COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 03/06 pp 35–58
Volume 2, Number 1, March 2022
DOI: 10.23919/CSMS.2022.0004

Differential Evolution with Level-Based Learning Mechanism

Kangjia Qiao, Jing Liang*, Boyang Qu, Kunjie Yu, Caitong Yue, and Hui Song

Abstract: To address complex single objective global optimization problems, a new Level-Based Learning Differential Evolution (LBLDE) is developed in this study. In this approach, the whole population is sorted from the best to the worst at the beginning of each generation. Then, the population is partitioned into multiple levels, and different levels are used to exert different functions. In each level, a control parameter is used to select excellent exemplars from upper levels for learning. In this case, the poorer individuals can choose more learning exemplars to improve their exploration ability, and excellent individuals can directly learn from the several best individuals to improve the quality of solutions. To accelerate the convergence speed, a difference vector selection method based on the level is developed. Furthermore, specific crossover rates are assigned to individuals at the lowest level to guarantee that the population can continue to update during the later evolutionary process. A comprehensive experiment is organized and conducted to obtain a deep insight into LBLDE and demonstrates the superiority of LBLDE in comparison with seven peer DE variants.

Key words: level-based learning; Differential Evolution (DE); parameter adaptation; exemplar selection

1 Introduction

Differential Evolution (DE)[1], similar to other Evolutionary Algorithms (EAs)[2, 3], is a population-based stochastic optimization method. DE has been used in a variety of engineer problems[4–6] and scientific researches[7–9] because of its simple operators, easy implementation, the use of a few control parameters, and high search efficiency. The excellent optimization performance of DE depends on its internal structure, which is an implicit self-adaptation system. Given a scale factor value, the solutions will converge to one specific region of the decision space during evolution, which means the influence of the perturbation vector formed by two random individuals from the whole population on the population decreases gradually. Thus, DE has the exploration ability in the early stage and the exploitation ability in the later stage[10].

Although DE has exhibited outstanding optimization performance, it is still influenced by its mutation strategy and three control parameters, i.e., scale factor (F), crossover rate (CR), and population size (NP)[11]. When the population cannot generate better solutions during a large number of generations due to some reasons, the same moves constituted by fixed F and mutation strategy will not be effective in subsequent generations[10]. The population may fall stagnant and maintains a low diversity. The parameter CR determines how many components of the mutation vector the trial vector can inherit to further affect the diversity of the population. A small NP can lead to a limited number of moves, which could cause premature convergence. In fact, the mutation strategy and control parameters dominate the numbers and quality of moves. The number of moves determines the diversity

• Kangjia Qiao, Jing Liang, Kunjie Yu, and Caitong Yue are with the School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, China. E-mail: qiaokangjia@yeah.net; liangjing@zzu.edu.cn; yukunjie1990@163.com; zzuyuecaitong@163.com.
• Boyang Qu is with the School of Electronic and Information, Zhongyuan University of Technology, Zhengzhou 450007, China. E-mail: qby1984@hotmail.com.
• Hui Song is with the School of Engineering, RMIT University, Melbourne 3000, Australia. E-mail: hui.song@rmit.edu.au.
* To whom correspondence should be addressed.
Manuscript received: 2021-12-16; revised: 2022-01-06; accepted: 2022-03-02

of the population, which is desired for the population in the early stage to explore new promising areas. The quality of moves affects the convergence speed of the population and the accuracy of the solutions. The balance between diversity and convergence is still the main issue to be studied and solved.

To guarantee the quality of moves, some methods increase the exploitation ability of DE[12]. Excellent individuals contain promising information; thus, the exploitation of these individuals can provide better search directions for other individuals. For example, Gong and Cai[13] proposed a rank-based mutation rule, where excellent individuals are provided more chances for other bad individuals to learn. The results show that the proposed rule can improve the performance of some basic mutation strategies with strong exploration ability. The local search technique[14] also contributes to the exploitation of DE. The best individual in the population searches its local surroundings to find a more promising area, which can significantly improve the quality of the optimal solution. These methods are essentially an elite mechanism. However, if too much emphasis is placed on the utilization of elite individuals, then the population may fall into the local optima. Thus, a certain degree of randomization is essential.

Modifying the structure of the original DE[15], such as the search logic and parameter selection, is also a popular method[10]. Some algorithms[16, 17] use multiple mutation strategies to enrich the search behavior of DE. These strategies are dynamically adjusted during the evolution to balance diversity and convergence. The mating selection is random, and the information of the population is shared globally; thus, premature convergence would happen. A structured population[18, 19] is used to reduce the speed of information flow to maintain diversity. For example, multipopulation-based DE evolves the individuals using the information of other individuals within the same population to improve diversity. The information exchange among subpopulations serves to accelerate convergence. In neighbor-based DE[20], each individual is allowed to communicate with its neighbors. Choosing neighbors and determining the number of neighbors are important[21]. The parameter settings also need to be considered. The most successful improvement on NP is achieved by the population size reduction technique[22−24], which gradually reduces NP to increase the exploitation ability. Random CR and F can enrich the moves. However, they lose the convergence rate[25, 26]. The self-adaptive or adaptive method[27−29] is a more promising approach to adjust the parameters because it reduces the sensitivity of the algorithm to the problem.

Inspired by the above, a new DE with the fitness-based population structure or Level-Based Learning DE (LBLDE) is developed in this study. The whole population is partitioned into multiple levels equally based on the sorted fitness values. Different levels will choose different numbers of individuals as learning exemplars, which guarantees the exploitation of excellent individuals and the exploration of the remaining individuals simultaneously. Moreover, an individual can select only the individuals that form the difference vector from the respective level or the higher level, which guarantees the high quality of the difference vector. The main contributions of this study are as follows:

(1) A novel DE variant LBLDE is proposed. The level-based learning mechanism assigns different numbers of learning exemplars for each level, which ensures the correct search directions of the population and considers both diversity and convergence speed.

(2) According to level, a difference vector selection method is proposed, which limits the number of moves and guarantees that each individual has the potential to become better to increase the convergence speed of the population.

(3) To exert the unique effect of individuals in different levels, different CR values are allocated to individuals at different levels. CR is assigned to 1 for the individuals with poor fitness, which can help the population continue to update in the late phase of the evolving process.

The level-based learning mechanism[30] was originally used in particle swarm optimization to address large-scale optimization problems. It helps improve the population diversity without adding an extra computing burden. Large-scale optimization problems require high diversity to avoid premature convergence[31]. Thus, the individuals of the first level in particle swarm optimization[30] do not evolve and enter into the next generation directly. Inspired by it, the level-based learning mechanism is introduced into DE to improve the performance of DE. In LBLDE, the individuals in the first level learn from several most excellent individuals to accelerate the convergence rate to solve complex global optimization problems. With the combination of the level-based learning mechanism, the difference vector selection method, and the *CR* allocation scheme, LBLDE can achieve a good performance.

The subsequent content of this paper is organized as follows. The literature review is introduced in Section 2. The proposed LBLDE algorithm is described in Section 3. Comprehensive experiments and results are presented in Section 4. Conclusions and future work are outlined in Section 5.

2 Literature Review

The proposed LBLDE is a novel DE variant that employs an elitism mechanism and a fitness-based population structure. The elitism includes the local search executed by the several best individuals in the first level and the level-based learning mechanism for all levels. Each level can be regarded as a subpopulation, and all subpopulations communicate in a fitness-based top-down way to maintain diversity. Moreover, from the viewpoint of multistrategy, the strategy adopted by the last level is similar to DE/current-to-rand/1, and that used by the other levels is similar to DE/current-to-*p*best/1. Both the multipopulation model and the multistrategy method modify the original structure of DE. Therefore, in this section, some DE variants are reviewed from two aspects: (1) the increase of exploitation in DE, and (2) the modification of the structure in DE.

2.1 Increase of exploitation in DE

2.1.1 Elite learning mechanism

Introducing the elite individuals into the mutation strategy can significantly increase the exploitation of DE. JADE[28] is a popular DE variant that proposes a successful mutation strategy DE/current-to-*p*best/1. In this strategy, each individual learned from one elite that is randomly selected from the top p individuals. The elite individuals provide promising search directions. Thus, JADE obtains better results on unimodal and simple multimodal problems[32]. JADE fully utilizes the information of global excellent solutions to improve the convergence, which causes the diversity in JADE to be too low to solve complex multimodal problems. To better balance diversity and convergence, some new mutation strategies are proposed, which can be divided into two categories. In the first category, the global elite and poor individuals are utilized to balance convergence and diversity. For example, Wang et al.[33] designed an improved DE/rand/2 mutation strategy. The new strategy divides the whole population into two parts: the elite part to ensure promising search directions and the non-elite part to increase population diversity. Yu et al.[5] used a constrained DE (CDE) to solve unmanned aerial vehicle problems. In CDE, the better the individual is, the higher the chosen probability it has, in which objective and constraint values are two criteria. A. W. Mohamed and A. K. Mohamed[34] introduced a new mutation rule, which divides the population into three parts based on their fitness. The three individuals for the mutation strategy are randomly selected from three parts, respectively. Unlike in the above DE variants that utilize the poor individuals to maintain diversity, the local elite is selected[35]. Thus, for each individual, two elites are selected from its neighbors and the whole population, respectively, to balance diversity and convergence. In the second category, the elite individuals are dynamically selected to gradually adjust the search behaviors of the algorithm. For example, Cai et al.[36] devised an adaptive guiding mechanism to dynamically adjust the selection range of learning exemplars, which meets the requirements of the algorithm for diversity and convergence in different stages. Yu et al.[37] developed a new DE variant that adaptively adjusted the greediness degree of the mutation strategy. Notably, the algorithms in the first category fix the selection ranges of elites for all individuals. Thus, the degree of exploitation is also fixed. Moreover, although the algorithms in the second category dynamically adjust the selection ranges of elites, the selection ranges for all individuals in each generation are the same, which ignores the attributes of each individual. Unlike these methods, the proposed method assigns different ranges of elites for each level, which not only balances diversity and convergence but also considers the attributes of each individual.

In addition, although the algorithms above have obtained better results on different test functions, their success is also highly dependent on their adaptive parameter settings, thus enriching the number of moves. Elite learning mechanisms and adaptation parameters work together to balance diversity and convergence. The adaptation parameters belong to the category of structural modification. Thus, they will be introduced in the next subsection. In addition, in these algorithms, the extra parameters that control the number or selection range of elites are difficult to set because different problems have different requirements for the convergence rate of the algorithm.

2.1.2 Local search technique

The local search technique is another effective method. Liang et al.[6] refined the optimal solution of the population at the end of each generation to improve its quality. Wang et al.[14] divided the evolutionary process into two stages. The standard DE is implemented in the first stage to explore the search space. Then, the chaos local search is used by the better individuals to accelerate the convergence rate. Wang and Tang[38] proposed a self-adaptive local search that considers the diversity and quality of solutions in the external archive. Memetic computing[39] is a popular pattern to hybridize the local search. For example, Li et al.[40] developed a Memetic Adaptive DE (MADE) to identify the parameters of multiple photovoltaic models. MADE adopts the Nelder Mead simplex method to refine the solution. Liu et al.[41] proposed a new memetic DE. The designed generalized fitness strategy uses three simple local search methods to enhance the convergence. Caponio et al.[42] proposed a super-fit memetic differential evolution, which first uses particle swarm optimization to evolve partial individuals to obtain a super-fit leader that would lead the population to evolve in the framework of DE. Then, two complementary local search techniques are adopted to detect promising search regions. Local search is also used to update F for generating better offspring in later generations[43].

The local search technique is of great significance to improve the accuracy of the solutions in the optimal region. However, their employment mode and times need to consider the characteristics of the problem and the total computing resources.

2.2 Modification of structure in DE

2.2.1 Multipopulation

DE has been improved by the structured population, which is inspired by structured EAs[44]. The two most famous structured EAs are the cellular model and the distributed model. In the cellular model, each individual has unique neighbors and can be allowed to communicate with only its neighbors to improve diversity. The distributed model divides the whole population into multiple subpopulations connected by one topology. The success of structured DE is rooted in the limitation of the pools of vectors contributing to the mutation. The main issue of multipopulation-based DE is how to allow the islands to communicate and thus return a degree of randomization. In the past two decades, various DE variants with a structured population have been proposed. Wu et al.[45] designed a new multipopulation model (MPEDE) that assigns one unique mutation strategy for each subpopulation. The subpopulation incorporates some individuals that are selected randomly from the whole population in each generation, and the best individual is inserted into each subpopulation. The subpopulation sizes are dynamically adjusted to exert the advantage of the fittest strategy. Then, Li et al.[46] proposed an improved MPEDE algorithm called MPMSDE, in which a new grouping method, an information sharing mechanism, and a new mutation strategy are proposed. Weber et al.[47] proposed a distributed DE with explorative-exploitation population families. The subpopulations form two parts: the subpopulations in the first part are connected by a ring topology and are dedicated to exploring the search space, while the other subpopulations in the second part are endowed with population size reduction technology to strengthen the exploitation ability. De Falco et al.[48] proposed a biological invasion-inspired migration scheme for distributed DE. In this new migration scheme, the excellent individuals in one subpopulation are inserted into each neighbor subpopulation, and these invasive individuals compete with the individuals of the neighbor subpopulation for survival. Then, an improved invasion-based model[49] endowed with three different parameter updating mechanisms was developed to further enhance its performance. Bouteldja and Batouche[50] presented a cellular DE that connects the subpopulations with a neighborhood criterion. Combined with the proposed multilevel thresholding method, this method is used to solve multilevel color image segmentation.

Multipopulation DE includes some parameters, such as the number of subpopulations, the number of neighbors, and migration interval, which are related to the performance of algorithms.

2.2.2 Multistrategy

In traditional DE, one basic mutation strategy characterized by exploration or exploitation is used for the whole search process, which limits the diversity of moves and cannot solve various kinds of problems. To overcome this shortcoming, multiple strategies have been used for DE. Qin et al.[51] proposed a self-adaptive DE, in which the proportion of four different basic strategies is dynamically adjusted according to the historical experience. Mallipeddi et al.[27] proposed a novel DE with the ensemble of parameters and mutation strategies, where each individual is assigned a mutation strategy and control parameters to increase the diversity of the population. An inheritance mechanism is designed for the offspring to utilize the strategies and parameters of the successful parent individuals. Similarly, Fan et al.[52] developed a DE with strategy adaptation and knowledge-based control parameters. Differently, a novel adaptation method is used to select the more promising strategy and parameters from the respective pool for the individuals who fail in evolution. Liang et al.[53] proposed an ensemble-based DE, where multiple different strategies are performed at two stages to exert distinct functions. Qiao et al.[54] developed a self-adaptive resource allocation-based DE to solve constrained optimization problems, where three different strategies with different preferences on constraints and objectives are self-adaptively employed. Wang et al.[55] proposed a composite DE, which employs three groups of parameter combinations and three distinct mutation strategies. Each parent individual generats three offspring by three mutation strategies, and the parent individual competes with three offspring to increase the selective pressure. Liu et al.[56] proposed a Two-Stage DE (TSDE) method to obtain better diversity and convergence rate. TSDE employs distinct mutation strategies and parameters to improve the exploration ability in the early stage and focuses on the convergence in the late stage. A MultiRole-based Differential Evolution algorithm (MRDE)[57] was proposed to take advantage of different mutation strategies. In MRDE, each group has three individuals, each individual of which has a special role that is allocated with a unique strategy for generating the mutation vector. Wang et al.[58] devised a novel DE variant with two different mutation strategies to solve the gait optimization of humanoid robots, which is a constrained optimization problem. These two strategies have equal selective probability, and one is devoted to improving diversity, while another could drive the population to approach the global optimal solution. Tan et al.[59] proposed a DE with an adaptive mutation operator based on fitness landscape, in which a random forest model is trained to study the relationships between fitness landscape features and three mutation strategies. Then, the trained model would recommend the most suitable strategy for the new problem.

In these algorithms, the utilization of multiple strategies is successful because different problems can be addressed by different strategies. These strategies are adjusted by the self-adaptive scheme to make the algorithm suitable for different evolutionary stages. However, the self-adaptive scheme not only needs additional parameters but also increases the computational complexity of the algorithm.

2.2.3 Parameter adaptation

Adaptation or self-adaptation of control parameters has been used to improve the performance of DE, and it is often accompanied by other improvement schemes. In JADE[28], an effective adaptation method was proposed, which considers both the experience of past generations and the last generation to update F and CR. The Cauchy distribution has a wider boundary for generating more diverse F, and normal distribution is used to produce CR. Later, this method is improved from different aspects. For example, Peng et al.[60] added a weighting strategy for CR. Li et al.[61] believed that CR and F should be updated in pairs. Thus, the authors clustered the two parameters used by successful individuals and then updated them. Tanabe[62] developed a success history based parameter adaptation method to avoid errors in a certain generation. Zhou et al.[63] devised a sorting CR scheme, in which the generated CR values are sorted in ascending order, and the better individual is assigned a smaller CR value to increase the exploitation ability of excellent individuals. Yu et al.[64] proposed a two-level parameter adaptation method. F is increased and CR is decreased in the exploration state to assist the algorithm to reach more spaces. Then, in the exploitation state, two parameters are changed reversely. Tirronen and Neri[65] proposed a DE with fitness diversity self-adaptation, where the diversity indicator measured by the fitness values of the population is used to adjust the values of F, CR, and NP. Zamuda et al.[66] introduced a structured population size reduction technique into a structured population DE. The population size reduction technique could gradually increase the exploitation ability of the population during the evolution. Rakshit et al.[67] introduced an adaptive memetic algorithm that combines DE with Q-learning. The action is to select an F value from 10 different F values between 0 and 1 for one individual.

Despite the success of the self-adaptive parameter adjustment method, they need additional parameters that should be adjusted carefully.

Apart from the above DE variants, some latest reviews[10, 68] have presented algorithmic design issues of DE. In addition, the above DE variants are designed from only one or two specific aspects, and they can still be improved due to unsatisfactory performance. Considering the impact of the parameters, structure, local search, and global search on the search ability of DE, a new LBLDE is proposed in this paper to further improve the performance of DE.

3 Proposed Algorithm

To design a DE with better performance, this paper proposes a new LBLDE algorithm. The level-based learning mechanism is introduced into DE to obtain higher diversity and reduce the probability of premature convergence. In addition, a new difference vector selection method and specific parameter setting are used in LBLDE to accelerate the convergence rate.

3.1 Level-based learning mechanism

Figure 1 shows how to partition the population into multiple levels. The population is first sorted from the best to the worst in the light of their fitness. The best individual's rank is 1, and the worst individual's rank is NP. The population is partitioned into NL (an integer) levels, and each level has the same number of individuals (LS, $LS = NP/NL$). The level that has the best individual is regarded as the first level, denoted L_1. The level that includes the worst individual is regarded as the last level, denoted as L_{NL}. Let $X_{i,j}\ (i = 1,2,\ldots,NL; j = 1,2,\ldots,LS)$ be the j-th individual in the i-th level. The purpose of LBLDE is to arrange suitable learning exemplars for each level. $p = (p_1, p_2, \ldots, p_{NL})$ controls the number of exemplars for different levels, and the top p_i individuals are defined as the exemplars of i-th level. $p_i\ (i = 1,2,\ldots,NL)$ is denoted as follows:

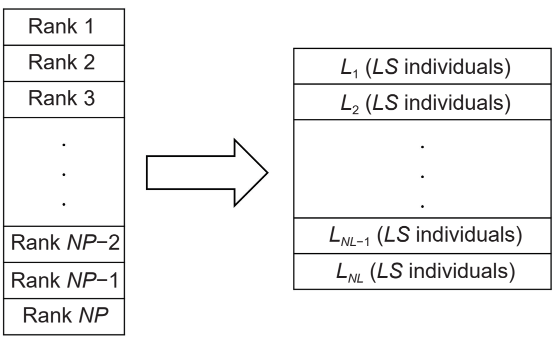

Fig. 1 Partitioning the whole population into *NL* levels.

$$p_i = \frac{(i-1)\times LS}{NP} \times 100\% \tag{1}$$

On the basis of Eq. (1), one individual can select any individual at a better level as its learning exemplar. Poor individuals can choose learning exemplars from the majority of individuals in the population, which can improve the population diversity. In addition, for $X_{1,j}$, it selects only the several best individuals at L_1. A very small value is assigned to p_1 ($p_1 = 0.05$), which means only the top 5% superior individuals can be selected for $X_{1,j}$ to conduct the local search. A small p_1 guarantees the exploitation ability of excellent individuals. The selection ranges of elites for the whole population are not fixed as in the previous algorithms[33−35]; thus, the population has more diverse search abilities. Therefore, the level-based learning mechanism will maintain a good balance between diversity and convergence.

