Underestimation-Assisted Global-Local Cooperative Differential Evolution and the Application to Protein Structure Prediction

Xiao-Gen Zhou, Chun-Xiang Peng, Jun Liu, Yang Zhang, and Gui-Jun Zhang

***Abstract*—Various mutation strategies show distinct advantages in differential evolution (DE). The cooperation of multiple strategies in the evolutionary process may be effective. This article presents an underestimation-assisted global and local cooperative DE to simultaneously enhance the effectiveness and efficiency. In the proposed algorithm, two phases, namely, the global exploration and the local exploitation, are performed in each generation. In the global phase, a set of trial vectors is produced for each target individual by employing multiple strategies with strong exploration capability. Afterward, an adaptive underestimation model with an self-adapted slope control parameter is proposed to evaluate these trial vectors, the best of which is selected as the candidate. In the local phase, the better-based strategies guided by individuals that are better than the target individual are designed. For each individual accepted in the global phase, multiple trial vectors are generated by using these strategies and filtered by the underestimation value. The cooperation between the global and local phases includes two aspects. First, both of them concentrate on generating better individuals for the next generation. Second, the global phase aims to locate promising regions quickly while the local phase serves as a local search for enhancing convergence. Moreover, a simple mechanism is designed to determine the parameter of DE adaptively in the searching process. Finally, the proposed approach is applied to predict the protein 3-D structure. The experimental studies on classical benchmark functions, CEC test sets, and protein structure prediction problem show that the proposed approach is superior to the competitors.**

***Index Terms*—Cooperation, differential evolution (DE), evolutionary algorithm (EA), protein structure prediction (PSP), underestimation.**

Manuscript received February 27, 2019; revised May 29, 2019 and July 14, 2019; accepted August 27, 2019. Date of publication August 30, 2019; date of current version May 29, 2020. The work of G.-J. Zhang was supported by the National Nature Science Foundation of China under Grant 61773346. The work of Y. Zhang was supported in part by the National Institute of General Medical Sciences Grant GM083107 and Grant GM116960, and in part by the National Science Foundation Grant DBI1564756. *(Corresponding authors: Yang Zhang; Gui-Jun Zhang)*

X.-G. Zhou is with the College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China, and also with the Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: zxg@zjut.edu.cn).

C.-X. Peng, J. Liu, and G.-J. Zhang are with the College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: pengcx@zjut.edu.cn; junl@zjut.edu.cn; zgj@zjut.edu.cn).

Y. Zhang is with the Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI 48109 USA, and also with the Department of Biological Chemistry, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: zhng@umich.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2019.2938531

I. Introduction

DIFFERENTIAL evolution (DE), proposed by Storn and Price [1], is a powerful and popular evolutionary algorithm (EA) for global optimization. Over the past two decades, DE has elicited considerable attention because of its effectiveness and efficiency. Various DE variants have been developed to solve a wide range of complex optimization problems in diverse scientific and engineering fields, such as flow shop scheduling [2], protein structure prediction (PSP) [3], power systems [4], and robust design [5]. Many other applications and improved DE approaches can be found in [6]–[8].

Similar to other EAs, DE simulates the biological evolutionary process through mutation, crossover, and selection operators to evolve the initial population to the global optimal solution. Amongst these operators, the mutation operator, which helps to explore the search space by perturbing individuals, substantially influences the performance of DE [9], [10]. Many different mutation strategies, such as ranking-based [11], triangular [12], centroid-based [13], and neighborhood mutations [14], have been proposed to enhance the search capability of DE. However, each of these mutation strategies seems to be suitable for different tasks. Some of them (e.g., rand-based mutation strategies) are effective in exploring search spaces, whereas others (e.g., strategies that use the best solution found so far) have strong exploitation capability. The exploration can help the algorithm to find the promising solution regions with good population diversity, whereas exploitation can enhance the convergence speed to find the optimal solution by executing the local search in the promising solution regions [15]. Therefore, cooperation between the mutation strategies with superior exploration capability and those that have good exploitation capability will improve the effectiveness and efficiency of DE.

Many approaches have been developed to improve the performance of DE by the cooperation of different mutation strategies. These algorithms can be roughly classified into three categories: 1) individual-specific strategy techniques; 2) evolutionary stage-specific strategy techniques; and 3) subpopulation-specific strategy techniques. Methods in the

first category aim to adaptively select mutation strategies for each individual from the strategy pool. The self-adaptive DE (SaDE) [16], DE with individual-dependent mechanism (IDE) [15], DE with ensemble of mutation strategies and parameters (EPSDE) [17], DE with strategy adaptation mechanism (SaM) [10], and DE with adaptive strategy selection [18] can be considered to belong to the first category. For methods in the second category, such as DE with zoning evolution of control parameters and adaptive mutation strategies (ZEPDE) [19], two-phase DE [20], abstract convex underestimation-assisted multistage DE (UMDE) [13], SaDE with discrete mutation control parameters [21], and DE with multistage strategies [22], divide the entire searching process into multiple stages and select suitable mutation strategies for each stage. Methods in the last category enhance the search capability of DE by utilizing various mutation strategies in different subpopulations. These methods include DE with self-adaptive multioperator [23], [24], DE with hybrid strategies and self-adaptive parameters [25], DE with role assignment [26], DE with multipopulation-based ensemble of mutation strategies [27], and SaDE with more strategies [28]. The experimental results have verified that the above methods can enhance the performance of DE.

In this article, an underestimation-assisted global and local cooperative DE (GLCDE) is proposed to further enhance the search capability of DE. To obtain an accurate underestimation of the objective function, an adaptive underestimation model is designed on the basis of the abstract convexity theory [29], [30], in which the slope control factor of the supporting vectors [31] is dynamically updated based on the evaluated trial individual. According to the underestimation, a global exploration phase and a local exploitation phase are performed for each generation. In the global exploration phase, multiple trial vectors are generated by several different explorative mutation strategies to explore diverse promising solution regions quickly. Subsequently, the underestimation of the objective function is calculated to select one candidate amongst the trial vectors. In the local phase, three new better-based mutation strategies that use the individuals better than the target individual are designed to balance the convergence speed and population diversity. Through these strategies, a set of trial vectors is produced for each accepted individual in the global phase and then filtered by their underestimation values. Additionally, a parameter adaption method is also utilized to automatically determine the parameters of DE in the evolutionary process. The proposed GLCDE is tested on the classical benchmark functions, CEC 2013, 2014, and 2017 test sets, and a real-world case. The experimental results indicate that GLCDE is superior to the compared advanced DE variants for most of the cases.

II. Background Information

Without loss of generality, in this article, a single objective optimization problem is defined as follows:

$$\text{Minimize} \quad f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \tag{1}$$

where Ω represents the feasible region of the search space and $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)^T$ is a D-dimensional decision vector. Each variable x_j is limited in the range $[L_j, U_j]$, where L_j and U_j denote the lower and upper bounds of x_j, respectively.

A. Classical DE Algorithm

1) Initialization: Denote the ith individual in the gth generation as $\boldsymbol{x}_i^g = (x_{i,1}^g, x_{i,2}^g, \ldots, x_{i,D}^g)$, and its initial value at $g = 0$ can be generated by

$$x_{i,j}^0 = L_j + \text{rand}(0, 1) \cdot (U_j - L_j), j = 1, 2, \ldots, D \tag{2}$$

where rand(0, 1) is a uniform distributed random number between 0 and 1. By generating $NP (NP \geq 4)$ individuals according to (2), we can get the initial population $P^0 = \{x_1^0, x_2^0, \ldots, x_{NP}^0\}$ with NP individuals.

2) Mutation: Each individual $\boldsymbol{x}_i^g$ in the current generation is regarded as the target vector, and a new individual called mutant vector can be created by

$$\boldsymbol{v}_i^g = \boldsymbol{x}_{r_1}^g + F(\boldsymbol{x}_{r_2}^g - \boldsymbol{x}_{r_3}^g) \tag{3}$$

where r_1, r_2, and r_3 are mutually different indices randomly selected from the range $\{1, \ldots, NP\}$, and none of them are equal to i; F is the scaling factor within the range $(0, 1]$. Some other widely used mutation strategies are given in the supplementary material.

3) Crossover: The mutant vector is recombined with the corresponding target vector to produce a trial vector $\boldsymbol{u}_i^g$. This process can be implemented by using the exponential recombination [32] or binomial recombination. The binomial recombination which is mostly used can be formulated as

$$u_{i,j}^g = \begin{cases} v_{i,j}^g, & \text{if } \text{rand}(0, 1) \leq CR \text{ or } j = j_{\text{rand}} \\ x_{i,j}^g, & \text{otherwise} \end{cases} \tag{4}$$

where $j = 1, 2, \ldots, D$, and $CR \in [0, 1]$ is the crossover rate. j_{rand} is a random integer selected from $\{1, 2, \ldots, D\}$.

4) Selection: The selection operator picks the better one from $\boldsymbol{x}_i^g$ and $\boldsymbol{u}_i^g$ to enter the next generation. It can be described as

$$\boldsymbol{x}_i^{g+1} = \begin{cases} \boldsymbol{u}_i^g, & \text{if } f(\boldsymbol{u}_i^g) \leq f(\boldsymbol{x}_i^g) \\ \boldsymbol{x}_i^g, & \text{otherwise.} \end{cases} \tag{5}$$

B. Underestimation Model

Abstract convex analysis indicates that every nonconvex function is the upper envelope of its affine minorants [30], [33]. With use of subdifferential-based [34] supporting functions to replace the affine minorants, a lower bound (underestimation) of the objective function can be achieved from below based on a set of supporting functions of the given points.

In DE and other population-based algorithms, each individual in the population is regarded as a given point and the underestimation of the objective problem can be constructed by the supporting functions of all individuals. The supporting function of a individual x_i^g can be defined as follows:

$$h_i^g(\boldsymbol{x}) = \min_{j \in J} \left(f(\boldsymbol{x}_i^g) - M(x_{i,j}^g - x_j) \right) \tag{6}$$

where $J = \{1, 2, \ldots, D+1\}$, $x_{i,D+1}^g = 1 - \sum_{j=1}^{D} x_{i,j}^g$ is a slack variable to simplify the supporting function, and M is the slope control parameter of the supporting function [34].

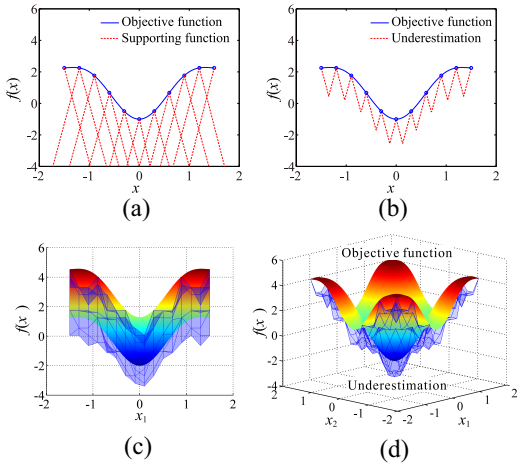


Fig. 1. Illustration of the underestimation, where (a) and (b) are the curves of the supporting functions and underestimation for the 1-D function, respectively. (c) and (d) are 3-D maps with 2-D view and 3-D view of a 2-D objective function plus its underestimation, respectively.

