IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 3, 2021, accepted September 29, 2021, date of publication October 13, 2021,
date of current version November 4, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3119616

Adaptive Differential Evolution With Information Entropy-Based Mutation Strategy

LIUJING WANG[1], XIAOGEN ZHOU[2], TENGYU XIE[1], JUN LIU[1], AND GUIJUN ZHANG[1]
[1]College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China
[2]Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI 48109, USA

Corresponding author: Guijun Zhang (zgj@zjut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61773346 and Grant 62173304, and in part by the Key Program of Natural Science Foundation of Zhejiang Province of China under Grant LZ20F030002.

ABSTRACT In order to balance the exploration and exploitation ability of differential evolution (DE), different mutation strategy for different evolutionary stages may be effective. An adaptive differential evolution with information entropy-based mutation strategy (DEIE) is proposed to divide the evolutionary process reasonably. In DEIE, the number of Markov states deduced from the crowding strategy is determined first and then the transition matrix between states is inferred from the historical evolutionary information. Based on the above-mentioned knowledge, the Markov state model is constructed. The evolutionary process is divided into exploration and exploitation stages dynamically using the information entropy derived from the Markov state model. Consequently, stage-specific mutation operation is employed adaptively. Experiments are conducted on CEC 2013, 2014, and 2017 benchmark sets and classical benchmark functions to assess the performance of DEIE. Moreover, the proposed approach is also used to solve the protein structure prediction problem efficiently.

INDEX TERMS Differential evolution, information entropy, mutation strategy, evolutionary stages, Markov state model.

I. INTRODUCTION

Differential evolution (DE), proposed by Storn and Price [1], is a competitive and popular population-based stochastic search algorithm. DE and its variants have made remarkable contribution to solving complex optimization problems [2]. Similar to other evolutionary algorithms, DE consists of three operations, i.e., mutation, crossover, and selection. The difference vectors of DE have adaptability for perturbation to the natural scales of the objective landscape in a random process [3]. This self-referential mutation provides DE with a tremendous speed advantage at the early stage. However, this property makes DE sensitive to the loss of diversity, then resulting in poor exploitation at the later evolutionary stage [4]. In terms of the mutation operator, various mutation strategies show distinct advantages in DE. Inappropriate mutation strategies may cause stagnation due to overexploration or premature convergence because of over exploitation [5].

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

Hence, how to balance exploration and exploitation is still an open issue in the evolutionary computation community. For exploration and exploitation conundrum in DE, it may be feasible to divide the evolutionary process into different stages for the balance between exploration and exploitation. Many approaches have been developed to improve the performance of DE by the division of evolutionary stages. For example, a fixed number of iterations are used as a division criterion, such as two stages proposed by Liu *et al.* [6] and three stages introduced by Cheng and Tran [7]. Subsequently, suitable mutation strategies are used in each stage. Although the performance of DE may be improved, empirical guidelines are sometimes unreliable and lack universality. Yu *et al.* [8] focused on new metrics that represent the relationship between the order of fitness value and distance to divide evolutionary process into two stages with corresponding parameter adjustment mechanisms and strategies. Tang *et al.* [5] presented a variant with individual-dependent mechanism, in which the search process is separated into two stages to design mutation strategy for specific stages. The algorithm enters the later stage according to the defined success rate. Fan and Yan [9] introduced a self-adaptive DE

called ZEPDE. After DE/rand/1 is employed at the first stage, the mutation strategy with zoning evolution is assigned to each individual in accordance with the selective probability at the second stage. Zhan *et al.* [10] proposed master-slave multipopulation distributed framework, three populations are co-evolved which different populations adaptively choose their suitable mutation strategies based on the evolutionary state estimation. The evolutionary state is estimated into two states by distance computations between two individuals with the best fitness value and the median fitness value. Li and Li [11] designed an evolutionary state estimation method based on the correlation coefficient between the population distribution in objective space and solution space. Then, the evolutionary process is divided into three kinds of state. Zhou and Zhang [12] proposed the underestimate model based on abstract convex theory, in which the variation in the average estimation error is used to divide the evolutionary process into three stages with corresponding strategy candidate pool.

Instead of being divided into irreversible multistage, it is better to dynamically divide the evolutionary stage based on the search behaviour of population. Therefore, better understanding is needed of the population dynamics. In the past few years, entropy was utilized as an evaluation criterion to measure certain properties of population or evolutionary process as follows. Based on the diversity defined by the genotypic and phenotypic entropies, Naghib and Nobakhti [13] designed a fully adaptive DE with the adaptive rule of the parameters. Wu *et al.* [14] proposed a diversity metric based on the crowding entropy to sustain the diversity of Pareto optimality. Similarly, Zhang *et al.* [15] utilized entropy diversity method to adaptively monitor population diversity. Chen *et al.* [16] coupled DE algorithm with entropy to solve multi-mode resource constrained project scheduling. Entropy based on activity durations is used as a measure of uncertainty to ensure the feasibility of the project despite the existence of unexpected events. Ali *et al.* [17] proposed a multi-level thresholding achieved by integrating the DE algorithm and Kapur entropy into image segmentation.

The motivation behind this research is to proposes an adaptive differential evolution with information entropy-based mutation strategy (DEIE), which realize a dynamic division of the evolutionary stages based on entropy and stage-specific mutation strategies adaption to obtain the trade-off between exploration and exploitation. To be specific, the Markov states are obtained in our method, and Markov state model is constructed by using the historical evolutionary information across generations to describe the frequency of state transition. Subsequently, the information entropy metric is proposed to estimate the extent that the population explores the solution space, which is mainly used for the dynamic division of the evolutionary stages. Then, the stage-specific mutation strategies are adopted to take their advantage based on exploration or exploitation stage. Compared to other DE variants, the contributions of this paper are: (1) Dynamic division of evolutionary stages of DE based on information entropy metric is realized in the hope of getting a trade-off between exploration and exploitation. (2) The information entropy metric is designed by using the historical evolutionary information across generations. The switching of evolutionary stage is realized in a statistical sense, while allowing the population to choose strategies adaptively in individual level. (3) DEIE can be extended to other real-life application. On the basis of stage division, the corresponding mutation strategies can be adjusted or replaced flexibly according to different application scenarios. Moreover, the proposed DEIE is tested on CEC 2013, 2014, and 2017 test sets, classical benchmark functions and a real-world case.