3.2 Difference vector selection method

DE evolves each individual based on differential information. In the traditional DE algorithm, the individuals that make up the difference vectors are randomly selected from the whole population to maintain population diversity. In this paper, LBLDE modifies the difference vector selection method to match the stratification.

The strategy DE/current-to-$p$best/1 is used in LBLDE. The mutation strategy formula is as follows:

$$V_{i,j} = X_{i,j} + F_{i,j} \times \left(X_{\text{best}}^{p_i} - X_{i,j}\right) + F_{i,j} \times \left(X_{r_1,j} - X_{r_2,j}\right) \tag{2}$$

where $X_{\text{best}}^{p_i}$ is the learning exemplar of the target individual and randomly selected from several elite individuals, i.e., the top p_i individuals. In addition, $F_{i,j}$ is the scaling factor. The difference vectors $X_{r_1,j}$ and $X_{r_2,j}$ are different from each other.

For $X_{i,j}$, $X_{r_1,j}$ is randomly selected from L_1 to L_{i-1} and $X_{r_2,j}$ is randomly selected from L_1 to L_i, which ensures the right search direction for $X_{i,j}$, and can speed the convergence rate of the population. For the first level, no better level exists, so these two individuals are randomly selected from the first level. If LBLDE uses the general difference vector selection method, then the population diversity of LBLDE will improve and the convergence speed will slow down.

3.3 Parameter adaptation method

The parameter combination is another key to improving the performance of DE. Selecting suitable parameters is important because they can enhance the robustness of the algorithm[28, 29, 64]. The adaptive method is well known to be able to dynamically adjust parameters to balance exploration and exploitation effectively. Therefore, LBLDE employs an effective parameter adaptive method proposed in JADE[28].

For $X_{i,j}$, its $CR_{i,j}$ and $F_{i,j}$ are independently generated based on Eqs. (3) and (4),

$$CR_{i,j} = rand_n(\mu_{CR}, 0.1) \tag{3}$$

$$F_{i,j} = rand_c(\mu_F, 0.1) \tag{4}$$

where μ_{CR} and μ_F are the mean values and 0.1 is the standard deviation. $CR_{i,j} > 1$ and $CR_{i,j} < 0$ are set as $CR_{i,j} = 1$ and $CR_{i,j} = 0$, respectively. In the same way, if $F_{i,j} > 1$, then $F_{i,j} = 1$. However, when $F_{i,j} \leqslant 0$, it will be regenerated. The initial μ_{CR} ($\mu_{CR\text{-}ini}$) and μ_F are 0.5 and then adjusted after every cycle of evolution using Eqs. (5) and (6),

$$\mu_{CR} = (1-c)\times\mu_{CR} + c\times mean_A(S_{CR}) \tag{5}$$

$$\mu_F = (1-c)\times\mu_F + c\times mean_L(S_F) \tag{6}$$

where c is a number ranging from 0 to 1; S_{CR} and S_F indicate the set of CR and F values of all successful individuals in the previous generation, respectively; $mean_A(\cdot)$ is the arithmetic mean as usual; and $mean_L(\cdot)$ indicates the flowing Lehmer mean,

$$mean_L(S_F) = \frac{\sum_{F\in S_F} F^2}{\sum_{F\in S_F} F} \tag{7}$$

For Eq. (3), if $\mu_{CR\text{-}ini}$ = 0.5, then μ_{CR} will first increase and maintain a high level until the end, which is harmful to LBLDE because DE needs small CR values to accelerate convergence in the later stage. Therefore, $\mu_{CR\text{-}ini}$ is set as a smaller value in LBLDE that is different from JADE. In this way, μ_{CR} has less possibility of increasing in the late stage. Furthermore, because the population diversity is high in LBLDE, μ_{CR} should be adaptively changed at the lower level. In Section 4.1, the experiment proves that $\mu_{CR\text{-}ini}$ = 0.35 is the optimal choice. Moreover, the CR values are set to 1 in the lower levels. As a result of this setting, almost all components of lower levels come from the mutation vectors. In the later stage, the influence of difference vectors on the population is small due to low population diversity; thus, the generated trial vectors will be close to excellent individuals. Hence, this setting can improve the exploitation ability of the population to continue to evolve in the late stage. The verification of these modifications is shown in Section 4.3.

Based on the above process, Algorithm 1 gives the pseudo-code of LBLDE.

Algorithm 1 Pseudo-code of LBLDE

Input: NP (population size), NL (the number of levels), LS (number of individuals in each level), $MaxFES$ (maximal number of the function evaluations)
Output: optimal solution X and its fitness $f(X)$

Set FES = 0, G = 0, μ_F = 0.5, μ_{CR} = 0.35, c = 0.l;
Create a random initial population P and evaluate each individual X in P;
FES=FES+NP;
while $FES \leqslant MaxFES$ **do**
  G = G+1;
  Sort P by the fitness in ascending order and partition it into NL levels equally;
  for i = 1 : NL **do**
    for j = 1 : LS **do**
      Generate $CR_{i,j} = rand_n(\mu_{CR}, 0.1)$ and $F_{i,j} = rand_c(\mu_F, 0.1)$;
      if i == 1 **then**
        p_i = 0.05;
      else
        $p_i = \frac{(i-1)\times LS\times 100\%}{NP}$
      end
      Randomly select an individual from top p_i individuals as exemplar $X^{p_i}_{best}$;
      Randomly select two individuals, $X_{r_1,j}$ from top p_{i-1} and $X_{r_2,j}$, from top p_i individuals;
      Generate the mutation vector $V_{i,j} = X_{i,j} + F_{i,j}\times(X^{p_i}_{best} - X_{i,j}) + F_{i,j}\times(X_{r_1,j} - X_{r_2,j})$;
      Generate the trial vector $U_{i,j}$ by implementing crossover operation between $X_{i,j}$ and $V_{i,j}$ and evaluate $U_{i,j}$;
      FES=FES+l;
      if $f(X_{i,j}) \leqslant f(U_{i,j})$ **then**
        $X_{i,j} \leftarrow X_{i,j}$;
      else
        if $i \neq NL$ **then**
          $X_{i,j} \leftarrow U_{i,j}$, $S_{CR} \leftarrow CR_{i,j}$, $S_F \leftarrow F_{i,j}$;
        else
          $X_{i,j} \leftarrow U_{i,j}$, $S_F \leftarrow F_{i,j}$;
        end
      end
    end
  end
end
$\mu_{CR} = (1-c)\times\mu_{CR} + c\times mean_A(S_{CR})$;
$\mu_F = (1-c)\times\mu_F + c\times mean_L(S_F)$;
end

4 Experimental Study

The CEC'2017 benchmark set[69] is adopted in this paper to derive deep insights into the performance of LBLDE. This test suite contains 30 test functions: unimodal functions (F1–F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20), and composition functions (F21–F30).

Four dimensions (D), namely 10D, 30D, 50D, and 100D, are used, which represent different degrees of difficulties. The maximum number of function evaluations ($MaxFES$) is 10 000D. NP = 100 for 10D, 30D, and 50D, NP = 160 for 100D, and NL = 4 for all dimensions. $\mu_{CR\text{-}ini}$ is 0.35, and the number of levels that CR = 1 (NLB) is set as 1. The discussion of these parameter settings is presented in Section 4.1. All algorithms are executed 51 independent times for a fair comparison. If $f(X)-f(X^*)<10^{-8}$, then the error is set as 0. X and X^* are the found and true optimal solutions, respectively.

The experiments are performed step by step as follows: (1) The parameter sensitivity is analyzed to obtain the optimal parameter configuration; (2) the superiority of LBLDE is demonstrated based on the comparison experiments between LBLDE and other peer DE variants. At the same time, the statistical test is performed using the Wilcoxon rank-sum test with α = 0.05. "+", "−" , and "=" indicate that the results obtained by LBLDE are significantly better than, worse than, and similar to those found by compared algorithms, respectively; and (3) the effectiveness of the proposed schemes in LBLDE is verified.

4.1 Parameter analysis

The parameters to be tuned in LBLDE include NL, NLB, and $\mu_{CR\text{-}ini}$, which influence the evolutionary trend of the population. Therefore, the parameter sensitivity analysis is first explored to obtain the best parameter configuration. All experiments in this subsection and Section 4.3 are implemented on the 30D case.

First, the influence of NL is discussed. A large NL will reduce the probability of excellent individuals being selected, leading to high population diversity. Four different NL values (i.e., 1, 4, 5, and 10) are used for compared experiments, where NL = 4 means that the population of LBLDE is divided into four levels. The boxplot figures of LBLDE with different NL values on nine different functions are provided in Fig. 2. These nine functions represent different kinds of functions. In Fig. 2b, with the increase of NL, the stability of the algorithm begins to decline, because F3

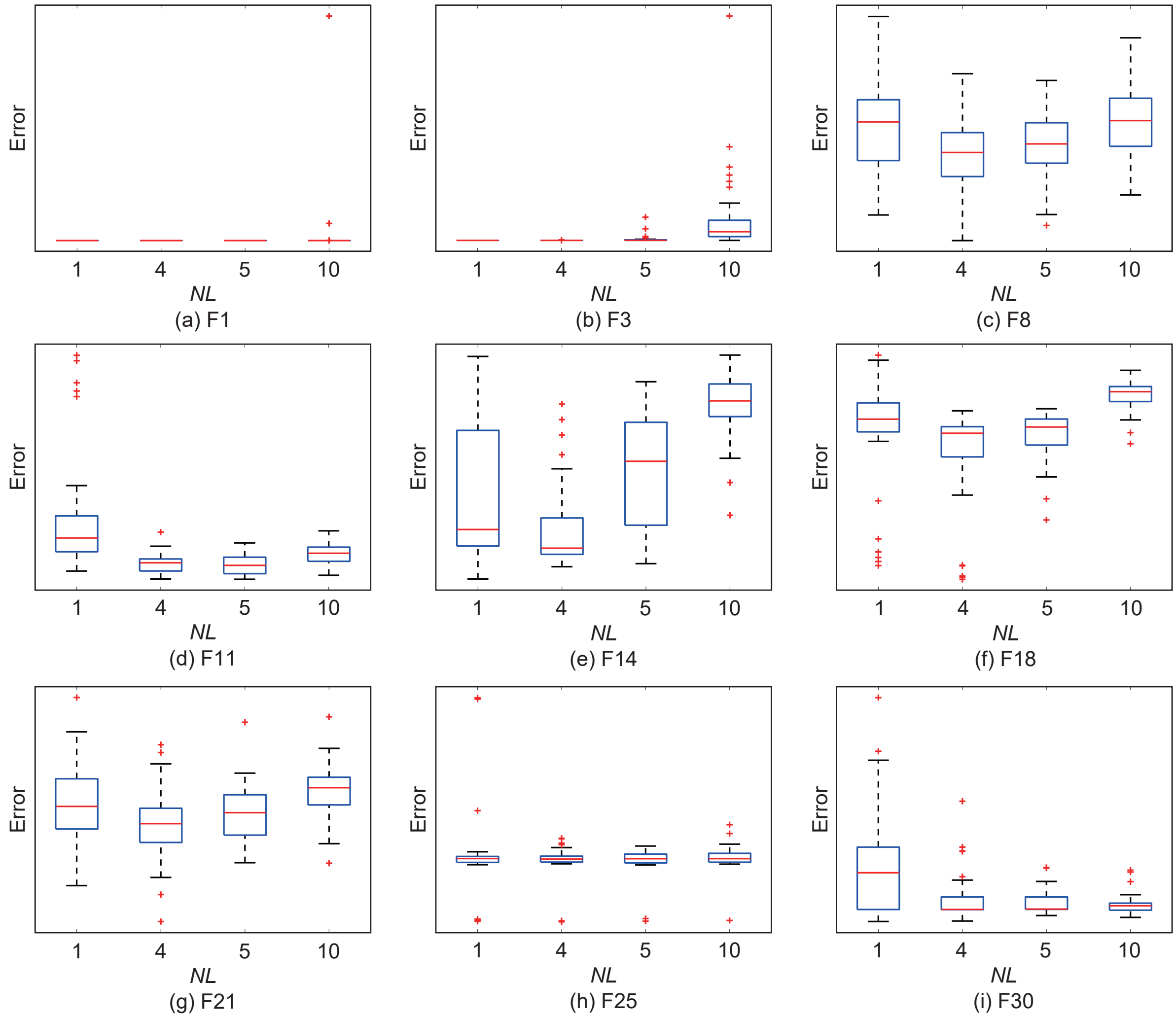

Fig. 2 Boxplot figures of LBLDE with different NL values on nine test functions.

is a unimodal function that needs excellent individuals to guide the evolution of the population, and a small *NL* leads more individuals to learn from the several best exemplars. For most of the other problems, LBLDE with *NL* = 4 performs best. The ranking of each LBLDE variant on 30 functions based on the Friedman test is presented in Table 1, where LBLDE with *NL* = 4 has the best ranking. Thus, *NL* = 4 will be the optimal choice.

Second, the influence of *NLB* is discussed, where *NLB* = 0 represents no level's *CR* is 1, and *NLB* = 2 indicates *CR* = 1 in the last two levels. The number of levels whose *CR* = 1 is discussed. From the experimental results presented in Fig. 3 and Table 2, when *NLB* = 0, the algorithm obtains the worst performance on most of the functions. For F3, F14, and F18, *NLB* = 1, 2, and 3 can obtain similar results. However, in the case of *NLB* = 1, LBLDE achieves better performances on other functions. Moreover, on the basis of the ranking in Table 2, LBLDE with *NLB* = 1 obtains the best ranking. Therefore, *NLB* is set to 1.

Table 1 Rankings of LBLDE with different *NL* values on the CEC'2017 test suite.

NL	Ranking
1	3
4	1
5	2
10	4

Lastly, the influence of the $\mu_{CR\text{-}ini}$ is discussed, where $\mu_{CR\text{-}ini}$ is set to 0.05, 0.15, 0.25, 0.35, 0.45, and 0.50, respectively. We aim to find a trade-off $\mu_{CR\text{-}ini}$ on most of the test functions. If $\mu_{CR\text{-}ini}$ is too small, then increasing μ_{CR} during the evolution is difficult. However, if $\mu_{CR\text{-}ini}$ is 0.5, then it may increase with a 50% probability and may not decrease in the late stage. LBLDE does not need a large *CR* value to maintain

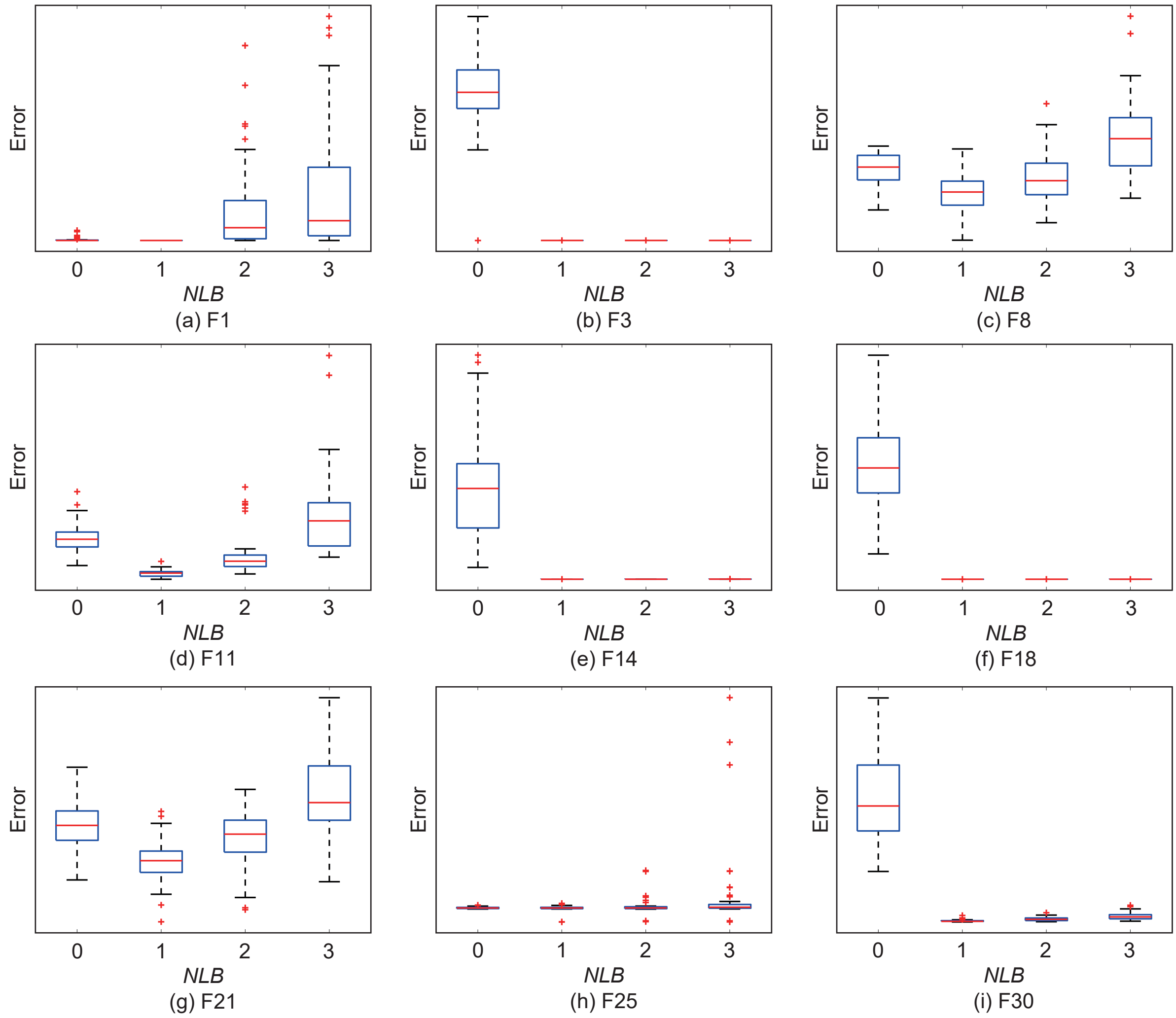

Fig. 3 Boxplot figures of LBLDE with different *NLB* values on nine test functions.

Table 2 Rankings of LBLDE with different *NLB* values on the CEC'2017 test suite.

NLB	Ranking
0	3
1	1
2	2
3	3

population diversity because of multilevels. Figure 4 presents the boxplot figures of LBLDE with different $\mu_{CR\text{-}ini}$ on nine test functions. For F8 and F21, as $\mu_{CR\text{-}ini}$ increases, the performance of the algorithm increases. However, a large $\mu_{CR\text{-}ini}$ causes LBLDE perform poorly on other functions. On the basis of the ranking provided in Table 3, $\mu_{CR\text{-}ini}$ = 0.35 leads to better results. Therefore, the combination of NL = 4, NLB = 1, and $\mu_{CR\text{-}ini}$ = 0.35 can lead to satisfactory performance by considering diversity and convergence and is chosen for the following experiments.