As shown in Fig. 1(a), after calculating the supporting functions of all individuals, we can obtain NP estimation values $h_i^g(x), i = 1, 2, \ldots, NP$ based on the NP supporting functions for each solution x in the feasible region. By considering the largest estimation value (i.e., the value closest to the real objective function value) as the underestimation of each point, an underestimation model of the objection function can be calculated as follows:

$$U(\boldsymbol{x}) = \max_{i=1,\ldots,NP} h_i^g(\boldsymbol{x}). \tag{7}$$

Fig. 1 shows two examples of the underestimation for a 1-D function [Fig. 1(b)] and a 2-D function [Fig. 1(c) and (d)]. As shown in the figure, the underestimation model is consistently below the objective function and covers the entire search space. In addition, the underestimation becomes more accurate as the evolution proceeds because the individuals used to compute the supporting functions become increasingly crowded (see Fig. S1 of the supplementary material). Other properties of the underestimation model can be found in [34]–[36].

III. Literature Review

Recent years, numerous attempts have been made to improve the performance of DE, such as employing multiple mutation strategies, designing new mutation strategies, and developing novel parameter control schemes. This section briefly reviews some of these methods.

Some studies mainly focused on utilizing more than one mutation strategy to breed new solutions. Mallipeddi *et al.* [17] developed EPSDE, in which a pool of mutation strategies and a pool of parameter values are randomly combined to produce trial individuals. Wang *et al.* [37] presented CoDE, in which three mutation strategies along with three sets of parameters are simultaneously used to generate three trial vectors, and the one with the best fitness value is selected as the candidate. Elsayed *et al.* [38] introduced a DE using a mix of different mutation operators. In this algorithm, the population is equally divided into four groups, and each group using their own mutation strategy. IDE, designed by Tang *et al.* [15], assigns four different mutation strategies to the superior and inferior individuals during the search process. Epitropakis *et al.* [39] proposed a hybrid DE that combines explorative and exploitive mutation strategies to balance their effects. Ali *et al.* [40] proposed an adaptive DE with dynamic population reduction (*s*TDE-*d*R), in which the population is adaptively divided into multiple tribes with different size according to their previous success, and multiple different mutation strategies are employed for each tribe. Pan *et al.* [41] proposed DE with self-adapting strategy and control parameters, in which a winning strategy list is used to store strategies that can generate better trial vectors. The mutation strategy is selected from the strategy list refilled by selecting strategies from the winning strategy list.

Numerous new mutation strategies have been proposed and incorporated into DE. Zhang and Sanderson [42] proposed JADE, in which a new mutation strategy named DE/current-to-*p*best/1 is presented. In this strategy, two random individuals, one selected from the top $p\%$ and the other from the archived inferior individuals, are applied to guide the evolution. Das *et al.* [43] presented an improved variant of the DE/target-to-best/1 that combines a global neighborhood model and a local neighborhood model by a weight factor based on the neighborhood individuals or the entire population. Tang *et al.* [44] proposed a new mutation strategy which adds three randomly selected vectors into DE/current-to-*p*best/1 to enhance the population diversity. Islam *et al.* [45] designed a new strategy called DE/current-to-gr_best/1, which adopts the best vector amongst the randomly selected individuals from the current population to replace the globally best vector in the classical DE/current-to-best/1. Cai and Wang [46] proposed a direction induced mutation strategy, in which the base and difference vectors for mutation are selected on the basis of the neighborhood information of the population. Yu *et al.* [47] presented a new strategy named DE/lbest/1. It divides the population into several groups, and the local best vector of each group is used to replace the global best vector in DE/best/1.

Various parameter control schemes have been developed to adaptively tune the scaling factor F, crossover rate CR, and population size NP during the evolution. Qiu *et al.* [48] designed a cross-generation adaptation mechanism to update F and CR for each individual. Tanabe and Fukunaga [49] proposed an improved version of JADE named SHADE, in which a new success-history-based scheme is applied to adjust F and CR adaptively. Brest *et al.* [50] proposed a DE with self-adaptive control parameters (jDE). In jDE, each individual is assigned to its own F and CR, and they are adjusted according to the probabilities τ_1 and τ_2. Tatsis and Parsopoulos [51] introduced an approach to tune F and CR according to the performance of algorithm and another method using gradient approximation and line search [52]. Sarker *et al.* [53] introduced a DE with dynamic parameters, in which the combination of different parameters with better performance have the higher chance been applied for the subsequent generations. Tan *et al.* [54] utilized the online discovered tradeoff surface and the desired population distribution density to adjust NP adaptively. Tanabe and Fukunaga [55] proposed

L-SHADE, which integrates a linear population size reduction technique into SHADE. In L-SHADE, *NP* is continuously reduced according to a linear function calculated by the number of function evaluations. Awad *et al.* [56] proposed a sinusoidal DE with ensemble of parameters and population reduction, in which *F* is adapted by using a Cauchy distribution and two sinusoidal formulas, and *NP* is gradually reduced on the basis of a niching-based reduction approach.

IV. Proposed GLCDE

Although many mutation strategies have been proposed to enhance the search capability of DE, it has been shown that some of them are good at exploring the search space with good exploration capability, and others are fit for local search with strong exploitation capability. To integrate the advantages of explorative and exploitation strategies, we propose GLCDE, an underestimation-assisted GLCDE. GLCDE is characterized by the adaptive underestimation model, global, and local cooperation scheme, and better-based mutation strategies. In the new underestimation model, the slope control parameter is adaptively updated to increase the accuracy of the underestimation. For each generation, two important phases are conducted according to the underestimation, namely, global exploration and local exploitation phases. The former phase aims to locate the promising solution area quickly, while the latter phase is performed as a local search to accelerate the convergence. In the local phase, the better-based mutation strategies which employ the individuals that are better than the target are designed for the second phase to guide the local search. Moreover, a parameter adaptive method is also presented to determine the parameters *F* and *CR* automatically.

A. Framework of GLCDE

The framework of GLCDE is described as Algorithm 1 in the supplementary material. First, the initial population P^0 is generated according to (2). For each generation, the global exploration phase and the local exploitation phase are performed. In the global exploitation phase, three trial vectors are generated by different explorative mutation strategies (DE/rand/1, DE/rand/2, and DE/current-to-rand/1) for each individual. Afterward, the adaptive underestimation model explained in Section IV-B is constructed to evaluate each trial vector. On the basis of the underestimation, the best one with the lowest underestimation value is chosen as the candidate $\boldsymbol{u}_i^g$. Then, the underestimation value is also utilized to guide the selection because the objective function is always above the underestimation model [30]. If the underestimation value $U(\boldsymbol{u}_i^g)$ is larger than the function value $f(\boldsymbol{x}_i^g)$ of the target vector, then the candidate $\boldsymbol{u}_i^g$ is directly discarded. Otherwise, the candidate $\boldsymbol{u}_i^g$ is evaluated by the objective function and updated according to (5). In the local phase, for each accepted trial individual in the first phase, three trial vectors are produced by the better-based mutation strategies introduced in Section IV-C. Subsequently, they are also filtered by the adaptive underestimation model and updated according to the selection process of the global phase. All trial individuals in the initial population are evaluated by the objective function.

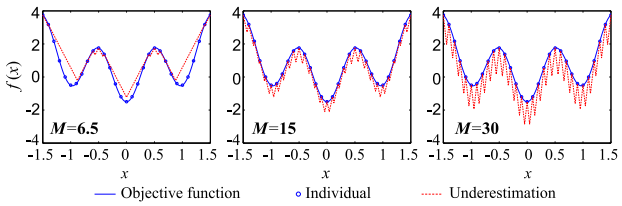


Fig. 2. Underestimation curves with different values of the slope control parameter *M* for a 1-D function.

For each trial individual $\boldsymbol{u}_i^0$ in the initial population, a slope control parameter M_i^0 can be computed. The largest of all M_i^0 derived from all trial individuals is considered as the initial value M^0. During the evolutionary process, *M* is adaptively updated according to all M_i^g, $i = 1, 2, \ldots, N$ calculated from the evaluated trial individuals of the current population, where *N* is the number of evaluated trial individuals. The methods for the initialization and updating of *M* also are introduced in Section IV-B. Furthermore, the DE parameters *F* and *CR* are automatically adjusted by using a simple adaptive scheme explained in Section IV-D during the searching process.

B. Adaptive Underestimation Model

In both global phase and local phase of GLCDE, multiple trial vectors are generated for each target individual by utilizing different mutation strategies and evaluated by the underestimation. Let $\boldsymbol{u}_i^g$ be one of the trial vectors for the target individual $\boldsymbol{x}_i^g$. As we discussed in [13], to reduce the computational complexity, only the supporting functions of the two individuals near $\boldsymbol{u}_i^g$ (determined by Euclidean distance) are calculated to obtain the underestimation value in GLCDE. Let $\boldsymbol{x}_a^g$ and $\boldsymbol{x}_b^g$ are the two individuals near $\boldsymbol{u}_i^g$, where $a \neq b \in \{1, \ldots, NP\}$. The underestimation value $U(\boldsymbol{u}_i^g)$ of $\boldsymbol{u}_i^g$ can be calculated by

$$U(\boldsymbol{u}_i^g) = \max\left(h_a^g(\boldsymbol{u}_i^g), h_b^g(\boldsymbol{u}_i^g)\right) \tag{8}$$

where $h_a^g(\boldsymbol{u}_i^g)$ and $h_b^g(\boldsymbol{u}_i^g)$ are the estimation values of $\boldsymbol{u}_i^g$ calculated by the supporting functions of $\boldsymbol{x}_a^g$ and $\boldsymbol{x}_b^g$ according to (6), respectively.

Based on the underestimation value of each trial vector, the best one with the lowest underestimation value is chosen as the candidate for each trial individual. However, *M* used to control the slope of the supporting functions (6) significantly influences the accuracy of the underestimation. For example, the underestimation models with $M = 6.5$, $M = 15$, and $M = 30$ for a 1-D function are depicted in Fig. 2. As shown in the figure, for $M = 6.5$, some regions of the underestimation are above the objective function while we want to get the lower bound of the objective function. For $M = 15$ and $M = 30$, although the underestimations are always below the objective function, the underestimation with $M = 15$ is more accurate than that of $M = 30$ because it is closer to the objective function. Therefore, a suitable *M* is crucial to achieving an accurate underestimation with small error.

We investigated *M* in [13] and concluded that $M = 10\,000$ is preferable for all benchmark functions. However, due to the different landscapes of different problems, different *M* may

be more suitable, and the same problem with various dimensions may require different M. Moreover, the population may converge to different regions as evolution proceeds; a specific M may be more effective than a constant one. Therefore, it is desirable to automatically generate the value of M at different stages of the evolutionary process. Motivated by these considerations, we propose an adaptive underestimation model in which M is gradually self-adapted by learning from the evaluated trial individuals.