II. PRELIMINARY

A. DIFFERENTIAL EVOLUTION

DE consist of mutation, crossover, and selection operations [1]. Starting from a random initial population including NP individuals, the better individual is retained whereas the inferior individual is eliminated.

The main operations of one DE variant, namely DE/rand/1/bin, are shown below.

1) *Initialization*: $\boldsymbol{P}^g = \{\boldsymbol{x}_1^g, \cdots, \boldsymbol{x}_i^g, \cdots, \boldsymbol{x}_{NP}^g\}$ called population is randomly produced from the solution domain, $\boldsymbol{x}_i^g = (x_{i,1}^g, x_{i,2}^g, \cdots, x_{i,D}^g)$, where $i \in [1, NP]$, and g is expressed as the gth generation.

2) *Mutation*: The two individuals randomly selected from the population is used as the perturbation of the base vector, and the perturbation is weighted to produce the mutant individual $\boldsymbol{v}_i^g$.

$$\boldsymbol{v}_i^g = \boldsymbol{x}_{rand_1}^g + F \cdot (\boldsymbol{x}_{rand_2}^g - \boldsymbol{x}_{rand_3}^g) \tag{1}$$

where $F > 0$, $rand_1$, $rand_2$, and $rand_3$ are chosen from $[1, NP]$, and they differ from i but also to each other.

3) *Crossover*: The binomial crossover operator copies the jth parameter of the mutant individual $\boldsymbol{v}_i^g$ to the corresponding element in the trial individual $\boldsymbol{u}_i^g$ according to the crossover rate. Otherwise, it is copied from the corresponding target individual $\boldsymbol{x}_i^g$.

$$u_{i,j}^g = \begin{cases} v_{i,j}^g, & \text{if } rand(0,1) \le CR \text{ or } j = j_{\text{rand}} \\ x_{i,j}^g, & \text{otherwise} \end{cases} \tag{2}$$

where CR as the crossover rate is chosen from $(0, 1)$; $rand(0, 1)$ is randomly generated from $[0, 1]$; $j \in [1, D]$; j_{rand} is a integer randomly yielded from $[1, D]$.

4) *Selection*: If the trial individual u_i^g achieves the better function value than that of the target individual x_i^g, u_i^g will replace x_i^g in the next generation, otherwise the x_i^g is still preserved.

$$\boldsymbol{x}_i^{g+1} = \begin{cases} \boldsymbol{u}_i^g, & \text{if } f(\boldsymbol{u}_i^g) \le f(\boldsymbol{x}_i^g) \\ \boldsymbol{x}_i^g, & \text{otherwise} \end{cases} \tag{3}$$

where $f(\boldsymbol{u}_i^g)$ and $f(\boldsymbol{x}_i^g)$ are the function value of $\boldsymbol{u}_i^g$ and $\boldsymbol{x}_i^g$, respectively.

B. INFORMATION ENTROPY

The concept of entropy derives from thermodynamics and is successfully applied to the different fields of science and engineering. Entropy introduced by Shannon [18] characterizes the uncertainty related to the occurrence of a random event, which is equal to its information content. In mathematics, we let $X = x_i, i \in n$ be a discrete random variable that represents the event of its occurrence, and the probability is denoted by p_i. Then, entropy function E can be defined as

$$E = -\sum_{i}^{n} p_i \ln p_i, \tag{4}$$

Information entropy increases with the increase in uncertainty. As a result, the measure reaches a peak value when all the outcomes are equiprobable. This implies that

$$\begin{aligned} p_i &= \frac{1}{n}, \\ E(p_1, \cdots, p_n) &\leq E(\frac{1}{n}, \cdots, \frac{1}{n}) = \ln n, \end{aligned} \tag{5}$$

III. LITERATURE REVIEW

Although DE performs well on a wide variety of problems, it has a series of problems related to stagnation, premature convergence, and so on [3], [19]. One direction of improvement is on the mutation scheme modification. Mutation is the most important step of DE as it produces a new individual in the population. Over the last few years, a lot of modifications in mutation scheme have been proposed.

Many researchers have worked towards the new mutation strategies, which helps to explore the search space by perturbing individuals, substantially to influence the performance of DE. Many different mutation strategies, such as ranking-based [20], archive-based [21], niche-based [22], centroid-based [12], and neighborhood mutations [23], have been proposed to enhance the search capability of DE. However, each of these mutation strategies, being more explorative or exploitative, seems to work for different tasks. Therefore, more attention has been paid to the multiple mutation operators which have strategies with both exploration ability and development ability.

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 promising directions: 1) individual-specific strategy techniques; 2) subpopulation-specific strategy techniques; and 3) evolutionary stage-specific strategy techniques.

Methods in the first category aim to adaptively select mutation strategies for each individual from the strategy pool. These individual-strategy matching methods mainly include probability model-based, surrogate-assisted, and so on. Probability model-based DE updates selection probabilities based on successful historical experience [24]. The self-adaptive DE (SaDE) [25], DE with ensemble of mutation strategies and parameters (EPSDE) [26], DE with strategy adaptation mechanism (SaM) [27], and DE with adaptive strategy selection (CACDE) [28] can be considered to belong to the this category. Surrogate-assisted DE utilizes valid simplified models to approximate the fitness function and is thus computationally inexpensive [29], [30]. These techniques of constructing surrogate model include kernel density estimation [31], Kriging model [32], abstract convex underestimation [33], and so on.

Methods in the second category realize multiple operators of DE by utilizing various mutation strategies in different subpopulations. The DE with self-adaptive multi-subpopulation [34], [35], DE with three small indicator subpopulations and one large reward subpopulation [36], DE with role assignment [37], and SHADE (success-history based adaptive DE) with subpopulation-based ensemble of mutation strategies are belong to this category.

