**7.75E-06**	8.17E-06
f_8	6.21E-20	5.57E-20	**4.03E-20**	9.58E-20
f_9	4.3616	2.8713	**2.72925**	95.872
f_{10}	9.68E-17	**2.40E-17**	2.51E-17	2.71E-17
f_{11}	0.451	**0.0388**	0.0588	0.66667
f_{12}	0.047914	0.027761	**2.45E-17**	3.51E-17
f_{13}	2.00E-77	6.04E-82	**1.12E-83**	1.02E-81
f_{14}	0.7413	**0.0522**	0.2025	0.19686
f_{15}	7.58E-17	4.18E-17	**2.60E-17**	2.66E-17
f_{16}	1171.1	**1005.3**	1217.5	1289.5
f_{17}	0.90079	**0**	**0**	**0**
f_{18}	1.0003	1.0003	**1.0002**	1.0003
f_{19}	0.1062	0.048304	**0.005448**	0.007202
f_{20}	4.27E-05	3.76E-05	**1.39E-05**	3.74E-06
f_{21}	8.07E+13	8.75E-10	**1.24E-16**	6.99E-18
f_{22}	7.02E-14	5.86E-15	**5.33E-15**	4.80E-15
f_{23}	1.1795	**1**	**1**	1.0448
f_{24}	2.5228	0.069937	**0.041924**	7.07E-06
f_{25}	-3898.2	**-3906.7**	-3900.4	-3889.8
f_{26}	2.22E-17	**0**	**0**	1.11E-17
f_{27}	1.07E-23	**1.17E-33**	3.63E-33	1.35E-24
f_{28}	2.23E-17	2.28E-17	**1.59E-17**	1.79E-17
f_{29}	**1.37E-15**	2.16E-15	2.28E-15	2.25E-15
f_{30}	**1.41E-15**	2.01E-15	2.20E-15	2.38E-15
f_{31}	-99.0	**-99.143**	-99.094	-99.102
f_{32}	0.093271	1.90E-05	**1.72E-05**	1.58E-05
w/t/l	26/1/5	18/1/13		22/2/8

Windows 11 operating systems (64 bits): 2.5 GHz Intel(R) Core (TM) i5 CPU, 8 GB RAM, and 512 GB Disk. To ensure a methodologically sound comparison, the simulation was conducted using MATLAB® 2016a, utilizing its standard library. To account for the inherent variability, each experiment was iterated 50 times with distinct random seeds. The outcomes presented encompass the best, worst, and mean results, the standard deviations, and the Wilcoxon rank-sum test, thereby enabling comprehensive comparisons.

A. BENCHMARK FUNCTIONS

Table 1 presents a comprehensive overview of the benchmark functions from [49], [50], [51], [52], which encapsulate multiple essential elements. These elements include the function range, ND, f^*_{min}, C, and formulae that precisely delineate the lower and upper bounds of the search spaces of each dimension, number of dimensions, global minimum values, characteristics, and functions. The table serves as a valuable reference for understanding and analyzing the intricate aspects of these benchmark functions.

Each function employed in this experiment exhibited unique properties. Functions that manifest multiple local optima are classified as multimodal functions (M), including f_{16}-f_{32}. Conversely, functions with a solitary and distinctive local optimum are categorized as unimodal functions (U), such as f_1-f_{15}.

The dimensionality of the search space is crucial in algorithmic problem solving [10], [22], [29], [31], [33], [34], [46]. In our experimental study, we specifically focused on investigating and comparing the performances of the methods on numerical functions [49], [50], [51], [52] characterized by a dimensionality of 100 except for f_{14}(for a dimensionality of 5) [49]. In addition, we attempt to set the number of dimensions to a higher value, such as 100; however, MATLAB does not respond to all the algorithms. To comprehensively overview the experimental setup, Table 2 presents the key parameters employed in the evaluation of the optimization problem of each algorithm.

These parameters encompass essential aspects such as the number of bee particles utilized during the employed bee and scout bee phases (*NF*), the *ND*, the management of scout bees through utilizing a threshold value (*limit*), the maximum iteration count employed as the termination condition for the methods (*maxiter*), and the number of groups (*n_gp*). In addition, the values of *NF*, *ND*, and *maxiter* are set to 20, 100, and 5000, respectively, for all algorithms. The limit value is set to $NF \times ND$ for all algorithms, except for the proposed algorithm, which is set to $\sqrt[3]{NF \times ND}$ based on experimental studies, as shown in Table 2.

Setting the limit value to $NF \times ND$, as in traditional ABC with a high value, ensures that no bee particle surpasses the limit in terms of its trail value. This is due to the continuous improvement in bee particle positions in each iteration during the employed and onlooker bee phases, where the bees select new food source positions to the *vleft* or *vright*. As a result, the trail value of bee particles remains consistently low. Thus, the primary objective of this study was to evaluate and determine the optimal value of the limit. The obtained results are presented in Table 3, including the optimum values for 32 functions (f_1-f_{32}) as the limit value varies from $NF \times ND$ to $\sqrt[4]{NF \times ND}$. Table 3 illustrates that each function typically performs better at achieving the average optimum result when the limit value is set to $\sqrt[3]{NF \times ND}$ than when the limit is set to $NF \times ND$, $\sqrt{NF \times ND}$, or $\sqrt[4]{NF \times ND}$ because the exploration and exploitation processes are better balanced. Thus, our proposed method uses $\sqrt[3]{NF \times ND}$ as the limit value for each of the benchmark functions.

TABLE 4. Unimodal functions.