Let $\boldsymbol{u}_i^{g*}$ be a trial individual that has been evaluated by the objective function. According to (6), its estimation value obtained from the supporting functions of the ith individual $\boldsymbol{x}_i^g$ in the current population can be described as follows:

$$\begin{aligned} h_i^g(\boldsymbol{u}_i^{g*}) &= \min_{j\in J}\left(f(\boldsymbol{x}_i^g) - M\left(x_{i,j}^g - u_{i,j}^{g*}\right)\right) \\ &= f(\boldsymbol{x}_i^g) - M\max_{j\in J}\left(x_{i,j}^g - u_{i,j}^{g*}\right). \end{aligned} \tag{9}$$

According to (9), M can be calculated by

$$M = \frac{f(\boldsymbol{x}_i^g) - h_i^g(\boldsymbol{u}_i^{g*})}{\max_{j\in J}\left(x_{i,j}^g - u_{i,j}^{g*}\right)}. \tag{10}$$

Suppose that the underestimation value of $\boldsymbol{u}_i^{g*}$ is equal to the corresponding function value (i.e., $U(\boldsymbol{u}_i^{g*}) = f(\boldsymbol{u}_i^{g*})$) when the underestimation model is sufficiently accurate. As shown in (8), the underestimation value $U(\boldsymbol{u}_i^{g*})$ can be computed by the supporting functions of either $\boldsymbol{x}_a^g$ or that of $\boldsymbol{x}_b^g$. If $h_a^g(\boldsymbol{u}_i^{g*}) > h_b^g(\boldsymbol{u}_i^{g*})$, that is, $U(\boldsymbol{u}_i^{g*}) = h_a^g(\boldsymbol{u}_i^{g*})$, a slope control parameter M_a^g can be computed according to (10) by replacing $h_a^g(\boldsymbol{u}_i^{g*})$ with $f(\boldsymbol{u}_i^{g*})$. Otherwise, M_b^g is calculated. The larger value between M_a^g and M_b^g is considered as the slope control parameter M_i^g derived from $\boldsymbol{u}_i^{g*}$, namely

$$M_i^g = \max_{t=a,b}\left(\frac{|f(\boldsymbol{x}_t^g) - f(\boldsymbol{u}_i^{g*})|}{\max_{j\in J}|x_{t,j}^g - u_{i,j}^{g*}|}\right). \tag{11}$$

Note that the absolute values of the numerator and the denominator should be taken in (11) to ensure that $M > 0$.

In the initial population of GLCDE, each trial individual is evaluated by the objective function. Based on all evaluated trial individuals, we can obtain NP values M_i^0, $i = 1, 2, \ldots, NP$ of the slope control parameter according to (11). Then the initial value M^0 of M can be determined as follows:

$$M^0 = \max_{i=1,\ldots,NP} M_i^0. \tag{12}$$

Since the underestimation is also utilized to guide the selection between the candidate individual and the target individual, not all trial individuals are evaluated by the objective function except for the initial population. The trial individual is only evaluated if its underestimation value is lower than the function value of the corresponding target individual. Suppose that N trial individuals are evaluated in the current generation. Then, N values of M can be calculated by (11), and the value M^{g+1} for the next generation can be updated by the following:

$$M^{g+1} = \begin{cases} M_{\max}^g, & \text{if } M_{\max}^g < M^g \\ M^g, & \text{otherwise} \end{cases} \tag{13}$$

where $M_{\max}^g = \max_{n=1,\ldots,N} M_n^g$. It should be note that M^g remains unchanged if $N = 0$.

C. Better-Based Mutation Strategy

A number of studies has indicated that best-based mutation strategies, such as DE/best/k, DE/current-to-best/k, and DE/rand-to-best/k have fast convergence because they use the best solution of the current population to guide the evolutionary search [16], [45]. These strategies have strong exploitation capability to promote the convergence speed. However, the population is easy to lose the diversity and the global exploration capability to explore new promising solution regions in many cases, thereby falling into a local optimal point. Hence, using these greedy strategies in the local phase of GLCDE may result in the incapability of the individuals to explore any better solution region of the search space, thus making them difficult to escape the stagnation.

Inspired by the social learning-based particle swarm optimization [57], three new better-based mutation strategies are proposed in this article to preserve the population diversity and exploitation capability simultaneously in the local phase. In the new strategies, all individuals that are better than each target in the current population are, respectively, grouped into an independent set. For each target, an individual is randomly selected from the corresponding set to guide the mutation. The new strategies can be described as follows.

1) *DE/better/1:*

$$\boldsymbol{v}_i^g = \boldsymbol{x}_{\text{better}}^g + F\cdot(\boldsymbol{x}_{r_1}^g - \boldsymbol{x}_{r_2}^g), \tag{14}$$

2) *DE/current-to-better/1:*

$$\boldsymbol{v}_i^g = \boldsymbol{x}_i^g + F\cdot(\boldsymbol{x}_{\text{better}}^g - \boldsymbol{x}_i^g) + F\cdot(\boldsymbol{x}_{r_1}^g - \boldsymbol{x}_{r_2}^g), \tag{15}$$

3) *DE/rand-to-better/1:*

$$\boldsymbol{v}_i^g = \boldsymbol{x}_{r_1}^g + F\cdot(\boldsymbol{x}_{\text{better}}^g - \boldsymbol{x}_{r_1}^g) + F\cdot(\boldsymbol{x}_{r_2}^g - \boldsymbol{x}_{r_3}^g), \tag{16}$$

where $\boldsymbol{x}_{\text{better}}^g$ is an individual randomly selected from all individuals better than $\boldsymbol{x}_i^g$. $\boldsymbol{x}_{r_1}^g$, $\boldsymbol{x}_{r_2}^g$, and $\boldsymbol{x}_{r_3}^g$ are three mutually different individuals randomly chosen from the entire population, and none of them are equal to $\boldsymbol{x}_i^g$ or $\boldsymbol{x}_{\text{better}}^g$. For the best individual, $\boldsymbol{x}_{\text{better}}^g$ is replaced by a random individual of the entire population when use these strategies. Compared to the best-based strategies, the target individuals are not always attracted toward the same globally best individual in the proposed better-based strategies, thereby preventing premature convergence. Therefore, the better-based strategies adopted in the local phase ensure that the promising regions explored in the global phase are exploited and better promising regions are explored.

D. DE Parameter Adaption

In addition to the mutation strategy, the parameters (i.e., F and CR) highly influence the performance of DE. Inappropriate control parameters combine with mutation strategies may cause stagnation due to over exploration or premature convergence because of over exploitation [15]. Inspired by the current approaches, a simple adaptive scheme is designed to

determine F and CR automatically. Take the update of CR as example, the method is described as follow.

As suggested in [17], the value of CR should be taken in the range $[0.1, 0.9]$. In the proposed method, the range is divided into $K = 8$ intervals with a step size of 0.1, and each interval is allocated a selection probability p^k which is initialized as $1/K$. For each target individual in the current generation, CR is randomly generated in the interval selected by a roulette wheel method according to the selection probability. At each generation, the number of CR that falls into the kth interval and the number of trial individuals produced by the CR in the kth interval that can successfully replace the target individual are recorded as NT_k^g and NS_k^g, respectively. After the initial learning generation (LG), the selection probability of each interval is updated according to the success rates of previous LG generations. For the kth interval, the selection probability is calculated by

$$p_k^G = \frac{\sum_{G=g-\mathrm{LG}}^{g-1} SR_k^G}{\sum_{i=1}^{K} \sum_{G=g-\mathrm{LG}}^{g-1} SR_i^G} \tag{17}$$

where

$$SR_k^G = \frac{NS_k^G}{NT_k^G} + \epsilon \tag{18}$$

is the success rate of the kth interval in the Gth generation. $\epsilon = 0.01$ is a small constant to prevent some intervals from being lost in the searching process due to the null selection probability caused by the poor performance in the previous stage. According to [16], $\mathrm{LG} = 20$ is suitable. Similarly, the parameter F is also generated on the basis of the approach for CR. However, the value of F is selected from the range $[0.4, 0.9]$ in the light of the suggestion in [17].

E. Runtime Complexity

For the adaptive underestimation model, the runtime complexity is mainly from the selection of the two individuals near each trial individual. Since they are measured by Euclidean distance, the runtime complexity is $O(NP^2 \cdot D)$. The global phase costs $O(3NP \cdot D)$ runtime as three trial vectors are created for each target individual. For the local phase, if all individuals conduct the local phase, the runtime complexity will be $O(\max(NP \cdot (NP-1), 3NP \cdot D))$ because the better individuals for each target should be determined in the better-based strategies. For the adaption of parameters F and CR adaption, the runtime is $O(NP)$. In summary, the total runtime complexity of GLCDE is $O(NP^2 \cdot D \cdot G_{\max})$ over $G_{\max}$ generations. According to the study in [13], [23], and [46], the runtime complexity of GLCDE is relatively small compared with that of expensive function evaluations. Therefore, the proposed GLCDE is accepted for the practical problems, especially for expensive-to-evaluate problems. The algorithm complexity on benchmark functions and experimental analysis of the efficiency for the real-world problem will be reported in Sections VI-F and VI-G, respectively.

F. Remarks

The presented GLCDE is based on our previous work in [13] and [36], but it significantly differs from them in five aspects: 1) the determination of slope control parameter M; 2) the purpose of underestimation model; 3) the employed mutation strategies; 4) different new mutation strategy is designed; and 5) the selection method of crossover rate CR and scaling factor F. Details on these differences are given in the supplementary material.

V. Application of GLCDE to PSP

As we know, the living organism contains a large number of proteins. Each protein performs a crucial role in the living organisms and is important to carry out the biological functions. The function of a protein is generally determined by its spatial (3-D) structure. The misfolding of the protein 3-D structures will lead to a wide variety of protein-folding diseases, such as cataract disease, mad cow disease, and Alzheimer's disease. Therefore, the information of high-resolution structure of proteins is essential to understand the function of the molecules and to design new drugs against these diseases. Currently, the 3-D structures of proteins can be determined by the experimental methods, such as nuclear magnetic resonance, X-ray crystallography, and cryo-EM. However, these experimental methods are usually costly and time-consuming [58]. Hence, the computational method, i.e., predicting the 3-D structures of proteins using the computer based on an optimization algorithm, becomes an important problem in computational biology [59]. On the basis of the thermodynamic hypothesis [60], the computational method aims to find the global minimum of an energy function as the structure with the lower energy is considered closer to the native. In other words, the PSP problem involves an optimization of the energy function [61]. The search capability of the optimization algorithm highly influences the prediction accuracy. Many approaches have been proposed for the PSP, among which the Rosetta developed by the Baker Lab [62] and AlphaFold(https://deepmind.com/blog/alphafold/) designed by Google are two state-of-the-art approaches and ranked as the top methods in the worldwide CASP [63] competitions.