For methods in the last category, the main idea is to divide entire searching process into multiple stages and select suitable mutation strategies for each stage. Some early works divided the whole process by setting a fixed number of iterations, such as two stages [6], three stages [7], and so on. Although the performance of DE may be improved, empirical guidelines are sometimes unreliable and lack universality. In order to accommodate the search characteristics in the evolutionary process of DE, researchers prefer to estimate the evolutionary states to distinguish the different stages. Yu *et al.* [8] discussed the relationship between the order of fitness value and distance to divide evolutionary process into two stages. Zhan *et al.* [10] proposed master-slave distributed framework based on the evolutionary state estimation. The evolutionary state is estimated to two states by distance computations between two individuals with the best fitness value and the median fitness value. Li and Li [11] designed an evolutionary state estimation method based on the correlation coefficient between the population distributions in objective space and solution space. Then, the evolutionary process is divided into three kinds of state.

It is distinct from the above evolutionary state-based adaptive operator selection methods realized by current population distribution estimation. In the proposed DEIE, an information entropy metric is designed by using the historical evolutionary information across generations, which reveal the trend of the movement of individuals in the search space. It is reasonable to estimate the extent that the population explores the solution space and then divided evolutionary process into two stages.

IV. DEIE ALGORITHM

This section introduces an adaptive differential evolution with information entropy-based mutation strategy, named DEIE, mainly including a dynamic stage division and a stage-specific mutation strategy adaptation technique.

The emergence of local fitness landscape on the multimodality is due to the individuals in the population are scattered at the exploration stage of evolution. The differences between individuals to each other are gradually reduced and

the distribution is concentrated. Thus, exploitation stage can be determined by the property of the unimodal basins of local fitness landscape [38]. Based on the above property, it can be seen that search dynamics in DE induces basin-to-basin transfer, where trial solutions may traverse from one attraction basin to another one [39]. In consideration of the search behaviour of DE, these basins are defined as Markov states with respect to the partition of the solution space. In this way, the Markov state model is constructed using the historical evolutionary information across generations to describe the frequency of state transition. Subsequently, the information entropy metric is proposed to estimate the extent that the population explores the solution space, which is mainly used for the dynamic division of the evolutionary stages. Moreover, the suitable mutation strategies are utilized to update offspring individuals for the different stages.

A. THE DETERMINATION OF MARKOV STATES

Due to the search behaviour of basin-to-basin transfer, several subdomains decomposed from the entire solution space are defined as Markov states in this paper. Inspired by the automatic clustering of the crowding strategy in the multimodal method [40], a learning process is designed to guide the entire population split into several subpopulations located different optima. In this way, the multiple solution subspaces based on final spatial positions of the individuals are generated, namely Markov states.

The learning process consists of archiving, crowding and clustering operations. After *Gen* iterations, K stable Markov states can be obtained.

A population $\boldsymbol{P}^g = \{\boldsymbol{x}_i^g\}, i = 1, 2, \cdots, NP$ is generated after initial operation, where NP is population size and g is generation count. The detailed procedure at one iteration is performed as follows.

1. Archiving operation

Mutation and crossover operations act on the target individual $\boldsymbol{x}_i^g$ and generate the trial individual $\boldsymbol{v}_i^g$. If $\boldsymbol{v}_i^g$ has the minimum Euclidean distance from $\boldsymbol{x}_j^g$ compared with the other individuals in $\boldsymbol{P}^g$, then $\boldsymbol{v}_i^g$ is added to the archive $\boldsymbol{A}_j^g$ of $\boldsymbol{x}_j^g$. Repeating above steps for i from 1 to NP, all trial individuals are fell into the corresponding archive.

2. Crowding operation

The purpose of crowding operation is to generate new population.

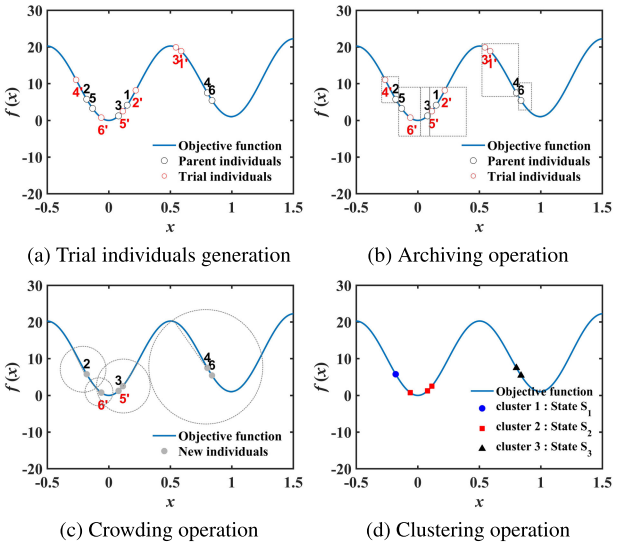
For each archive $\boldsymbol{A}_j^g$, the final optimal individual $\boldsymbol{o}_j^g$ and the corresponding radius r_j^g are calculated.

$$\boldsymbol{o}_j^g = \arg\min_{t=1,\cdots,l} f(\boldsymbol{A}_{j,t}^g), \tag{6}$$

$$r_j^g = \arg\max_{t=1,\cdots,l} d(\boldsymbol{A}_{j,t}^g, \boldsymbol{o}_j^g), \tag{7}$$

where $\boldsymbol{A}_{j,t}^g$ is t th trial individual of archive $\boldsymbol{A}_j^g$, and $f(\boldsymbol{A}_{j,t}^g)$ is the function value for $\boldsymbol{A}_{j,t}^g$, $d(\boldsymbol{A}_{j,t}^g, \boldsymbol{o}_j^g)$ is the Euclidean distance between $\boldsymbol{A}_{j,t}^g$ and $\boldsymbol{o}_j^g$, the size of $\boldsymbol{A}_j^g$ is l. Then, $\boldsymbol{x}_j^g$ is replaced by $\boldsymbol{o}_j^g$ in order to update population.

3. Clustering operation



(a) Trial individuals generation (b) Archiving operation

(c) Crowding operation (d) Clustering operation

FIGURE 1. Example of the learning of Markov states.