4.2 Comparisons with state-of-the-art DE variants

Seven peer DE algorithms including EAGDE[70], EFADE[71], AMECoDEs[35], TSDE[56], RNDE[21], MPEDE[45], and TVDE[72], are used for comparison. EAGDE adopts the fitness-based population structure; EFADE designs the triangular mutation operator to balance diversity and convergence; AMECoDEs and RNDE adopt the neighbor-based population structure, and elite information is used; TSDE is a two-stage algorithm that adopts multiple strategies to enhance the search abilities of the population; MPEDE is a multipopulation algorithm, and it self-adaptively adjusts the utilization of multiple strategies; and TVDE designs a time-varying strategy to gradually increase the utilization of excellent individuals. These algorithms adopt different mechanisms. Thus, the comparison between LBLDE and these algorithms can verify the superiority of LBLDE. The parameters of the

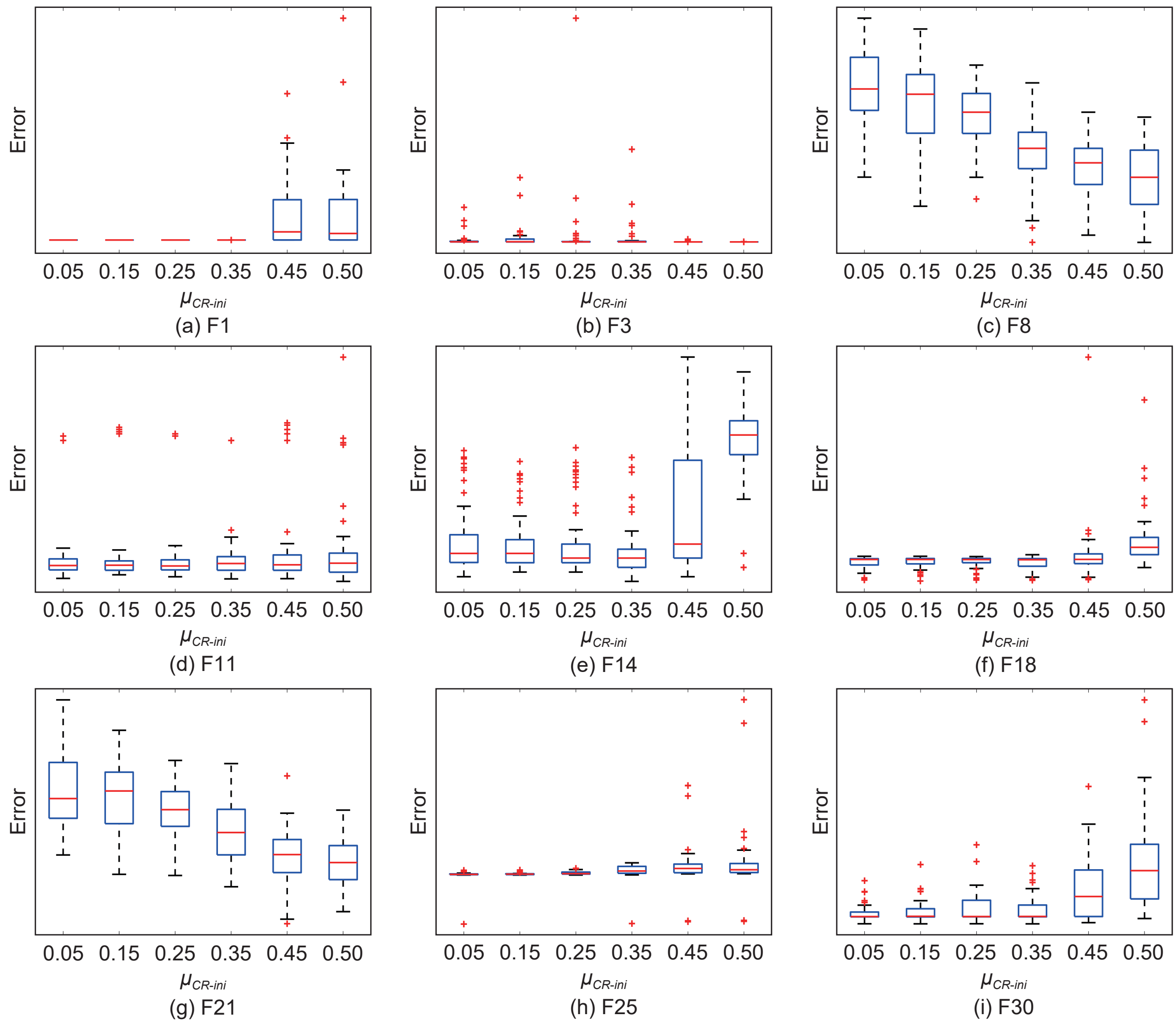

Fig. 4 Boxplot figures of LBLDE with different $\mu_{CR\text{-}ini}$ values on nine test functions.

Table 3 Rankings of LBLDE with different $\mu_{CR\text{-}ini}$ values on the CEC'2017 test suite.

$\mu_{CR\text{-}ini}$	Ranking
0.05	2
0.15	6
0.25	4
0.35	1
0.45	3
0.50	5

Table 4 Parameter settings of compared algorithms.

Algorithm	Parameter
EAGDE[70]	$p=0.1$, $N_{min}=12$
EFADE[71]	$\varepsilon=0.0001$, $p_1=0.5$
AMECoDEs[35]	$p=0.1$, $\varepsilon=0.001$
RNDE[21]	$F=0.5$
TSDE[56]	$[F=1, CR=0.1]$, $[F=1, CR=0.9]$, $[F=0.8, CR=0.2]$
MPEDE[45]	$\lambda=0.2$, $ng=20$
TVDE[72]	$Freq=0.05$, $G=10\ 000$

compared algorithms are provided in Table 4. Tables 5–8 provide the average error values (average) and related standard deviation (std) of all eight algorithms on four different dimensions. Table 9 provides the statistical results regarding average and std on the $10D$, $30D$, $50D$, and $100D$ cases. Figure 5 shows the average error curves over all 51 runs under the function evaluations (FES) for nine test functions.

Table 5 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite ($10D$).

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F1	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$6.47\times10^{1}\pm3.74\times10^{1}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$7.20\times10^{-10}\pm2.92\times10^{-9}$(=)	$9.94\times10^{1}\pm7.10\times10^{2}$(=)	**$0.00\times10^{0}\pm0.00\times10^{0}$**
F2	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$1.03\times10^{2}\pm7.06\times10^{2}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$**
F3	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$1.86\times10^{0}\pm1.11\times10^{0}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$**
F4	**$0.00\times10^{0}\pm0.00\times10^{0}$(−)**	$2.26\times10^{-1}\pm1.27\times10^{-1}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$(−)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(−)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(−)**	$4.45\times10^{-8}\pm3.07\times10^{-7}$(−)	$4.68\times10^{0}\pm1.42\times10^{0}$(+)	$3.17\times10^{-3}\pm1.14\times10^{-2}$
F5	$4.65\times10^{0}\pm2.10\times10^{0}$(+)	$1.13\times10^{1}\pm1.81\times10^{0}$(+)	$6.77\times10^{0}\pm2.76\times10^{0}$(+)	$3.65\times10^{0}\pm1.65\times10^{0}$(+)	$9.46\times10^{0}\pm1.78\times10^{0}$(+)	$6.07\times10^{0}\pm1.67\times10^{0}$(+)	$3.54\times10^{0}\pm1.91\times10^{0}$(+)	$2.62\times10^{0}\pm1.05\times10^{0}$
F6	$4.88\times10^{-8}\pm2.13\times10^{-7}$(+)	$6.10\times10^{-3}\pm2.16\times10^{-3}$(+)	$9.36\times10^{-6}\pm2.29\times10^{-6}$(+)	$0.00\times10^{0}\pm0.00\times10^{0}$(=)	$0.00\times10^{0}\pm0.00\times10^{0}$(=)	$2.30\times10^{-5}\pm5.68\times10^{-6}$(+)	$3.39\times10^{-5}\pm1.36\times10^{-4}$(+)	$0.00\times10^{0}\pm0.00\times10^{0}$
F7	$1.76\times10^{1}\pm3.77\times10^{0}$(+)	$2.47\times10^{1}\pm2.52\times10^{0}$(+)	$1.71\times10^{1}\pm3.41\times10^{0}$(+)	$1.43\times10^{1}\pm1.58\times10^{0}$(+)	$2.10\times10^{1}\pm2.35\times10^{0}$(+)	$1.76\times10^{1}\pm1.58\times10^{0}$(+)	$1.38\times10^{1}\pm1.98\times10^{0}$(=)	$1.34\times10^{1}\pm8.98\times10^{-1}$
F8	$4.37\times10^{0}\pm1.59\times10^{0}$(+)	$1.20\times10^{1}\pm2.38\times10^{0}$(+)	$7.35\times10^{0}\pm2.72\times10^{0}$(+)	$4.45\times10^{0}\pm2.22\times10^{0}$(+)	$1.01\times10^{1}\pm2.27\times10^{0}$(+)	$6.33\times10^{0}\pm1.58\times10^{0}$(+)	$3.73\times10^{0}\pm1.78\times10^{0}$(+)	$2.88\times10^{0}\pm1.08\times10^{0}$
F9	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$4.89\times10^{-4}\pm5.38\times10^{-4}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	**$0.00\times10^{0}\pm0.00\times10^{0}$(=)**	$1.78\times10^{-2}\pm8.91\times10^{-2}$(=)	**$0.00\times10^{0}\pm0.00\times10^{0}$**
F10	$1.40\times10^{2}\pm1.37\times10^{2}$(=)	$4.48\times10^{2}\pm9.15\times10^{1}$(+)	$1.87\times10^{2}\pm1.48\times10^{2}$(+)	$1.25\times10^{2}\pm9.56\times10^{1}$(=)	$5.00\times10^{2}\pm1.22\times10^{2}$(+)	$2.81\times10^{2}\pm1.17\times10^{2}$(+)	$1.31\times10^{2}\pm1.14\times10^{2}$(=)	$7.58\times10^{1}\pm5.83\times10^{1}$
F11	$2.93\times10^{-1}\pm4.58\times10^{-1}$(+)	$4.78\times10^{0}\pm8.57\times10^{-1}$(+)	$7.02\times10^{-1}\pm8.03\times10^{-1}$(+)	$3.71\times10^{-1}\pm5.61\times10^{-1}$(+)	$3.90\times10^{-2}\pm1.95\times10^{-1}$(=)	$2.32\times10^{0}\pm5.86\times10^{-1}$(+)	$2.72\times10^{0}\pm3.98\times10^{0}$(+)	$1.01\times10^{-7}\pm7.00\times10^{-7}$
F12	$3.70\times10^{1}\pm6.92\times10^{1}$(+)	$3.54\times10^{2}\pm1.12\times10^{2}$(+)	$1.44\times10^{1}\pm3.66\times10^{1}$(+)	$8.85\times10^{0}\pm2.91\times10^{1}$(+)	$7.08\times10^{-1}\pm2.17\times10^{0}$(−)	$1.08\times10^{1}\pm1.27\times10^{1}$(+)	$2.61\times10^{3}\pm3.96\times10^{3}$(+)	$7.86\times10^{0}\pm1.82\times10^{1}$
F13	$3.32\times10^{0}\pm2.43\times10^{0}$(=)	$1.19\times10^{1}\pm2.01\times10^{0}$(+)	$4.39\times10^{0}\pm2.17\times10^{0}$(+)	$2.65\times10^{0}\pm2.35\times10^{0}$(=)	**$2.31\times10^{0}\pm2.50\times10^{0}$(=)**	$5.88\times10^{0}\pm2.11\times10^{0}$(+)	$8.60\times10^{0}\pm1.16\times10^{1}$(+)	$2.40\times10^{0}\pm1.87\times10^{0}$
F14	$6.53\times10^{-1}\pm6.78\times10^{-1}$(+)	$5.22\times10^{0}\pm1.56\times10^{0}$(+)	$1.90\times10^{0}\pm1.30\times10^{0}$(+)	$2.15\times10^{-1}\pm4.13\times10^{-1}$(+)	$2.46\times10^{-9}\pm1.76\times10^{-8}$(=)	$4.48\times10^{0}\pm1.67\times10^{0}$(+)	$1.35\times10^{1}\pm1.13\times10^{1}$(+)	**$0.00\times10^{0}\pm0.00\times10^{0}$**
F15	$2.38\times10^{-1}\pm3.22\times10^{-1}$(+)	$2.29\times10^{0}\pm5.61\times10^{-1}$(+)	$1.44\times10^{-1}\pm3.39\times10^{-1}$(+)	$6.09\times10^{-2}\pm2.05\times10^{-1}$(+)	$6.30\times10^{-2}\pm1.46\times10^{-1}$(+)	$7.20\times10^{-1}\pm2.65\times10^{-1}$(+)	$2.74\times10^{0}\pm6.65\times10^{0}$(+)	**$2.84\times10^{-2}\pm7.83\times10^{-2}$**
F16	$4.31\times10^{-1}\pm2.65\times10^{-1}$(+)	$3.24\times10^{0}\pm1.13\times10^{0}$(+)	$4.13\times10^{-1}\pm2.09\times10^{-1}$(+)	**$2.32\times10^{-1}\pm1.92\times10^{-1}$(=)**	$4.59\times10^{-1}\pm2.56\times10^{-1}$(+)	$2.58\times10^{0}\pm9.10\times10^{-1}$(+)	$6.18\times10^{1}\pm9.18\times10^{1}$(+)	$2.87\times10^{-1}\pm1.34\times10^{-1}$
F17	$1.72\times10^{0}\pm1.36\times10^{0}$(+)	$1.06\times10^{1}\pm2.04\times10^{0}$(+)	$1.67\times10^{0}\pm3.03\times10^{0}$(+)	$3.07\times10^{-1}\pm3.38\times10^{-1}$(+)	$4.51\times10^{-1}\pm4.00\times10^{-1}$(+)	$6.93\times10^{0}\pm2.23\times10^{0}$(+)	$2.05\times10^{1}\pm2.82\times10^{1}$(+)	**$2.45\times10^{-2}\pm6.10\times10^{-2}$**
F18	$5.99\times10^{-1}\pm2.79\times10^{0}$(+)	$5.96\times10^{0}\pm1.58\times10^{0}$(+)	$1.98\times10^{-1}\pm3.49\times10^{-1}$(+)	$5.46\times10^{-2}\pm1.10\times10^{-1}$(+)	$1.07\times10^{-1}\pm1.62\times10^{-1}$(=)	$2.98\times10^{0}\pm1.43\times10^{0}$(+)	$2.21\times10^{1}\pm1.17\times10^{1}$(+)	**$3.01\times10^{-2}\pm7.26\times10^{-2}$**
F19	$2.00\times10^{-2}\pm1.86\times10^{-2}$(+)	$9.50\times10^{-1}\pm2.79\times10^{-1}$(+)	$5.17\times10^{-2}\pm4.78\times10^{-2}$(+)	$1.11\times10^{-2}\pm1.10\times10^{-2}$(=)	**$3.85\times10^{-4}\pm2.72\times10^{-3}$(−)**	$5.39\times10^{-1}\pm1.60\times10^{-1}$(+)	$1.57\times10^{0}\pm1.65\times10^{0}$(+)	$7.03\times10^{-3}\pm8.32\times10^{-3}$

(to be continued)

Table 5 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (10*D*). (continued)

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F20	1.59×10^{-1} ± 1.58×10^{-1}(+)	1.58×10^{0} ± 8.80×10^{-1}(+)	1.60×10^{-1} ± 2.24×10^{-1}(+)	**6.12×10^{-3} ± 4.37×10^{-2}(=)**	8.57×10^{-2} ± 1.54×10^{-1}(+)	8.41×10^{-1} ± 5.07×10^{-1}(+)	2.61×10^{1} ± 4.14×10^{1}(+)	1.22×10^{-2} ± 6.12×10^{-2}
F21	1.75×10^{2} ± 4.91×10^{1}(+)	1.56×10^{2} ± 5.76×10^{1}(−)	1.30×10^{2} ± 4.90×10^{1}(=)	1.59×10^{2} ± 5.41×10^{1}(=)	1.53×10^{2} ± 5.65×10^{1}(=)	**1.15×10^{2} ± 3.75×10^{1}(=)**	1.96×10^{2} ± 3.20×10^{1}(+)	1.62×10^{2} ± 5.20×10^{1}
F22	1.00×10^{2} ± 2.52×10^{-1}(+)	8.80×10^{1} ± 3.46×10^{1}(−)	9.11×10^{1} ± 2.59×10^{1}(−)	8.60×10^{1} ± 3.41×10^{1}(−)	9.61×10^{1} ± 1.96×10^{1}(=)	**8.85×10^{1} ± 3.18×10^{1}(−)**	1.03×10^{2} ± 4.09×10^{1}(+)	1.00×10^{2} ± 1.05×10^{-11}
F23	3.05×10^{2} ± 1.71×10^{0}(+)	3.12×10^{2} ± 2.20×10^{0}(+)	3.07×10^{2} ± 3.37×10^{0}(+)	3.06×10^{2} ± 2.03×10^{0}(+)	3.09×10^{2} ± 3.21×10^{0}(+)	3.06×10^{2} ± 1.69×10^{0}(+)	3.06×10^{2} ± 2.85×10^{0}(+)	**3.03×10^{2} ± 1.91×10^{0}**
F24	3.24×10^{2} ± 4.59×10^{1}(+)	2.76×10^{2} ± 1.09×10^{2}(−)	**2.62×10^{2} ± 1.11×10^{2}(−)**	2.85×10^{2} ± 9.79×10^{1}(+)	2.76×10^{2} ± 1.04×10^{2}(−)	2.74×10^{2} ± 1.03×10^{2}(−)	3.17×10^{2} ± 6.41×10^{1}(+)	2.83×10^{2} ± 9.14×10^{1}
F25	4.07×10^{2} ± 1.86×10^{1}(=)	**4.00×10^{2} ± 9.13×10^{0}(=)**	4.07×10^{2} ± 1.83×10^{1}(=)	4.03×10^{2} ± 1.49×10^{1}(=)	4.09×10^{2} ± 1.95×10^{1}(=)	4.04×10^{2} ± 1.58×10^{1}(=)	4.30×10^{2} ± 2.18×10^{1}(+)	4.13×10^{2} ± 2.16×10^{1}
F26	**3.00×10^{2} ± 0.00×10^{0}(=)**	3.00×10^{2} ± 4.46×10^{-4}(+)	3.00×10^{2} ± 1.48×10^{-13}(+)	**3.00×10^{2} ± 0.00×10^{0}(=)**	**3.00×10^{2} ± 0.00×10^{0}(=)**	**3.00×10^{2} ± 0.00×10^{0}(+)**	4.61×10^{2} ± 3.55×10^{2}(+)	**3.00×10^{2} ± 0.00×10^{0}**
F27	3.89×10^{2} ± 2.44×10^{-1}(=)	**3.89×10^{2} ± 7.76×10^{-1}(−)**	3.89×10^{2} ± 5.01×10^{-1}(=)	3.89×10^{2} ± 1.38×10^{0}(−)	3.92×10^{2} ± 2.35×10^{0}(+)	3.89×10^{2} ± 2.50×10^{-1}(=)	3.96×10^{2} ± 4.69×10^{0}(+)	3.90×10^{2} ± 2.16×10^{0}
F28	5.06×10^{2} ± 1.38×10^{2}(+)	3.17×10^{2} ± 6.74×10^{1}(−)	3.08×10^{2} ± 4.28×10^{1}(−)	**3.06×10^{2} ± 3.97×10^{1}(−)**	3.11×10^{2} ± 5.56×10^{1}(=)	3.12×10^{2} ± 5.84×10^{1}(−)	5.13×10^{2} ± 1.29×10^{2}(+)	3.34×10^{2} ± 9.55×10^{1}
F29	2.34×10^{2} ± 4.71×10^{0}(−)	2.52×10^{2} ± 4.97×10^{0}(+)	2.36×10^{2} ± 4.97×10^{0}(−)	**2.32×10^{2} ± 2.77×10^{0}(−)**	2.39×10^{2} ± 4.58×10^{0}(=)	2.50×10^{2} ± 5.72×10^{0}(+)	2.56×10^{2} ± 3.13×10^{1}(+)	2.40×10^{2} ± 3.11×10^{0}
F30	8.05×10^{4} ± 2.45×10^{5}(=)	7.09×10^{2} ± 1.22×10^{2}(+)	1.64×10^{4} ± 1.14×10^{5}(+)	4.06×10^{2} ± 3.59×10^{1}(−)	4.00×10^{2} ± 1.23×10^{1}(−)	**3.95×10^{2} ± 4.38×10^{-1}(−)**	1.47×10^{5} ± 3.47×10^{5}(+)	4.17×10^{2} ± 6.74×10^{1}
+/=/−	18/10/2	23/2/5	18/7/5	11/13/6	10/15/5	18/7/5	25/5/0	−

Table 6 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (30*D*).