Functions	Algorithms							
	ABC	DABC	ABCADE	MTABC	ABCLGII	AABC	EMABC-NS	GRABC
f_1								
Worst	3.68E-13	1.15E-13	9.1584	545.79	0.79537	2.97E-15	9.43E-15	**4.90E-17**
Best	6.91E-15	3.43E-15	2.50E-07	0.55019	0.068303	2.03E-15	1.85E-15	**1.01E-17**
Avg	5.81E-14	2.31E-14	0.58389	85.933	0.31371	2.32E-15	2.61E-15	**2.10E-17**
Std	9.50E-14	2.70E-14	2.0493	127.23	0.23993	2.31E-16	1.62E-15	**8.67E-18**
f_2								
Worst	3.66E-16	4.22E-16	4.03E-15	1.26E-06	3.71E-09	1.11E-16	8.66E-17	**2.77E-17**
Best	6.11E-17	4.99E-17	2.09E-17	3.55E-17	1.88E-17	4.33E-17	1.82E-17	**1.96E-19**
Avg	1.13E-16	1.12E-16	2.42E-16	1.17E-07	1.88E-10	6.72E-17	4.97E-17	**1.10E-17**
Std	6.81E-17	8.41E-17	8.92E-16	3.57E-07	8.30E-10	1.92E-17	1.90E-17	**9.58E-17**
f_3								
Worst	1.94E-07	8.43E-08	1.3078	61.384	2.9048	8.39E-11	5.84E-06	**3.92E-17**
Best	1.83E-08	5.46E-09	0.004309	11.552	0.65348	8.14E-12	2.19E-14	**5.12E-19**
Avg	5.84E-08	2.57E-08	0.41863	34.11	1.6259	2.74E-11	2.94E-07	**1.33E-17**
Std	4.31E-08	2.14E-08	0.45273	13.963	0.59551	2.11E-11	1.30E-06	**1.17E-17**
f_4								
Worst	62.907	64.006	41.365	36.451	43.86	56.663	49.426	**2.27E-16**
Best	50.825	50.223	30.908	26.055	23.401	46.485	39.694	**4.73E-17**
Avg	56.848	57.219	35.724	32.272	32.028	51.488	44.522	**1.53E-16**
Std	3.351	3.8085	2.8153	3.0281	5.7326	2.7247	3.2700	**4.92E-17**
f_5								
Worst	1.04E-13	3.50E-13	1.5632	444.41	1.0563	2.73E-15	9399.3	**2.32E-15**
Best	4.28E-15	3.84E-15	2.86E-08	0.28353	0.03725	1.80E-15	30.936	**1.66E-15**
Avg	2.93E-14	4.24E-14	0.1987	72.894	0.30215	2.33E-15	1434.7	**2.10E-15**
Std	2.72E-14	7.98E-14	0.42949	115.79	0.27622	2.66E-16	2655.4	**2.67E-16**
f_6								
Worst	-574	-574	**-575**	-557	**-575**	**-575**	**-575**	**-575**
Best	-575	-575	**-575**	-575	**-575**	**-575**	**-575**	**-575**
Avg	-574.9	-574.95	**-575**	-572.2	**-575**	**-575**	**-575**	**-575**
Std	0.30779	0.22361	**0**	4.2128	**0**	**0**	**0**	**0**
f_7								
Worst	0.7052	0.7187	1.8105	1.12E+05	39.387	0.41903	0.5624	**3.74E-05**
Best	0.26506	0.32757	0.17902	0.25737	1.2865	0.22402	0.2212	**1.95E-06**
Avg	0.48672	0.47337	0.50447	6411.8	4.1771	0.348	0.3310	**7.75E-06**
Std	0.12567	0.1057	0.44358	25048	8.3927	0.048957	0.0782	**1.40E-06**
f_8								
Worst	6.79E-16	5.41E-16	4.63E-16	0.045047	1.00E-13	5.15E-16	4.76E-16	**7.64E-19**
Best	3.02E-16	2.91E-16	1.93E-16	7.29E-17	3.20E-16	2.38E-16	1.85E-16	**7.99E-23**
Avg	4.62E-16	4.51E-16	2.86E-16	0.002252	8.25E-15	4.39E-16	2.84E-16	**4.03E-20**
Std	9.69E-17	7.00E-17	5.23E-17	0.010073	2.34E-14	7.71E-17	8.02E-17	**1.85E-19**
f_9								
Worst	72.203	22.301	1680.2	1.59E+05	861.32	27.835	197.68	**10.89455**
Best	0.046747	0.51883	8.2414	134.35	58.143	1.6431	2.0493	**0.182135**
Avg	7.8748	11.431	167.27	22158	256.04	10.07	78.711	**2.72925**
Std	15.773	6.6161	370.56	43654	162.65	7.2324	62.691	**3.59135**
f_{10}								
Worst	3.86E-15	1.49E-14	0.025666	0.4357	0.007124	8.88E-13	2.68E-15	**4.25E-17**
Best	2.26E-15	2.94E-15	1.55E-11	0.000137	0.000282	3.41E-15	1.62E-15	**7.31E-18**
Avg	3.18E-15	5.13E-15	0.001687	0.037567	0.002384	7.25E-14	2.19E-15	**2.51E-17**
Std	3.68E-16	3.32E-15	0.005733	0.1008	0.001876	1.95E-13	2.48E-16	**8.07E-17**
f_{11}								
Worst	5.1401	0.99767	99.098	11666	39.639	8.4331	21.940	**0.2489**
Best	0.005513	0.004376	0.38305	11.875	8.8611	0.056562	0.0626	**6.78E-05**
Avg	0.66376	0.15768	18.4	999.36	21.067	0.83938	8.3981	**0.0588**
Std	1.2916	0.3841	27.73	2870.5	7.1427	1.84	7.3250	**0.007584**
f_{12}								
Worst	0.066503	0.092992	3.8277	50.378	1.3336	0.4501	2.7583	**2.86E-17**
Best	0.034548	0.037588	0.15293	0.10793	0.25607	0.1534	0.1361	**1.48E-17**
Avg	0.047541	0.058855	0.78054	8.6742	0.52995	0.28126	0.5058	**2.45E-17**
Std	0.007922	0.015018	0.9561	13.153	0.26069	0.075427	0.6361	**3.47E-18**
f_{13}								
Worst	2.6101E-101	2.62E-98	5.51E-99	5.38E-98	3.30E-97	**1.62E-101**	1.44E-97	2.19E-82
Best	1.7094E-127	7.05E-128	4.41E-127	3.51E-123	5.34E-99	**1.69E-145**	8.10E-118	3.25E-119
Avg	1.6531E-102	1.33E-99	2.76E-100	2.71E-99	1.96E-98	**1.53E-102**	7.57E-99	1.12E-83
Std	1.9596E-102	5.84E-99	1.23E-99	1.20E-98	7.44E-98	**1.48E-102**	3.22E-98	4.91E-83
f_{14}								
Worst	3.8069	13.909	24.065	18.773	17.894	4.149	100.92	**0.973**
Best	0.059808	0.29531	2.922	0.56222	0.38503	0.27215	0.0092	**0.078**
Avg	1.6214	3.1622	7.5359	5.3665	5.6274	1.6052	12.655	**0.2025**
Std	1.0877	3.3767	5.273	5.3938	4.818	0.90317	24.933	**0.347**

TABLE 4. *(Continued.)* **Unimodal functions.**

f_{15}								
Worst	1.88E-13	2.84E-14	1.1762	279.07	1.6036	2.75E-15	2.77E-15	**4.02E-17**
Best	3.30E-15	2.99E-15	5.01E-12	0.098239	0.009903	2.00E-15	1.75E-15	**1.33E-17**
Avg	3.31E-14	9.33E-15	0.13899	22.883	0.29067	2.37E-15	2.24E-15	**2.60E-17**
Std	4.52E-14	7.76E-15	0.30261	62.103	0.39804	2.16E-16	2.61E-16	**6.99E-18**

B. COMPARISONS

Comparative analyses of the traditional ABC [10], DABC [22], MTABC [29], ABCADE [31], AABC [33], ABCLGII [34], and EMABC-NS [46] algorithms for unimodal and multimodal functions are presented in Tables 4 and 5, respectively. A comprehensive evaluation of the methods' performances reveals that the GRABC algorithm surpasses alternative approaches in terms of solution quality and robustness (except for f_{13}, f_{19}, f_{20}, and f_{25}, as shown in Tables 4 and 5). This superiority is evident through meticulous consideration of fundamental metrics, including the best, worst, mean results, standard deviations, and $w/t/l$ (the number of win, tie, and lose between two algorithms on all the functions according to the Wilcoxon rank-sum test) [51], [52] given 100 dimensions. Table 6 shows the results of the Wilcoxon rank-sum test at a significance level of $=$ 0.05. The symbols $+$, $-$, and $\sim$ indicate that the proposed algorithm is significantly better, significantly worse, and nonsignificant, respectively, than the compared algorithms on one function [51], [52].