Objects to be clustered are individuals of new population and the clustering criterion is based on their location. Starting from the individual with minimum function value, the clustering step is performed with $\boldsymbol{x}_s^{g+1}$ as the center and r_s^{g+1} as the radius in turn according to the ascending order, where $s = 1, 2, \cdots, NP$. Each individual $\boldsymbol{x}_i^{g+1}$ of the new population is assigned to the corresponding cluster on the basis of a distance criterion.

$$d(\boldsymbol{x}_i^{g+1}, \boldsymbol{x}_s^{g+1}) < r_s^{g+1}, \tag{8}$$

where $d(\boldsymbol{x}_i^{g+1}, \boldsymbol{x}_s^{g+1})$ is the Euclidean distance between $\boldsymbol{x}_i^{g+1}$ and $\boldsymbol{x}_s^{g+1}$.

The clustering process is completed when all the individuals are assign to a corresponding cluster. Denote these clusters as the Markov states S represented by center and K as the number of Markov states. After *Gen* iterations, the number K tends to be stable.

Fig.1 shows the learning of Markov states of the one-dimensional Rastrigin function at one iteration. The population is composed of six individuals. the parent individuals are represented by black circle $\boldsymbol{P}^g = \{1, 2, 3, 4, 5, 6\}$, whereas the trial individuals are marked in red circle $\{1', 2', 3', 4', 5', 6'\}$. The indexes are used to mark their respective order. Six archives, namely, $\boldsymbol{A}_1^g = \{1, 2', 5'\}, \boldsymbol{A}_2^g = \{2, 4'\}, \boldsymbol{A}_3^g = \{3\}, \boldsymbol{A}_4^g = \{4, 1', 3'\}, \boldsymbol{A}_5^g = \{5, 6'\}, \boldsymbol{A}_6^g = \{6\}$, with a parent individual and the corresponding trial individuals are shown in Fig.1b. The population updated using the optimal individuals, namely, $\boldsymbol{P}^{g+1} = \{\boldsymbol{o}_1^g : 5', \boldsymbol{o}_2^g : 2, \boldsymbol{o}_3^g : 3, \boldsymbol{o}_4^g : 4, \boldsymbol{o}_5^g : 6', \boldsymbol{o}_6^g : 6\}$, is the new population as shown in Fig.1c. The new population is finally divided into three subpopulations using clustering operation in Fig.1d. Notably, the above-mentioned learning process is not strictly the classical clustering method. In this part, we only focus on dividing the entire population into several subpopulations and define the final clustering partition as the Markov state.

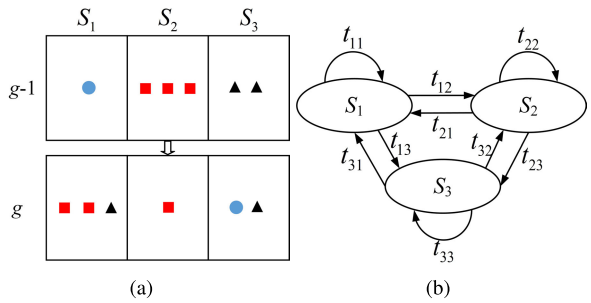


FIGURE 2. Illustration of the transition matrix.

B. THE TRANSITION MATRIX OF MARKOV STATE MODEL

These Markov states correspond to different solution subspaces have been fixed, as discussed before. For the purpose of building Markov state model, we take the population data available and assign each individual in the population to a Markov state according to the minimal Euclidean distance. The state assignment of each individual in consecutive generations can be traced. The historical evolutionary information across generations may reveal the trend of the movement of individuals in the search space and can thus be used to guide the evolutionary process.

The transition matrix of Markov state model is illustrated in Fig.2. Considering that Markov states have been learned in advance, the state assignments at each generation can be obtained shown in Fig. 2a. In line with the temporal ordering of state transition of individuals at the two adjacent generations, a transition matrix can be constructed as shown in Fig. 2b.

For a Markov state model with K states, the count of states transition can be obtained using historical evolutionary information across generations to build the transition matrix $\boldsymbol{T}^g$. The element t_{ij}^g in row i and column j in this matrix corresponds to the observed frequency of transitions from state i at the $(g-1)$th generation to state j at the gth generation. Therefore, each element t_{ij}^g in the transition matrix $\boldsymbol{T}^g$ can be described as

$$t_{ij}^g = \frac{N(i^{g-1} \to j^g)}{N(i^{g-1})}, \tag{9}$$

where t_{ij}^g is an element in transition matrix $\boldsymbol{T}^g$, $N(i^{g-1} \to j^g)$ is the number of state transition from state i at the $(g-1)$th generation to state j at the gth generation, and $N(i^{g-1})$ is the number of individuals located in state i at the $(g-1)$th generation. For example, there are 3 red individuals in state 2 at generation $g-1$ ($N(2^{g-1}) = 3$), two of which move to state 1 at generation g ($N(2^{g-1} \to 1^g) = 2$). So $t_{21}^g = 2/3$.

C. INFORMATION ENTROPY-BASED MUTATION STRATEGY ADAPTION

It is well known that the important operation of DE is its mutation operation [41]. The mutation strategy utilized by DE largely governs its tendency to discover promising regions or detect the optima. The population behaviour in different evolutionary stages influences the selection of mutation strategies to a certain extent.

However, there is a problem that how to estimate the evolutionary stages and employ the stage-specific mutation strategy appropriately.

The mutation strategy adaption is performed as follows.

1) INFORMATION ENTROPY METRIC

The information entropy based on the Markov state model describes the extent that the population explores the solution space. Given the above-mentioned analysis, the information entropy can be used to estimate the evolutionary stages.

On the basis of the transition matrix, the probability that the evolutionary process is undergoing a transition between any given pair of states can be estimated by

$$p_{ij}^g = \frac{t_{ij}^g}{\sum_m^K \sum_n^K t_{mn}^g}, \tag{10}$$

where $\sum_{i=1}^K \sum_{j=1}^K p_{ij}^g = 1$.

Then, the information entropy E^g across all possible transitions can be calculated by summing the Shannon entropy for each individual transition at each iteration

$$E^g = -\sum_i^K \sum_j^K p_{ij}^g \ln p_{ij}^g, \tag{11}$$

where E^g is denoted as the information entropy of the gth generation.