In this part, the proposed algorithm is applied to solve the PSP problem, where the fragment assembly technique [64] is employed to improve the prediction accuracy and reduce the computational cost. Since the mutation and crossover operators are based on the fragment exchange and assembly rather than the standard operators of DE, we call the proposed approach global and local cooperative EA (GLCEA) in this application. As shown in Fig. 3, for the input target amino acid sequence, the fragment library with homologous fragments (sequence identity $> 30\%$) removed is first generated by the ROBETTA full-chain PSP server (http://robetta.bakerlab.org). Then the initial population is produced by randomly picking up the fragment of each residue position from the corresponding fragment library to assemble NP conformations. For each conformation in the population, the global and local phases are conducted to generate the trial conformation. In each phase, each mutation strategy is converted to the corresponding one

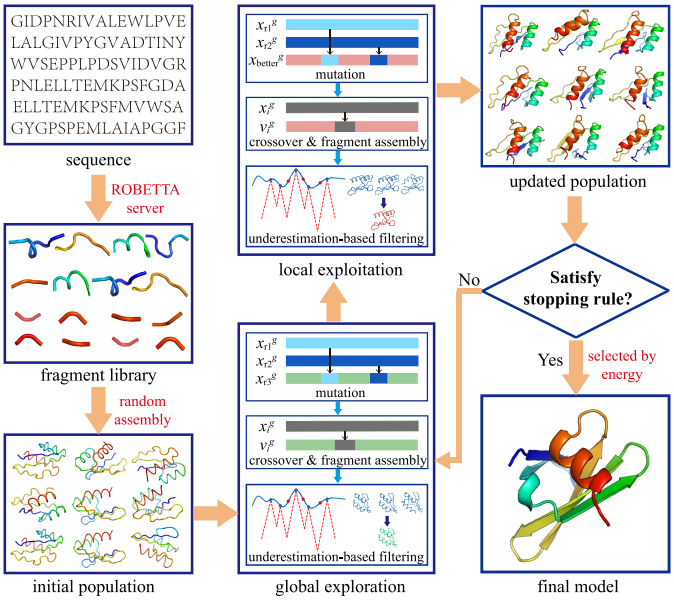


Fig. 3. Pipeline of the proposed GLCEA for PSP problem.

based on fragments exchange between different conformations. For example, in DE/rand/1, three different conformations $x_{r_1}^g$, $x_{r_2}^g$, and $x_{r_3}^g$ are randomly selected from the current population, and the mutation conformation v_i^g is created by the replacement of two random residue positions in the third conformation $x_{r_3}^g$ with the corresponding fragments from the first conformation $x_{r_1}^g$ and the second conformation $x_{r_2}^g$, respectively. However, in DE/better/1, a conformation x_{better}^g which have lower energy than the target conformation is first picked up from the conformations. Then, two different conformations $x_{r_1}^g$ and $x_{r_2}^g$ which also differ from x_{better}^g and the target conformation x_i^g are randomly chosen from the population. Two fragments are randomly extracted from $x_{r_1}^g$ and $x_{r_2}^g$ to replace the corresponding fragments in x_{better}^g, respectively. The conformation is evaluated by the Rosetta score3 energy function [62].

After the mutation conformations generation, the crossover is conducted on the mutation conformations to generate trial conformations. In the crossover, a fragment is randomly chosen from the target conformation x_i^g to replace the corresponding position in the mutation conformation v_i^g. Moreover, to improve the quality and diversity of the conformation, a random fragment assembly is also performed for the mutation conformations. As introduced in Section IV, three different trial conformations are simultaneously generated for each target conformation in both local and global phases. To select the best one from the three trial conformations, the underestimation value of the energy rather than the real energy is calculated because the energy evaluation is usually computational expensive [61]. Based on the coordinates of all C_α atoms of each conformation (i.e., the dimension $D = 3L$, where L is the sequence length), the adaptive underestimation model described in Section IV-A is constructed to measure the quality of each trial conformation, and the best one with lower underestimation value is selected as the candidate offspring conformation. The offspring conformation will be accepted to the new population if it yield lower energy than the target conformation. By iterating the global exploration, local exploitation, and population updating with specific times, the conformation with the lowest energy in the last generation will be selected as the predicted final model. In the above process, the local exploitation is only performed when the offspring conformation has lower energy than the target conformation.

As described in the above process, the mutation and crossover operators of DE are replaced by the corresponding one according to the fragment exchange and assembly between different conformations. Following this strategy, other EAs [65], [66] also can be utilized to the PSP problem.

VI. Experimental Study

In this section, 23 classical benchmark functions selected from [13] and [47], as well as the entire CEC 2013, 2014, and 2017 test sets are used to demonstrate the performance of GLCDE. In the 23 classical benchmark functions, f_1–f_{10} are unimodal, whereas the others are multimodal. Their mathematical expressions are given in Table S1 of the supplementary material. Details of the CEC 2013, 2014, and 2017 test functions can be found in [67]–[69], respectively.

Two main parameters of GLCDE must be set, namely, the population size NP and the learning generation LG. In the following experiments, NP is set to 50, and LG is set as 20 according to the suggestion in [16]. For each approach, 30 and 51 independent runs are conducted for the classical benchmark functions [16] and CEC test sets [69], respectively. The average and standard deviation of the function error $f(\boldsymbol{x}) - f(\boldsymbol{x}^*)$ obtained within the maximum function evaluations (MaxFES) are recorded to evaluate the performance, where $\boldsymbol{x}$ represents the best solution found within the MaxFES in a single run and $\boldsymbol{x}^*$ is the global optimum solution of the test function. In addition, the Wilcoxon signed-rank test is conducted at the 5% significance level to reveal the significant difference between any two approaches. The symbols "+," "≈," and "−" are employed to indicate when the performance of GLCDE is significantly better than, nearly equal to, and remarkably worse than the competitor, respectively. The MaxFES is set to $2000 \times D$, $3000 \times D$, and $3000 \times D$ for the 30-D, 50-D, and 100-D classical benchmark functions [47], respectively. For all CEC test functions, the MaxFES is set to $10\,000 \times D$ as suggested by Awad *et al.* [69].

A. Comparison With State-of-the-Art DE Variants

In this section, the proposed GLCDE is compared with five state-of-the-art DE variants, i.e., EPSDE [17], CoDE [37], SaDE [16], SHADE [49], and UMDE [13] on the 23 classical benchmark functions. For a fair comparison, the parameters of these algorithms are set in the light of their original papers.

Tables S2–S4 of the supplementary material summarize the results of the 30-D, 50-D, and 100-D benchmark functions. The data reveal that GLCDE consistently outperforms the five competitors in most of the cases. Furthermore, the statistically significant results between GLCDE and each competitor are listed in Table I. Specifically, for 30-D problems, GLCDE is significantly better than EPSDE, CoDE, SaDE, SHADE, and

TABLE I
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND FIVE STATE-OF-THE-ART DE VARIANTS ON 30-D, 50-D, AND 100-D CLASSICAL BENCHMARK FUNCTIONS

GLCDE v.s.	$D = 30$			$D = 50$			$D = 100$		
	$+$	$\approx$	$-$	$+$	$\approx$	$-$	$+$	$\approx$	$-$
EPSDE	19	4	0	20	2	1	22	0	1
CoDE	18	2	3	15	5	3	18	1	4
SaDE	17	4	2	14	6	3	17	3	3
SHADE	18	2	3	11	7	5	15	2	6
UMDE	9	10	4	10	8	5	12	5	6

UMDE on 19, 18, 17, 18, and 9 cases, respectively. CoDE, SaDE, SHADE, and UMDE remarkably outperform GLCDE only on 3, 2, 3, and 4 functions, respectively. EPSDE is not significantly superior to GLCDE on any case. The average convergence curves depicted in Fig. S2 of the supplementary material clearly indicate that GLCDE shows faster convergence speed than the control methods for all six representative functions except for f_{12}.

Generally, problems with higher dimension are more difficult to locate the global optimum. However, it is impressive that the performance of GLCDE is not affected by the increase of problem's dimension. For the 50-D functions, GLCDE performs dramatically better than EPSDE, CoDE, SaDE, SHADE, and UMDE on 20, 15, 14, 11, and 10 problems, respectively. EPSDE, CoDE, SaDE, SHADE, and UMDE exhibits remarkably better performance than GLCDE on 1, 3, 3, 5, and 5 functions, respectively. For 100-D problems, GLCDE obtains obviously better performance on 22, 18, 17, 15, and 12 cases compared to EPSDE, CoDE, SaDE, SHADE, and UMDE, respectively. The five competitors significantly outperform GLCDE on 1, 4, 3, 6, and 6 functions, respectively.

B. Comparison With Up-to-Date DE Variants

In this section, GLCDE is compared with nine up-to-date DE variants published in recent years. We first compare GLCDE with ZEPDE [21], SHADE [49], SinDE [70], and IDE [15] on the CEC 2013 test set. The parameters of these four approaches are set in the light of their original papers. Tables S5 and S6 of the supplementary material list the detailed results of the 30-D and 50-D problems, respectively. It can be found that GLCDE shows better performance than the four competitive approaches. In addition, the statistically significant results calculated by Wilcoxon test are given in Table II. For the 30-D problems, GLCDE obtains obviously better results than ZEPDE, SHADE, SinDE, and IDE on 19, 16, 17, and 15 out of 28 cases, respectively. ZEPDE, SHADE, SinDE, and IDE perform remarkably better than GLCDE on 3, 7, 8, and 6 cases, respectively. For the 50-D functions, GLCDE achieves significantly better results on 17, 18, 17, and 14 functions compared with ZEPDE, SHADE, SinDE, and IDE, respectively. ZEPDE, SHADE, SinDE, and IDE dynamically outperform GLCDE on 6, 7, 7, and 9 cases, respectively.

Furthermore, GLCDE is compared with *s*TDE-*d*R [40], UMDE [13], MVC_E_S_C [71], ETI-SHADE [72], and UMS-SHADE [31] over the CEC 2014 functions. All parameter settings of these competitors keep the same as their published papers. The detailed results for the 30-D and 50-D cases are

TABLE II
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND ZEPDE, SHADE, SINDE, AND IDE ON 30-D, AND 50-D CEC 2013 FUNCTIONS

GLCDE v.s.	$D = 30$			$D = 50$		
	$+$	$\approx$	$-$	$+$	$\approx$	$-$
ZEPDE	19	6	3	17	5	6
SHADE	16	5	7	18	3	7
SinDE	17	3	8	17	4	7
IDE	15	7	6	14	5	9

TABLE III
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND *s*TDE-*d*R, UMDE, MVC_E_S_C, ETI-SHADE, AND UMS-SHADE ON 30-D, AND 50-D CEC 2014 FUNCTIONS

GLCDE v.s.	$D = 30$			$D = 50$		
	$+$	$\approx$	$-$	$+$	$\approx$	$-$
*s*TDE-*d*R	17	10	3	15	6	9
UMDE	13	11	6	13	7	10
MVC_E_S_C	18	6	6	21	4	5
ETI-SHADE	22	6	2	25	3	2
UMS-SHADE	12	11	7	13	6	11

summarized in Tables S7 and S8 of the supplementary material, respectively. As seen, GLCDE gets lower mean values on most of the problems compared to the four competitive approaches. In addition, the results provided by Wilcoxon test are given in Table III. Compared to *s*TDE-*d*R, UMDE, MVC_E_S_C, ETI-SHADE, and UMS-SHADE, GLCDE provides significantly better performance on 17, 13, 18, 22, and 12 out of 30 functions for 30-D problems, respectively. The results of *s*TDE-*d*R, UMDE, MVC_E_S_C, ETI-SHADE, and UMS-SHADE are remarkably better than GLCDE on 3, 6, 6, 2, and 7 cases, respectively. For 50-D problems, *s*TDE-*d*R, UMDE, MVC_E_S_C, ETI-SHADE, and UMS-SHADE performs dynamically better than GLCDE on 9, 10, 5, 2, 11 cases, respectively. However, GLCDE obtains obviously better results than *s*TDE-*d*R, UMDE, MVC_E_S_C, ETI-SHADE, and UMS-SHADE on 15, 13, 21, 25, and 13 cases, respectively.