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F1	**0.00×10^{0} ± 0.00×10^{0} (=)**	1.64×10^{2} ± 7.00×10^{1} (+)	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00**$\times10^{0}$ ± 0.00×10^{0}
F2	7.33×10^{0} ± 1.49×10^{1} (=)	1.29×10^{15} ± 3.72×10^{15} (+)	1.22×10^{0} ± 6.08×10^{0} (=)	**0.00×10^{0} ± 0.00×10^{0} (−)**	**0.00×10^{0} ± 0.00×10^{0} (−)**	**0.00×10^{0} ± 0.00×10^{0} (−)**	6.57×10^{5} ± 3.04×10^{6} (−)	4.25×10^{6} ± 3.03×10^{7}
F3	**0.00×10^{0} ± 0.00×10^{0} (−)**	1.22×10^{3} ± 6.32×10^{2} (+)	**0.00×10^{0} ± 0.00×10^{0} (−)**	**0.00×10^{0}** ± 0.00×10^{0} (−)	**0.00×10^{0} ± 0.00×10^{0} (−)**	4.17×10^{-10} ± 2.98×10^{-9} (−)	5.42×10^{-6} ± 3.68×10^{-5} (−)	1.19×10^{-1} ± 3.52×10^{-1}
F4	5.94×10^{1} ± 2.04×10^{0} (+)	8.36×10^{1} ± 2.86×10^{0} (+)	5.78×10^{1} ± 8.40×10^{0} (+)	4.29×10^{1} ± 2.71×10^{1} (=)	3.16×10^{1} ± 2.99×10^{1} (=)	5.32×10^{1} ± 1.75×10^{1} (=)	7.11×10^{1} ± 1.38×10^{1} (+)	**3.02**$\times10^{1}$ ± 3.19×10^{1}
F5	3.40×10^{1} ± 2.11×10^{1} (+)	1.06×10^{2} ± 8.98×10^{0} (+)	2.75×10^{1} ± 1.00×10^{1} (=)	3.63×10^{1} ± 1.08×10^{1} (+)	9.09×10^{1} ± 7.40×10^{0} (+)	2.92×10^{1} ± 8.07×10^{0} (+)	2.42×10^{1} ± 8.09×10^{0} (=)	**2.38**$\times10^{1}$ ± 3.98×10^{0}
F6	5.57×10^{-6} ± 8.26×10^{-6} (+)	1.18×10^{-2} ± 2.56×10^{-3} (+)	1.14×10^{-8} ± 3.73×10^{-8} (+)	**0.00×10^{0} ± 0.00×10^{0} (=)**	**0.00×10^{0} ± 0.00×10^{0} (=)**	2.68×10^{-9} ± 1.92×10^{-8} (=)	3.01×10^{-8} ± 1.55×10^{-7} (+)	**0.00**$\times10^{0}$ ± 0.00×10^{0}
F7	1.09×10^{2} ± 2.77×10^{1} (+)	1.58×10^{2} ± 8.40×10^{0} (+)	5.73×10^{1} ± 9.33×10^{0} (=)	6.75×10^{1} ± 9.56×10^{0} (+)	1.28×10^{2} ± 8.36×10^{0} (+)	5.44×10^{1} ± 7.42×10^{0} (=)	5.53×10^{1} ± 8.74×10^{0} (=)	**5.39**$\times10^{1}$ ± 4.98×10^{0}
F8	4.25×10^{1} ± 2.47×10^{1} (+)	1.10×10^{2} ± 7.56×10^{0} (+)	**2.46×10^{1} ± 1.06×10^{1} (=)**	3.60×10^{1} ± 1.08×10^{1} (+)	9.38×10^{1} ± 7.10×10^{0} (+)	3.05×10^{1} ± 7.63×10^{0} (+)	2.59×10^{1} ± 8.36×10^{0} (=)	2.55×10^{1} ± 5.13×10^{0}
F9	2.85×10^{-2} ± 1.08×10^{-1} (=)	6.96×10^{1} ± 2.74×10^{1} (+)	**0.00×10^{0} ± 0.00×10^{0} (=)**	3.74×10^{-2} ± 1.23×10^{-1} (=)	**0.00×10^{0} ± 0.00×10^{0} (=)**	1.96×10^{-2} ± 8.96×10^{-2} (=)	**0.00×10^{0} ± 0.00×10^{0} (=)**	8.91×10^{-3} ± 6.36×10^{-2}
F10	4.58×10^{3} ± 3.69×10^{2} (+)	4.15×10^{3} ± 3.15×10^{2} (+)	**1.67**$\times10^{3}$ ± 6.00×10^{2} (−)	2.00×10^{3} ± 4.58×10^{2} (=)	4.37×10^{3} ± 2.86×10^{2} (+)	2.65×10^{3} ± 4.19×10^{2} (+)	2.38×10^{3} ± 6.63×10^{2} (+)	1.89×10^{3} ± 2.25×10^{2}
F11	1.44×10^{1} ± 1.77×10^{1} (=)	6.49×10^{1} ± 2.02×10^{1} (+)	2.07×10^{1} ± 2.47×10^{1} (=)	2.32×10^{1} ± 2.22×10^{1} (+)	3.05×10^{1} ± 2.60×10^{1} (+)	1.98×10^{1} ± 1.02×10^{1} (+)	2.45×10^{1} ± 2.47×10^{1} (+)	**8.55**$\times10^{0}$ ± 3.20×10^{0}
F12	1.03×10^{4} ± 1.16×10^{4} (+)	2.68×10^{4} ± 1.45×10^{4} (+)	1.10×10^{3} ± 3.83×10^{2} (−)	9.51×10^{3} ± 1.13×10^{4} (+)	1.07×10^{4} ± 6.44×10^{3} (+)	**9.33×10^{2} ± 4.19×10^{2} (−)**	1.05×10^{4} ± 1.77×10^{4} (+)	2.97×10^{3} ± 3.45×10^{3}
F13	2.14×10^{1} ± 7.15×10^{0} (+)	1.13×10^{2} ± 1.08×10^{1} (+)	2.00×10^{1} ± 7.30×10^{0} (=)	2.89×10^{1} ± 1.05×10^{1} (+)	5.99×10^{1} ± 1.87×10^{1} (+)	2.08×10^{1} ± 7.94×10^{0} (=)	5.35×10^{3} ± 6.05×10^{3} (+)	**1.82**$\times10^{1}$ ± 7.13×10^{0}
F14	2.41×10^{1} ± 7.43×10^{0} (+)	5.09×10^{1} ± 4.62×10^{0} (+)	2.56×10^{1} ± 6.90×10^{0} (+)	1.31×10^{1} ± 8.59×10^{0} (+)	4.03×10^{1} ± 6.12×10^{0} (+)	1.31×10^{1} ± 1.00×10^{1} (+)	3.31×10^{1} ± 7.01×10^{0} (+)	**8.90**$\times10^{0}$ ± 6.54×10^{0}

(to be continued)

Table 6 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (30*D*). (continued)

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F15	$5.47\times10^{0} \pm 2.12\times10^{0}$ (=)	$3.88\times10^{1} \pm 5.72\times10^{0}$ (+)	$6.57\times10^{0} \pm 3.61\times10^{0}$ (=)	$9.00\times10^{0} \pm 3.67\times10^{0}$ (+)	$1.59\times10^{1} \pm 7.11\times10^{0}$ (+)	$8.43\times10^{0} \pm 3.94\times10^{0}$ (+)	$3.20\times10^{1} \pm 3.79\times10^{1}$ (+)	$\mathbf{5.42}\times10^{0} \pm 2.00\times10^{0}$
F16	**$5.40\times10^{1} \pm 7.32\times10^{1}$ (−)**	$5.53\times10^{2} \pm 1.64\times10^{2}$ (+)	$4.66\times10^{2} \pm 2.32\times10^{2}$ (+)	$4.09\times10^{2} \pm 1.83\times10^{2}$ (+)	$5.59\times10^{2} \pm 1.50\times10^{2}$ (+)	$3.46\times10^{2} \pm 1.82\times10^{2}$ (=)	$2.41\times10^{2} \pm 2.06\times10^{2}$ (−)	$3.09\times10^{2} \pm 1.57\times10^{2}$
F17	$3.22\times10^{1} \pm 6.70\times10^{0}$ (=)	$1.13\times10^{2} \pm 3.65\times10^{1}$ (+)	$4.53\times10^{1} \pm 2.36\times10^{1}$ (+)	$7.15\times10^{1} \pm 6.46\times10^{1}$ (+)	$5.82\times10^{1} \pm 1.93\times10^{1}$ (+)	$5.50\times10^{1} \pm 2.39\times10^{1}$ (+)	$5.89\times10^{1} \pm 4.09\times10^{1}$ (+)	$\mathbf{3.19}\times10^{1} \pm 1.23\times10^{1}$
F18	$2.58\times10^{1} \pm 1.97\times10^{0}$ (+)	$4.88\times10^{1} \pm 6.74\times10^{0}$ (+)	$2.40\times10^{1} \pm 5.03\times10^{0}$ (+)	$6.91\times10^{1} \pm 4.24\times10^{1}$ (+)	$5.30\times10^{1} \pm 4.33\times10^{1}$ (+)	$2.38\times10^{1} \pm 7.67\times10^{0}$ (+)	$3.30\times10^{3} \pm 4.56\times10^{3}$ (+)	$\mathbf{1.91}\times10^{1} \pm 6.57\times10^{0}$
F19	**$5.00\times10^{0} \pm 1.57\times10^{0}$ (−)**	$2.49\times10^{1} \pm 2.80\times10^{0}$ (+)	$5.21\times10^{0} \pm 1.59\times10^{0}$ (−)	$5.62\times10^{0} \pm 2.37\times10^{0}$ (−)	$1.59\times10^{1} \pm 3.01\times10^{0}$ (+)	$6.66\times10^{0} \pm 1.94\times10^{0}$ (−)	$1.38\times10^{1} \pm 8.34\times10^{0}$ (+)	$7.58\times10^{0} \pm 2.14\times10^{0}$
F20	**$2.14\times10^{1} \pm 7.46\times10^{0}$ (−)**	$9.26\times10^{1} \pm 4.52\times10^{1}$ (+)	$8.19\times10^{1} \pm 4.90\times10^{1}$ (+)	$9.83\times10^{1} \pm 8.32\times10^{1}$ (+)	$4.19\times10^{1} \pm 3.83\times10^{1}$ (+)	$6.99\times10^{1} \pm 5.13\times10^{1}$ (+)	$1.11\times10^{2} \pm 6.51\times10^{1}$ (+)	$2.23\times10^{1} \pm 4.00\times10^{1}$
F21	$2.31\times10^{2} \pm 1.48\times10^{1}$ (=)	$3.07\times10^{2} \pm 7.31\times10^{0}$ (+)	$2.28\times10^{2} \pm 8.30\times10^{0}$ (=)	$2.39\times10^{2} \pm 1.13\times10^{1}$ (+)	$2.91\times10^{2} \pm 8.87\times10^{0}$ (+)	$2.28\times10^{2} \pm 7.28\times10^{0}$ (=)	$2.28\times10^{2} \pm 8.34\times10^{0}$ (=)	$\mathbf{2.25}\times10^{2} \pm 5.21\times10^{0}$
F22	$1.00\times10^{2} \pm 1.58\times10^{-13}$ (+)	$1.00\times10^{2} \pm 4.04\times10^{-4}$ (+)	**$1.00\times10^{2} \pm 1.11\times10^{-13}$ (−)**	$1.00\times10^{2} \pm 1.00\times10^{-13}$ (=)	$1.00\times10^{2} \pm 1.00\times10^{-13}$ (=)	$1.00\times10^{2} \pm 1.00\times10^{-13}$ (=)	$2.61\times10^{2} \pm 6.61\times10^{2}$ (=)	$1.00\times10^{2} \pm 1.22\times10^{-13}$
F23	$3.73\times10^{2} \pm 9.13\times10^{0}$ (+)	$4.46\times10^{2} \pm 7.71\times10^{0}$ (+)	$3.71\times10^{2} \pm 9.86\times10^{0}$ (=)	$3.88\times10^{2} \pm 1.15\times10^{1}$ (+)	$4.35\times10^{2} \pm 8.39\times10^{0}$ (+)	$3.79\times10^{2} \pm 1.22\times10^{1}$ (+)	$3.73\times10^{2} \pm 6.59\times10^{0}$ (+)	$\mathbf{3.69}\times10^{2} \pm 5.63\times10^{0}$
F24	$4.46\times10^{2} \pm 7.36\times10^{0}$ (+)	$5.29\times10^{2} \pm 1.05\times10^{1}$ (+)	$4.53\times10^{2} \pm 1.12\times10^{1}$ (+)	$4.59\times10^{2} \pm 1.32\times10^{1}$ (+)	$5.08\times10^{2} \pm 1.03\times10^{1}$ (+)	$4.44\times10^{2} \pm 7.44\times10^{0}$ (+)	$4.46\times10^{2} \pm 8.28\times10^{0}$ (+)	$\mathbf{4.41}\times10^{2} \pm 5.84\times10^{0}$
F25	$3.87\times10^{2} \pm 3.03\times10^{-2}$ (−)	**$3.87\times10^{2} \pm 2.91\times10^{-2}$ (=)**	$3.87\times10^{2} \pm 6.15\times10^{-2}$ (−)	$3.87\times10^{2} \pm 1.23\times10^{-1}$ (−)	$3.87\times10^{2} \pm 9.07\times10^{-2}$ (−)	$3.87\times10^{2} \pm 6.40\times10^{-2}$ (−)	$3.87\times10^{2} \pm 1.46\times10^{-1}$ (=)	$3.87\times10^{2} \pm 8.00\times10^{-1}$
F26	$1.21\times10^{3} \pm 1.11\times10^{2}$ (=)	$1.93\times10^{3} \pm 4.15\times10^{2}$ (+)	$1.13\times10^{3} \pm 1.27\times10^{2}$ (−)	$1.28\times10^{3} \pm 3.18\times10^{2}$ (+)	$1.72\times10^{3} \pm 9.78\times10^{1}$ (+)	$1.20\times10^{3} \pm 1.06\times10^{2}$ (=)	**$1.04\times10^{3} \pm 3.11\times10^{2}$ (−)**	$1.16\times10^{3} \pm 1.99\times10^{2}$
F27	**$4.93\times10^{2} \pm 9.97\times10^{0}$ (−)**	$5.01\times10^{2} \pm 9.76\times10^{0}$ (+)	$5.01\times10^{2} \pm 6.09\times10^{0}$ (+)	$4.99\times10^{2} \pm 9.46\times10^{0}$ (+)	$4.93\times10^{2} \pm 1.48\times10^{1}$ (=)	$5.01\times10^{2} \pm 6.16\times10^{0}$ (+)	$5.06\times10^{2} \pm 4.97\times10^{0}$ (+)	$4.96\times10^{2} \pm 9.72\times10^{0}$
F28	$3.25\times10^{2} \pm 4.68\times10^{1}$ (+)	$3.87\times10^{2} \pm 3.19\times10^{1}$ (+)	**$3.16\times10^{2} \pm 4.10\times10^{1}$ (−)**	$3.41\times10^{2} \pm 5.36\times10^{1}$ (=)	$3.20\times10^{2} \pm 4.34\times10^{1}$ (−)	$3.27\times10^{2} \pm 5.04\times10^{1}$ (+)	$3.44\times10^{2} \pm 5.43\times10^{1}$ (=)	$3.25\times10^{2} \pm 4.51\times10^{1}$
F29	$4.29\times10^{2} \pm 2.64\times10^{1}$ (=)	$5.64\times10^{2} \pm 5.38\times10^{1}$ (+)	$4.36\times10^{2} \pm 2.35\times10^{1}$ (=)	$4.58\times10^{2} \pm 7.10\times10^{1}$ (=)	$5.58\times10^{2} \pm 3.10\times10^{1}$ (+)	$4.56\times10^{2} \pm 2.30\times10^{1}$ (+)	$4.80\times10^{2} \pm 4.88\times10^{1}$ (+)	$\mathbf{4.27}\times10^{2} \pm 2.94\times10^{1}$
F30	$2.02\times10^{3} \pm 5.12\times10^{1}$ (+)	$2.18\times10^{3} \pm 1.31\times10^{2}$ (+)	$2.03\times10^{3} \pm 1.26\times10^{2}$ (=)	$2.08\times10^{3} \pm 8.16\times10^{1}$ (+)	$2.14\times10^{3} \pm 9.25\times10^{1}$ (+)	**$2.00\times10^{3} \pm$ 5.41e+01 (=)**	$3.28\times10^{3} \pm 1.00\times10^{3}$ (+)	$2.00\times10^{3} \pm 5.63\times10^{1}$
+/=/−	15/9/6	29/1/0	9/13/8	18/8/4	20/6/4	14/11/5	17/9/4	−

Table 7 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (50*D*).

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F1	$1.63 \times 10^{2} \pm 4.37 \times 10^{2}$(−)	$2.25 \times 10^{5} \pm 1.08 \times 10^{5}$(+)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(−)**	$3.65 \times 10^{2} \pm 1.04 \times 10^{3}$(−)	$1.70 \times 10^{-7} \pm 3.52 \times 10^{-7}$(−)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(−)**	$3.78 \times 10^{3} \pm 4.78 \times 10^{3}$(=)	$1.69 \times 10^{3} \pm 2.02 \times 10^{3}$
F2	$1.45 \times 10^{9} \pm 9.63 \times 10^{9}$(−)	$4.65 \times 10^{40} \pm 2.77 \times 10^{41}$(+)	$1.91 \times 10^{2} \pm 1.34 \times 10^{3}$(−)	**$7.94 \times 10^{1} \pm 5.72 \times 10^{1}$(−)**	$1.04 \times 10^{5} \pm 7.34 \times 10^{5}$(−)	$5.23 \times 10^{8} \pm 3.13 \times 10^{9}$(−)	$4.53 \times 10^{16} \pm 3.22 \times 10^{17}$(+)	$2.28 \times 10^{9} \pm 1.63 \times 10^{10}$
F3	$1.81 \times 10^{-3} \pm 4.31 \times 10^{-3}$(−)	$4.60 \times 10^{4} \pm 2.16 \times 10^{4}$(+)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(−)**	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(−)**	$3.42 \times 10^{-3} \pm 4.28 \times 10^{-3}$(−)	$9.49 \times 10^{-5} \pm 5.39 \times 10^{-4}$(−)	$7.17 \times 10^{0} \pm 7.32 \times 10^{0}$(−)	$1.01 \times 10^{2} \pm 9.58 \times 10^{1}$
F4	$8.79 \times 10^{1} \pm 4.86 \times 10^{1}$(=)	$1.23 \times 10^{2} \pm 2.99 \times 10^{1}$(+)	$4.85 \times 10^{1} \pm 4.34 \times 10^{1}$(−)	**$4.51 \times 10^{1} \pm 4.17 \times 10^{1}$(−)**	$5.16 \times 10^{1} \pm 3.63 \times 10^{1}$(−)	$4.57 \times 10^{1} \pm 4.72 \times 10^{1}$(−)	$6.79 \times 10^{1} \pm 4.58 \times 10^{1}$(−)	$9.50 \times 10^{1} \pm 4.31 \times 10^{1}$
F5	$8.34 \times 10^{1} \pm 5.71 \times 10^{1}$(=)	$2.65 \times 10^{2} \pm 1.58 \times 10^{1}$(+)	**$3.63 \times 10^{1} \pm 6.01 \times 10^{0}$(−)**	$8.18 \times 10^{1} \pm 2.06 \times 10^{1}$(+)	$2.15 \times 10^{2} \pm 1.38 \times 10^{1}$(+)	$5.77 \times 10^{1} \pm 1.17 \times 10^{1}$(−)	$4.50 \times 10^{1} \pm 1.21 \times 10^{1}$(−)	$6.14 \times 10^{1} \pm 8.00 \times 10^{0}$
F6	$1.56 \times 10^{-4} \pm 3.84 \times 10^{-4}$(+)	$5.88 \times 10^{-2} \pm 9.26 \times 10^{-3}$(+)	$1.08 \times 10^{-5} \pm 1.48 \times 10^{-5}$(+)	$1.20 \times 10^{-7} \pm 3.85 \times 10^{-7}$(+)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(=)**	$1.48 \times 10^{-3} \pm 3.79 \times 10^{-3}$(+)	$1.01 \times 10^{-7} \pm 2.91 \times 10^{-7}$(+)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$**
F7	$2.58 \times 10^{2} \pm 5.85 \times 10^{1}$(+)	$3.66 \times 10^{2} \pm 1.57 \times 10^{1}$(+)	**$8.11 \times 10^{1} \pm 6.68 \times 10^{0}$(−)**	$1.31 \times 10^{2} \pm 1.81 \times 10^{1}$(+)	$2.69 \times 10^{2} \pm 1.54 \times 10^{1}$(+)	$1.08 \times 10^{2} \pm 1.30 \times 10^{1}$(+)	$9.58 \times 10^{1} \pm 1.17 \times 10^{1}$(−)	$1.03 \times 10^{2} \pm 9.44 \times 10^{0}$
F8	$7.73 \times 10^{1} \pm 5.41 \times 10^{1}$(=)	$2.61 \times 10^{2} \pm 1.32 \times 10^{1}$(+)	**$3.33 \times 10^{1} \pm 6.27 \times 10^{0}$(−)**	$8.14 \times 10^{1} \pm 1.77 \times 10^{1}$(+)	$2.12 \times 10^{2} \pm 1.29 \times 10^{1}$(+)	$5.42 \times 10^{1} \pm 1.17 \times 10^{1}$(−)	$4.40 \times 10^{1} \pm 1.10 \times 10^{1}$(−)	$6.15 \times 10^{1} \pm 8.66 \times 10^{0}$
F9	$6.96 \times 10^{-1} \pm 1.15 \times 10^{0}$(+)	$9.28 \times 10^{2} \pm 2.21 \times 10^{2}$(+)	$6.39 \times 10^{-2} \pm 1.68 \times 10^{-1}$(=)	$4.70 \times 10^{0} \pm 5.81 \times 10^{0}$(+)	$3.37 \times 10^{-2} \pm 1.10 \times 10^{-1}$(−)	$9.81 \times 10^{-1} \pm 8.69 \times 10^{-1}$(+)	**$0.00 \times 10^{0} \pm 0.00 \times 10^{0}$(−)**	$3.61 \times 10^{-1} \pm 9.33 \times 10^{-1}$