The value of E^g is normalized using $E_{\max}$ and $E_{\min}$ as follows

$$\bar{E}^g = \frac{E^g - E_{\min}}{E_{\max} - E_{\min}}, \tag{12}$$

Shannon showed that this quantity achieves its maximum value when p_{ij}^g are equal to each other, and the value of $E_{\max}$ can be calculated using

$$E_{\max} = \frac{1}{K} \ln \frac{1}{K^2}, \tag{13}$$

and $E_{\min} = 0$ because individuals no longer have state transition when the evolution process stabilizes or terminates.

2) STAGE ESTIMATION

In accordance with the aforementioned property of information entropy, the evolutionary stages can be estimated as follows:

$$\Psi = \begin{cases} \text{exploration stage}, & \text{if } rand(0,1) < \bar{E}^g \\ \text{exploitation stage}, & \text{otherwise} \end{cases} \tag{14}$$

where Ψ represents the estimated evolutionary stage.

There are several reasons for the above stage division.

(1) From the perspective of population, the different stages may coexist in the same generation, and each individual stage should be estimated separately.

(2) The large $\bar{E}^g$ caused by a case that the population is frequently transferred between several states, which indicates that the scope of solution space is explored extensively. For this case, the individual stage is likely estimated to be the exploration stage because the population explores different regions. The mutation strategy DE/rand/1 with good exploration capability is more suitable for this stage. On the contrary, the population concentrates on some parts of the solution space for exploitation when the value of $\bar{E}^g$ is small. The individual stage can be estimated to the exploitation stage, and the mutation strategy DE/best/1 with good exploitation capability can be employed to detect the optima.

(3) When the population is in a certain state (the local optimal solution region), the individuals are no longer transferred between states. In this case, the value of the information entropy is zero, which results in rapid convergence of the population.

3) MUTATION STRATEGY AND CONTROL PARAMETER SELECTION

$$\boldsymbol{v}_i^g = \begin{cases} \boldsymbol{x}_{rand_1}^g + F_i^g \cdot (\boldsymbol{x}_{rand_2}^g - \boldsymbol{x}_{rand_3}^g), \\ \text{if } \boldsymbol{x}_i^g \text{ in exploration stage} \\ \boldsymbol{x}_{best}^g + F_i^g \cdot (\boldsymbol{x}_{rand_2}^g - \boldsymbol{x}_{rand_3}^g), \\ \text{otherwise,} \quad \boldsymbol{x}_i^g \text{ in exploitation stage} \end{cases} \tag{15}$$

where mutant individual $\boldsymbol{v}_i^g$ is generated according to the stage of target individual $\boldsymbol{x}_i^g$. When $\boldsymbol{x}_i^g$ in exploration stage, the mutation strategy DE/rand/1 is employed, when $\boldsymbol{x}_i^g$ in exploitation stage, the mutation strategy DE/best/1 is employed. $\boldsymbol{x}_{best}^g$ is the best individual in the population at generation g. $F_i^g \in (0, 1]$ is scaling factor for $\boldsymbol{x}_i^g$, $rand_1$, $rand_2$, and $rand_3$ are randomly chosen from $[1, NP]$, and they differ from i but also to each other.

A simple selection strategy for control parameters F and CR based on current stage division inspired by [8] is designed for comparison. An explorative individual will demand a high F and CR, whereas an exploitative will require the opposite. The F and CR values for each individual of gth generation are assigned as follows.

$$F_i^g = \begin{cases} F_i^{g-1} + rand(0, 0.1)\bar{E}^g, \\ \text{if } \boldsymbol{x}_i^g \text{ in exploration stage} \\ F_i^{g-1} - rand(0, 0.1)\bar{E}^g, \\ \text{otherwise,} \quad \boldsymbol{x}_i^g \text{ in exploitation stage} \end{cases} \tag{16}$$

$$CR_i^g = \begin{cases} CR_i^{g-1} + rand(0, 0.1)\bar{E}^g, \\ \text{if } \boldsymbol{x}_i^g \text{ in exploration stage} \\ CR_i^{g-1} - rand(0, 0.1)\bar{E}^g, \\ \text{otherwise,} \quad \boldsymbol{x}_i^g \text{ in exploitation stage} \end{cases} \tag{17}$$

D. DEIE ALGORITHM DESCRIPTION

The DEIE algorithm is described as Algorithm 1. After initialization, the K Markov states are learned firstly through the learning process of Gen generations. In the following iteration process, the individuals in population have state transition caused by the search behaviour of the population. In line with the temporal ordering of state transition of individuals at the two adjacent generations, the state transition probability is calculated, and further the information entropy is calculated to observe the population dynamics. Based on the information entropy metric, the evolutionary stages of current individual can be estimated, then the suitable mutation strategy is selected for different stages.

Notably, the Markov state is formed at the end of the learning process of Gen generation, and the number of Markov states K and the representative center point are obtained, where the DE/rand/1 mutation strategy is employed in the learning process. Based on the K Markov states, the information entropy is calculated in the following iteration process. In addition, the infeasible solution is simply discarded and replaced with a new solution regenerated within the domain.

Algorithm 1 Pseudocode of DEIE Algorithm

Require: population size (NP), scaling factor (F), crossover rate (CR), learning period (Gen).
Ensure: Final population ($\boldsymbol{P}^g$).

```
Initialization: generate initial population (g = 0, P^g = {x_1^g, x_2^g, ..., x_NP^g}), evaluate the function value of each individual in P^g, and set the relevant parameters of DEIE algorithm;
while the termination criterion is not satisfied do
    for g <= Gen do
        Execute the learning process in Section 4.1;
    end for
    Obtain the number Markov states K;
    Determine Markov states of current population S = {S_1, ..., S_t, ..., S_NP}, S_t ∈ {1, ..., K};
    Set initial T^g = 0 and Ē^g = 0;
    for g > Gen do
        for i = 1 to NP do
            if rand(0, 1) < Ē^g then
                Generate mutant individual v_i^g via mutation strategy DE/rand/1;
            else
                Generate mutant individual v_i^g via mutation strategy DE/best/1;
            end if
            Generate trial individual u_i^g via binomial crossover operation;
            Select new individual x_i^{g+1} by compare trial individual u_i^g with target individual x_i^g;
            Assign x_i^{g+1} to corresponding Markov state;
        end for
        Update T^g and Ē^g;
        g = g + 1;
    end for
end while
```

TABLE 1. Results of mean and standard deviation of the function error obtained by SHADE, ZEPDE, IDE, SinDE, and DEIE for CEC 2013 benchmark set at $D = 30$.