Table 4 lists the worst values, best values, average values, and standard deviations (stds) derived from 50 random seed runs for unimodal functions (f_1-f_{15}) of the ABC, DABC, ABCADE, MTABC, ABCLGII, AABC, EMABC-NS and GRABC algorithms. For $f_1, f_2, f_3, f_4, f_5, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}$, f_{14}, and f_{15}, GRABC outperforms the comparison methods. In addition, GRABC performs better than the other methods except for ABCADE and AABC (equally) for f_6. However, for f_{13}, AABC outperforms the other methods, including GRABC.

Table 5 lists the values of the worst values, best values, average values, and standard deviations (stds) derived from 50 random seed runs for multimodal functions (f_{16}-f_{32}). For f_{16}-f_{18}, f_{21}, f_{22}, f_{24}, and f_{26}-f_{32}, GRABC outperforms the comparison methods. In addition, GRABC performs better than all the other methods except for ABC, DABC, and AABC (equally) for f_{23}. However, for f_{25}, ABC outperforms the other methods, including GRABC.

Convergence graphs depicting the performance of the techniques (as illustrated in Figs. 4 and 5) are generated with 100 dimensions except for f_{14}, where ND is set to 5. The analysis reveals that the convergence rate of the GRABC algorithm is notably superior to that of the other methods, as the number of iterations reaches or surpasses the specific thresholds for each function except for f_{13} and $f_{25,}$ for which AABC and ABC, respectively, outperform GRABC.

For unimodal functions, as shown in Fig. 4, the convergence rate of GRABC is better than that of the comparison methods for f_1-f_{12}. Specifically, the iteration thresholds are 500, 100, 100, 100, 500, 500, 400, 500, 600, 500, 700, 200, 100, and 400 for f_1-f_{12} and f_{14}-$f_{15,}$ respectively. According to Table 4, GRABC converges to the result of 1.12E-83, which is faster than the other algorithms for f_{13}, at iteration 100. However, it is stuck in a local optimum. As a result, AABC outperforms the comparison algorithms, including GRABC, with 1.53E-102 as the average result.

For multimodal functions, as shown in Fig. 5, the convergence rate of GRABC is better for f_{16}-f_{18}, f_{21}-f_{24}, and $f_{26–}f_{32}$ than that of the comparison methods. Specifically, the iteration thresholds are 1400, 800, 1700, 100, 300, 400, 100, 300, 100, 2100, 500, 600, 2000, and 100 for f_{16}-f_{24} and f_{26}-$f_{32,}$ respectively.

According to Table 5, GRABC converges to 0.005448, 1.39E-05, and 3096.7 faster than the other algorithms do at iterations 600, 1700, and 600 for f_{19}, f_{20}, and f_{25}, respectively. However, it remains in a local optimum. As a result, EMABC-NS, EMABC-NS, and ABC outperform the comparison algorithms, including GRABC, with 2.09E-15, 2.15E-12, and 3912.4 as the average results for f_{19}, f_{20}, and f_{25}, respectively.

C. COMPUTATIONAL TIME COMPLEXITY

For a problem f, assume that $O(f)$ is the computational time complexity of evaluating its function value. NF is the population size. The time complexity of traditional ABC is $(ONF{\times}f + NF{\times}f) = O(NF{\times}f)$ [53]. For GRABC, it will replace *vright* or *vleft* for x, so $O(1.5{\times}f)$ is the computational time complexity of evaluating its function value. Therefore, the complexity is $O(NF{\times}1.5{\times}f + NF{\times}1.5{\times}f)$ on average. Therefore, the total computational complexity of GRABC is $O(1.5{\times}NF{\times}f)$ at each iteration. In addition, Table 7 shows the average computational time of all algorithms on 32 functions with 100 dimensions. The results show that the average time complexity (in seconds) of GRABC is greater than that of ABC in the range from ABC to 2$\times$ABC. However, the proposed algorithm converges quickly to the optimum value (at approximately the 3000$^{\text{th}}$ iteration), as shown in Figs. 4 and 5.

VI. DISCUSSION

Swarm intelligence-based optimization techniques initiate the solution space traversal process by generating randomized initial solutions. These techniques effectively harness the intricate interactions among the particles within the population to obtain an optimal or near-optimal solution. However, as the dimensionality of the problem increases, the probability of encountering a local optimum increases for each algorithm. This predicament significantly impacts the attainment of the overall optimum value.

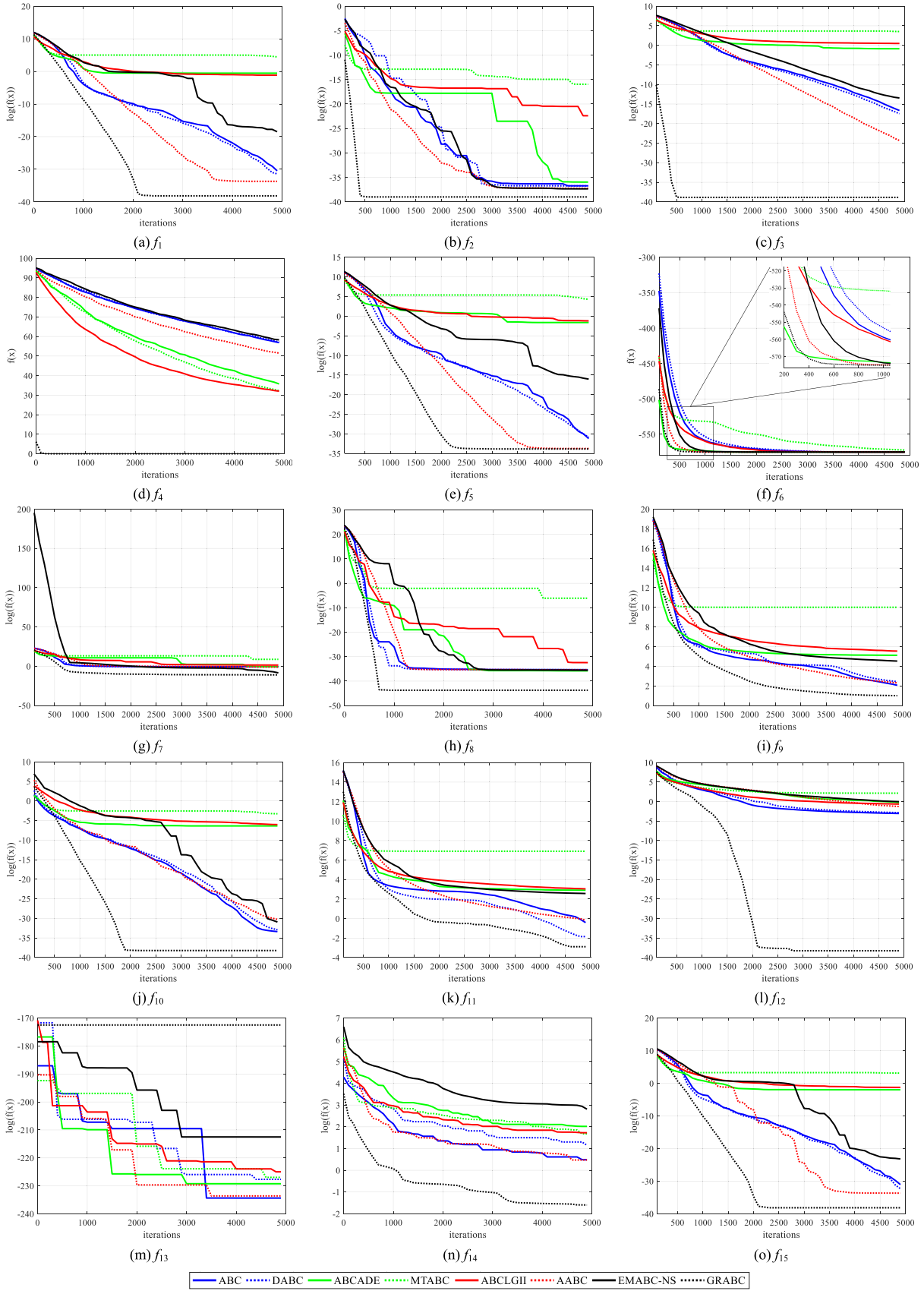


FIGURE 4. The convergence graph of the methods with $ND = 100$ for f_1-f_{15} (unimodal functions).