(to be continued)

Table 7 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (50*D*). (continued)

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F10	$9.66 \times 10^{3} \pm 3.93 \times 10^{2}$(+)	$8.93 \times 10^{3} \pm 3.46 \times 10^{2}$(+)	$3.23 \times 10^{3} \pm 5.51 \times 10^{2}$(=)	$4.44 \times 10^{3} \pm 7.84 \times 10^{2}$(+)	$8.69 \times 10^{3} \pm 3.86 \times 10^{2}$(+)	$4.85 \times 10^{3} \pm 7.71 \times 10^{2}$(+)	$4.71 \times 10^{3} \pm 1.24 \times 10^{3}$(+)	**$3.12 \times 10^{3} \pm 5.40 \times 10^{2}$**
F11	$4.29 \times 10^{1} \pm 8.94 \times 10^{0}$(−)	$1.40 \times 10^{2} \pm 1.34 \times 10^{1}$(+)	$6.69 \times 10^{1} \pm 1.67 \times 10^{1}$(+)	$5.66 \times 10^{1} \pm 1.45 \times 10^{1}$(=)	$6.22 \times 10^{1} \pm 9.13 \times 10^{0}$(+)	$1.05 \times 10^{2} \pm 2.33 \times 10^{1}$(+)	**$4.11 \times 10^{1} \pm 7.12 \times 10^{0}$(−)**	$5.03 \times 10^{1} \pm 1.22 \times 10^{1}$
F12	$5.63 \times 10^{4} \pm 3.26 \times 10^{4}$(+)	$8.86 \times 10^{5} \pm 6.34 \times 10^{5}$(+)	**$5.32 \times 10^{3} \pm 3.71 \times 10^{3}$(−)**	$2.92 \times 10^{4} \pm 2.18 \times 10^{4}$(=)	$5.04 \times 10^{4} \pm 2.97 \times 10^{4}$(+)	$8.62 \times 10^{3} \pm 8.38 \times 10^{3}$(−)	$6.36 \times 10^{4} \pm 1.15 \times 10^{5}$(=)	$3.90 \times 10^{4} \pm 3.09 \times 10^{4}$
F13	**$8.16 \times 10^{1} \pm 4.78 \times 10^{1}$(−)**	$5.66 \times 10^{2} \pm 1.88 \times 10^{2}$(+)	$8.87 \times 10^{1} \pm 3.72 \times 10^{1}$(−)	$3.27 \times 10^{3} \pm 3.96 \times 10^{3}$(+)	$2.02 \times 10^{3} \pm 2.77 \times 10^{3}$(+)	$9.50 \times 10^{1} \pm 5.89 \times 10^{1}$(−)	$2.92 \times 10^{3} \pm 4.08 \times 10^{3}$(+)	$1.84 \times 10^{2} \pm 2.03 \times 10^{2}$
F14	**$3.75 \times 10^{1} \pm 5.77 \times 10^{0}$(=)**	$1.06 \times 10^{2} \pm 9.57 \times 10^{0}$(+)	$5.60 \times 10^{1} \pm 1.33 \times 10^{1}$(+)	$6.84 \times 10^{1} \pm 4.18 \times 10^{1}$(+)	$8.33 \times 10^{1} \pm 1.04 \times 10^{1}$(+)	$6.10 \times 10^{1} \pm 1.36 \times 10^{1}$(+)	$9.10 \times 10^{2} \pm 3.68 \times 10^{3}$(+)	$3.78 \times 10^{1} \pm 9.64 \times 10^{0}$
F15	$3.55 \times 10^{1} \pm 8.71 \times 10^{0}$(=)	$1.26 \times 10^{2} \pm 1.21 \times 10^{1}$(+)	$9.40 \times 10^{1} \pm 4.10 \times 10^{1}$(+)	$1.99 \times 10^{2} \pm 3.77 \times 10^{2}$(+)	$7.24 \times 10^{1} \pm 2.93 \times 10^{1}$(+)	$7.51 \times 10^{1} \pm 5.19 \times 10^{1}$(+)	$1.79 \times 10^{3} \pm 1.62 \times 10^{3}$(+)	**$3.47 \times 10^{1} \pm 6.20 \times 10^{0}$**
F16	**$4.85 \times 10^{2} \pm 1.91 \times 10^{2}$(−)**	$1.24 \times 10^{3} \pm 2.44 \times 10^{2}$(+)	$6.17 \times 10^{2} \pm 1.89 \times 10^{2}$(−)	$1.07 \times 10^{3} \pm 3.50 \times 10^{2}$(+)	$1.30 \times 10^{3} \pm 1.82 \times 10^{2}$(+)	$9.17 \times 10^{2} \pm 3.28 \times 10^{2}$(+)	$6.79 \times 10^{2} \pm 2.64 \times 10^{2}$(=)	$7.38 \times 10^{2} \pm 1.87 \times 10^{2}$
F17	**$2.30 \times 10^{2} \pm 1.60 \times 10^{2}$(−)**	$9.29 \times 10^{2} \pm 1.52 \times 10^{2}$(+)	$4.47 \times 10^{2} \pm 2.09 \times 10^{2}$(=)	$6.70 \times 10^{2} \pm 1.92 \times 10^{2}$(+)	$7.81 \times 10^{2} \pm 1.89 \times 10^{2}$(+)	$5.69 \times 10^{2} \pm 2.01 \times 10^{2}$(+)	$4.66 \times 10^{2} \pm 2.25 \times 10^{2}$(=)	$4.30 \times 10^{2} \pm 1.39 \times 10^{2}$
F18	$1.41 \times 10^{2} \pm 1.11 \times 10^{2}$(+)	$3.10 \times 10^{3} \pm 3.57 \times 10^{3}$(+)	$1.60 \times 10^{2} \pm 7.96 \times 10^{1}$(+)	$3.33 \times 10^{3} \pm 3.27 \times 10^{3}$(+)	$2.00 \times 10^{3} \pm 1.67 \times 10^{3}$(+)	$1.29 \times 10^{2} \pm 1.08 \times 10^{2}$(=)	$3.14 \times 10^{4} \pm 8.94 \times 10^{4}$(+)	**$1.06 \times 10^{2} \pm 1.07 \times 10^{2}$**
F19	$1.51 \times 10^{1} \pm 3.37 \times 10^{0}$(+)	$6.18 \times 10^{1} \pm 6.63 \times 10^{0}$(+)	$5.83 \times 10^{1} \pm 1.99 \times 10^{1}$(+)	$2.53 \times 10^{1} \pm 1.41 \times 10^{1}$(+)	$3.45 \times 10^{1} \pm 9.29 \times 10^{0}$(+)	$3.86 \times 10^{1} \pm 1.65 \times 10^{1}$(+)	$1.06 \times 10^{4} \pm 6.33 \times 10^{3}$(+)	**$1.31 \times 10^{1} \pm 3.45 \times 10^{0}$**
F20	**$1.49 \times 10^{2} \pm 1.37 \times 10^{2}$(−)**	$6.06 \times 10^{2} \pm 1.61 \times 10^{2}$(+)	$4.85 \times 10^{2} \pm 1.51 \times 10^{2}$(+)	$5.17 \times 10^{2} \pm 2.02 \times 10^{2}$(+)	$5.90 \times 10^{2} \pm 1.97 \times 10^{2}$(+)	$4.10 \times 10^{2} \pm 1.80 \times 10^{2}$(+)	$2.89 \times 10^{2} \pm 1.81 \times 10^{2}$(=)	$3.24 \times 10^{2} \pm 1.47 \times 10^{2}$
F21	$2.67 \times 10^{2} \pm 3.75 \times 10^{1}$(=)	$4.71 \times 10^{2} \pm 1.19 \times 10^{1}$(+)	**$2.36 \times 10^{2} \pm 7.04 \times 10^{0}$(−)**	$2.80 \times 10^{2} \pm 1.46 \times 10^{1}$(+)	$4.19 \times 10^{2} \pm 1.42 \times 10^{1}$(+)	$2.54 \times 10^{2} \pm 1.20 \times 10^{1}$(−)	$2.48 \times 10^{2} \pm 1.07 \times 10^{1}$(−)	$2.59 \times 10^{2} \pm 7.71 \times 10^{0}$
F22	$8.39 \times 10^{3} \pm 3.63 \times 10^{3}$(+)	$8.60 \times 10^{3} \pm 2.85 \times 10^{3}$(+)	$3.07 \times 10^{3} \pm 1.74 \times 10^{3}$(=)	$4.68 \times 10^{3} \pm 1.42 \times 10^{3}$(+)	$6.59 \times 10^{3} \pm 4.24 \times 10^{3}$(+)	$3.08 \times 10^{3} \pm 2.68 \times 10^{3}$(+)	$4.56 \times 10^{3} \pm 1.79 \times 10^{3}$(+)	**$2.63 \times 10^{3} \pm 1.93 \times 10^{3}$**
F23	$4.72 \times 10^{2} \pm 1.44 \times 10^{1}$(−)	$6.85 \times 10^{2} \pm 1.58 \times 10^{1}$(+)	**$4.53 \times 10^{2} \pm 6.95 \times 10^{0}$(−)**	$5.13 \times 10^{2} \pm 2.53 \times 10^{1}$(+)	$6.42 \times 10^{2} \pm 1.53 \times 10^{1}$(+)	$4.82 \times 10^{2} \pm 1.78 \times 10^{1}$(=)	$4.74 \times 10^{2} \pm 1.38 \times 10^{1}$(−)	$4.84 \times 10^{2} \pm 1.16 \times 10^{1}$
F24	$5.50 \times 10^{2} \pm 1.12 \times 10^{1}$(=)	$7.67 \times 10^{2} \pm 1.88 \times 10^{1}$(+)	**$5.29 \times 10^{2} \pm 1.20 \times 10^{1}$(−)**	$5.80 \times 10^{2} \pm 1.69 \times 10^{1}$(+)	$7.01 \times 10^{2} \pm 1.80 \times 10^{1}$(+)	$5.41 \times 10^{2} \pm 1.31 \times 10^{1}$(−)	$5.51 \times 10^{2} \pm 1.23 \times 10^{1}$(=)	$5.51 \times 10^{2} \pm 1.08 \times 10^{1}$
F25	$4.95 \times 10^{2} \pm 2.92 \times 10^{1}$(−)	**$4.85 \times 10^{2} \pm 4.87 \times 10^{0}$(−)**	$5.15 \times 10^{2} \pm 3.10 \times 10^{1}$(−)	$5.25 \times 10^{2} \pm 3.62 \times 10^{1}$(−)	$5.23 \times 10^{2} \pm 3.14 \times 10^{1}$(−)	$5.19 \times 10^{2} \pm 3.44 \times 10^{1}$(−)	$4.88 \times 10^{2} \pm 2.48 \times 10^{1}$(−)	$5.58 \times 10^{2} \pm 3.48 \times 10^{1}$
F26	$1.56 \times 10^{3} \pm 1.28 \times 10^{2}$(−)	$3.74 \times 10^{3} \pm 1.61 \times 10^{2}$(+)	**$1.36 \times 10^{3} \pm 7.97 \times 10^{1}$(−)**	$1.95 \times 10^{3} \pm 2.09 \times 10^{2}$(+)	$3.08 \times 10^{3} \pm 1.40 \times 10^{2}$(+)	$1.60 \times 10^{3} \pm 1.33 \times 10^{2}$(−)	$1.50 \times 10^{3} \pm 1.75 \times 10^{2}$(−)	$1.75 \times 10^{3} \pm 1.35 \times 10^{2}$
F27	$5.15 \times 10^{2} \pm 1.31 \times 10^{1}$(=)	$5.20 \times 10^{2} \pm 1.57 \times 10^{1}$(+)	$5.31 \times 10^{2} \pm 1.58 \times 10^{1}$(+)	$5.40 \times 10^{2} \pm 2.78 \times 10^{1}$(+)	$5.15 \times 10^{2} \pm 1.05 \times 10^{1}$(=)	$5.49 \times 10^{2} \pm 2.40 \times 10^{1}$(+)	$5.21 \times 10^{2} \pm 7.93 \times 10^{0}$(+)	**$5.10 \times 10^{2} \pm 1.38 \times 10^{1}$**
F28	$4.67 \times 10^{2} \pm 1.79 \times 10^{1}$(−)	**$4.59 \times 10^{2} \pm 4.36 \times 10^{-1}$(−)**	$4.86 \times 10^{2} \pm 2.41 \times 10^{1}$(=)	$4.82 \times 10^{2} \pm 2.33 \times 10^{1}$(−)	$4.74 \times 10^{2} \pm 2.20 \times 10^{1}$(−)	$4.89 \times 10^{2} \pm 2.59 \times 10^{1}$(=)	$4.66 \times 10^{2} \pm 1.70 \times 10^{1}$(−)	$4.95 \times 10^{2} \pm 1.52 \times 10^{1}$
F29	**$3.49 \times 10^{2} \pm 3.24 \times 10^{1}$(=)**	$8.27 \times 10^{2} \pm 1.23 \times 10^{2}$(+)	$4.14 \times 10^{2} \pm 3.92 \times 10^{1}$(+)	$5.60 \times 10^{2} \pm 1.65 \times 10^{2}$(+)	$7.10 \times 10^{2} \pm 8.71 \times 10^{1}$(+)	$4.52 \times 10^{2} \pm 1.36 \times 10^{2}$(+)	$4.09 \times 10^{2} \pm 6.14 \times 10^{1}$(+)	**$3.76 \times 10^{2} \pm 7.24 \times 10^{1}$**
F30	$5.98 \times 10^{5} \pm 2.50 \times 10^{4}$(+)	**$5.86 \times 10^{5} \pm 1.60 \times 10^{4}$(−)**	$6.67 \times 10^{5} \pm 9.17 \times 10^{4}$(+)	$6.01 \times 10^{5} \pm 2.73 \times 10^{4}$(+)	$6.08 \times 10^{5} \pm 3.77 \times 10^{4}$(+)	$6.88 \times 10^{5} \pm 1.36 \times 10^{5}$(+)	$6.20 \times 10^{5} \pm 3.04 \times 10^{-4}$(+)	$5.93 \times 10^{5} \pm 2.20 \times 10^{4}$
+/=/−	9/9/12	27/0/3	10/5/15	22/2/6	21/2/7	15/3/12	12/6/12	−

Table 8 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (100*D*).

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F1	$6.07\times10^{3} \pm 6.70\times10^{3}$(=)	$5.10\times10^{6} \pm 1.97\times10^{6}$(+)	**$0.00\times10^{0} \pm 0.00\times10^{0}$(−)**	$5.09\times10^{3} \pm 5.66\times10^{3}$(=)	$1.56\times10^{1} \pm 3.27\times10^{1}$(−)	**$0.00\times10^{0} \pm 0.00\times10^{0}$(−)**	$3.05\times10^{-3} \pm 6.19\times10^{-3}$(−)	$4.80\times10^{3} \pm 4.58\times10^{3}$
F2	$1.17\times10^{45} \pm 8.33\times10^{45}$(+)	$1.19\times10^{104} \pm 3.49\times10^{104}$(+)	$1.69\times10^{34} \pm 1.21\mathrm{e}\times10^{35}$(+)	$1.90\times10^{35} \pm 1.36\times10^{35}$(+)	**$8.59\times10^{25} \pm 6.13\times10^{26}$(−)**	$1.60\times10^{46} \pm 1.14\times10^{47}$(+)	$6.84\times10^{48} \pm 2.83\times10^{49}$(+)	$5.94\times10^{33} \pm 4.24\times10^{34}$
F3	$1.38\times10^{3} \pm 7.99\times10^{2}$(−)	$3.14\times10^{5} \pm 4.83\times10^{4}$(+)	$1.07\times10^{4} \pm 5.33\times10^{4}$(−)	**$2.07\times10^{-1} \pm 8.18\times10^{-1}$(−)**	$2.28\times10^{2} \pm 3.55\times10^{2}$(−)	$1.75\times10^{1} \pm 4.14\times10^{1}$(−)	$9.07\times10^{4} \pm 3.08\times10^{4}$(+)	$1.83\times10^{4} \pm 8.24\times10^{3}$
F4	$2.14\times10^{2} \pm 2.01\times10^{1}$(=)	$2.83\times10^{2} \pm 2.59\times10^{1}$(+)	**$6.66\times10^{1} \pm 6.20\times10^{1}$(−)**	$1.52\times10^{2} \pm 4.78\times10^{1}$(−)	$1.77\times10^{2} \pm 4.86\times10^{1}$(−)	$6.69\times10^{1} \pm 6.46\times10^{1}$(−)	$2.09\times10^{2} \pm 9.28\times10^{0}$(=)	$2.03\times10^{2} \pm 4.78\times10^{1}$

(to be continued)

Table 8 Results (average±std) of LBLDE with peer DE algorithms on the CEC'2017 test suite (100*D*). (continued)