Fun	D	SHADE Mean(Std Dev)	ZEPDE Mean(Std Dev)	IDE Mean(Std Dev)	SinDE Mean(Std Dev)	DEIE Mean(Std Dev)
F_1	30	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	2.27E−13(1.53E−28)$^{+}$	**0.00E+00(0.00E+00)**
F_2	30	**2.66E+04(1.13E+04)**$^{-}$	1.97E+05(7.53E+04)$^{-}$	1.68E+06(4.23E+05)$^{+}$	2.66E+06(8.33E+05)$^{+}$	2.40E+05(1.44E+05)
F_3	30	8.80E+05(1.96E+06)$^{+}$	1.50E+06(2.07E+06)$^{+}$	1.38E+05(1.85E+05)$^{-}$	**1.01E+05(3.77E+05)**$^{-}$	5.88E+05(1.18E+05)
F_4	30	1.61E−03(**1.41E−03**)$^{-}$	7.66E−01(4.78E−01)$^{+}$	6.85E+03(1.10E+03)$^{+}$	8.28E+03(1.54E+03)$^{+}$	2.12E−03(1.76E−03)
F_5	30	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	1.14E−13(7.65E−29)$^{+}$	**0.00E+00(0.00E+00)**
F_6	30	4.28E+01(5.52E+00)$^{+}$	4.34E+01(3.18E−13)$^{+}$	4.34E+01(2.62E−04)$^{+}$	4.34E+01(1.44E−14)$^{+}$	**1.46E+01(1.68E+00)**
F_7	30	2.33E+01(9.32E+00)$^{+}$	1.37E+01(4.88E+00)$^{+}$	3.18E+00(1.55E+00)$^{+}$	**6.10E−01(5.97E−01)**$^{-}$	2.89E+00(1.42E+00)
F_8	30	**2.09E+01(1.68E−01)**$^{\approx}$	2.11E+01(1.17E−01)$^{+}$	2.11E+01(2.44E−02)$^{+}$	2.11E+01(3.59E−02)$^{+}$	**2.09E+01(4.76E−02)**
F_9	30	5.54E+01(1.98E+00)$^{+}$	3.74E+01(5.85E+00)$^{+}$	3.56E+01(5.54E+00)$^{-}$	**3.48E+01(4.34E+00)**$^{-}$	3.58E+01(1.28E+00)
F_{10}	30	7.37E−02(3.67E−02)$^{+}$	1.37E−01(6.96E−02)$^{+}$	4.38E−02(2.17E−02)$^{+}$	7.93E−02(3.57E−02)$^{+}$	**3.82E−02(9.73E−03)**
F_{11}	30	**0.00E+00(0.00E+00)**$^{-}$	3.65E−01(6.12E−01)$^{-}$	**0.00E+00(0.00E+00)**$^{-}$	5.92E+00(2.86E+00)$^{+}$	5.06E+00(1.60E+00)
F_{12}	30	5.86E+01(1.11E+01)$^{-}$	6.04E+01(1.76E+01)$^{-}$	6.89E+01(8.82E+00)$^{-}$	**5.61E+01(1.41E+01)**$^{-}$	1.79E+02(1.27E+01)
F_{13}	30	1.45E+02(1.95E+01)$^{-}$	**1.32E+02(3.62E+01)**$^{-}$	1.34E+02(2.28E+01)$^{-}$	1.39E+02(3.41E+01)$^{-}$	1.80E+02(1.19E+01)
F_{14}	30	**3.45E−02(1.93E−02)**$^{-}$	4.83E+00(2.70E+00)$^{+}$	1.17E+02(8.38E+01)$^{+}$	2.34E+02(9.23E+01)$^{+}$	1.14E−01(2.45E−02)
F_{15}	30	6.82E+03(4.41E+02)$^{+}$	6.59E+03(9.36E+03)$^{+}$	**6.54E+03(5.91E+02)**$^{-}$	6.80E+03(1.00E+03)$^{+}$	6.58E+03(2.14E+02)
F_{16}	30	1.28E+00(2.07E−01)$^{+}$	**7.82E−01(6.74E−01)**$^{-}$	1.59E+00(2.36E−01)$^{+}$	2.08E+00(3.66E−01)$^{+}$	1.02E+00(3.01E−01)
F_{17}	30	**5.08E+01(4.27E−14)**$^{\approx}$	5.11E+01(1.60E−01)$^{+}$	5.92E+01(1.41E+00)$^{+}$	6.52E+01(3.47E+00)$^{+}$	**5.08E+01(3.69E+00)**
F_{18}	30	1.37E+02(1.29E+01)$^{+}$	**1.03E+02(1.19E+01)**$^{-}$	1.68E+02(1.27E+01)$^{+}$	1.41E+02(2.27E+01)$^{+}$	1.09E+02(1.01E+01)
F_{19}	30	2.64E+00(2.83E−01)$^{-}$	3.71E+00(7.55E−01)$^{-}$	**2.24E+00(3.66E−01)**$^{-}$	4.85E+00(8.82E−01)$^{-}$	1.12E+02(1.13E+00)
F_{20}	30	1.93E+01(7.70E−01)$^{+}$	1.97E+01(7.88E−01)$^{+}$	1.93E+01(4.47E−01)$^{+}$	**1.92E+01(7.52E−01)**$^{\approx}$	**1.92E+01(3.01E−01)**
F_{21}	30	8.45E+02(3.63E+02)$^{+}$	6.33E+02(4.48E+02)$^{+}$	7.32E+02(3.82E+02)$^{+}$	5.84E+02(4.22E+02)$^{+}$	**3.53E+02(4.99E+01)**
F_{22}	30	**1.33E+01(7.12E+00)**$^{-}$	4.23E+02(5.75E+02)$^{+}$	6.88E+01(2.03E+01)$^{-}$	3.51E+02(2.72E+02)$^{+}$	3.25E+02(5.44E+01)
F_{23}	30	7.63E+03(6.58E+02)$^{+}$	7.02E+03(8.73E+02)$^{-}$	7.32E+03(6.92E+02)$^{-}$	**6.59E+03(8.47E+02)**$^{-}$	7.56E+03(3.12E+02)
F_{24}	30	2.34E+02(1.01E+01)$^{+}$	2.35E+02(1.09E+01)$^{+}$	2.02E+02(1.14E+00)$^{+}$	2.00E+02(**1.34E−01**)$^{\approx}$	**2.00E+02(5.12E+00)**
F_{25}	30	3.40E+02(3.09E+01)$^{+}$	3.23E+02(1.31E+01)$^{+}$	3.03E+02(1.09E+01)$^{-}$	**2.97E+02(1.33E+01)**$^{-}$	3.09E+02(7.18E+00)
F_{26}	30	2.58E+02(8.08E+01)$^{+}$	2.27E+02(6.20E+01)$^{+}$	2.23E+02(4.46E+01)$^{+}$	2.76E+02(5.96E+01)$^{+}$	**2.09E+02(3.00E+01)**
F_{27}	30	9.36E+02(3.07E+02)$^{+}$	9.38E+02(1.40E+02)$^{+}$	**3.58E+02(3.30E+01)**$^{-}$	4.75E+02(1.55E+02)$^{-}$	5.91E+02(1.63E+02)
F_{28}	30	4.58E+02(4.13E+02)$^{+}$	**4.00E+02(0.00E+00)**$^{\approx}$	**4.00E+02(0.00E+00)**$^{\approx}$	**4.00E+02(0.00E+00)**$^{\approx}$	**4.00E+02(0.00E+00)**
$+/\approx/-$		14/4/10	17/3/8	13/3/12	16/3/9	
p-value		0.1615	0.0300	0.9571	0.0443	

