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.

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 zooming 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 multiparameter [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:

\[
\text{Minimize } f(x), \quad x \in \Omega \tag{1}
\]

where \( \Omega \) represents the feasible region of the search space and \( x = (x_1, x_2, \ldots, x_D)^T \) is a \( D \)-dimensional decision vector. Each variable \( x_i \) 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 \( i \)-th individual in the \( g \)-th generation as \( x_i^g = (x_{i,1}^g, x_{i,2}^g, \ldots, x_{i,D}^g) \), and its initial value at \( g = 0 \) can be generated by

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

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

2) Mutation: Each individual \( x_i^g \) in the current generation is regarded as the target vector, and a new individual called mutant vector can be created by

\[
v_i^g = x_i^{g+1} + F (x_{r_2}^g - x_{r_3}^g) \tag{3}
\]

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 \( u_i^g \). 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. \( j_{\text{rand}} \) is a random integer selected from \([1, 2, \ldots, D]\).

4) Selection: The selection operator picks the better one from \( x_i^{g+1} \) and \( u_i^g \) to enter the next generation. It can be described as

\[
x_{i}^{g+1} = \begin{cases} u_i^g, & \text{if } f(u_i^g) \leq f(x_i^{g+1}) \\ x_i^{g+1}, & \text{otherwise} \end{cases} \tag{5}
\]

B. Underestimation Model

Abstract convex analysis indicates that every nonconvex function is the upper envelope of its affine minorsants [30], [33]. With use of subdifferential-based [34] supporting functions to replace the affine minorsants, 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:

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

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].
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 exploitative mutation strategies to balance their effects. Ali et al. [40] proposed an adaptive DE with dynamic population reduction (sDE-dR), 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-pbest/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-pbest/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 $r_1$ and $r_2$. 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

As shown in Fig. 1(a), after calculating the supporting functions of all individuals, we can obtain $NP$ estimation values $h_i^x(x)$, $i = 1, 2, \ldots, NP$ 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:

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

![Image](https://via.placeholder.com/273x658)

**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.
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 individual. However, the candidate for each trial individual. However, the best one with the lowest underestimation value is chosen as the candidate for each trial individual. The underestimation value of \( u^i \) can be calculated by

\[
U(u^i) = \max(h_1^R(u^i), h_2^R(u^i))
\]

where \( h_1^R(u^i) \) and \( h_2^R(u^i) \) are the estimation values of \( u^i \) calculated by the supporting functions of \( x^a \) and \( x^b \), 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, \( U \) used to control the slope of the supporting functions (6) significantly influences the accuracy of the underestimation. For example, the underestimation models with \( M = 6.5 \) or \( M = 15 \) 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 \( M = 15 \), 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.

For each trial individual \( u^i \) in the initial population, a slope control parameter \( M_i \) derived from all trial individuals is considered as the initial value \( M_0 \). During the evolutionary process, \( M \) is adaptively updated according to (3). In the local phase, for each individual, \( M \) is continuously reduced according to a linear function calculated by the number of evaluated trial individuals. All trial individuals that are better than the target vector, then the candidate for each trial individual. However, the candidate for each trial individual. However, the best one with the lowest underestimation value is chosen as the candidate for each trial individual. The underestimation value of \( U(u^i) \) can be calculated by

\[
U(u^i) = \max(h_1^R(u^i), h_2^R(u^i))
\]

where \( h_1^R(u^i) \) and \( h_2^R(u^i) \) are the estimation values of \( u^i \) calculated by the supporting functions of \( x^a \) and \( x^b \), 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 \( M = 6.5 \) or \( M = 15 \) 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 \( M = 15 \), 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 \( M = 10000 \) is preferable for all benchmark functions. However, due to the different landscapes of different problems, different \( M \) may