C. Comparison With CEC DE Winners

GLCDE is further compared with the DE winners of CEC 2014–2017 competitions. First, we compare GLCDE with five DE winners (i.e., LSHADE-EpSin [73], UMOEAII [74], [75], MC-SHADE [76], LSHADE-ND [77], L-SHADE [55], and iLSHADE [78]) in CEC 2014–2016 competitions over the CEC 2014 benchmark set. The parameters of these competitors are set on the basis of their published literatures. The results of 30-D and 50-D problems are displayed in Tables S9 and S10 of the supplementary material. Clearly, GLCDE attains the better or similar mean values on most of the cases when compared with the competitors. In addition, Table IV summarizes the significant results obtained by Wilcoxon test. Compared to LSHADE-EpSin, UMOEAII, MC-SHADE, LSHADE-ND, L-SHADE, and iLSHADE, GLCDE performs significantly better on 10, 12, 23, 19, 16, and 12 cases for 30-D functions, respectively. LSHADE-EpSin, UMOEAII, MC-SHADE, LSHADE-ND, L-SHADE, and iLSHADE obviously outperform GLCDE on 10, 10, 3, 1, 4, and 7 cases, respectively. For 50-D problems, LSHADE-EpSin, UMOEAII, MC-SHADE, LSHADE-ND, L-SHADE, and iLSHADE obtain remarkably

TABLE IV
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND LSHADE-EPSIN, UMOEAII, MC-SHADE, LSHADE-ND, L-SHADE, AND ILSHADE ON 30-D, AND 50-D CEC 2014 FUNCTIONS

GLCDE v.s.	$D = 30$			$D = 50$		
	$+$	$\approx$	$-$	$+$	$\approx$	$-$
LSHADE-EpSin	10	10	10	13	6	11
UMOEAII	12	8	10	13	6	11
MC-SHADE	23	4	3	26	2	2
LSHADE-ND	19	10	1	19	4	7
L-SHADE	16	10	4	23	3	4
iLSHADE	12	11	7	16	6	8

TABLE V
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND LSHADE-CNEPSIN, LSHADE_SPACMA, AND IDEBESTNSIZE ON 30-D, AND 50-D CEC 2017 FUNCTIONS

GLCDE v.s.	$D = 30$			$D = 50$		
	$+$	$\approx$	$-$	$+$	$\approx$	$-$
LSHADE-cnEpSin	13	6	10	14	5	10
LSHADE_SPACMA	13	6	10	12	7	10
IDEbestNsize	18	5	6	22	0	7

better performance than GLCDE on 11, 11, 2, 7, 4, and 8 cases, respectively. However, they are significantly worse than GLCDE on 13, 13, 26, 19, 23, and 16 cases, respectively.

In addition, the comparison between GLCDE and three top DE algorithms (i.e., LSHADE-cnEpSin [79], LSHADE_SPACMA [80], and IDEbestNsize [81]) in the CEC 2017 competition is conducted on the CEC 2017 test set. All parameters of these competitors are set in accordance with their published papers. The results of the 30-D and 50-D problems are given in Tables S11 and S12 of the supplementary material, respectively. Due to the numerical instability, the function F_2 is removed according to the suggestion in [69]. The data shows GLCDE attains better or similar performance compared with the three compared approaches. Furthermore, Table V reports the significant results achieved by Wilcoxon test. For 30-D functions, GLCDE performs obviously worse than LSHADE-cnEpSin, LSHADE_SPACMA, and IDEbestNsize on 10, 10, and 6 cases, respectively. But GLCDE gets markedly better performance than LSHADE-cnEpSin, LSHADE_SPACMA, and IDEbestNsize on 13, 13, and 18 cases, respectively. GLCDE keeps this remarkable advantage for 50-D functions. Specifically, GLCDE is dramatically better than LSHADE-cnEpSin, LSHADE_SPACMA, and IDEbestNsize on 14, 12, and 22 functions, respectively. However, LSHADE-cnEpSin, LSHADE_SPACMA, and IDEbestNsize significantly outperform GLCDE on 10, 10, and 7 cases, respectively.

D. Comparison With Surrogate-Based DE Variants

Similar to the underestimation model, the surrogate model [82] is usually integrate into EAs to reduce the function evaluations for the expensive-to-evaluate problems because it is much cheaper compared to the function evaluations [83]–[85]. Here, GLCDE is compared with five surrogate-assisted DE approaches, i.e., CSM-SHADE [86], GPEME [87], ESMDE [88], LLUDE [36], and UMS-SHADE [31], over the 30-D CEC 2013 test set. The parameters of these comparison approaches are set on the basis of their original papers. All algorithms are stopped when the number of function evaluations reaches 300 000.

TABLE VI
SIGNIFICANCE TEST RESULTS BETWEEN GLCDE AND CSM-SHADE, GPEME, ESMDE, LLUDE, AND UMS-SHADE ON 30-D CEC 2013 FUNCTIONS

GLCDE v.s.	CSM-SHADE	GPEME	ESMDE	LLUDE	UMS-SHADE
$+$	16	26	25	18	12
$\approx$	3	0	0	6	7
$-$	9	2	3	4	9

Table S13 of the supplementary material reports the mean and standard deviation of the function error for each problem. The data shows that the proposed GLCDE achieves better results with lower function errors compared to these five algorithms. Also, the significant test results between GLCDE and each comparison algorithm are summarized in Table VI. As shown in the table, the results provided by GLCDE is significantly better than CSM-SHADE, GPEME, ESMDE, LLUDE, and UMS-SHADE on 16, 26, 25, 18, and 12 out of 28 functions, respectively. However, CSM-SHADE, GPEME, ESMDE, LLUDE, and UMS-SHADE significantly outperform GLCDE only on 9, 2, 3, 4, and 9 cases, respectively. The comparison between GLCDE and these algorithms on the PSP problem will be discussed in Section VI-G.

E. Effects of GLCDE Components

GLCDE consists of four main components: 1) the adaptive underestimation model; 2) the underestimation-based global and local cooperative scheme; 3) the better-based mutation strategy; and 4) the parameter adaption of DE. In order to verify the effect of each component, various experiments are conducted on all classical benchmark functions at $D = 30$ in this section.

1) Adaptive Underestimation Model: The adaptive underestimation model is characterized by the adaption of the parameter M. Therefore, we first investigate the effect of the M adaption, then study the contribution of the whole underestimation model. GLCDE is first compared with the GLCDE using three fixed M, i.e., $M = 5000$, 10 000, and 15 000. These three GLCDE methods are, respectively, represented as GLCDE($M = 5000$), GLCDE($M = 10\,000$), and GLCDE($M = 15\,000$), and they utilize the same parameter settings with GLCDE for fair comparison. The detailed results achieved by these four GLCDE methods are listed in Table S14 of the supplementary material. It is observed that GLCDE using adaptive M is superior to the three GLCDE variants with the fixed M on most of cases. Additionally, the significant test results and Friedman rankings [50] given in Table VII also indicate GLCDE consistently performs better than the other three GLCDE variants and obtains the first ranking.

In GLCDE, the underestimation model is used to select the best candidate from multiple trial vectors. To identify the effect of the underestimation model, GLCDE is compared with two GLCDE variants, i.e., GLCDE-rand and GLCDE-FES. In both global and local phases of GLCDE-rand, only one strategy is randomly selected from the three mutation strategies to produce a trial individual for each target. In GLCDE-FES,

TABLE VII
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE VARIANTS WITH FIXED M

	GLCDE(M=5000)	GLCDE(M=10000)	GLCDE(M=15000)	GLCDE
+	18	13	16	-
≈	5	9	6	-
−	0	1	1	-
Ranking	3.35	2.24	2.91	1.50

TABLE VIII
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE-RAND, GLCDE-FES

	GLCDE-rand	GLCDE-FES	GLCDE
+	13	17	-
≈	8	5	-
−	2	1	-
Ranking	2.04	2.59	1.37

TABLE IX
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE VARIANTS WITH GLOBAL OR LOCAL PHASE ONLY

	GLCDE-global	GLCDE-local	GLCDE
+	15	14	-
≈	6	5	-
−	2	4	-
Ranking	2.59	1.96	1.46

three trial vectors are produced by the three different mutation strategies for each target in both global and local phases. But the trial vectors are filtered according to their function values rather than the underestimation values. The parameter settings of GLCDE-rand and GLCDE-FES are the same as those of GLCDE. The results of each function provided by these three methods are listed in Table S15 of the supplementary material. The data indicates that GLCDE exhibits better performance than GLCDE-rand and GLCDE-FES. Moreover, the Wilcoxon and Friedman results given in Table VIII also show that GLCDE is obviously better than GLCDE-rand and GLCDE-FES.

2) Global and Local Cooperation Scheme: The contribution of global and local cooperation scheme can be demonstrated by the comparison between GLCDE and two GLCDE variants, i.e., GLCDE-global and GLCDE-local. In GLCDE-global, only the global phase is performed for each generation, while only local phase is employed in GLCDE-local. These three algorithms utilize the same parameter settings. Table S16 of the supplementary material shows the detailed results. As seen, GLCDE achieves better results compared to GLCDE-global and GLCDE-local. Additionally, according to the significant test results and Friedman rankings displayed in Table IX, we can find that GLCDE significantly outperforms GLCDE-global and GLCDE-local on the majority of cases and obtains the best ranking.

3) Better-Based Mutation Strategy: In order to verify the contribution of the proposed better-based mutation strategy, GLCDE is compare with three GLCDE variants, i.e., GLCDE-RND, GLCDE-best, and GLCDE-better3. In GLCDE-RND, the three rand-based strategies (i.e., DE/rand/1, DE/rand/2, and DE/current-to-rand) employed in the global phase are also employed in the local phase, while the three best-based strategies (i.e., DE/best/1, DE/rand-to-best/1, and DE/current-to-best/1) are utilized in the local phase of GLCDE-best. In GLCDE-better3, only one mutation strategy (i.e., DE/rand-to-better/1) is used three times to generate three trial vectors for each target individual in the local phase. These four GLCDE algorithms use the same parameter settings. The detailed results of them are summarized in Table S17 of the supplementary material. Obviously, GLCDE obtains better performance than the other GLCDE variants on the majority of cases. From the statistically significant results and Friedman test results reported in Table X, GLCDE performs dramatically better than the three competitors and gets the best ranking.