Problem	EAGDE	EFADE	AMECoDEs	TSDE	RNDE	MPEDE	TVDE	LBLDE
F5	$1.18\times10^{2}\pm2.15\times10^{1}(-)$	$8.17\times10^{2}\pm2.92\times10^{1}(+)$	$1.40\times10^{2}\pm1.41\times10^{1}(-)$	$2.19\times10^{2}\pm2.94\times10^{1}(+)$	$6.19\times10^{2}\pm2.51\times10^{1}(+)$	$1.47\times10^{2}\pm2.50\times10^{1}(-)$	**$9.58\times10^{1}\pm2.20\times10^{1}(-)$**	$1.85\times10^{2}\pm2.23\times10^{1}$
F6	$1.10\times10^{-3}\pm1.01\times10^{-3}(+)$	$5.76\times10^{-1}\pm5.55\times10^{-2}(+)$	$1.52\times10^{-2}\pm1.54\times10^{-2}(+)$	$3.01\times10^{-3}\pm7.65\times10^{-3}(+)$	$1.11\times10^{-5}\pm7.94\times10^{-5}(+)$	$1.26\times10^{-1}\pm1.05\times10^{-1}(+)$	$4.56\times10^{-6}\pm3.79\times10^{-6}(+)$	**0.00**$\times10^{0}\pm0.00\times10^{0}$
F7	$5.11\times10^{2}\pm2.44\times10^{2}(=)$	$1.10\times10^{3}\pm3.10\times10^{1}(+)$	$2.40\times10^{2}\pm1.95\times10^{1}(-)$	$3.61\times10^{2}\pm3.97\times10^{1}(+)$	$7.21\times10^{2}\pm2.21\times10^{1}(+)$	$3.01\times10^{2}\pm3.51\times10^{1}(=)$	**$1.97\times10^{2}\pm2.42\times10^{1}(-)$**	$3.10\times10^{2}\pm2.96\times10^{1}$
F8	$1.44\times10^{2}\pm1.16\times10^{2}(-)$	$8.10\times10^{2}\pm2.34\times10^{1}(+)$	$1.38\times10^{2}\pm1.33\times10^{1}(-)$	$2.10\times10^{2}\pm3.52\times10^{1}(+)$	$6.07\times10^{2}\pm2.50\times10^{1}(+)$	$1.51\times10^{2}\pm2.48\times10^{1}(-)$	**$9.95\times10^{1}\pm2.42\times10^{1}(-)$**	$1.86\times10^{2}\pm1.83\times10^{1}$
F9	$4.46\times10^{0}\pm4.48\times10^{0}(=)$	$8.33\times10^{3}\pm1.96\times10^{3}(+)$	$1.18\times10^{1}\pm5.47\times10^{0}(-)$	$7.24\times10^{2}\pm5.02\times10^{2}(+)$	$5.17\times10^{0}\pm7.61\times10^{0}(=)$	$3.28\times10^{1}\pm1.88\times10^{1}(-)$	**$1.77\times10^{-2}\pm6.79\times10^{-2}(-)$**	$8.04\times10^{1}\pm1.65\times10^{2}$
F10	$2.55\times10^{4}\pm6.13\times10^{2}(+)$	$2.44\times10^{4}\pm4.17\times10^{2}(+)$	**$9.90\times10^{3}\pm1.19\times10^{3}(-)$**	$1.20\times10^{4}\pm1.39\times10^{3}(=)$	$2.24\times10^{4}\pm4.90\times10^{2}(+)$	$1.10\times10^{4}\pm1.21\times10^{3}(-)$	$1.42\times10^{4}\pm3.08\times10^{3}(+)$	$1.16\times10^{4}\pm7.39\times10^{2}$
F11	**$1.29\times10^{2}\pm4.69\times10^{1}(-)$**	$8.48\times10^{2}\pm7.46\times10^{1}(+)$	$9.29\times10^{2}\pm1.98\times10^{2}(+)$	$2.02\times10^{2}\pm5.90\times10^{1}(=)$	$5.04\times10^{2}\pm1.24\times10^{2}(+)$	$7.75\times10^{2}\pm2.48\times10^{2}(+)$	$5.14\times10^{2}\pm1.15\times10^{2}(+)$	$2.00\times10^{2}\pm5.37\times10^{1}$
F12	$2.59\times10^{5}\pm9.76\times10^{4}(=)$	$9.20\times10^{6}\pm7.85\times10^{6}(+)$	**$2.30\times10^{4}\pm8.74\times10^{3}(-)$**	$2.26\times10^{5}\pm7.85\times10^{4}(-)$	$1.88\times10^{5}\pm6.38\times10^{4}(-)$	$3.86\times10^{4}\pm2.32\times10^{4}(-)$	$2.48\times10^{5}\pm1.16\times10^{5}(-)$	$3.03\times10^{5}\pm1.32\times10^{5}$
F13	$3.09\times10^{3}\pm2.84\times10^{3}(=)$	$5.23\times10^{3}\pm4.92\times10^{3}(+)$	$1.16\times10^{3}\pm1.14\times10^{3}(-)$	$2.34\times10^{3}\pm3.19\times10^{3}(=)$	$2.47\times10^{3}\pm2.26\times10^{3}(=)$	**$7.14\times10^{2}\pm8.94\times10^{2}(-)$**	$1.94\times10^{3}\pm1.75\times10^{3}(=)$	$2.30\times10^{3}\pm1.98\times10^{3}$
F14	**$9.24\times10^{1}\pm1.87\times10^{1}(-)$**	$2.43\times10^{4}\pm1.91\times10^{4}(+)$	$5.60\times10^{2}\pm1.21\times10^{2}(+)$	$6.75\times10^{3}\pm3.96\times10^{3}(+)$	$1.37\times10^{4}\pm9.88\times10^{3}(+)$	$4.77\times10^{2}\pm1.06\times10^{2}(+)$	$6.65\times10^{4}\pm1.07\times10^{5}(+)$	$1.39\times10^{2}\pm2.58\times10^{1}$
F15	**$1.97\times10^{2}\pm9.98\times10^{1}(=)$**	$1.53\times10^{3}\pm1.19\times10^{3}(+)$	$3.39\times10^{2}\pm9.23\times10^{1}(+)$	$2.28\times10^{3}\pm3.28\times10^{3}(+)$	$8.18\times10^{2}\pm9.68\times10^{2}(+)$	$3.47\times10^{2}\pm1.66\times10^{2}(+)$	$1.19\times10^{3}\pm1.26\times10^{3}(+)$	$2.69\times10^{2}\pm2.14\times10^{2}$
F16	**$1.84\times10^{3}\pm5.99\times10^{2}(-)$**	$4.69\times10^{3}\pm3.26\times10^{2}(+)$	$2.53\times10^{3}\pm3.54\times10^{2}(+)$	$2.99\times10^{3}\pm6.67\times10^{2}(+)$	$4.43\times10^{3}\pm2.42\times10^{2}(+)$	$2.80\times10^{3}\pm6.34\times10^{2}(+)$	$2.86\times10^{3}\pm7.88\times10^{2}(+)$	$2.19\times10^{3}\pm4.33\times10^{2}$
F17	$1.62\times10^{3}\pm6.67\times10^{2}(=)$	$3.08\times10^{3}\pm2.46\times10^{2}(+)$	$1.59\times10^{3}\pm2.84\times10^{2}(=)$	$2.05\times10^{3}\pm4.24\times10^{2}(+)$	$2.74\times10^{3}\pm3.61\times10^{2}(+)$	$1.99\times10^{3}\pm4.73\times10^{2}(+)$	$2.02\times10^{3}\pm5.44\times10^{2}(+)$	**1.50**$\times10^{3}\pm3.33\times10^{2}$
F18	$4.99\times10^{3}\pm3.12\times10^{3}(=)$	$2.30\times10^{5}\pm1.58\times10^{5}(+)$	**$4.55\times10^{2}\pm4.05\times10^{2}(-)$**	$3.52\times10^{4}\pm1.71\times10^{4}(+)$	$5.87\times10^{4}\pm3.84\times10^{4}(+)$	$9.62\times10^{2}\pm8.80\times10^{2}(-)$	$1.78\times10^{5}\pm8.62\times10^{4}(+)$	$3.86\times10^{3}\pm2.39\times10^{3}$
F19	$2.48\times10^{2}\pm1.09\times10^{3}(-)$	$2.69\times10^{3}\pm5.09\times10^{3}(+)$	**$2.36\times10^{2}\pm5.56\times10^{1}(-)$**	$2.85\times10^{3}\pm4.01\times10^{3}(+)$	$1.34\times10^{3}\pm1.48\times10^{3}(+)$	$2.36\times10^{2}\pm5.85\times10^{1}(-)$	$1.51\times10^{3}\pm1.61\times10^{3}(+)$	$3.29\times10^{2}\pm5.54\times10^{2}$
F20	$2.11\times10^{3}\pm6.34\times10^{2}(+)$	$2.62\times10^{3}\pm2.39\times10^{2}(+)$	$2.14\times10^{3}\pm2.65\times10^{2}(+)$	$1.97\times10^{3}\pm4.79\times10^{2}(+)$	$2.73\times10^{3}\pm2.40\times10^{2}(+)$	$1.99\times10^{3}\pm3.73\times10^{2}(+)$	$2.14\times10^{3}\pm4.94\times10^{2}(+)$	**1.41**$\times10^{3}\pm3.09\times10^{2}$
F21	$3.62\times10^{2}\pm6.67\times10^{1}(-)$	$1.05\times10^{3}\pm2.08\times10^{1}(+)$	$3.57\times10^{2}\pm1.42\times10^{1}(-)$	$4.32\times10^{2}\pm2.93\times10^{1}(+)$	$8.34\times10^{2}\pm2.47\times10^{1}(+)$	$3.62\times10^{2}\pm1.71\times10^{1}(-)$	**$3.33\times10^{2}\pm2.37\times10^{1}(-)$**	$3.88\times10^{2}\pm2.05\times10^{1}$
F22	$2.61\times10^{4}\pm6.49\times10^{2}(+)$	$2.53\times10^{4}\pm5.78\times10^{2}(+)$	**$1.10\times10^{4}\pm1.24\times10^{3}(-)$**	$1.30\times10^{4}\pm1.41\times10^{3}(=)$	$2.32\times10^{4}\pm3.34\times10^{3}(+)$	$1.21\times10^{4}\pm1.20\times10^{3}(-)$	$1.50\times10^{4}\pm2.27\times10^{3}(+)$	$1.28\times10^{4}\pm9.00\times10^{2}$
F23	$8.19\times10^{2}\pm1.82\times10^{2}(+)$	$1.10\times10^{3}\pm1.82\times10^{1}(+)$	$6.52\times10^{2}\pm1.12\times10^{1}(+)$	$7.04\times10^{2}\pm2.95\times10^{1}(+)$	$9.95\times10^{2}\pm1.77\times10^{1}(+)$	$6.93\times10^{2}\pm3.24\times10^{1}(+)$	**$6.18\times10^{2}\pm1.53\times10^{1}(-)$**	$6.28\times10^{2}\pm1.57\times10^{1}$
F24	$9.82\times10^{2}\pm2.40\times10^{1}(-)$	$1.60\times10^{3}\pm2.27\times10^{1}(+)$	$9.88\times10^{2}\pm1.28\times10^{1}(-)$	$1.08\times10^{3}\pm3.59\times10^{1}(+)$	$1.43\times10^{3}\pm2.60\times10^{1}(+)$	$1.03\times10^{3}\pm2.73\times10^{1}(=)$	**$9.58\times10^{2}\pm2.02\times10^{1}(-)$**	$1.03\times10^{3}\pm2.29\times10^{1}$
F25	**$7.20\times10^{2}\pm4.61\times10^{1}(-)$**	$9.49\times10^{2}\pm1.05\times10^{2}(+)$	$7.39\times10^{2}\pm4.62\times10^{1}(-)$	$7.64\times10^{2}\pm5.95\times10^{1}(-)$	$7.58\times10^{2}\pm4.86\times10^{1}(-)$	$7.42\times10^{2}\pm4.68\times10^{1}(-)$	$7.44\times10^{2}\pm4.70\times10^{1}(-)$	$8.19\times10^{2}\pm4.06\times10^{1}$
F26	$4.15\times10^{3}\pm2.34\times10^{2}(-)$	$1.09\times10^{4}\pm2.44\times10^{2}(+)$	$4.11\times10^{3}\pm1.63\times10^{2}(-)$	$5.37\times10^{3}\pm3.99\times10^{2}(+)$	$8.72\times10^{3}\pm2.99\times10^{2}(+)$	$4.48\times10^{3}\pm2.97\times10^{2}(-)$	**$3.82\times10^{3}\pm2.10\times10^{2}(-)$**	$4.93\times10^{3}\pm2.66\times10^{2}$
F27	$5.95\times10^{2}\pm1.75\times10^{1}(+)$	$6.45\times10^{2}\pm2.80\times10^{1}(+)$	$6.63\times10^{2}\pm2.72\times10^{1}(+)$	$6.97\times10^{2}\pm3.23\times10^{1}(+)$	$6.45\times10^{2}\pm2.25\times10^{1}(+)$	$7.03\times10^{2}\pm2.49\times10^{1}(+)$	$6.23\times10^{2}\pm1.57\times10^{1}(+)$	**5.84**$\times10^{2}\pm2.08\times10^{1}$
F28	$5.41\times10^{2}\pm3.10\times10^{1}(-)$	$6.42\times10^{2}\pm4.55\times10^{1}(+)$	**$5.28\times10^{2}\pm4.14\times10^{1}(-)$**	$5.55\times10^{2}\pm2.95\times10^{1}(-)$	$5.66\times10^{2}\pm2.47\times10^{1}(=)$	$5.33\times10^{2}\pm3.60\times10^{1}(-)$	$5.35\times10^{2}\pm2.30\times10^{1}(-)$	$5.68\times10^{2}\pm2.43\times10^{1}$
F29	**$1.50\times10^{3}\pm3.38\times10^{2}(-)$**	$3.43\times10^{3}\pm2.68\times10^{2}(+)$	$2.00\times10^{3}\pm3.07\times10^{2}(+)$	$2.51\times10^{3}\pm4.90\times10^{2}(+)$	$3.14\times10^{3}\pm2.41\times10^{2}(+)$	$2.48\times10^{3}\pm5.35\times10^{2}(+)$	$1.78\times10^{3}\pm4.85\times10^{2}(=)$	$1.74\times10^{3}\pm2.71\times10^{2}$
F30	$2.86\times10^{3}\pm7.50\times10^{2}(+)$	$5.12\times10^{3}\pm1.22\times10^{3}(+)$	$3.13\times10^{3}\pm1.30\times10^{3}(+)$	$2.87\times10^{3}\pm3.36\times10^{2}(+)$	$4.05\times10^{3}\pm1.39\times10^{3}(+)$	**$2.65\times10^{3}\pm2.5910^{2}(-)$**	$3.67\times10^{3}\pm1.30\times10^{3}(+)$	$2.71\times10^{3}\pm9.72\times10^{2}$
+/=/−	8/9/13	30/0/0	11/1/18	20/5/5	21/3/6	11/2/17	15/3/12	−

(1) Unimodal functions (F1–F3): Except for EFADE, MPEDE, and EAGDE, LBLDE and the remaining four algorithms obtain the best results on 10*D* functions. When D = 30, 50, and 100, LBLDE

Table 9 Statistical compared results of all compared algorithms.

Compared algorithms	Indicator	10*D*	30*D*	50*D*	100*D*
	+	**18**	**15**	**4**	**8**
LBLDE vs. EAGDE	=	10	9	0	9
	−	2	6	26	13
	+	**23**	**29**	**27**	**30**
LBLDE vs. EFADE	=	2	1	0	0
	−	5	0	3	0
	+	**18**	**9**	**10**	**11**
LBLDE vs. AMECoDEs	=	7	13	5	1
	−	5	8	15	18
	+	**11**	**18**	**22**	**20**
LBLDE vs. TSDE	=	13	8	2	5
	−	6	4	6	5
	+	**10**	**20**	**21**	**21**
LBLDE vs. RNDE	=	15	6	2	3
	−	5	4	7	6
	+	**18**	**14**	**15**	**11**
LBLDE vs. MPEDE	=	7	11	3	2
	−	5	5	12	17
	+	**25**	**28**	**12**	**15**
LBLDE vs. TVDE	=	5	1	6	3
	−	0	1	12	12

outperforms EFADE and TVDE and is worse than other algorithms because *NL* is a fixed value, and the number of individuals in the first level is too small. If *NL* is set to 2 as in Table 1, then LBLDE can also obtain better results.

(2) Simple multimodal functions (F4–F10): LBLDE is significantly better than the other seven algorithms on the 10*D* and 30*D* functions. When D = 50, LBLDE achieves better results than EAGDE, EFADE, TSDE, RNDE, and TVDE and has a competitive performance with MPEDE. LBLDE is worse than AMECoDEs on four functions. In the case of D = 100, LBLDE also outperforms EAGDE, EFADE, TSDE, and RNDE and is outperformed by AMECoDEs, MPEDE, and TVDE. Considering all cases, LBLDE achieves better performance than EFADE, TSDE, and RNDE on these functions, thus proving the superiority of LBLDE.

(3) Hybrid functions (F11–F20): When D = 10 and 30, LBLDE obtains the best results on most of the functions, and LBLDE is significantly superior to all the other compared algorithms. When D = 50, EAGDE and LBLDE perform best on five and three functions, respectively. LBLDE outperforms EFADE, AMECoDEs, TSDE, RNDE, MPEDE, and TVDE on 10, 6, 8, 10, 7, and 5 test functions, respectively. When D = 100, EAGDE, AMECoDES, and LBLDE obtain the minimum average error values on 4, 3, and 2 test functions, respectively. LBLDE performs better than, similar to, and worse than EAGDE on 1, 5, and 4 test functions, respectively. Compared with EFADE, TSDE, RNDE, TVDE, and MPEDE, LBLDE has superior performance.

(4) Composition functions (F21–F30): This group of functions is rather complex, and no algorithm is better than others on all dimensions. When D = 10 and 30, LBLDE is better than or similar to other algorithms. In the case of 50*D*, only EAGDE outperforms LBLDE, and AMECoDEs, MPEDE, and TVDE have similar performance as LBLDE. Moreover, when D = 100, LBLDE is inferior to AMECoDEs and MPEDE on six test functions, respectively. However, LBLDE is better than or at least equal to the other five algorithms.

According to Table 9, the statistical compared results show that LBLDE significantly outperforms EAGDE, EFADE, AMECoDEs, TSDE, RNDE, MPEDE, and TVDE on 18, 23, 18, 11, 10, 18, and 25 functions when D = 10, and on 15, 29, 9, 18, 20, 14, and 28 functions when D = 30, respectively. LBLDE is inferior to EAGDE, EFADE, AMECoDEs, TSDE, RNDE, MPEDE, and TVDE on 2, 5, 5, 6, 5, 5, and 0 functions when D = 10, and on 6, 0, 8, 4, 6, 5, and 1 functions when D = 30, respectively. LBLDE yields the best results on most of the simple multimodal functions, hybrid functions, and composition functions. When D = 50 and 100, the superiority of LBLDE is affected. This result occurred because the problems require higher diversity as the dimension increases. However, the difference vector selection strategy and parameter setting in LBLDE are more biased toward convergence. To be specific, in the case of 50*D*, EAGDE, AMECoDEs, and LBLDE obtain the best results on 6, 10, and 9 test functions, respectively. LBLDE is similar to TVDE and better than the other four algorithms. When D = 100, LBLDE is outperformed by EAGDE, AMECoDEs, and MPEDE. LBLDE still has superior performance than other algorithms. Considering all cases, LBLDE outperforms EFADE, TSDE, RNDE, MPEDE, and TVDE and shows comparable performance to EAGDE and AMECoDEs. Table 10 provides the rankings of all algorithms on each test dimension and the average rankings of all algorithms on four test dimensions. LBLDE ranks the second,

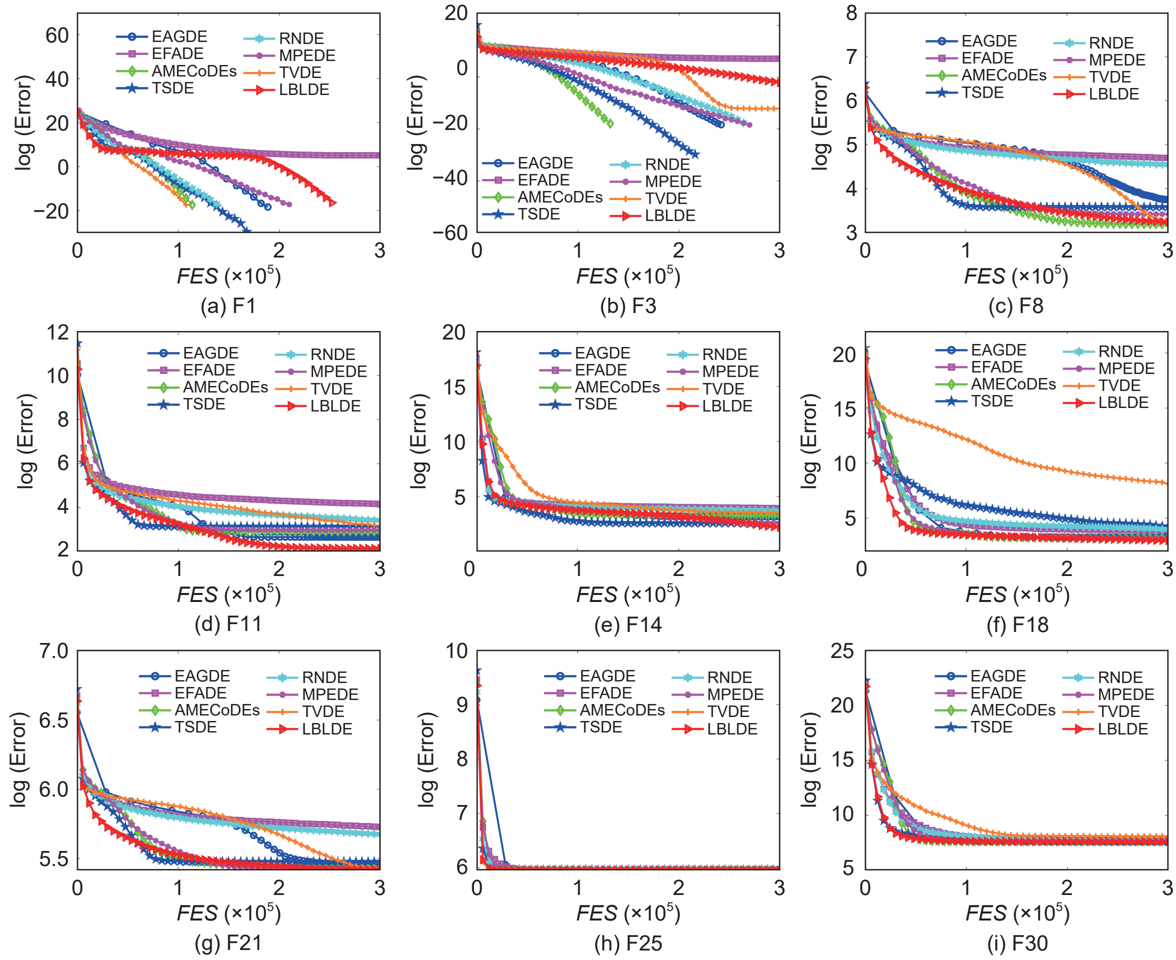

Fig. 5 Error curves of eight DE variants during the evolution on nine test functions.