![Fig. 2. Underestimation curves with different values of the slope control parameter \( M \) for a 1-D function.](image-url)
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 \( u_{i}^{gold} \) 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 \( i \)-th individual \( x_{i}^{g} \) in the current population can be described as follows:

\[
\begin{align*}
&h_{i}^{g}(u_{i}^{gold}) = \min_{j \in J} \left( f(x_{j}^{g}) - M \left( x_{ij}^{g} - u_{ij}^{gold} \right) \right) \\
&= f(x_{i}^{g}) - M \max_{j \in J} \left( x_{ij}^{g} - u_{ij}^{gold} \right).
\end{align*}
\]

According to (9), \( M \) can be calculated by

\[
M = \frac{f(x_{i}^{g}) - h^{g}(u_{i}^{gold})}{\max_{j \in J} \left( x_{ij}^{g} - u_{ij}^{gold} \right)}.
\]

(10)

Suppose that the underestimation value of \( u_{i}^{gold} \) is equal to the corresponding function value (i.e., \( U(u_{i}^{gold}) = f(u_{i}^{gold}) \)) when the underestimation model is sufficiently accurate. As shown in (8), the underestimation value \( U(u_{i}^{gold}) \) can be computed by the supporting functions of either \( x_{i}^{g} \) or that of \( x_{i}^{g} \). If \( h_{i}^{g}(u_{i}^{gold}) > h_{i}^{g}(x_{i}^{g}) \), that is, \( U(u_{i}^{gold}) = h_{i}^{g}(x_{i}^{g}) \), a slope control parameter \( M_{s}^{g} \) can be computed according to (10) by replacing \( h_{i}^{g}(u_{i}^{gold}) \) with \( f(x_{i}^{g}) \). Otherwise, \( M_{s}^{g} \) is calculated. The larger value between \( M_{s}^{g} \) and \( M_{s}^{g} \) is considered as the slope control parameter \( M_{s}^{g} \) derived from \( u_{i}^{gold} \), namely

\[
M_{s}^{g} = \max_{i=1,\ldots,N} \left( \frac{f(x_{i}^{g}) - f(u_{i}^{gold})}{\max_{j \in J} \left( x_{ij}^{g} - u_{ij}^{gold} \right)} \right).
\]

(11)

Note that the absolute values of the numerator and the denominator should be 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 \( M_{0}^{i}, i = 1, 2, \ldots, NP \) of the slope control parameter according to (11). Then the initial value \( M_{0} \) of \( M \) can be determined as follows:

\[
M_{0} = \max_{i=1,\ldots,NP} M_{0}^{i}.
\]

(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:

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

(13)

where \( M_{\text{max}}^{g} = \max_{i=1,\ldots,N} M^{g}_{i} \). It should be noted 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 exploitation 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) \( \text{DE/better/1} \):

\[
v_{ij}^{g} = x_{ij}^{\text{better}} + F \cdot (x_{ij}^{g} - x_{ij}^{\text{better}}),
\]

(14)

2) \( \text{DE/current-to-better/1} \):

\[
v_{ij}^{g} = x_{ij}^{g} + F \cdot (x_{ij}^{\text{better}} - x_{ij}^{g}) + F \cdot (x_{ij}^{g} - x_{ij}^{\text{better}}),
\]

(15)

3) \( \text{DE/rand-to-better/1} \):

\[
v_{ij}^{g} = x_{ij}^{g} + F \cdot (x_{ij}^{\text{better}} - x_{ij}^{g}) + F \cdot (x_{ij}^{\text{better}} - x_{ij}^{g}),
\]

(16)

where \( x_{ij}^{\text{better}} \) is an individual randomly selected from all individuals better than \( x_{ij}^{g} \). If \( x_{ij}^{g}, x_{ij}^{\text{better}}, x_{ij}^{\text{better}}, x_{ij}^{\text{better}} \) are three mutually different individuals randomly chosen from the entire population, and none of them are equal to \( x_{ij}^{g} \) or \( x_{ij}^{\text{better}} \). For the best individual, \( x_{ij}^{\text{better}} \) is replaced by a random individual of the entire population when using 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 exploitation 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 $p^G_k$ 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 $k$th interval and the number of trial individuals produced by the $CR$ in the $k$th interval that can successfully replace the target individual are recorded as $NT^G_k$ and $NS^G_k$, 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 $k$th interval, the selection probability is calculated by

$$p^G_k = \frac{\sum_{G=g-LG}^{g-1} SR^G_k}{\sum_{i=1}^{K} \sum_{G=g-LG}^{g-1} SR^G_i}$$