TABLE 5. Multimodal functions.

Functions	Algorithms							
	ABC	DABC	ABCADE	MTABC	ABCLGII	AABC	EMABC-NS	GRABC
f_{16}								
Worst	1994.2	2458.8	3085.1	3491.1	2980.4	2589.9	2963.2	**1421.8**
Best	719.62	1556.9	1923.2	1562	1116.6	1193.6	785.93	**691.40**
Avg	1241.7	1947.2	2449.2	2499.3	2236.3	1639.9	1979.3	**1217.5**
Std	299.18	293.23	324.47	525.59	488.69	355.3	605.91	**253.89**
f_{17}								
Worst	4.0811	5.5825	49.465	27.893	6.0746	4.5862	16.9594	**0**
Best	3.91E-09	0.99498	15.674	9.5636	0.81741	1.25E-05	3.9927	**0**
Avg	1.6028	2.3212	24.093	17.79	2.7853	1.2293	8.6224	**0**
Std	1.3224	1.2193	7.7369	5.0716	1.288	1.0456	3.2709	**0**
f_{18}								
Worst	1.0087	1.0085	1.0282	1.0211	1.0029	1.007	1.0022	**1.0011**
Best	1.003	1.0026	1.0035	1.0011	1.0005	1.0018	1	**1**
Avg	1.0064	1.0049	1.0116	1.0052	1.0012	1.0041	1.0004	**1.0002**
Std	0.001643	0.001658	0.006955	0.004674	0.000541	0.001356	0.00037	**0.000328**
f_{19}								
Worst	1.4113	3.845	3.65E+06	3.53E+06	1197.6	2.1192	**2.77E-15**	0.097511
Best	6.25E-05	0.002784	0.019328	3.0898	21.423	0.004822	**1.42E-15**	1.59E-15
Avg	0.16167	0.44904	1.83E+05	1.88E+05	213.24	0.34997	**2.09E-15**	0.005448
Std	0.32898	0.87932	8.16E+05	7.87E+05	279.7	0.52001	**3.07E-15**	0.002173
f_{20}								
Worst	0.066236	0.031323	0.45417	0.31016	0.037822	0.005484	**2.27E-11**	0.000105
Best	0.00101	0.003394	0.000262	0.016959	0.002837	0.000157	**1.00E-14**	1.70E-12
Avg	0.014899	0.017819	0.14206	0.12561	0.013365	0.001488	**2.15E-12**	1.39E-05
Std	0.018475	0.008116	0.14564	0.09345	0.009461	0.00138	**6.01E-12**	2.55E-06
f_{21}								
Worst	7.72E+44	3.72E+43	7.62E+38	5.25E+35	8.54E+40	5.85E+44	7.28E+34	**1.47E-15**
Best	1.51E+28	2.07E+36	2.00E+28	7.16E+26	1.33E+30	8.34E+34	1.07E+26	**4.22E-19**
Avg	3.98E+43	5.90E+42	4.27E+37	5.69E+34	4.43E+39	2.95E+43	3.76E+33	**1.24E-16**
Std	1.72E+44	1.19E+43	1.70E+38	1.53E+35	1.91E+40	1.31E+44	1.62E+34	**3.54E-16**
f_{22}								
Worst	1.49E-07	3.20E-07	2.4809	4.5787	0.1842	0.003426	0.7216	**7.99E-15**
Best	2.05E-08	1.71E-08	0.96034	1.2118	0.032689	9.69E-12	2.92E-13	**4.44E-15**
Avg	6.01E-08	7.83E-08	1.5814	2.8955	0.10784	0.000171	0.2451	**5.33E-15**
Std	3.80E-08	8.16E-08	0.43654	0.88902	0.045153	0.000766	0.2770	**1.58E-15**
f_{23}								
Worst	**1**	**1**	902.78	9316.6	118.73	**1**	5.4867	**1**
Best	**1**	**1**	3.332	21.832	14.079	**1**	1	**1**
Avg	**1**	**1**	87.754	1746.7	36.391	**1**	2.2782	**1**
Std	**0**	**0**	211.37	2716.7	26.42	**0**	1.2713	**0**
f_{24}								
Worst	6.0999	6.2158	4.6182	5.7999	5.8999	4.8999	4.5999	**0.099873**
Best	4.6146	4.8999	3.2999	3.5999	4.1999	3.8999	3.2999	**1.62E-16**
Avg	5.629	5.6108	3.8559	4.2999	5.0749	4.4399	3.9899	**0.041924**
Std	0.44652	0.36539	0.39463	0.49418	0.41279	0.26439	0.3307	**0.004921**
f_{25}								
Worst	**-3888.3**	**-3888.3**	-3817.7	-3833.2	-3845.8	**-3888.3**	-3860.1	**-3888.3**
Best	**-3916.6**	**-3916.6**	**-3916.6**	**-3916.6**	**-3916.6**	**-3916.6**	**-3916.6**	**-3916.6**
Avg	**-3912.4**	-3909.5	-3882.7	-3878	-3895.3	-3910.9	-3906.6	-3900.4
Std	**8.0755**	9.7296	23.559	23.208	19.766	8.4178	14.6562	9.5416
f_{26}								
Worst	0.019988	0.019794	0.42029	1.1176	0.080321	2.16E-13	0.1821	**0**
Best	5.55E-16	5.55E-16	0.000108	0.035738	0.001033	1.11E-16	**0**	**0**
Avg	0.001862	0.00099	0.068144	0.55649	0.016394	1.65E-14	0.0092	**0**
Std	0.005749	0.004426	0.1099	0.31702	0.022698	4.91E-14	0.0407	**0**
f_{27}								
Worst	1.48E-23	2.91E-23	2.86E-23	1.74E-23	2.41E-23	2.10E-23	1.05E-22	**1.08E-32**
Best	2.14E-27	2.38E-26	1.41E-26	1.59E-25	3.59E-25	7.90E-26	3.20E-25	**2.00E-34**
Avg	5.44E-24	4.32E-24	5.89E-24	5.20E-24	6.17E-24	4.08E-24	1.55E-23	**3.63E-33**
Std	5.10E-24	7.20E-24	8.10E-24	4.23E-24	6.25E-24	5.02E-24	2.25E-23	**4.20E-33**
f_{28}								
Worst	2.77E-17	3.06E-17	2.67E-17	3.43E-17	3.61E-17	2.73E-17	2.89E-17	**2.35E-17**
Best	1.67E-17	1.62E-17	1.66E-17	1.56E-17	1.80E-17	1.67E-17	1.71E-17	**2.08E-18**
Avg	2.12E-17	2.09E-17	2.08E-17	2.19E-17	2.27E-17	2.05E-17	2.22E-17	**1.59E-17**
Std	2.88E-18	3.73E-18	2.90E-18	4.10E-18	4.49E-18	3.16E-18	3.53E-18	**1.56E-17**
f_{29}								
Worst	1.29E-14	9.11E-14	0.031101	0.043204	0.000553	2.93E-15	7.74E-10	**2.55E-15**
Best	2.32E-15	2.68E-15	7.27E-09	1.52E-05	4.15E-05	1.78E-15	1.21E-15	**1.07E-15**
Avg	3.53E-15	8.38E-15	0.001618	0.011275	0.000219	2.38E-15	3.87E-11	**2.28E-15**
Std	2.25E-15	1.96E-14	0.006942	0.015604	0.000155	3.16E-16	1.73E-10	**1.81E-16**