Table 10 Rankings of all algorithms on each test dimension case and the average rankings of all algorithms on four test dimensions.

Algorithm	Ranking				Average ranking
	D=10	*D*=30	*D*=50	*D*=100	
EAGDE	5	4	2	4	3
EFADE	7	7	8	8	8
AMECoDES	4	2	1	1	1
TSDE	1	5	6	6	5
RNDE	3	6	7	7	6
MPEDE	6	3	5	2	4
TVDE	8	6	4	5	7
LBLDE	2	1	3	3	2

first, third, and third on 10*D*, 30*D*, 50*D*, and 100*D* functions, respectively. When the performance of all algorithms in four dimensions is considered, LBLDE achieves the second average ranking, while AMECoDEs obtains the first average ranking. The error iteration curves of LBLDE on some 30*D* test functions are plotted in Fig. 5. LBLDE has similar convergence trends with other algorithms. For F8 (Fig. 5c) and F14 (Fig. 5e) in particular, LBLDE can continue to evolve when other algorithms fall into stagnation.

4.3 Effectiveness of the proposed schemes

The effectiveness of proposed schemes, which are (1) the level-based learning mechanism, (2) the difference vector selection method based on the level, and (3) the CR allocation mechanism for different levels, are verified. LBLDE-1 indicates that LBLDE does not use the level-based learning mechanism, LBLDE-2 refers to LBLDE that does not use the difference vector selection method based on the level, and LBLDE-3 represents LBLDE that does not use the CR allocation scheme. The compared results of LBLDE and its three variants are provided in Table 11. LBLDE obtains the best results on most of the functions. In the last row of Table 11, the "+/ = /−" of 25/5/0, 15/13/2, and 25/5/0 demonstrate the superiority of LBLDE and the effectiveness of the three schemes. Next, a detailed analysis is provided.

(1) A new selection method of difference vectors corresponding to levels is proposed in Section 3.2. Unlike the traditional random selection method, the new method prevents good individuals from being influenced by poor individuals. To clarify the difference between them, the diversity (DP^G) and success rate (SR^G) in the G generation are calculated[35],

$$DP^G = \frac{1}{NP} \times \sqrt{\sum_{i=1}^{NP} \left\| X_i^G - \frac{1}{NP} \times \sum_{j=1}^{NP} X_j^G \right\|^2} \tag{8}$$

$$SR^G = \frac{N_S^G}{NP} \tag{9}$$

where N_S^G is the number of successful individuals.

Table 11 Results (average±std) of LBLDE with its variants on the CEC'2017 test suite (30*D*).

Problem	LBLDE-1	LBLDE-2	LBLDE-3	LBLDE
F1	$1.38\times10^2 \pm 2.58\times10^2$(+)	$1.31\times10^1 \pm 9.33\times10^1$(=)	$6.78\times10^1 \pm 1.56\times10^2$(+)	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$
F2	$7.44\times10^{11} \pm 2.12\times10^{12}$(+)	$7.25\times10^8 \pm 5.05\times10^9$(+)	$5.53\times10^{10} \pm 1.34\times10^{11}$(+)	$\mathbf{4.25\times10^6 \pm 3.03\times10^7}$
F3	$4.49\times10^4 \pm 9.49\times10^3$(+)	$1.45\times10^0 \pm 3.61\times10^0$(+)	$4.47\times10^4 \pm 1.30\times10^4$(+)	$\mathbf{1.19\times10^{-1} \pm 3.52\times10^{-1}}$
F4	$4.90\times10^1 \pm 3.35\times10^1$(=)	$4.28\times10^1 \pm 3.11\times10^1$(=)	$5.31\times10^1 \pm 3.16\times10^1$(+)	$\mathbf{3.02\times10^1 \pm 3.19\times10^1}$
F5	$3.56\times10^1 \pm 4.05\times10^0$(+)	$2.82\times10^1 \pm 5.49\times10^0$(+)	$3.44\times10^1 \pm 5.10\times10^0$(+)	$\mathbf{2.38\times10^1 \pm 3.98\times10^0}$
F6	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$**(=)**	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$**(=)**	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$**(=)**	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$
F7	$6.35\times10^1 \pm 4.65\times10^0$(+)	$5.60\times10^1 \pm 3.77\times10^0$(+)	$6.14\times10^1 \pm 4.53\times10^0$(+)	$\mathbf{5.39\times10^1 \pm 4.98\times10^0}$
F8	$3.45\times10^1 \pm 5.16\times10^0$(+)	$2.94\times10^1 \pm 4.13\times10^0$(+)	$3.20\times10^1 \pm 4.41\times10^0$(+)	$\mathbf{2.55\times10^1 \pm 5.13\times10^0}$
F9	$1.06\times10^1 \pm 7.88\times10^0$(+)	$\mathbf{0.00\times10^0 \pm 0.00\times10^0}$**(=)**	$7.51\times10^{-1} \pm 1.30\times10^0$(+)	$8.91\times10^{-3} \pm 6.36\times10^{-2}$
F10	$1.99\times10^3 \pm 1.86\times10^2$(=)	$1.96\times10^3 \pm 2.22\times10^2$(=)	$1.96\times10^3 \pm 2.33\times10^2$(=)	$\mathbf{1.89\times10^3 \pm 2.25\times10^2}$
F11	$8.33\times10^1 \pm 2.47\times10^1$(+)	$1.13\times10^1 \pm 1.42\times10^1$(=)	$4.18\times10^1 \pm 1.34\times10^1$(+)	$\mathbf{8.55\times10^0 \pm 3.20\times10^0}$
F12	$2.70\times10^5 \pm 1.91\times10^5$(+)	$\mathbf{2.06\times10^3 \pm 1.68\times10^3}$**(=)**	$1.55\times10^5 \pm 1.45\times10^5$(+)	$2.97\times10^3 \pm 3.45\times10^3$
F13	$2.95\times10^4 \pm 1.64\times10^4$(+)	$2.22\times10^1 \pm 7.47\times10^0$(+)	$1.41\times10^4 \pm 1.04\times10^4$(+)	$\mathbf{1.82\times10^1 \pm 7.13\times10^0}$
F14	$3.36\times10^4 \pm 1.54\times10^4$(+)	$2.04\times10^1 \pm 6.99\times10^0$(+)	$2.59\times10^4 \pm 1.55\times10^4$(+)	$\mathbf{8.90\times10^0 \pm 6.54\times10^0}$
F15	$1.04\times10^4 \pm 6.10\times10^3$(+)	$7.51\times10^0 \pm 2.13\times10^0$(+)	$4.70\times10^3 \pm 3.23\times10^3$(+)	$\mathbf{5.42\times10^0 \pm 2.00\times10^0}$
F16	$4.64\times10^2 \pm 1.32\times10^2$(+)	$\mathbf{2.56\times10^2 \pm 1.42\times10^2}$**(=)**	$5.20\times10^2 \pm 1.13\times10^2$(+)	$3.09\times10^2 \pm 1.57\times10^2$
F17	$7.47\times10^1 \pm 1.81\times10^1$(+)	$3.99\times10^1 \pm 7.38\times10^0$(+)	$7.90\times10^1 \pm 2.68\times10^1$(+)	$\mathbf{3.19\times10^1 \pm 1.23\times10^1}$
F18	$1.90\times10^5 \pm 8.39\times10^4$(+)	$2.25\times10^1 \pm 3.13\times10^0$(+)	$1.63\times10^5 \pm 6.03\times10^4$(+)	$\mathbf{1.91\times10^1 \pm 6.57\times10^0}$
F19	$9.93\times10^3 \pm 5.78\times10^3$(+)	$9.66\times10^0 \pm 1.38\times10^0$(+)	$5.17\times10^3 \pm 4.58\times10^3$(+)	$\mathbf{7.58\times10^0 \pm 2.14\times10^0}$
F20	$1.30\times10^2 \pm 6.02\times10^1$(+)	$3.74\times10^1 \pm 3.23\times10^1$(+)	$1.29\times10^2 \pm 6.16\times10^1$(+)	$\mathbf{2.23\times10^1 \pm 4.00\times10^1}$
F21	$2.35\times10^2 \pm 5.34\times10^0$(+)	$2.29\times10^2 \pm 3.86\times10^0$(+)	$2.34\times10^2 \pm 5.24\times10^0$(+)	$\mathbf{2.25\times10^2 \pm 5.21\times10^0}$
F22	$1.00\times10^2 \pm 2.71\times10^0$(+)	$\mathbf{1.00\times10^2 \pm 1.00\times10^{-13}}$**(=)**	$1.00\times10^2 \pm 1.58\times10^0$(=)	$1.00\times10^2 \pm 1.22\times10^{-13}$
F23	$3.82\times10^2 \pm 4.90\times10^0$(+)	$3.74\times10^2 \pm 5.92\times10^0$(+)	$3.79\times10^2 \pm 5.95\times10^0$(+)	$\mathbf{3.69\times10^2 \pm 5.63\times10^0}$
F24	$4.52\times10^2 \pm 5.04\times10^0$(+)	$4.42\times10^2 \pm 5.42\times10^0$(=)	$4.49\times10^2 \pm 5.94\times10^0$(+)	$\mathbf{4.41\times10^2 \pm 5.84\times10^0}$
F25	$3.87\times10^2 \pm 2.23\times10^{-1}$(=)	$\mathbf{3.87\times10^2 \pm 2.09\times10^{-1}}$**(−)**	$3.87\times10^2 \pm 2.65\times10^{-1}$(=)	$3.87\times10^2 \pm 8.00\times10^{-1}$
F26	$1.31\times10^3 \pm 9.98\times10^1$(+)	$\mathbf{1.13\times10^3 \pm 2.64\times10^2}$**(=)**	$1.25\times10^3 \pm 2.07\times10^2$(+)	$1.16\times10^3 \pm 1.99\times10^2$
F27	$5.08\times10^2 \pm 5.11\times10^0$(+)	$\mathbf{4.93\times10^2 \pm 7.70\times10^0}$**(−)**	$5.07\times10^2 \pm 3.92\times10^0$(+)	$4.96\times10^2 \pm 9.72\times10^0$
F28	$\mathbf{3.16\times10^2 \pm 3.97\times10^1}$**(=)**	$3.24\times10^2 \pm 4.91\times10^1$(=)	$3.21\times10^2 \pm 4.23\times10^1$(=)	$3.25\times10^2 \pm 4.51\times10^1$
F29	$4.90\times10^2 \pm 3.20\times10^1$(+)	$4.38\times10^2 \pm 3.64\times10^1$(+)	$4.92\times10^2 \pm 3.34\times10^1$(+)	$\mathbf{4.27\times10^2 \pm 2.94\times10^1}$
F30	$1.05\times10^4 \pm 3.24\times10^3$(+)	$\mathbf{1.98\times10^3 \pm 3.61\times10^1}$**(=)**	$7.56\times10^3 \pm 2.11\times10^3$(+)	$2.00\times10^3 \pm 5.63\times10^1$
+/=/−	25/5/0	15/13/2	25/5/0	−

Figure 6 shows the iterative curves of population diversity and error of LBLDE obtained on F5, F14, and F24 by two differential vector selection methods. Note that "*FES*" is used as the abscissa in Fig. 5 because different algorithms have different generations. In this subsection, LBLDE and its variants have the same generations. Thus, the "Generation" is used as the abscissa. F5 is a simple multimodal function, F14 indicates the hybrid function, and F24 represents the composition function. LBLDE_r indicates the selection of difference vectors in a random way, and LBLDE_l refers to the difference vector selection method based on level. Figure 6 shows that LBLDE_r has higher diversity than LBLDE_l from the beginning to the end of the evolutionary procedure. As a result of the slower convergence speed of LBLDE_r in the late stage, the eventually evolved results obtained by LBLDE_r are worse than those obtained by LBLDE_l.

(2) In Section 3.3, some changes are made on several parameters. The first one is that $\mu_{CR\text{-}ini}$ is reduced, and the best trade-off value is 0.35 obtained in Section 4.1. A validation experiment is performed on F1, F6, and F16, which represent three different trends on μ_{CR}. The curves of μ_{CR} with different initial values on three distinct functions during the process of evolution are drawn in Fig. 7. As analyzed in Section 3.3, if $\mu_{CR\text{-}ini}$ =

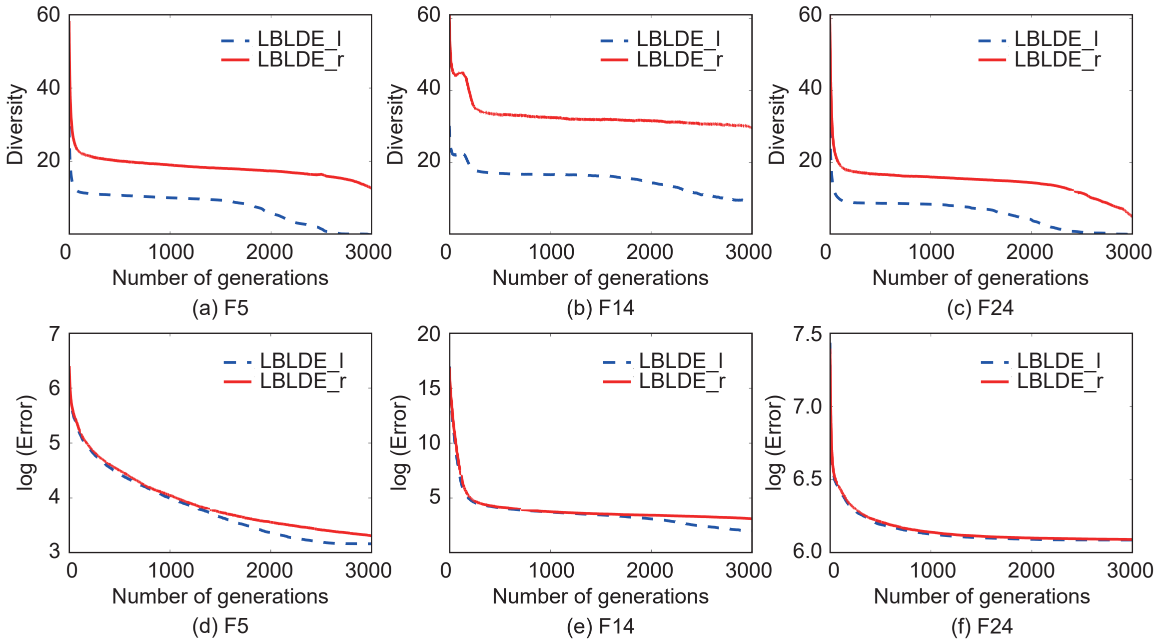

Fig. 6 Population diversity ((a)−(c)) and log (Error) ((d)−(f)) curves obtained on F5, F14, and F24 by two difference vector selection methods (LBLDE_l: LBLDE with difference vector selection method based on level; LBLDE_r: LBLDE with random selection method of difference vectors).

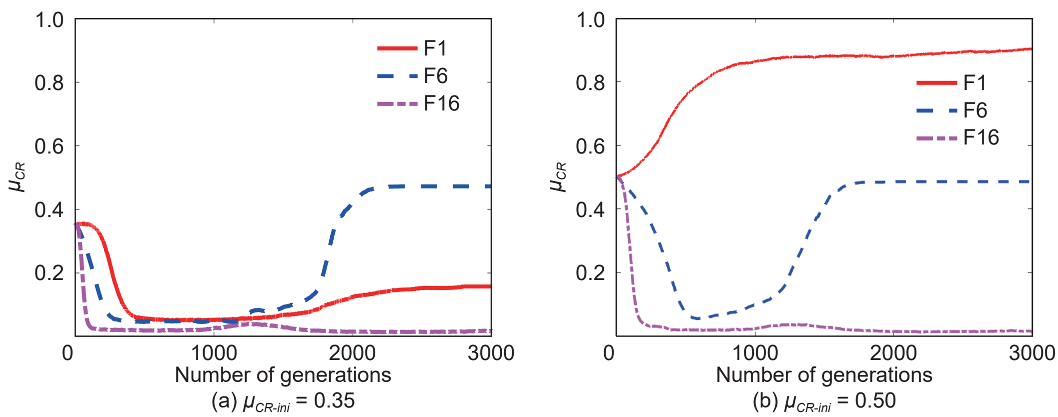

Fig. 7 Curves of μ_{CR} with different initial values (a) $\mu_{CR\text{-}ini}$ = 0.35 and (b) $\mu_{CR\text{-}ini}$ = 0.5 during the evolution on three distinct functions.

0.5, then μ_{CR} may fluctuate at a higher level for F1. This situation is detrimental to the population in the late stage of evolution because the population needs small CR values to increase its convergence rate. Moreover, from Fig. 4, the performance of the algorithm with $\mu_{CR\text{-}ini}$ = 0.35 on F1 is better than that with $\mu_{CR\text{-}ini}$ = 0.5.

(3) The second modification is that CR in the lowest level is set as 1 to guarantee that the population can continue to evolve in the late stage. To verify this idea, for a total of 3000 generations, the previous half is regarded as the early stage, while the remaining half is the late stage. The compared experiments are implemented on four LBLDE variants, which are LBLDE_e (CR = 1 in the early stage), LBLDE_l (CR = 1 in the late stage), LBLDE_el (CR = 1 in both early and late stages), and LBLDE_w (CR = 1 in neither early nore late stage), respectively.

The results on F5, F14, and F24 are shown in Fig. 8, where diversity curves show that the population diversity of LBLDE_e is slightly lower than that of LBLDE_l. For LBLDE_e and LBLDE_el, the population of the former may fall into stagnation, and the population of the latter can continue to evolve given the decline in diversity. However, LBLDE_l obtains high diversity in the early stage, while its convergence speed is slower than that of LBLDE_el in the late stage. The success rate curves also show that the success rate of LBLDE_el increases suddenly in the late stage. This increase occurs because poor individuals converge to the vicinity of the optimal solution quickly, which increases the exploitation ability of excellent individuals. From the fitness curves, LBLDE_el can continue to converge in the late stage and find a better solution. Therefore, LBLDE_el is proved to be the best one.

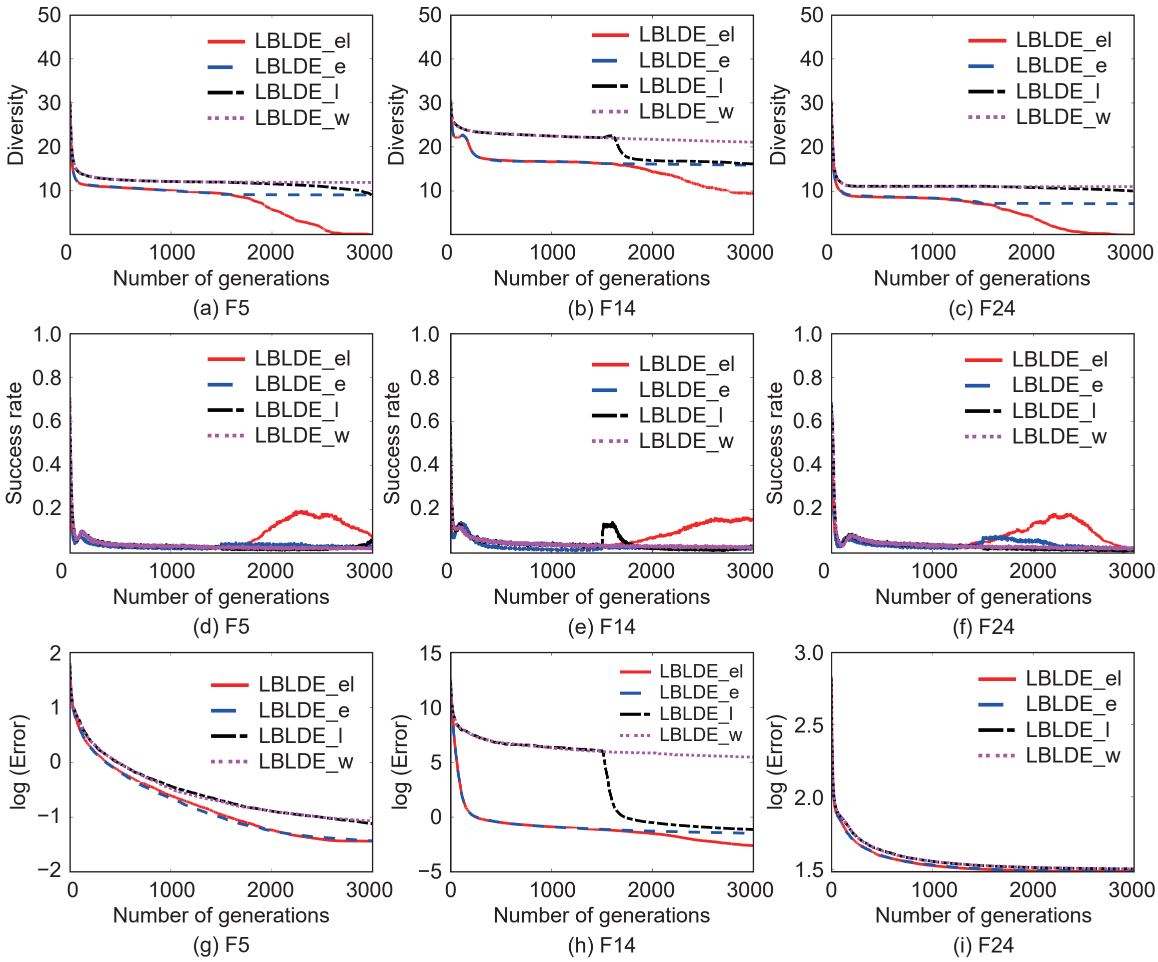

Fig. 8 Population diversity ((a)−(c)), success rate ((d)−(f)), and log (Error) ((g)−(i)) curves obtained by four LBLDE variants on F5, F14, and F24. LBLDE_el: *CR* = 1 of the last level in both early and late stages; LBLDE_e: *CR* = 1 of the last level in early stage; LBLDE_l: *CR* = 1 of the last level in late stage; and LBLDE_w: *CR* value of the last level is not set as 1 in both early and late stages.