(17)

where

$$SR^G_k = \frac{NS^G_k}{NT^G_k} + \epsilon$$

(18)

is the success rate of the $k$th interval in the $G$th 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], 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 $O(NP^2 \cdot D \cdot G_{\text{max}})$ over $G_{\text{max}}$ 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
Based on fragments exchange between different conformations. For example, in DE/rand/1, three different conformations \( x_{g1} \), \( x_{g2} \), and \( x_{g3} \) are randomly selected from the current population, and the mutation conformation \( v_{g} \) is created by the replacement of two random residue positions in the third conformation \( x_{g3} \) with the corresponding fragments from the first conformation \( x_{g1} \) and the second conformation \( x_{g2} \), respectively. However, in DE/better/1, a conformation \( x_{gbetter} \) which have lower energy than the target conformation is first picked up from the conformations. Then, two different conformations \( x_{gb} \) and \( x_{gb} \) which also differ from \( x_{gbetter} \) and the target conformation \( x_{gbetter} \) are randomly chosen from the population. Two fragments are randomly extracted from \( x_{gbetter} \) and \( x_{gbetter} \) to replace the corresponding fragments in \( x_{gbetter} \), respectively. The conformation is evaluated by the Rosetta score3 energy function [62].

After the mutation conformation 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 \( x_{gbetter} \) to replace the corresponding position in the mutation conformation \( v_{gbetter} \). 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 \( C_\alpha \) 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 \( f(x) - f(x^*) \) obtained within the maximum function evaluations (MaxFES) are recorded to evaluate the performance, where \( x \) represents the best solution found within the MaxFES in a single run and \( x^* \) 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 × D, 3000 × D, and 3000 × 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 × 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
UMDE on 19, 18, 17, 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 sTDE-dR [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 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 sTDE-dR, 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 sTDE-dR, UMDE, MVC_E_S_C, ETI-SHADE, and UMS-SHADE are remarkably better than GLCDE on 3, 6, 2, and 7 cases, respectively. For 50-D problems, sTDE-dR, 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 sTDE-dR, 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, UMÖEAII, MC-SHADE, LSHADE-NDO, L-SHADE, AND LSHADE ON 30-D, AND 50-D CEC 2014 FUNCTIONS

<table>
<thead>
<tr>
<th>GLCDE v.s.</th>
<th>$D = 30$</th>
<th>$D = 50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSHADE-EpSin</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>UMÖEAII</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>MC-SHADE</td>
<td>23</td>
<td>4</td>
</tr>
<tr>
<td>LSHADE-NDO</td>
<td>19</td>
<td>10</td>
</tr>
<tr>
<td>L-SHADE</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>LSHADE</td>
<td>12</td>
<td>11</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GLCDE v.s.</th>
<th>$D = 30$</th>
<th>$D = 50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSHADE-cnEpSin</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>LSHADE_SPACMA</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>IDEbestNsize</td>
<td>18</td>
<td>5</td>
</tr>
</tbody>
</table>

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 according to 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 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,
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 reveal that GLCDE is the most effective one among these four GLCDE algorithms.

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. From the statistically significant results and Friedman test results presented in Table X, GLCDE performs dramatically better than the three competitors and gets the best ranking.