TABLE 5. *(Continued.)* Multimodal functions.

f_{30}								
Worst	3.81E-13	3.41E-13	0.18266	35.828	0.076428	1.16E-11	0.0110	**2.33E-15**
Best	3.05E-15	4.52E-15	8.49E-07	0.001782	0.002748	1.86E-15	1.38E-15	**1.22E-15**
Avg	6.34E-14	4.50E-14	0.01732	2.2542	0.020266	5.83E-13	0.0015	**2.20E-15**
Std	1.15E-13	9.83E-14	0.042128	7.9804	0.017935	2.59E-12	0.0037	**1.50E-16**
f_{31}								
Worst	-96.631	-96.191	-93.778	-95.099	-97.495	-97.588	-97.6128	**-98.842**
Best	-97.744	-97.732	-96.341	-97.817	-98.691	-98.32	-99.0629	**-99.251**
Avg	-96.975	-97.009	-95.266	-96.798	-98.291	-97.97	-98.3025	**-99.094**
Std	0.28846	0.39086	0.60517	0.6448	0.28447	0.18197	0.3774	**0.14893**
f_{32}								
Worst	0.57742	0.62746	0.2868	0.33859	0.58395	0.38393	0.4034	**4.89E-05**
Best	0.3253	0.36555	0.14151	0.1245	0.30341	0.21015	0.1731	**2.08E-06**
Avg	0.45189	0.45906	0.20671	0.19828	0.42975	0.31557	0.2523	**1.72E-05**
Std	0.066616	0.066718	0.037151	0.058658	0.086735	0.056106	0.00507	**1.36E-05**

TABLE 6. Statistical test (Wilcoxon rank-sum test).

Functions	Algorithms							
	ABC	DABC	ABCADE	MTABC	ABCLGII	AABC	EMABC-NS	GRABC
f_1	+	+	+	+	+	+	+	
f_2	+	+	+	+	+	+	+	
f_3	+	+	+	+	+	+	+	
f_4	+	+	+	+	+	+	+	
f_5	+	+	+	+	+	+	+	
f_6	~	~	~	+	~	~	~	
f_7	+	+	+	+	+	+	+	
f_8	+	+	+	+	+	+	+	
f_9	+	+	+	+	+	+	+	
f_{10}	+	+	+	+	+	+	+	
f_{11}	+	+	+	+	+	+	+	
f_{12}	+	+	+	+	+	+	+	
f_{13}	-	-	-	-	-	-	-	
f_{14}	+	+	+	+	+	+	+	
f_{15}	+	+	+	+	+	+	+	
f_{16}	~	+	+	+	+	+	+	
f_{17}	+	+	+	+	+	+	+	
f_{18}	+	+	+	+	+	+	+	
f_{19}	+	+	+	+	+	+	-	
f_{20}	+	+	+	+	+	+	-	
f_{21}	+	+	+	+	+	+	+	
f_{22}	+	+	+	+	+	+	+	
f_{23}	~	~	+	+	+	~	+	
f_{24}	+	+	+	+	+	+	+	
f_{25}	-	-	+	+	+	-	~	
f_{26}	+	+	+	+	+	+	+	
f_{27}	+	+	+	+	+	+	+	
f_{28}	~	~	~	~	~	~	~	
f_{29}	+	+	+	+	+	~	+	
f_{30}	+	+	+	+	+	+	+	
f_{31}	+	+	+	+	+	+	+	
f_{32}	+	+	+	+	+	+	+	
w/t/l	26/4/2	27/3/2	29/2/1	30/1/1	29/2/1	26/4/2	26/3/3	

Dancing behavior is a pivotal function within the ABC algorithm, serving as a catalyst for the exchange of positional information about food sources. Although the basic ABC algorithm inherently lacks directionality, incorporating directional information has emerged as a critical requirement for achieving a successful solution during this stage. Notably, the artificial ABC hive does not participate in the dissemination of this knowledge. Consequently, the local search capability of the fundamental ABC algorithm notably decreases, impeding the convergence rate and leading to a protracted process of convergence resulting from this specific predicament.

To enhance the performance of the ABC algorithm, the MTABC strives to intricately balance exploration and exploitation, aiming to optimize the desired optimum value search. However, despite these earnest endeavors, the convergence rate remains disappointingly low. In stark contrast, ABCLGII exhibits a conspicuously superior convergence rate by exclusively deploying bees within the visual scope, which diligently scour the environment to locate and forage food