E. RUNTIME COMPLEXITY OF DEIE

Based on the above procedures, the runtime complexity of DEIE depends on the following analysis. In terms of Markov states, the runtime complexity $O(NP \cdot NP \cdot D)$ comes from calculating the Euclidean distance between NP individuals to each other. For state assignments, there is $O(NP \cdot K \cdot D)$ for computing the Euclidean distance between K centers and NP individuals. The generation of offspring individuals needs $O(NP \cdot D)$ similar to the basic DE. For Markov state model construction, transition matrix is constructed after statistics with runtime complexity denoted as $O(NP)$. $O(K \cdot K)$ runtime is used to calculate the information entropy. Hence, the total runtime complexity of DEIE is $O(\max(NP \cdot NP \cdot D \cdot Gen, NP \cdot K \cdot D \cdot G_{\max}, NP \cdot D \cdot G_{\max}), NP \cdot G_{\max}, K \cdot K \cdot G_{\max})$. Given that the number of states K is less than the population size NP, the final runtime complexity is $O(NP \cdot K \cdot D \cdot G_{\max})$. $G_{\max}$ is the maximum number of generations of whole algorithm. The original DE algorithm is $O(NP \cdot D \cdot G_{\max})$. According to the study in [43]–[45], the runtime complexity of DEIE is relatively small compared with that of expensive function evaluations. Therefore, the proposed DEIE is accepted for the practical problems, especially for expensive-to-evaluate problems.

V. EXPERIMENTAL RESULTS

To evaluate the performance of DEIE, CEC 2013 [42], CEC 2014 [43], and CEC2017 sets [44] are utilized in the following diverse experiments.

The experimental results are presented in four subsections. In Section V-A, the performance of DEIE is evaluated against that of eleven top-ranked DE variants. Section V-B presents the experiment of component analysis. The parameter study is described in Section V-C. Section V-D provides the real-life application of DEIE. Moreover, just to show that the proposed DEIE works well on different test sets, 21 classical benchmark functions [45] are also used to test the performance compared with that of four state-of-the-art DE and three classical EAs. Some experiments about classic benchmark functions are shown in the supplementary file.

All algorithms cease when the number of function evaluation (FEs) accumulates to exceed the maximum number of FES (MaxFEs), or the function error reaches the predefined accuracy within the given MaxFEs. $(f(x) - opti)$ represents the function error, where $f(x)$ is expressed as the function value of solution x generated by the current algorithm and $opti$ presents the global optimum. In the following experiments, the predefined accuracy value is $1.00E - 08$. The results

TABLE 2. Results of mean and standard deviation of the function error obtained by L-SHADE, MC-SHADE, iLSHADE, ADDE and DEIE for CEC 2014 benchmark set at $D = 30$.