In addition, the proposed better-based mutation strategies are compared with the DE algorithms using DE/current-to-pbest [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-pbest/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 ADAPTATION TECHNIQUES

<table>
<thead>
<tr>
<th>GLC-SaDE</th>
<th>GLC-JDE</th>
<th>GLC-SHADE</th>
<th>GLC-JADE</th>
<th>GLCDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>12</td>
<td>17</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>≈</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Ranking</td>
<td>3.00</td>
<td>4.24</td>
<td>2.37</td>
<td>3.17</td>
</tr>
</tbody>
</table>

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, 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 \( K^+ \), \( K^- \), and \( K_{\infty} \) indicate that the algorithm obtains the best result among all algorithms, and \( K^+ \) and \( K^- \) indicate that the algorithm is significantly worse than and almost similar to the best algorithm, respectively. The results show that GLCDE achieves 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 is improved from 4.24 to 2.13 when it is combined to GLCDE.

<table>
<thead>
<tr>
<th>GLCDE1</th>
<th>GLCDE2</th>
<th>GLCDE3</th>
<th>GLCDE12</th>
<th>GLCDE13</th>
<th>GLCDE23</th>
<th>GLCDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K^+ )</td>
<td>2</td>
<td>6</td>
<td>5</td>
<td>11</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>( K^- )</td>
<td>20</td>
<td>13</td>
<td>18</td>
<td>11</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>Ranking</td>
<td>6.48</td>
<td>4.22</td>
<td>4.91</td>
<td>3.07</td>
<td>4.24</td>
<td>2.96</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GLCDE12</th>
<th>GLCDE13</th>
<th>GLCDE23</th>
<th>GLCDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K^+ )</td>
<td>2</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>( K^- )</td>
<td>13</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>Ranking</td>
<td>6.48</td>
<td>4.24</td>
<td>2.96</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>NP=30</th>
<th>NP=40</th>
<th>NP=50</th>
<th>NP=60</th>
<th>NP=80</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K^+ )</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>( K^- )</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>( K_{\infty} )</td>
<td>14</td>
<td>14</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Ranking</td>
<td>3.76</td>
<td>3.17</td>
<td>2.35</td>
<td>3.17</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>LG=20</th>
<th>LG=30</th>
<th>LG=40</th>
<th>LG=50</th>
<th>LG=60</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K^+ )</td>
<td>16</td>
<td>14</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>( K^- )</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>( K_{\infty} )</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>Ranking</td>
<td>2.00</td>
<td>2.00</td>
<td>3.17</td>
<td>3.70</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>NP=30</th>
<th>NP=40</th>
<th>NP=50</th>
<th>NP=60</th>
<th>NP=80</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K^+ )</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>( K^- )</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>( K_{\infty} )</td>
<td>14</td>
<td>14</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Ranking</td>
<td>3.76</td>
<td>3.17</td>
<td>2.35</td>
<td>3.17</td>
</tr>
</tbody>
</table>

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 GLCDE 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 XVII

<table>
<thead>
<tr>
<th>ALGORITHM COMPLEXITY OF GLCDE</th>
<th>$T_0$</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$(T_2 - T_1)/T_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D = 10$</td>
<td>0.511</td>
<td>3.875</td>
<td>38.667</td>
<td></td>
</tr>
<tr>
<td>$D = 30$</td>
<td>0.087</td>
<td>1.585</td>
<td>11.532</td>
<td>114.333</td>
</tr>
<tr>
<td>$D = 50$</td>
<td>2.495</td>
<td>18.394</td>
<td>182.747</td>
<td></td>
</tr>
</tbody>
</table>

TABLE XVIII

<table>
<thead>
<tr>
<th>AVERAGE RESULTS OF ROSETTA, SAEA, COEA, UMEA, UMS-COEA, AND GLCEA WITHIN THE MAXFES</th>
<th>Rosetta</th>
<th>SaEA</th>
<th>CoEA</th>
<th>UMEA</th>
<th>UMS-COEA</th>
<th>GLCEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSD</td>
<td>8.02</td>
<td>10.38</td>
<td>10.70</td>
<td>9.60</td>
<td>8.48</td>
<td>6.55</td>
</tr>
<tr>
<td>TM-score</td>
<td>0.40</td>
<td>0.31</td>
<td>0.33</td>
<td>0.36</td>
<td>0.36</td>
<td>0.48</td>
</tr>
<tr>
<td>Rosetta energy</td>
<td>−15.59</td>
<td>−2.03</td>
<td>−2.82</td>
<td>−4.40</td>
<td>−10.84</td>
<td>−28.76</td>
</tr>
</tbody>
</table>