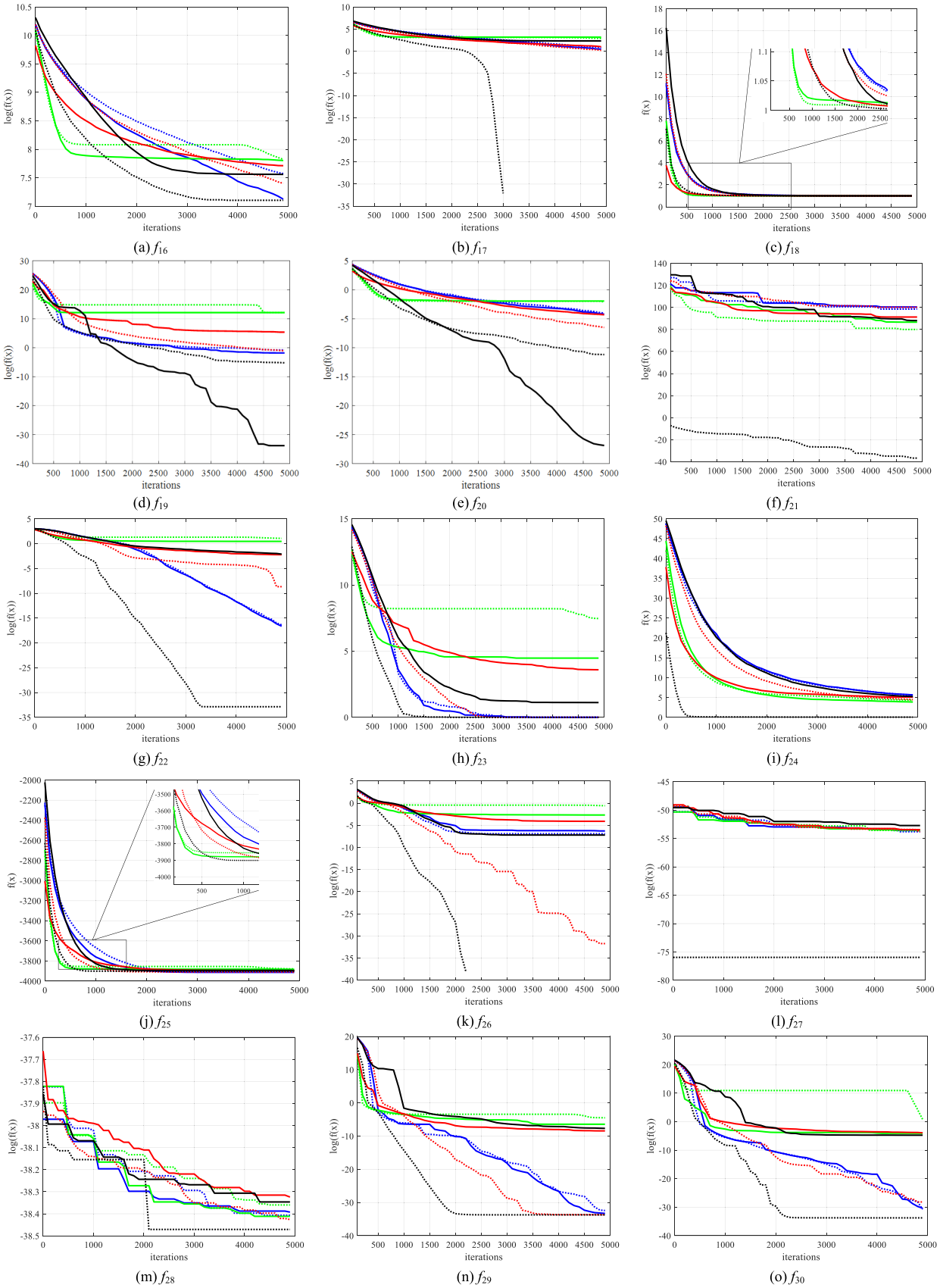
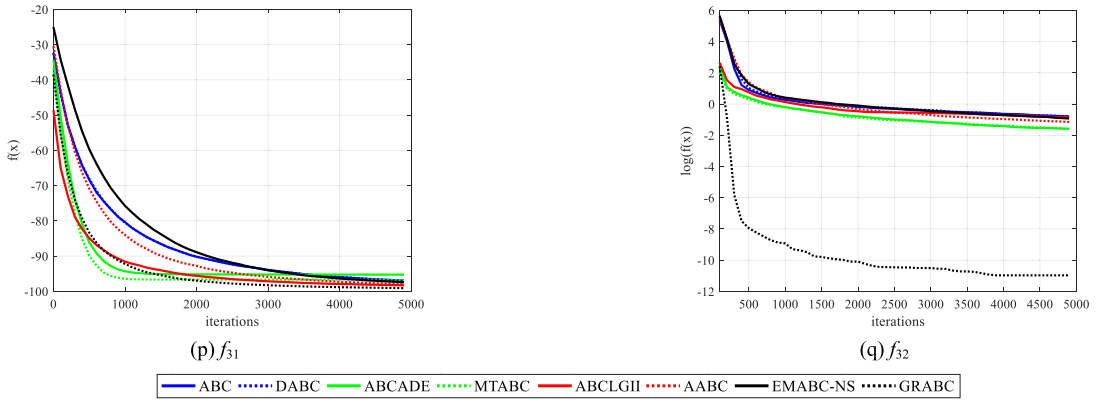


FIGURE 5. The convergence graph of the methods with $ND = 100$ for f_{16}-f_{32} (multimodal).



(p) f_{31} (q) f_{32}

FIGURE 5. *(Continued.)* **The convergence graph of the methods with $ND = 100$ for f_{16}-f_{32} (multimodal).**

TABLE 7. **Average computational time (seconds).**

Functions	Algorithms							
	ABC	DABC	ABCADE	MTABC	ABCLGII	AABC	EMABC-NS	GRABC
f_1	3.65	4.88	5.28	4.11	4.95	3.39	3.54	6.11
f_2	5.19	4.75	5.65	6.97	6.35	4.78	5.65	8.12
f_3	3.28	3.42	4.29	4.52	4.99	3.54	4.44	5.80
f_4	3.23	3.24	5.77	4.32	4.65	3.37	4.21	5.87
f_5	3.39	3.39	6.25	4.37	5.00	3.69	4.14	6.03
f_6	3.45	4.85	5.22	5.83	6.90	8.19	4.22	5.13
f_7	4.78	4.98	5.27	7.65	7.95	6.61	6.26	6.79
f_8	4.64	5.03	6.64	7.43	7.84	5.41	5.83	6.78
f_9	3.93	3.62	5.35	6.72	7.06	5.08	5.38	5.79
f_{10}	6.12	5.66	10.5	8.68	10.0	8.50	6.01	8.47
f_{11}	3.36	3.21	5.88	5.94	5.35	4.43	3.93	5.18
f_{12}	4.21	4.46	6.86	7.14	7.95	6.02	5.58	6.61
f_{13}	5.59	5.90	7.74	9.32	8.29	8.30	6.76	8.85
f_{14}	3.82	3.54	4.63	9.15	5.91	7.90	4.73	5.93
f_{15}	3.75	3.69	5.80	5.98	5.25	4.43	4.09	5.38
f_{16}	3.75	3.77	4.46	4.65	5.49	3.84	5.34	5.95
f_{17}	3.35	3.41	4.08	4.25	4.83	3.55	4.76	5.38
f_{18}	3.66	3.63	4.35	4.51	5.08	3.78	5.07	5.97
f_{19}	3.22	3.30	4.03	4.17	4.72	3.45	4.31	5.24
f_{20}	3.43	3.45	4.18	4.35	4.86	3.61	4.35	5.53
f_{21}	4.82	4.84	5.76	5.85	6.24	4.87	6.33	6.87
f_{22}	3.68	3.54	4.26	4.46	4.94	3.73	4.88	5.66
f_{23}	3.83	3.84	4.54	4.65	5.21	3.91	4.71	5.71
f_{24}	3.31	3.39	4.11	4.24	4.85	3.51	4.46	5.37
f_{25}	4.86	5.01	5.85	5.94	6.27	5.01	5.70	7.16
f_{26}	3.76	3.75	4.48	4.71	5.52	3.97	5.66	6.09
f_{27}	4.50	4.49	5.28	5.42	5.78	4.59	5.43	7.14
f_{28}	3.94	4.00	4.74	4.82	5.55	4.21	4.42	5.84
f_{29}	7.33	7.45	8.28	8.85	9.66	7.87	9.44	11.50
f_{30}	7.44	7.51	8.24	8.62	9.59	8.03	9.56	11.48
f_{31}	5.46	5.44	6.18	6.52	7.06	5.70	6.03	8.39
f_{32}	4.62	4.65	5.37	5.67	6.27	4.94	5.71	7.23

sources. Conversely, the DABC adopts a targeted approach by employing a meticulously tailored direct search equation, facilitating swift localization and acquisition of food sources, thereby resulting in accelerated convergence.