TABLE X
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE-RND, GLCDE-BEST, AND GLCDE-BETTER3

	GLCDE-RND	GLCDE-best	GLCDE-better3	GLCDE
+	20	20	14	-
≈	3	2	7	-
−	0	1	2	-
Ranking	3.41	2.70	2.50	1.39

TABLE XI
FRIEDMAN TEST RESULTS OF DE WITH DIFFERENT STRATEGIES

	DE/better/1	DE/current-to-better/1	DE/rand-to-better/1
Ranking	3.89	2.57	4.43
	DE/current-to-$p$best/1	DE/centroid/2	DE/lbest/1
Ranking	3.74	2.83	3.54

TABLE XII
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE VARIANTS WITH FIXED F AND CR

	GLCDE(CR=0.1)	GLCDE(CR=0.5)	GLCDE(CR=0.9)	GLCDE
+	14	14	21	-
≈	6	4	1	-
−	3	5	1	-
Ranking	2.41	2.33	3.72	1.54

In addition, the proposed better-based mutation strategies are compared with the DE algorithms using DE/current-to-$p$best [42], DE/centroid/2 [13], and DE/lbest/1 [47]. For fair comparison, they employ the same parameter settings: $NP = 50$, $CR = 0.5$, and $F = 0.5$. The detailed results provided by these DE algorithms are displayed in Table S18 of the supplementary material. It can be found that DE/current-to-better/1 is better than the other mutation strategies on the majority of cases. Additionally, the results of Friedman test presented in Table XI indicates DE/current-to-better/1 achieves the best ranking, followed by DE/centroid/2, DE/lbest/1, DE/current-to-$p$best/1, DE/better/1, and DE/rand-to-better/1.

4) Parameter Adaption of DE: The effect of the parameter adaption can be studied by the comparison between GLCDE and its three variants with fixed settings of F and CR. According to the suggestion in [1] and [47], F is set as 0.5, and CR is set to 0.1, 0.5, and 0.9, respectively. These three GLCDE variants are, respectively, named as GLCDE($CR = 0.1$), GLCDE($CR = 0.5$), and GLCDE($CR = 0.9$). The detailed results attained by them are reported in Table S19 of the supplementary material. Clearly, GLCDE provides better results compared to the three GLCDE variants with fixed F and CR. Meanwhile, the results computed by Wilcoxon and Friedman test given in Table XII also reveals that GLCDE is the most effective one among these four GLCDE algorithms.

TABLE XIII
SIGNIFICANCE TEST RESULTS AND FRIEDMAN RANKINGS BETWEEN GLCDE AND GLCDE VARIANTS WITH ADVANCED PARAMETER ADAPTION TECHNIQUES

	GLC-SaDE	GLC-jDE	GLC-SHADE	GLC-JADE	GLCDE
$+$	12	17	13	13	-
$\approx$	5	4	5	4	-
$-$	6	2	5	6	-
Ranking	3.00	4.24	2.37	3.17	2.20

TABLE XIV
KRUSKAL–WALLIS TEST RESULTS AND FRIEDMAN RANKINGS FOR GLCDE WITH DIFFERENT COMBINATIONS OF THE COMPONENTS

	GLCDE1	GLCDE2	GLCDE3	GLCDE12	GLCDE13	GLCDE23	GLCDE
K_+	2	8	5	11	8	10	18
$K_\approx$	1	2	0	1	2	1	1
K_-	20	13	18	11	13	12	4
Ranking	6.48	4.22	4.91	3.07	4.24	2.96	2.13

TABLE XV
KRUSKAL–WALLIS TEST RESULTS AND FRIEDMAN RANKINGS FOR GLCDE WITH DIFFERENT POPULATION SIZE (*NP*)

	NP=30	NP=40	NP=50	NP=60	NP=80	NP=100
K_+	6	7	15	12	9	8
$K_\approx$	3	2	2	1	1	2
K_-	14	14	6	10	13	13
Ranking	3.76	3.17	2.35	3.17	4.15	4.39

TABLE XVI
KRUSKAL–WALLIS TEST RESULTS AND FRIEDMAN RANKINGS FOR GLCDE WITH DIFFERENT LG

	LG=20	LG=30	LG=40	LG=50	LG=60
K_+	16	14	7	4	3
$K_\approx$	2	3	3	1	1
K_-	4	6	12	17	18
Ranking	2.00	2.00	3.17	3.70	4.13

Furthermore, GLCDE is also compared with four GLCDE using the parameter adaption methods proposed in SaDE [16], jDE [50], SHADE [49], and JADE [42] to further verify the performance of our proposed parameter adaption approach. The four GLCDE algorithms are named as GLC-SaDE, GLC-jDE, GLC-SHADE, and GLC-JADE, respectively. Their parameters of the parameter adaption methods are set according to the corresponding publication. Table S20 of the supplementary material reports the results of each function obtained by the five GLCDE algorithms. It is clear that GLCDE achieves lower mean function errors than the four competitors on most of the cases. The Wilcoxon test results presented in Table XIII indicates that GLCDE attains the significantly better results on 12, 17, 13, and 13 functions compared to GLC-SaDE, GLC-jDE, GLC-SHADE, and GLC-JADE, respectively. On the basis of the Friedman test results, GLCDE also gets the best ranking, followed by GLC-SHADE, GLC-SaDE, GLC-JADE, and GLC-jDE.

5) Sensitivity Analysis: The sensitivity analysis is conducted to reveal the most crucial components of the proposed GLCDE. In this experiment, GLCDE is compared with different combinations of the three components (i.e., GLCDE1, GLCDE2, GLCDE3, GLCDE12, GLCDE13, and GLCDE23), where 1, 2, and 3 represents the adaptive underestimation model, better-based mutation strategies, and parameter adaptive scheme of DE, respectively. In these algorithms, the global and local cooperation scheme is still included. Table S21 of the supplementary material gives the results achieved by these seven algorithms for all functions. As seen, GLCDE which uses all components performs better than other algorithms using some of the components. The total results calculated by Kruskal–Wallis test [89] and Friedman test are summarized in Table XIV, where K_+ means that the algorithm obtains the best result among all algorithms, and K_- and $K_\approx$ indicate that the algorithm is significantly worse than and almost similar to the best algorithm, respectively. The results show that GLCDE obtains the best performance since it achieves the best results on 18 out of 23 functions and gets the best ranking. In addition, the results are obviously improved when the rest one component is added to GLCDE12, GLCDE13, and GLCDE23 (see Table S22 of the supplementary material). This indicates that each component play an important role in the proposed GLCDE. However, the contribution of the better-based mutation strategies may be larger than other components as the ranking of GLCDE13 are improved from 4.24 to 2.13 when it is combined to GLCDE.

F. Parameter Study

In this section, all 30-D classical benchmark functions are utilized to analyze the sensitivity of population size and LG.

1) Population Size: In order to investigate the impact of *NP*, six frequently used settings, i.e., 30, 40, 50, 60, 80, and 100, are employed in GLCDE. The rest parameter settings are the same as that described at the start of Section VI. Table S23 of the supplementary material reports the results of each function. It is clear that CLCDE with $NP = 50$ obtains better performance compared to GLCDE using other *NP* settings. Furthermore, the Kruskal–Wallis test results and Friedman rankings are summarized in Table XV. It indicates that $NP = 50$ achieves the best results on 15 out of 23 functions, and obtains the best ranking. From these data, we can conclude that $NP = 50$ is more suitable for GLCDE, although large *NP* will increase the computation complexity as described in Section IV-E.

2) Learning Generation: In this experiment, the influence of LG on the performance of GLCDE is studied. The parameter settings given in the beginning of Section VI are employed, except for LG, which varies from 20 to 60 with a step of 10 according to the suggestion in [16]. The mean and standard deviation of the function error for each function are listed in Table S24 of the supplementary material. The data shows that GLCDE with LG $= 20$ performs better than GLCDE using other LG values. Table XVI gives the results obtained by Kruskal–Wallis and Friedman tests. The Kruskal–Wallis test results reveal that LG $= 20$ attains the best results in 16 out of 23 functions. In addition, LG $= 20$ and LG $= 30$ get the same Friedman rankings which are better than other competitors. Therefore, LG $= 20$ is a better choice for GLCDE.

G. Algorithm Complexity

The algorithm complexity of the proposed GLCDE is evaluated according to the method in CEC competitions [67]. Table

TABLE XVII
ALGORITHM COMPLEXITY OF GLCDE

	T_0	T_1	$\widehat{T}_2$	$(\widehat{T}_2 - T_1)/T_0$
$D = 10$	0.087	0.511	3.875	38.667
$D = 30$		1.585	11.532	114.333
$D = 50$		2.495	18.394	182.747

XV lists the calculated algorithm complexity on $D = 10$, 30, and 50, where T_0 is running time of the following test problem:

```
for i=1 : 1000000
 x=0.55+(double)i; x=x+1; x=x./2; x=x*x;
 x=sqrt(x); x=log(x); x=exp(x); y=x/x;
end
```

T_1 is the computational time to run the test problem F_{14} in CEC 2013 [67] at D dimensions with 200 000 function evaluations, and T_2 represents the computational time for GLCDE to optimize F_{14} at D dimensions with 200 000 function evaluations. $\widehat{T}_2$ is the mean value of T_2 for 5 runs. Based on the data given in Table XVII, T_1 and $\widehat{T}_2$ increase linearly with the number of dimensions, and $(\widehat{T}_2 - T_1)/T_0$ grows linearly.

H. Performance on PSP

In this experiment, we compare GLCEA with CoDE [37], SaDE [16], UMS-CoDE [31], UMDE [13], and Rosetta [62] over ten nonredundant proteins with various lengths of the amino acid sequence. The parameters of these methods are kept the same as in the corresponding publications. The fragment length is set to 9 in each algorithm. In all DE variants, the mutation and crossover operations are performed by the fragment exchange between different conformations as described in Section V. Therefore, CoDE, SaDE, UMS-CoDE, and UMDE are, respectively, renamed as CoEA, SaEA, UMS-CoEA, and UMEA in this experiment. Each protein is predicted over 30 independent runs. For each run, the conformation with the lowest energy is considered as its predicted model, and the model with the lowest energy among the 30 runs is selected as the final model. In order to evaluate the prediction accuracy, the TM-score [90] and the root-mean-squared-deviation (RMSD) between the predicted and the native structures are calculated after the optimal rigid-body superposition of C_α atoms. The range of TM-score is $(0, 1]$, and the higher value is preferable while RMSD is opposite.

Tables S25–S27 of the supplementary material report the RMSD, TM-score, and energy of the final model predicted by the six algorithms within MaxFES = 300 000, respectively. It should be note that the results of the top six proteins differ from those of [31] because the final model is selected by the energy without using any information of native according to the rules of CASP [63]. As shown in the table, GLCEA achieves better models for the majority of proteins. From the average results reported in Table XVIII, the average RMSD, TM-score, and energy of GLCEA are 6.55 Å, 0.48, and -28.86, which are 22.44%, 20.00%, and 84.47% better than the best of the compared algorithms, respectively. The distribution of RMSD for all decoys generated in the prediction process is shown in Fig. S3 of the supplementary material. Fig. S4 of the supplementary material displays a comparison between the predicted model and the native structure on two representative cases.