TABLE XIX

<table>
<thead>
<tr>
<th>AVERAGE RESULTS OF ROSETTA, SAEA, COEA, UMEA, UMS-COEA, AND GLCEA WITHIN THE RUNTIME</th>
<th>Rosetta</th>
<th>SaEA</th>
<th>CoEA</th>
<th>UMEA</th>
<th>UMS-COEA</th>
<th>GLCEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSD</td>
<td>8.04</td>
<td>11.34</td>
<td>9.47</td>
<td>8.49</td>
<td>7.79</td>
<td>6.68</td>
</tr>
<tr>
<td>TM-score</td>
<td>0.43</td>
<td>0.32</td>
<td>0.35</td>
<td>0.39</td>
<td>0.38</td>
<td>0.46</td>
</tr>
<tr>
<td>Rosetta energy</td>
<td>−25.28</td>
<td>−1.10</td>
<td>−7.62</td>
<td>−17.82</td>
<td>−24.79</td>
<td>−33.55</td>
</tr>
</tbody>
</table>

XV lists the calculated algorithm complexity on $D = 10$, 30, and 50, where $T_0$ is running time of the following test problem: \[
\begin{align*}
&\text{for } i = 1 : 1000000 \\
&x = 0.55 + (\text{double})i; x = x + 1; x = x / 2; x = x \times x; \\
&x = \text{sqrt}(x); x = \log(x); x = \exp(x); y = x / x; \\
&\text{end}
\end{align*}
\]

$T_1$ 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. $\hat{T}_2$ is the mean value of $T_2$ for 5 runs. Based on the data given in Table XVII, $T_1$ and $\hat{T}_2$ increase linearly with the number of dimensions, and $(\hat{T}_2 - T_1)/T_0$ 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 $C_{\alpha}$ 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 noted 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.

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, ESMDE, LLUDE, 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, ESMDE, LLUDE, 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
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.

TABLE XX

<table>
<thead>
<tr>
<th></th>
<th>CSM-SHAEA</th>
<th>GPEME</th>
<th>ESMEA</th>
<th>LLUEA</th>
<th>UMS-SHAEA</th>
<th>GLCEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSD</td>
<td>7.22</td>
<td>8.36</td>
<td>9.40</td>
<td>7.24</td>
<td>7.06</td>
<td>6.65</td>
</tr>
<tr>
<td>TM-score</td>
<td>0.41</td>
<td>0.37</td>
<td>0.30</td>
<td>0.30</td>
<td>0.44</td>
<td>0.48</td>
</tr>
<tr>
<td>Rosetta energy</td>
<td>−14.69</td>
<td>−13.68</td>
<td>2.79</td>
<td>0.12</td>
<td>−19.92</td>
<td>−28.76</td>
</tr>
</tbody>
</table>

TABLE XXI

<table>
<thead>
<tr>
<th></th>
<th>CSM-SHAEA</th>
<th>GPEME</th>
<th>ESMEA</th>
<th>LLUEA</th>
<th>UMS-SHAEA</th>
<th>GLCEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSD</td>
<td>7.18</td>
<td>8.44</td>
<td>9.45</td>
<td>9.29</td>
<td>7.13</td>
<td>6.68</td>
</tr>
<tr>
<td>TM-score</td>
<td>0.40</td>
<td>0.35</td>
<td>0.30</td>
<td>0.32</td>
<td>0.45</td>
<td>0.46</td>
</tr>
<tr>
<td>Rosetta energy</td>
<td>−26.04</td>
<td>−19.68</td>
<td>−15.97</td>
<td>−17.57</td>
<td>−26.78</td>
<td>−34.55</td>
</tr>
</tbody>
</table>

REFERENCES


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 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.

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-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 Jiao tong 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.