Moreover, to broaden the exploration horizons and uncover a diverse spectrum of new food sources, ABCADE incorporates the DE operator. This integration enables purposefully crafted equations to discover novel food sources to be generated. Similarly, the AABC employs a discerning strategy of selecting bees with high fitness values to pinpoint food sources throughout both the employed and onlooker bee phases. This strategic selection process effectively nurtures diversity during the exploration of food sources.

In the case of EMABC-NS, the neighborhood technique is used to increase the performance of finding and foraging for food sources in the employed and onlooker bee phases, respectively. This method aims to improve the exploration process.

All of the comparison methods discussed above are dedicated to improving the equation utilized for the identification

and acquisition of food sources in both the employed and onlooker bee phases. Nevertheless, conventional ABC and their variants function as single-dimensional undirected search algorithms. As a consequence, they do not encompass bidirectional search algorithms, such as *vleft* and *vright*, within the chosen dimension. Consequently, these algorithms encounter difficulties in efficiently exploring and exploiting the optimal value and thus predisposing individuals toward becoming ensnared in local optima.

In this paper, GRABC, a novel approach that employs vector reflection to develop equations for the efficient discovery and foraging of food sources in both the *vleft* and *vright* directions within the selected dimension, is introduced. The developed equations are applicable to both the employed and onlooker bee phases. Furthermore, we propose the incorporation of an inertial weight value, strategically determined to enhance the influence of the convergence rate. This weight is initialized with a large value and progressively decreases over subsequent iterations. Additionally, we present an equation that establishes a limit value, aiding in the identification of bees trapped in local optima during the scout bee phase. According to our strategies, vector reflection is highly effective in helping bee particles find and forage food sources with a high convergence rate, as illustrated in (23) and (24). However, this strategy could become trapped in a local optimum. Therefore, an improved scout bee phase will be utilized to address this situation. Through extensive experimentation, this method showed superior performance to alternative techniques.

VII. CONCLUSION

In this paper, GRABC, a novel variant of the standard ABC algorithm, is presented. Its efficacy is assessed using 32 numerical benchmark functions featuring unimodal and multimodal features. GRABC incorporates vector reflection and inertial weighting to formulate equations (*vleft* and *vright*) for both the employed and onlooker bee phases, notably accelerating the convergence rate. Furthermore, the integration of grouping bees fosters enhanced diversity during the exploration of food sources. Additionally, an equation is devised to compute the new positions of scout bees considering the proper limit value. Empirical findings compellingly demonstrate that the GRABC algorithm outperforms other algorithms in terms of solution quality and convergence characteristics, as shown by metrics such as the worst, best, average results, and standard deviation. A thorough comparative analysis established the evident superiority of GRABC over the fundamental ABC, MTABC, ABCLGII, ABCADE, DABC, AABC, and EMABC-NS algorithms. To apply GRABC (i.e., vector reflection) with another SI, equations (*vleft* and *vright*) in the employed and onlooker bee phases can be adapted to enhance the exploitation process. Moreover, grouping bees can also be applied to another SI for more diverse searching.

Employing this method for complex real-world continuous optimization problems is advantageous for further enhancing the applicability of GRABC. Notable applications include clustering, data mining, and the design and optimization of communication networks. Moving forward, our ongoing endeavors revolve around the seamless integration of our techniques with complementary operators, such as the DE operator or GA operator, to accelerate convergence toward the optimal value.

REFERENCES

[1] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm Evol. Comput.*, vol. 33, pp. 1–17, Apr. 2017.

[2] K. S. Tang, K. F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Process. Mag.*, vol. 13, no. 6, pp. 22–37, Dec. 1996.

[3] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, pp. 341–359, Dec. 1997.

[4] M. Dorigo and T. Stutzle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.

[5] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, Perth, WA, Australia, Sep. 1995, pp. 1942–1948.

[6] M. I. Solihin, and M. F. Zanil, "Performance comparison of Cuckoo search and differential evolution algorithm for constrained optimization," *Proc. IOP Conf. Ser., Mater. Sci. Eng.*, vol. 160, no. 1, pp. 1–7, 2016.

[7] X.-S. Yang, *Nature-Inspired Optimization Algorithms*, 1st ed. London, U.K.: Elsevier, 2014.

[8] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Syst. Mag.*, vol. 22, no. 3, pp. 52–67, Jun. 2002.

[9] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Mar. 2008.

[10] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Erciyes Univ., Kayseri, Türkiye, Tech. Rep. TR06, 2005.

[11] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Oct. 2007.

[12] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 687–697, Jan. 2008.

[13] H. U. R. Habib, U. Subramaniam, A. Waqar, B. S. Farhan, K. M. Kotb, and S. Wang, "Energy cost optimization of hybrid renewables based V2G microgrid considering multi objective function by using artificial bee colony optimization," *IEEE Access*, vol. 8, pp. 62076–62093, 2020.

[14] H. U. R. Habib, A. Waqar, A. K. Junejo, M. M. Ismail, M. Hossen, M. Jahangiri, A. Kabir, S. Khan, and Y.-S. Kim, "Optimal planning of residential microgrids based on multiple demand response programs using ABC algorithm," *IEEE Access*, vol. 10, pp. 116564–116626, 2022.

[15] S. F. Hussain, A. Pervez, and M. Hussain, "Co-clustering optimization using artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 97, Dec. 2020, Art. no. 106725.

[16] G. Yao, Y. Wu, X. Huang, Q. Ma, and J. Du, "Clustering of typical wind power scenarios based on K-means clustering algorithm and improved artificial bee colony algorithm," *IEEE Access*, vol. 10, pp. 98752–98760, 2022.

[17] Y. Yang and D. Liu, "A hybrid discrete artificial bee colony algorithm for imaging satellite mission planning," *IEEE Access*, vol. 11, pp. 40006–40017, 2023.

[18] S. Lin, F. Li, X. Li, K. Jia, and X. Zhang, "Improved artificial bee colony algorithm based on multi-strategy synthesis for UAV path planning," *IEEE Access*, vol. 10, pp. 119269–119282, 2022.

[19] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Syst. Appl.*, vol. 37, no. 8, pp. 5682–5687, Aug. 2010.

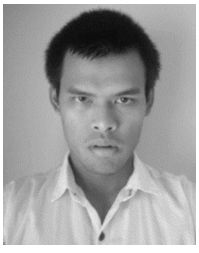
[20] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Inf. Sci.*, vol. 192, pp. 120–142, Jun. 2012.

[21] W. Gao and S. Liu, "A modified artificial bee colony algorithm," *Comput. Oper. Res.*, vol. 39, pp. 687–697, Jan. 2012.

[22] M. S. Kıran and O. Fındık, "A directed artificial bee colony algorithm," *Appl. Soft Comput.*, vol. 26, pp. 454–462, Jan. 2015.

[23] M. Maeda and S. Tsuda, "Reduction of artificial bee colony algorithm for global optimization," *Neurocomputing*, vol. 148, pp. 70–74, Jan. 2015.