5 Conclusion and Future Works

To solve complex global optimization problems, we propose a novel DE variant, called LBLDE, which chooses DE/current-to-*p*best/1 to determine the population's evolutionary direction, which has a great advantage in solving unimodal problems[32]. Nevertheless, such a strategy could lead the population to a local area on complex problems due to low diversity. Consequently, the level-based learning mechanism is used in LBLDE for improving the population diversity effectively. In accordance with the requirement of each level, the method used to select the difference vectors is changed to guarantee the population convergence speed. Moreover, different *CR* values are allocated to different levels for exerting their unique functions.

Thirty functions in the CEC'2017 test suite provide a fair platform to evaluate the performance of LBLDE. Seven DE variants are used to compare with LBLDE. The results show that LBLDE has a superior or similar performance in comparison with the other seven algorithms, thus demonstrating the superiority of the proposed LBLDE.

In the future, we will study the design of adaptive methods to adjust the number of levels. Other methods to improve population diversity will be studied to assist the algorithm in solving high-dimensional problems. In addition, we will extend LBLDE to solve complex multiobjective optimization problems, such as multimodal multiobjective optimization problems[73], constrained multiobjective optimization problems[74], and large-scale multiobjective optimization problems[75].

Acknowledgment

This work was supported in part by the National Natural Science Fund for Outstanding Young Scholars of China (No. 61922072), the National Natural Science Foundation of China (Nos. 61876169, 61276238, 61806179, and 61976237), and Key Research and Development and Promotion Projects in Henan Province (No. 192102210098).

References

[1] R. Storn and K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[2] F. Zhao, S. Di, J. Cao, J. Tang, and Jonrinaldi, A novel cooperative multi-stage hyper-heuristic for combination optimization problems, *Complex System Modeling and Simulation*, vol. 1, no. 2, pp. 91–108, 2021.

[3] W. Gong, Z. Liao, X. Mi, L. Wang, and Y. Guo, Nonlinear equations solving with intelligent optimization algorithms: A survey, *Complex System Modeling and Simulation*, vol. 1, no. 1, pp. 15–32, 2021.

[4] W. Deng, H. Liu, J. Xu, H. Zhao, and Y. Song, An improved quantuminspired differential evolution algorithm for deep belief network, *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 7319–7327, 2020.

[5] X. Yu, C. Li, and J. Zhou, A constrained differential evolution algorithm to solve uav path planning in disaster scenarios, *Knowledge-Based Systems*, vol. 204, pp. 106209–106220, 2020.

[6] J. Liang, K. Qiao, M. Yuan, K. Yu, B. Qu, S. Ge, Y. Li, and G. Chen, Evolutionary multi-task optimization for parameters extraction of photovoltaic models, *Energy Conversion and Management*, vol. 207, pp. 112509–112524, 2020.

[7] J. Liang, K. Qiao, C. Yue, K. Yu, B. Qu, R. Xu, Z. Li, and Y. Hu, A clustering-based differential evolution algorithm for solving multimodal multi-objective optimization problems, *Swarm and Evolutionary Computation*, vol. 60, pp. 100788–100802, 2021.

[8] W. Deng, S. Shang, X. Cai, H. Zhao, Y. Zhou, H. Chen, and W. Deng, Quantum differential evolution with cooperative coevolution framework and hybrid mutation strategy for large scale optimization, *Knowledge-Based Systems*, vol. 224, pp. 107080–107094, 2021.

[9] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, and E. Goodman, Push and pull search for solving constrained multi-objective optimization problems, *Swarm and Evolutionary Computation*, vol. 44, pp. 665–679, 2019.

[10] M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, Differential evolution: A review of more than two decades of research, *Engineering Applications of Artificial Intelligence*, vol. 90, pp. 103479–103504, 2020.

[11] K. R. Opara and J. Arabas, Differential evolution: A survey of theoretical analyses, *Swarm and Evolutionary Computation*, vol. 44, pp. 546–558, 2019.

[12] E. Cantú-Paz, Migration policies, selection pressure, and parallel evolutionary algorithms, *Journal of Heuristics*, vol. 7, no. 4, pp. 311–334, 2001.

[13] W. Gong and Z. Cai, Differential evolution with ranking-based mutation operators, *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2066–2081, 2013.

[14] P. C. Wang, X. Qian, and X. H. Hu, A novel differential evolution algorithm based on chaos local search, in *Proc. of International Conference on Information Engineering and Computer Science,* Wuhan, China, 2009, pp. 1–4.

[15] D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, Parallel differential evolution, in *Proceedings of the Congress on Evolutionary Computation,* Portland, OR, USA, 2004, pp. 2023–2029.

[16] S. Das, S. S. Mullick, and P. N. Suganthan, Recent

advances in differential evolution—an updated survey, *Swarm and Evolutionary Computation*, vol. 27, pp. 1–30, 2016.

[17] Y. Wang, Z. Cai, and Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[18] M. Di Carlo, M. Vasile, and E. Minisci, Adaptive multi-population inflationary differential evolution, *Soft Computing*, vol. 24, no. 5, pp. 3861–3891, 2020.

[19] E. Alba and M. Tomassini, Parallelism and evolutionary algorithms, *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 443–462, 2002.

[20] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, Enhancing differential evolution utilizing proximitybased mutation operators, *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 99–119, 2011.

[21] H. Peng, Z. Guo, C. Deng, and Z. Wu, Enhancing differential evolution with random neighbors based strategy, *Journal of Computational Science*, vol. 26, pp. 501–511, 2018.

[22] R. Tanabe and A. S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in *Proc. of IEEE Congress on Evolutionary Computation,* Beijing, China, 2014, pp. 1658–1665.

[23] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi, LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems, in *Proc. of 2017 IEEE Congress on Evolutionary Computation,* Donostia, Spain, 2017, pp. 145–152.

[24] J. Brest and M. S. Maučec, Population size reduction for the differential evolution algorithm, *Applied Intelligence*, vol. 29, no. 3, pp. 228–247, 2008.

[25] Z. Meng and C. Yang, Hip-DE: Historical population based mutation strategy in differential evolution with parameter adaptive mechanism, *Information Sciences*, vol. 562, pp. 44–77, 2021.

[26] D. Zaharie, Influence of crossover on the behavior of differential evolution algorithms, *Applied Soft Computing*, vol. 9, no. 3, pp. 1126–1138, 2009.

[27] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, *Applied Soft Computing*, vol. 11, no. 2, pp. 1679–1696, 2011.

[28] J. Zhang and A. C. Sanderson, JADE: Adaptive differential evolution with optional external archive, *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[29] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, Selfadapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[30] Q. Yang, W. N. Chen, J. D. Deng, Y. Li, T. Gu, and J. Zhang, A level-based learning swarm optimizer for large-scale optimization, *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 578–594, 2018.

[31] Q. Yang, W. N. Chen, T. Gu, H. Jin, W. Mao, and J. Zhang, An adaptive stochastic dominant learning swarm optimizer for high-dimensional optimization, *IEEE Transactions on Cybernetics*, vol. 52, no. 3, pp. 1–17, 2022.

[32] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, Ensemble of differential evolution variants, *Information Sciences*, vol. 423, pp. 172–186, 2018.

[33] S. Wang, Y. Li, H. Yang, and H. Liu, Self-adaptive differential evolution algorithm with improved mutation strategy, *Soft Computing*, vol. 22, no. 10, pp. 3433–3447, 2018.

[34] A. W. Mohamed and A. K. Mohamed, Adaptive guided differential evolution algorithm with novel mutation for numerical optimization, *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 2, pp. 253–277, 2019.

[35] L. Cui, G. Li, Z. Zhu, Q. Lin, K.-C. Wong, J. Chen, N. Lu, and J. Lu, Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism, *Information Sciences*, vol. 422, pp. 122–143, 2018.

[36] Y. Cai, C. Shao, Y. Zhou, S. Fu, H. Zhang, and H. Tian, Differential evolution with adaptive guiding mechanism based on heuristic rules, *IEEE Access*, vol. 7, pp. 58023–58040, 2019.

[37] W. J. Yu, J. J. Li, J. Zhang, and M. Wan, Differential evolution using mutation strategy with adaptive greediness degree control, in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation,* Vancouver, Canada, 2014, pp. 73–80.

[38] X. Wang and L. Tang, Multi-objective optimization using a hybrid differential evolution algorithm, in *Proc. of IEEE Congress on Evolutionary Computation,* Brisbane, Australia, 2012, pp. 1–8.

[39] M. Leon, N. Xiong, D. Molina, and F. Herrera, A novel memetic framework for enhancing differential evolution algorithms via combination with alopex local search, *International Journal of Computational Intelligence Systems*, vol. 12, no. 2, pp. 795–808, 2019.

[40] S. Li, W. Gong, X. Yan, C. Hu, D. Bai, and L. Wang, Parameter estimation of photovoltaic models with memetic adaptive differential evolution, *Solar Energy*, vol. 190, pp. 465–474, 2019.

[41] W. Liu, X. Wang, and X. Li, Memetic differential evolution for vehicle routing problem with time windows, in *Proc. of International Conference in Swarm Intelligence,* Berlin, Germany, 2012, pp. 358–365.

[42] A. Caponio, F. Neri, and V. Tirronen, Super-fit control adaptation in memetic differential evolution frameworks, *Soft Computing*, vol. 13, no. 8, pp. 811–831, 2009.

[43] F. Neri and V. Tirronen, Scale factor local search in differential evolution, *Memetic Computing*, vol. 1, no. 2, pp. 153–171, 2009.

[44] N. Lynn, M. Z. Ali, and P. N. Suganthan, Population topologies for particle swarm optimization and differential evolution, *Swarm and Evolutionary Computation*, vol. 39, pp. 24–35, 2018.

[45] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, Differential evolution with multi-population based ensemble of mutation strategies, *Information Sciences*, vol. 329, pp. 329–345, 2016.

[46] X. Li, L. Wang, Q. Jiang, and N. Li, Differential evolution algorithm with multi-population cooperation and multi-strategy integration, *Neurocomputing* , vol. 421, pp. 285–302, 2021.

[47] M. Weber, F. Neri, and V. Tirronen, Distributed differential evolution with explorative-exploitative population families, *Genetic Programming and Evolvable Machines*, vol. 10, no. 4, pp. 343–372, 2009.

[48] I. De Falco, A. Della Cioppa, D. Maisto, U. Scafuri, and E. Tarantino, Biological invasion-inspired migration in distributed evolutionary algorithms, *Information Sciences*, vol. 207, pp. 50–65, 2012.

[49] I. De Falco, A. Della Cioppa, D. Maisto, and U. Scafuri, An adaptive invasion-based model for distributed differential evolution, *Information Sciences*, vol. 278, pp. 653–672, 2014.

[50] M. A. Bouteldja and M. Batouche, A study on differential evolution and cellular differential evolution for multilevel color image segmentation, in *Proc. of Intelligent Systems and Computer Vision,* Fez, Morocco, 2017, pp. 1–8.

[51] A. K. Qin, V. L. Huang, and P. N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[52] Q. Fan, W. Wang, and X. Yan, Differential evolution algorithm with strategy adaptation and knowledge-based control parameters, *Artificial Intelligence Review*, vol. 51, no. 2, pp. 219–253, 2019.

[53] J. Liang, K. Qiao, K. Yu, S. Ge, B. Qu, R. Xu, and K. Li, Parameters estimation of solar photovoltaic models via a self-adaptive ensemblebased differential evolution, *Solar Energy*, vol. 207, pp. 336–346, 2020.

[54] K. Qiao, J. Liang, K. Yu, M. Yuan, B. Qu, and C. Yue, Self-adaptive resources allocation-based differential evolution for constrained evolutionary optimization, *Knowledge-Based Systems*, vol. 235, pp. 107653–107667, 2022.

[55] B. C. Wang, H. X. Li, J. P. Li, and Y. Wang, Composite differential evolution for constrained evolutionary optimization, *IEEE Transactions on Systems*, *Man*, *and Cybernetics*: *Systems*, vol. 49, no. 7, pp. 1482–1495, 2019.

[56] Z. Z. Liu, Y. Wang, S. Yang, and Z. Cai, Differential evolution with a two-stage optimization mechanism for numerical optimization, in *Proc. of IEEE Congress on Evolutionary Computation,* Vancouver, Canada, 2016, pp. 3170–3177.

[57] L. Gui, X. Xia, F. Yu, H. Wu, R. Wu, B. Wei, Y. Zhang, X. Li, and G. He, A multi-role based differential evolution, *Swarm and Evolutionary Computation*, vol. 50, pp. 100508–100513, 2019.

[58] Y. Wang, J. P. Li, X. Xue, and B. C. Wang, Utilizing the correlation between constraints and objective function for constrained evolutionary optimization, *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 29–43, 2020.

[59] Z. Tan, K. Li, and Y. Wang, Differential evolution with adaptive mutation strategy based on fitness landscape analysis, *Information Sciences*, vol. 549, pp. 142–163, 2021.

[60] F. Peng, K. Tang, G. Chen, and X. Yao, Multi-start jade with knowledge transfer for numerical optimization, in *Proc. of IEEE Congress on Evolutionary Computation,* Trondheim, Norway, 2009, pp. 1889–1895.

[61] Z. Li, J. Guo, and S. Yang, Improving the JADE algorithm by clustering successful parameters, *International Journal of Wireless and Mobile Computing*, vol. 11, no. 3, pp. 190–197, 2016.

[62] R. Tanabe and A. Fukunaga, Success-history based parameter adaptation for differential evolution, in *Proc. of IEEE Congress on Evolutionary Computation,* Cancun, Mexico, 2013, pp. 71–78.

[63] Y. Z. Zhou, W. C. Yi, L. Gao, and X. Y. Li, Adaptive differential evolution with sorting crossover rate for continuous optimization problems, *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2742–2753, 2017.

[64] W. J. Yu, M. Shen, W. N. Chen, Z. H. Zhan, Y. J. Gong, Y. Lin, O. Liu, and J. Zhang, Differential evolution with two-level parameter adaptation, *IEEE Transactions on Cybernetics*, vol. 44, no. 7, pp. 1080–1099, 2014.

[65] V. Tirronen and F. Neri, Differential evolution with fitness diversity self-adaptation, in *Proc. of Nature-Inspired Algorithms for Optimisation,* Berlin, Germany, 2009, pp. 199–234.

[66] A. Zamuda, J. Brest, and E. M. Montes, Structured population size reduction differential evolution with multiple mutation strategies on CEC 2013 real parameter optimization, in *Proc. of IEEE Congress on Evolutionary Computation,* Cancun, Mexico, 2013, pp. 1925–1931.

[67] P. Rakshit, A. Konar, P. Bhowmik, I. Goswami, S. Das, L. C. Jain, and A. K. Nagar, Realization of an adaptive memetic algorithm using differential evolution and Q-learning: A case study in multirobot path planning, *IEEE Transactions on Systems*, *Man*, *and Cybernetics*: *Systems*, vol. 43, no. 4, pp. 814–831, 2013.

[68] R. D. Al-Dabbagh, F. Neri, N. Idris, and M. S. Baba, Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy, *Swarm and Evolutionary Computation*, vol. 43, pp. 284–311, 2018.

[69] N. Awad, M. Ali, J. Liang, B. Qu, and P. Suganthan, Evaluation criteria for the CEC 2017 special session and competition on single objective realparameter numerical optimization, Technology Report, Nanyang Technological University, Singapore, 2016.

[70] A. K. Mohamed and A. W. Mohamed, Real-parameter unconstrained optimization based on enhanced AGDE algorithm, in *Machine Learning Paradigms: Theory and Application,* Hassanien, ed. Cham, Switzerland: Springer, 2019, pp. 431–450.

[71] A. W. Mohamed and P. N. Suganthan, Real-parameter

unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation, *Soft Computing*, vol. 22, no. 10, pp. 3215–3235, 2018.
[72] G. Sun, G. Xu, and N. Jiang, A simple differential evolution with timevarying strategy for continuous optimization, *Soft Computing*, vol. 24, no. 4, pp. 2727–2747, 2020.
[73] C. Yue, B. Qu, and J. Liang, A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems, *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 805–817, 2018.
[74] K. Qiao, K. Yu, B. Qu, J. Liang, H. Song, and C. Yue, An evolutionary multitasking optimization framework for constrained multi-objective optimization problems, *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 2, pp. 263–277, 2022.
[75] H. Li, F. He, Y. Chen, and Y. Pan, Mlfs-ccde: Multi-objective large-scale feature selection by cooperative coevolutionary differential evolution, *Memetic Computing*, vol. 13, no. 1, pp. 1–18, 2021.

Kunjie Yu received the BEng degree from Zhengzhou Institute of Light Industry, China in 2012, and the PhD degree in control science and engineering from East China University of Science and Technology, Shanghai, China in 2017. Currently, he is an associate professor at the School of Electrical Engineering, Zhengzhou University. His current research interests include evolutional computation, constrained optimization, multi-objective optimization, and their applications in chemical process, photovoltaic system, and energy system.

Caitong Yue received the BEng and PhD degrees from Zhengzhou University, China in 2014 and 2020, respectively. He studied at the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore from June 2019 to June 2020. He is currently a lecturer at the School of Electrical Engineering, Zhengzhou University, China. His research interests include multimodal multiobjective optimization, pattern recognition, neural network, and particle swarm optimization.

Hui Song received the BEng degree from Henan University of Technology, China in 2011, the MEng degree from Zhengzhou University, China in 2014, and the PhD degree from RMIT University, Australia in 2019. She is currently a research fellow at the School of Engineering, RMIT University. Her research interests mainly include evolutionary computation, machine learning, time-series analytics, evolutionary multi-task learning, and EV charging/discharging.

Kangjia Qiao received the BEng degree from Jianghan University, China in 2018, and the MEng degree from Zhengzhou University, China in 2021. He is currently a PhD candidate at Zhengzhou University, China. His current research interests include differential evolution, multitasking optimization, constrained optimization, and multi-modal multi-objective optimization.

Jing Liang received the BEng degree from Harbin Institute of Technology, China in 2003, and the PhD degree from Nanyang Technological University, Singapore in 2009. She is currently a professor at the School of Electrical Engineering, Zhengzhou University, China. Her main research interests are evolutionary computation, swarm intelligence, multi-objective optimization, and neural network. She serves as an associate editor for *IEEE Transactions on Evolutionary Computation* and the *Swarm and Evolutionary Computation*.

Boyang Qu received the BEng and PhD degrees from Nanyang Technological University, Singapore in 2008 and 2012, respectively. He is a professor at the School of Electronic and Information, Zhongyuan University of Technology, China. His research interests include machine learning, neural network, genetic and evolutionary algorithms, swarm intelligence, and multi-objective optimization. He is an associate editor for *Swarm and Evolutionary Computation*.