TABLE XVIII
AVERAGE RESULTS OF ROSETTA, SAEA, COEA, UMEA, UMS-COEA, AND GLCEA WITHIN THE MAXFES

	Rosetta	SaEA	CoEA	UMEA	UMS-CoEA	GLCEA
RMSD	8.02	10.38	10.70	9.60	8.48	6.55
TM-score	0.40	0.31	0.33	0.36	0.36	0.48
Rosetta energy	-15.59	-2.03	-2.82	-4.40	-10.84	-28.76

TABLE XIX
AVERAGE RESULTS OF ROSETTA, SAEA, COEA, UMEA, UMS-COEA, AND GLCEA WITHIN THE RUNTIME

	Rosetta	SaEA	CoEA	UMEA	UMS-CoEA	GLCEA
RMSD	8.04	11.54	9.47	8.49	7.79	6.68
TM-score	0.43	0.32	0.35	0.39	0.38	0.46
Rosetta energy	-25.28	-1.10	-7.62	-17.82	-24.79	-33.55

In order to verify the efficiency of GLCEA, we further compare the final model of GLCEA obtained within 2 h (the average runtime required by Rosetta) with those of the five algorithms. The results for each algorithm on each protein are summarized in Tables S28–S30 of the supplementary material, respectively. It is clear that the structures predicted by GLCEA are better than that generated by the comparison algorithms on most of the proteins. Table XIX lists the average results of all proteins. The results indicate that GLCEA gets an average RMSD of 6.68 Å, which is 20.36% lower than that of the best comparison algorithms (8.04 Å). When considering TM-score, GLCEA is also the best algorithm among these six algorithms as it achieves the highest TM-score (0.46). The superior performance is attributed to GLCEA can generate conformations with lower energy (-33.55) compared to other algorithms.

The proposed GLCEA is further compared with CSM-SHADE, GPEME, ESMDE, LLUDE, and UMS-SHADE on the ten proteins. The parameters of these control methods are determined in the light of their published papers. Since the mutation and crossover operations of them are performed by the fragment exchange and fragment assembly, CSM-SHADE, ESMDE, LLUDE, and UMS-SHADE are called CSM-SHAEA, ESMEA, LLUEA, and UMS-SHAEA, respectively. All algorithms are stopped when the number of energy function evaluations reaches 300 000 for each independent run. Tables S31–S33 of the supplementary material shows the results of the final model on each protein, respectively. It is clear that the models predicted by GLCEA are better than the comparison algorithms for most of the cases. The average results of all proteins listed in Table XX reveal that GLCEA attains the lowest RMSD (6.55 Å). In terms of TM-score, the average result of GLCEA is 0.48, which is higher than all compared algorithms. The average energy of GLCEA (-28.76) is also lower than the compared methods.

The final models of GLCEA predicted within 2 h are also compared with those generated by CSM-SHAEA, GPEME, ESMEA, LLUEA, and UMS-SHAEA. The RMSD, TM-score, and energy of each test protein are given in Tables S34–S36 of the supplementary material, respectively. The data indicates that GLCEA models provide better results compared to the

TABLE XX
AVERAGE RESULTS OF CSM-SHAEA, GPEME, ESMEA, LLUEA, UMS-SHAEA, AND GLCEA WITHIN THE MAXFES

	CSM-SHAEA	GPEME	ESMEA	LLUEA	UMS-SHAEA	GLCEA
RMSD	7.72	8.36	9.40	9.24	7.06	6.55
TM-score	0.41	0.37	0.30	0.30	0.44	0.48
Rosetta energy	−16.49	−13.88	2.79	0.12	−19.92	−28.76

TABLE XXI
AVERAGE RESULTS OF CSM-SHAEA, GPEME, ESMEA, LLUEA, UMS-SHAEA, AND GLCEA WITHIN THE RUNTIME

	CSM-SHAEA	GPEME	ESMEA	LLUEA	UMS-SHAEA	GLCEA
RMSD	7.58	8.44	9.45	9.29	7.13	6.68
TM-score	0.40	0.35	0.30	0.32	0.43	0.46
Rosetta energy	−26.04	−19.68	−15.97	−17.57	−26.78	−33.55

control methods for the majority of proteins. Furthermore, the average results given in Table XXI reveal that GLCEA obtains an average TM-score of 0.46, which is the highest among these approaches and 6.5% higher than that of the best control method. Also, the average RMSD and energy of GLCEA are 6.68 Å and −33.55, which are obviously lower than that of the competitors.

VII. CONCLUSION

An improved DE, called GLCDE, is presented in this article to enhance the effectiveness and efficiency of DE. In GLCDE, two phases, the global exploration phase and the local exploitation phase are performed for each generation. The global phase is performed for each target individual by using multiple explorative mutation strategies, while in the local phase, the better-based mutation strategies which apply individuals better than the target individual are designed to refine all individuals accepted in the global phase. In both global and local phases, a set of trial vectors is produced by various mutation strategies and assessed by an adaptive underestimation model, in which the slope control parameter of the supporting functions is automatically adjusted to obtain an accurate underestimation. The global phase aims to locate the promising regions quickly, and the local phase helps the approach to enhance the convergence speed. A simple parameter adaption scheme is also designed to determine F and CR adaptively during the searching process. Moreover, we applied GLCDE to predicted the 3-D structure of the protein.

The performance of GLCDE is demonstrated by comparing with state-of-the-art DE variants, up-to-date DE methods, and the top DE algorithms in the CEC 2014–2017 competitions over the classical benchmark functions, CEC 2013, 2014, and 2017 test sets. The results indicate that GLCDE is obviously superior to or at least comparable with the competitors in the majority of cases. The effect of each components of GLCDE is also investigated by various experiments. In addition, GLCDE is utilized in the PSP problem, termed GLCEA, to verify the effectiveness and efficiency for the real-world application. The results show that the structures predicted by GLCEA are more accurate than those of the competitors because GLCEA can identify lower energy conformations.

The proposed GLCDE (or GLCEA) is successfully applied to the real-world problem with 324 (108×3, protein 1THX) dimensions. However, as discussed in Section IV-E, the runtime complexity of GLCDE depends on the dimension of the problem and the population size. The computational time required to construct the underestimation model will increase with the growth of the problem dimensionality and population size. Therefore, it is very important to simplify the approach to obtain an efficient and effective underestimation model for large-scale problems. Moreover, integrating the population reduction mechanisms, such as [40] and [56] into GLCDE will be an interesting direction for future research.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[2] V. Santucci, M. Baioletti, and A. Milani, "Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 682–694, Oct. 2016.
[3] G.-J. Zhang, X.-G. Zhou, X.-F. Yu, X.-H. Hao, and L. Yu, "Enhancing protein conformational space sampling using distance profile-guided differential evolution," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 6, pp. 1288–1301, Nov./Dec. 2017.
[4] P. P. Biswas, P. N. Suganthan, and G. A. J. Amaratunga, "Minimizing harmonic distortion in power system with optimal design of hybrid active power filter using differential evolution," *Appl. Soft Comput.*, vol. 61, pp. 486–496, Dec. 2017.
[5] X. Qiu, J.-X. Xu, Y. Xu, and K.-C. Tan, "A new differential evolution algorithm for minimax optimization in robust design," *IEEE Trans. Cybern.*, vol. 48, no. 5, pp. 1355–1368, May 2018.
[6] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.
[7] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.
[8] F. Neri and V. Tirronen, "Recent advances in differential evolution: A survey and experimental analysis," *Artif. Intell. Rev.*, vol. 33, nos. 1–2, pp. 61–106, Feb. 2010.
[9] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 99–119, Feb. 2011.
[10] W. Gong, Z. Cai, C. X. Ling, and H. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 2, pp. 397–413, Apr. 2011.
[11] W. Gong and Z. Cai, "Differential evolution with ranking-based mutation operators," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2066–2081, Dec. 2013.
[12] A. W. Mohamed and P. N. Suganthan, "Real-parameter unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation," *Soft Comput.*, vol. 22, no. 10, pp. 3215–3235, May 2018.
[13] X.-G. Zhou and G.-J. Zhang, "Abstract convex underestimation assisted multistage differential evolution," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2730–2741, Sep. 2017.
[14] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 5, pp. 601–614, Oct. 2012.
[15] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 560–574, Aug. 2015.
[16] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.
[17] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, Mar. 2011.
[18] W. Gong, A. Fialho, Z. Cai, and H. Li, "Adaptive strategy selection in differential evolution for numerical optimization: An empirical study," *Inf. Sci.*, vol. 181, no. 24, pp. 5364–5386, Dec. 2011.

[19] Q. Fan and X. Yan, "Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 219–232, Jan. 2016.

[20] M.-Y. Cheng and D.-H. Tran, "Two-phase differential evolution for the multiobjective optimization of time–cost tradeoffs in resource-constrained construction projects," *IEEE Trans. Eng. Manag.*, vol. 61, no. 3, pp. 450–461, Aug. 2014.

[21] Q. Fan and X. Yan, "Self-adaptive differential evolution algorithm with discrete mutation control parameters," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1551–1572, Feb. 2015.

[22] X.-G. Zhou, G.-J. Zhang, X.-H. Hao, L. Yu, and D.-W. Xu, "Differential evolution with multi-stage strategies for global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 2550–2557.

[23] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, New Orleans, LA, USA, 2011, pp. 1041–1048.

[24] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Multi-operator based evolutionary algorithms for solving constrained optimization problems," *Comput. Oper. Res.*, vol. 38, no. 12, pp. 1877–1896, Dec. 2011.

[25] W. Yi, L. Gao, and Y. Zhou, "A new differential evolution algorithm with a hybrid mutation operator and self-adapting control parameters for global optimization problems," *Appl. Intell.*, vol. 42, no. 4, pp. 642–660, Jun. 2015.

[26] X. Zhou, Z. Wu, H. Wang, and S. Rahnamayan, "Enhancing differential evolution with role assignment scheme," *Soft Comput.*, vol. 18, no. 11, pp. 2209–2225, 2014.

[27] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Inf. Sci.*, vol. 329, no. 2, pp. 329–345, Feb. 2016.

[28] J. Brest, B. Boskovic, A. Zamuda, I. Fister, and E. Mezura-Montes, "Real parameter single objective optimization using self-adaptive differential evolution algorithm with more strategies," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, Jun. 2013, pp. 337–383.

[29] A. M. Rubinov, "Abstract convexity: Examples and applications," *Optimization*, vol. 47, nos. 1–2, pp. 1–33, 2000.

[30] G. Beliakov, "Geometry and combinatorics of the cutting angle method," *Optimization*, vol. 52, nos. 4–5, pp. 379–394, Jul. 2003.