[24] Z. Li, W. Wang, Y. Yan, and Z. Li, "PS–ABC: A hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8881–8895, Dec. 2015.
[25] X. Yan, Y. Zhu, H. Chen, and H. Zhang, "A novel hybrid artificial bee colony algorithm with crossover operator for numerical optimization," *Natural Comput.*, vol. 14, no. 1, pp. 169–184, Mar. 2015.
[26] X. Zhou, Z. Wu, H. Wang, and S. Rahnamayan, "Gaussian bare-bones artificial bee colony algorithm," *Soft Comput.*, vol. 20, no. 3, pp. 907–924, Mar. 2016.
[27] W. Gao, L. Huang, J. Wang, S. Liu, and C. Qin, "Enhanced artificial bee colony algorithm through differential evolution," *Appl. Soft Comput.*, vol. 48, pp. 137–150, Nov. 2016.
[28] L. Cui, G. Li, Q. Lin, Z. Du, W. Gao, J. Chen, and N. Lu, "A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation," *Inf. Sci.*, vols. 367–368, pp. 1012–1044, Nov. 2016.
[29] X. Song, Q. Yan, and M. Zhao, "An adaptive artificial bee colony algorithm based on objective function value information," *Appl. Soft Comput.*, vol. 55, pp. 384–401, Jun. 2017.
[30] S. S. Jadon, R. Tiwari, H. Sharma, and J. C. Bansal, "Hybrid artificial bee colony algorithm with differential evolution," *Appl. Soft Comput.*, vol. 58, pp. 11–24, Sep. 2017.
[31] Z. Liang, K. Hu, Q. Zhu, and Z. Zhu, "An enhanced artificial bee colony algorithm with adaptive differential operators," *Appl. Soft Comput.*, vol. 58, pp. 480–494, Sep. 2017.
[32] G. Li, L. Cui, X. Fu, Z. Wen, N. Lu, and J. Lu, "Artificial bee colony algorithm with gene recombination for numerical function optimization," *Appl. Soft Comput.*, vol. 52, pp. 146–159, Mar. 2017.
[33] W.-J. Yu, Z.-H. Zhan, and J. Zhang, "Artificial bee colony algorithm with an adaptive greedy position update strategy," *Soft Comput.*, vol. 22, no. 2, pp. 437–451, Jan. 2018.
[34] Q. Lin, M. Zhu, G. Li, W. Wang, L. Cui, J. Chen, and J. Lu, "A novel artificial bee colony algorithm with local and global information interaction," *Appl. Soft Comput.*, vol. 62, pp. 702–735, Jan. 2018.
[35] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Appl. Math. Comput.*, vol. 214, no. 1, pp. 108–132, Aug. 2009.
[36] D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas, "CIXL2: A crossover operator for evolutionary algorithms based on population features," *J. Artif. Intell. Res.*, vol. 24, pp. 1–48, Jul. 2005.
[37] S. Xiao, W. Wang, H. Wang, D. Tan, Y. Wang, X. Yu, and R. Wu, "An improved artificial bee colony algorithm based on elite strategy and dimension learning," *Mathematics*, vol. 7, no. 3, p. 289, Mar. 2019.
[38] W.-L. Xiang, Y.-Z. Li, R.-C. He, X.-L. Meng, and M.-Q. An, "An improved artificial bee colony algorithm with fitness-based information," *IEEE Access*, vol. 7, pp. 41052–41065, 2019.
[39] W. Gao, Z. Wei, Y. Luo, and J. Cao, "Artificial bee colony algorithm based on Parzen window method," *Appl. Soft Comput. J.*, vol. 74, pp. 679–692, Jan. 2019.
[40] X. Zhou, Y. Wu, M. Zhong, and M. Wang, "Artificial bee colony algorithm based on multiple neighborhood topologies," *Appl. Soft Comput.*, vol. 111, Nov. 2021, Art. no. 107697.
[41] H. Peng, C. Deng, and Z. Wu, "Best neighbor-guided artificial bee colony algorithm for continuous optimization problems," *Soft Comput.*, vol. 23, no. 18, pp. 8723–8740, Sep. 2019.
[42] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Appl. Math. Comput.*, vol. 217, no. 7, pp. 3166–3173, Dec. 2010.
[43] H. Wang, W. Wang, S. Xiao, Z. Cui, M. Xu, and X. Zhou, "Improving artificial bee colony algorithm using a new neighborhood selection mechanism," *Inf. Sci.*, vol. 527, pp. 227–240, Jul. 2020.
[44] Q. Jiang, J. Cui, Y. Ma, L. Wang, Y. Lin, X. Li, T. Feng, and Y. Wu, "Improved adaptive coding learning for artificial bee colony algorithms," *Appl. Intell.*, vol. 52, no. 7, pp. 7271–7319, May 2022.
[45] C. Wang, P. Shang, and P. Shen, "An improved artificial bee colony algorithm based on Bayesian estimation," *Complex Intell. Syst.*, vol. 8, no. 6, pp. 4971–4991, Dec. 2022.
[46] X. Li, S. Zhang, L. Yang, and P. Shao, "Neighborhood-search-based enhanced multi-strategy collaborative artificial bee colony algorithm for constrained engineering optimization," *Soft Comput.*, vol. 27, no. 19, pp. 13991–14017, Oct. 2023.
[47] T. Ye, H. Wang, T. Zeng, M. G. H. Omran, F. Wang, Z. Cui, and J. Zhao, "An improved two-archive artificial bee colony algorithm for many-objective optimization," *Expert Syst. Appl.*, vol. 236, Feb. 2024, Art. no. 121281.
[48] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 524–541, Aug. 2015.
[49] Q. Askari, M. Saeed, and I. Younas, "Heap-based optimizer inspired by corporate rank hierarchy for global optimization," *Expert Syst. Appl.*, vol. 161, Dec. 2020, Art. no. 113702.
[50] G. Wu, R. Mallipeddi, and P. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization," Nanyang Technol. Univ., Singapore, Tech. Rep. 1-18, 2016.
[51] Y. Wang, Z. Cai, L. Guo, G. Li, Y. Yu, and S. Gao, "A spherical evolution algorithm with two-stage search for global optimization and real-world problems," *Inf. Sci.*, vol. 665, Apr. 2024, Art. no. 120424.
[52] A. Guo, Y. Wang, L. Guo, R. Zhang, Y. Yu, and S. Gao, "An adaptive position-guided gravitational search algorithm for function optimization and image threshold segmentation," *Eng. Appl. Artif. Intell.*, vol. 121, May 2023, Art. no. 106040.
[53] H. Wang, Z. Wu, S. Rahnamayan, H. Sun, Y. Liu, and J.-S. Pan, "Multi-strategy ensemble artificial bee colony algorithm," *Inf. Sci.*, vol. 279, pp. 587–603, Sep. 2014.



SONGYUT PHOEMPHON received the Bachelor of Science and Master of Science degrees (Hons.) from Khon Kaen University, in 2014 and 2016, respectively, and the Ph.D. degree in computer science from the Department of Computer Science, Faculty of Science, Khon Kaen University, in 2020. He is currently a Lecturer with the Institute of Digital Arts and Science (DIGITECH), Suranaree University of Technology, Thailand. He also acquired invaluable industry experience during his internship with the National Electronics and Computer Technology Center (NECTEC), Thailand. His research interests include mobile computing, mobile and wireless sensor networks, mobile ad hoc networks, machine learning and intelligent systems, computer networks, and distributed systems.