[31] X. G. Zhou and G. J. Zhang, "Differential evolution with underestimation-based multimutation strategy," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1353–1364, Apr. 2019.

[32] X. Qiu, K. C. Tan, and J.-X. Xu, "Multiple exponential recombination for differential evolution," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 995–1006, Apr. 2017.

[33] L. M. Batten and G. Beliakov, "Fast algorithm for the cutting angle method of global optimization," *J. Glob. Optim.*, vol. 24, no. 2, pp. 149–161, Oct. 2002.

[34] G. Beliakov, "Extended cutting angle method of global optimization," *Pac. J. Optim.*, vol. 4, no. 1, pp. 152–176, 2008.

[35] X.-G. Zhou, G.-J. Zhang, X.-H. Hao, and L. Yu, "A novel differential evolution algorithm using local abstract convex underestimate strategy for global optimization," *Comput. Oper. Res.*, vol. 75, no. 11, pp. 132–149, Nov. 2016.

[36] X.-G. Zhou, G.-J. Zhang, X.-H. Hao, D.-W. Xu, and L. Yu, "Enhanced differential evolution using local Lipschitz underestimate strategy for computationally expensive optimization problems," *Appl. Soft Comput.*, vol. 48, pp. 169–181, Nov. 2016.

[37] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, Feb. 2011.

[38] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "An improved self-adaptive differential evolution algorithm for optimization problems," *IEEE Trans. Ind. Inf.*, vol. 9, no. 1, pp. 89–99, Feb. 2013.

[39] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Balancing the exploration and exploitation capabilities of the differential evolution algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, Jun. 2008, pp. 2686–2693.

[40] M. Z. Ali, N. H. Awad, P. N. Suganthan, and R. G. Reynolds, "An adaptive multipopulation differential evolution with dynamic population reduction," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2768–2779, Sep. 2017.

[41] Q.-K. Pan, P. N. Suganthan, L. Wang, L. Gao, and R. Mallipeddi, "A differential evolution algorithm with self-adapting strategy and control parameters," *Comput. Oper. Res.*, vol. 38, no. 1, pp. 394–408, Jan. 2011.

[42] J. Q. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[43] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.

[44] L. Tang, Y. Zhao, and J. Liu, "An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 209–225, Apr. 2014.

[45] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 482–500, Apr. 2012.

[46] Y. Cai and J. Wang, "Differential evolution with neighborhood and direction information for numerical optimization," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2202–2215, Dec. 2013.

[47] W.-J. Yu *et al.*, "Differential evolution with two-level parameter adaptation," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1080–1099, Jul. 2014.

[48] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.

[49] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, 2013, pp. 71–78.

[50] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[51] V. A. Tatsis and K. E. Parsopoulos, "Differential evolution with grid-based parameter adaptation," *Soft Comput.*, vol. 21, no. 8, pp. 2105–2127, Apr. 2017.

[52] V. A. Tatsis and K. E. Parsopoulos, "Dynamic parameter adaptation in metaheuristics using gradient approximation and line search," *Appl. Soft Comput.*, vol. 74, pp. 368–384, Jan. 2019.

[53] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 689–707, Oct. 2014.

[54] K. C. Tan, T. H. Lee, and E. F. Khor, "Evolutionary algorithms with dynamic population size and local exploration for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 5, no. 6, pp. 565–588, Dec. 2001.

[55] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, Jun. 2014, pp. 1658–1665.

[56] N. H. Awad, M. Z. Ali, and P. N. Suganthan, "Ensemble of parameters in a sinusoidal differential evolution with niching-based population reduction," *Swarm Evol. Comput.*, vol. 39, pp. 141–156, Apr. 2018.

[57] C. Sun, Y. Jin, R. Cheng, J. Ding, and J. Zeng, "Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 644–660, Aug. 2017.

[58] M. Dorn, M. B. E. Silva, L. S. Buriol, and L. C. Lamb, "Three-dimensional protein structure prediction: Methods and computational strategies," *Comput. Biol. Chem.*, vol. 53, pp. 251–276, Dec. 2014.

[59] Y. Zhang, "Protein structure prediction: When is it useful?" *Current Opin. Struct. Biol.*, vol. 19, no. 2, pp. 145–155, Apr. 2009.

[60] C. B. Anfinsen, "Principles that govern the folding of protein chains," *Science*, vol. 181, no. 4096, pp. 223–230, 1973.

[61] Y. Zhang, "Progress and challenges in protein structure prediction," *Current Opin. Struct. Biol.*, vol. 18, no. 3, pp. 342–348, Jun. 2008.

[62] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker, "Protein structure prediction using Rosetta," *Method Enzymol.*, vol. 383, pp. 66–93, Dec. 2004.

[63] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, and A. Tramontano, "Critical assessment of methods of protein structure prediction (CASP)-round XII," *Proteins*, vol. 86, no. S1, pp. 7–15, 2018.

[64] D. Xu and Y. Zhang, "Toward optimal fragment generations for ab initio, protein structure assembly," *Proteins*, vol. 81, no. 2, pp. 229–239, Feb. 2013.

[65] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "A new genetic algorithm for solving optimization problems," *Eng. Appl. Artif. Intell.*, vol. 27, pp. 57–69, Jan. 2014.

[66] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 211–224, Jun. 2004.

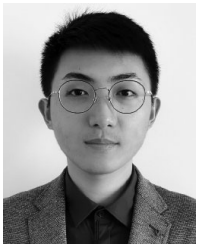
[67] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," Comput. Intell. Lab., Dept. Econ., Nanyang Technol. Univ., Singapore, Rep. 201212, 2013.
[68] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization," Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou, China, and Nanyang Technol. Univ., Singapore, Rep. 201311, Dec. 2013.
[69] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective bound constrained real-parameter numerical optimization," School Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Rep. 201611, Nov. 2016.
[70] A. Draa, S. Bouzoubia, and I. Boukhalfa, "A sinusoidal differential evolution algorithm for numerical optimisation," *Appl. Soft Comput.*, vol. 27, pp. 99–126, Feb. 2015.
[71] S. X. Zhang, S. Y. Zheng, and L. M. Zheng, "An efficient multiple variants coordination framework for differential evolution," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2780–2793, Sep. 2017.
[72] W. Du, S. Y. S. Leung, Y. Tang, and A. V. Vasilakos, "Differential evolution with event-triggered impulsive control," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 244–257, Jan. 2017.
[73] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, "An ensemble sinusoidal parameter adaptation incorporated with L-shade for solving CEC2014 benchmark problems," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 2958–2965.
[74] S. M. Elsayed, R. A. Sarker, D. L. Essam, and N. M. Hamza, "Testing united multi-operator evolutionary algorithms on the CEC2014 real-parameter numerical optimization," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 1650–1657.
[75] S. M. Elsayed, N. M. Hamza, and R. A. Sarker, "Testing united multi-operator evolutionary algorithms—II on single objective optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 2966–2973.
[76] A. Viktorin, M. Pluhacek, and R. Senkerik, "Success-history based adaptive differential evolution algorithm with multi-chaotic framework for parent selection performance on CEC2014 benchmark set," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 4797–4803.
[77] K. M. Sallam, R. A. Sarker, D. L. Essam, and S. M. Elsayed, "Neurodynamic differential evolution algorithm and solving CEC2015 competition problems," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 1033–1040.
[78] J. Brest, M. S. Maučec, and B. Bošković, "iL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 1188–1195.
[79] N. H. Awad, M. Z. Ali, and P. N. Suganthan, "Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems," in *Proc. IEEE Congr. Evol. Comput.*, Donostia-San Sebastián, Spain, 2017, pp. 372–379.
[80] 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. IEEE Congr. Evol. Comput.*, Donostia-San Sebastián, Spain, 2017, pp. 145–152.
[81] P. Bujok and J. Tvrdík, "Enhanced individual-dependent differential evolution with population size adaptation," in *Proc. IEEE Congr. Evol. Comput.*, Donostia-San Sebastián, Spain, 2017, pp. 1358–1365.
[82] L. Pan, C. He, Y. Tian, H. Wang, X. Zhang, and Y. Jin, "A classification based surrogate-assisted evolutionary algorithm for expensive many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 74–88, Feb. 2019.
[83] N. H. Awad, M. Z. Ali, R. Mallipeddi, and P. N. Suganthan, "An improved differential evolution algorithm using efficient adapted surrogate model for numerical optimization," *Inf. Sci.*, vols. 451–452, pp. 326–347, Jul. 2018.
[84] Y. Wang, D.-Q. Yin, S. Yang, and G. Sun, "Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints," *IEEE Trans. Cybern.*, vol. 49, no. 5, pp. 1642–1656, May 2019.
[85] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2664–2677, Sep. 2017.
[86] W. Gong, A. Zhou, and Z. Cai, "A multioperator search strategy based on cheap surrogate models for evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 746–758, Oct. 2015.
[87] B. Liu, Q. Zhang, and G. G. E. Gielen, "A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 180–192, Apr. 2014.
[88] R. Mallipeddi and M. Lee, "An evolving surrogate model-based differential evolution algorithm," *Appl. Soft Comput.*, vol. 34, pp. 770–787, Sep. 2015.
[89] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *J. Amer. Stat. Assoc.*, vol. 47, no. 260, pp. 583–621, 1952.
[90] Y. Zhang and J. Skolnick, "Scoring function for automated assessment of protein structure template quality," *Proteins*, vol. 57, no. 4, pp. 702–710, Dec. 2004.

Xiao-Gen Zhou received the Ph.D. degree in control science and engineering from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China, in 2018.

He is currently a Post-Doctoral Fellow with the Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA. His current research interests include intelligent information processing, optimization theory and algorithm design, and bioinformatics.

Chun-Xiang Peng is currently pursuing the Ph.D. degree in control science and engineering from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

His current research interests include intelligent information processing, intelligent optimization, and bioinformatics.



Jun Liu is currently pursing the Ph.D. degree in control science and engineering from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

His current research interests include intelligent information processing, intelligent optimization, and bioinformatics.

Yang Zhang received the Ph.D. degree in physics from the Institute of Particle Physics, Central China Normal University, Wuhan, China, in 1996.

From 1996 to 1998, he acted as an Alexander von Humboldt Fellow with the Physics Department, Free University Berlin, Berlin, Germany. He is a Professor with the Department of Computational Medicine and Bioinformatics and Department of Biological Chemistry, University of Michigan, Ann Arbor, MI, USA. His current research interests include lab is in protein design, protein folding, and protein structure prediction. The I-TASSER developed by his lab was ranked as one of the best methods for automated protein structure prediction in the past decade of the worldwide CASP competitions.

Prof. Zhang was a recipient of the U.S. National Science Foundation Career Award, the Alfred P. Sloan Award, and the Dean's Basic Science Research Award, and was selected as the Thomson Reuters Highly Cited Researcher in 2015–2018.

Gui-Jun Zhang received the Ph.D. degree in control theory and control engineering from Shanghai Jiaotong University, Shanghai, China, in 2004.

He is currently a Professor with the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China. His current research interests include intelligent information processing, optimization theory and algorithm design, and bioinformatics.