Fun	D	L-SHADE Mean(Std Dev)	MC-SHADE Mean(Std Dev)	iLSHADE Mean(Std Dev)	ADDE Mean(Std Dev)	DEIE Mean(Std Dev)
F_1	30	**0.00E+00(0.00E+00)**$^-$	2.92E+03(2.72E+03)$^+$	**0.00E+00(0.00E+00)**$^-$	**0.00E+00(0.00E+00)**$^-$	2.52E+03(7.40E+03)
F_2	30	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**
F_3	30	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**
F_4	30	**0.00E+00(0.00E+00)**$^-$	7.82E−02(5.58E−01)$^-$	**0.00E+00(0.00E+00)**$^-$	**0.00E+00(0.00E+00)**$^-$	2.22E+00(1.62E+00)
F_5	30	**2.01E+01(3.72E−02)**$^\approx$	**2.02E+01(2.39E−02)**$^\approx$	**2.01E+01(1.00E−01)**$^\approx$	2.03E+01(2.85E−02)$^+$	**2.01E+01(2.54E−02)**
F_6	30	1.38E−07(9.98E−07)$^-$	1.23E+00(2.37E+00)$^+$	**0.00E+00(0.00E+00)**$^-$	1.85E−03(3.10E−02)$^-$	6.65E−01(2.55E−01)
F_7	30	**0.00E+00(0.00E+00)**$^\approx$	2.90E−04(2.07E−03)$^+$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**$^\approx$	**0.00E+00(0.00E+00)**
F_8	30	**0.00E+00(0.00E+00)**$^-$	**0.00E+00(0.00E+00)**$^-$	**0.00E+00(0.00E+00)**$^-$	**0.00E+00(0.00E+00)**$^-$	5.72E+00(1.08E+00)
F_9	30	**6.78E+00(1.50E+00)**$^-$	1.95E+01(2.95E+00)$^+$	6.91E+00(2.00E+00)$^-$	1.40E+01(2.87E+00)$^+$	8.64E+00(4.11E+00)
F_{10}	30	1.63E−02(1.59E−02)$^+$	1.10E−02(1.46E−02)$^+$	1.10E−02(1.27E−02)$^+$	3.39E−01(2.22E−01)$^+$	1.02E−02(9.47E−02)
F_{11}	30	1.23E+03(1.81E+02)$^-$	1.57E+03(1.91E+02)$^-$	1.17E+03(2.80E+02)$^-$	**1.72E+03(3.27E+02)**$^-$	6.20E+03(2.75E+02)
F_{12}	30	1.61E−01(2.31E−02)$^+$	2.10E−01(2.98E−02)$^+$	1.48E−01(5.14E−02)$^+$	4.06E−01(6.44E−02)$^+$	**1.35E−01(2.36E−02)**
F_{13}	30	1.24E−01(1.76E−02)$^+$	2.06E−01(3.00E−02)$^+$	9.50E−02(2.12E−02)$^+$	1.42E−01(1.53E−02)$^+$	**3.13E−02(4.87E−02)**
F_{14}	30	2.42E−01(3.00E−02)$^+$	2.19E−01(3.59E−02)$^+$	1.98E−01(3.46E−02)$^+$	2.23E−01(4.22E−02)$^+$	**1.93E−01(2.86E−02)**
F_{15}	30	2.15E+00(2.50E−01)$^+$	3.00E+00(3.91E−01)$^+$	**1.84E+00(2.72E−01)**$^+$	2.52E+00(6.21E−01)$^+$	1.84E+00(1.09E−01)
F_{16}	30	8.50E+00(4.62E−01)$^+$	9.42E+00(4.43E−01)$^+$	**8.02E+00(9.81E−01)**$^-$	9.49E+00(2.77E−01)$^+$	8.12E+00(2.63E−01)
F_{17}	30	1.88E+02(7.55E+01)$^+$	1.24E+03(3.93E+02)$^+$	1.31E+02(6.52E+01)$^+$	4.57E+02(2.13E+02)$^+$	**1.05E+02(7.81E+00)**
F_{18}	30	5.91E+00(2.92E+00)$^-$	7.81E+01(3.68E+01)$^-$	**3.78E+00(1.45E+00)**$^-$	1.85E+01(6.36E+00)$^-$	5.59E+01(3.09E+01)
F_{19}	30	3.68E+00(6.87E−01)$^+$	4.50E+00(8.29E−01)$^+$	2.29E+00(7.55E−01)$^+$	3.64E+00(5.23E−01)$^+$	**2.20E+00(3.20E−01)**
F_{20}	30	3.08E+00(1.48E+00)$^-$	1.99E+01(1.27E+01)$^-$	2.41E+00(1.09E+00)$^-$	5.22E+00(1.19E+00)$^+$	3.91E+01(3.83E+01)
F_{21}	30	8.68E+01(9.02E+01)$^+$	3.15E+02(1.58E+02)$^+$	**5.07E+01(6.12E+01)**$^-$	7.69E+01(7.23E+01)$^+$	5.84E+01(9.58E+01)
F_{22}	30	2.76E+01(1.80E+01)$^+$	1.37E+02(6.64E+01)$^+$	2.90E+01(2.42E+01)$^+$	2.86E+01(4.60E+00)$^+$	**2.32E+01(1.01E+01)**
F_{23}	30	3.15E+02(1.72E−13)$^\approx$	3.15E+02(0.00E+00)$^\approx$	3.15E+02(0.00E+00)$^\approx$	3.15E+02(0.00E+00)$^\approx$	3.15E+02(4.06E−12)
F_{24}	30	2.24E+02(1.07E+00)$^+$	2.25E+02(2.02E+00)$^+$	2.20E+02(6.09E+00)$^\approx$	2.23E+02(4.60E+00)$^+$	2.20E+02(5.48E−01)
F_{25}	30	2.03E+02(4.98E−02)$^\approx$	2.05E+02(1.80E+00)$^+$	2.03E+02(4.17E−02)$^\approx$	2.03E+02(4.76E+00)$^\approx$	2.03E+02(2.41E−01)
F_{26}	30	**1.00E+02(1.57E−02)**$^\approx$	1.02E+02(1.40E+01)$^+$	**1.00E+02(2.06E−02)**$^\approx$	**1.00E+02(2.68E−02)**$^\approx$	**1.00E+02(1.64E−02)**
F_{27}	30	3.00E+02(0.00E+00)$^\approx$	3.40E+02(4.76E+01)$^+$	3.01E+02(5.21E+00)$^+$	3.00E+02(0.00E+00)$^\approx$	3.00E+02(4.16E+00)
F_{28}	30	8.40E+02(1.42E+01)$^+$	8.00E+02(3.02E+01)$^+$	8.44E+02(1.48E+01)$^+$	8.19E+02(1.69E+01)$^+$	7.87E+02(9.57E+00)
F_{29}	30	7.17E+02(5.17E+00)$^+$	7.35E+02(3.86E+01)$^+$	7.16E+02(3.31E+00)$^+$	7.17E+02(2.68E+00)$^+$	6.69E+02(1.30E+02)
F_{30}	30	1.25E+03(6.18E+02)$^+$	1.56E+03(6.43E+02)$^+$	1.20E+03(5.55E+02)$^+$	**8.16E+02(4.96E+02)**$^-$	9.84E+02(5.32E+02)
$+/\approx/-$		14/8/8	21/4/5	12/8/10	16/7/7	
p-value		0.6849	0.0049	0.6389	0.8078	

are averaged using 51 independent runs for each function of each algorithm. The parameters NP, F, and CR of DEIE are set to 50, 0.5, and 0.5, respectively. The number of Markov state K is determined automatically at the end of the learning process. And the iteration number of learning process Gen is set to 100. In addition, the parameters settings of other algorithms are identical to their original papers.

A. COMPARISON OF DEIE WITH TOP-RANKED DE VARIANTS

To evaluate the overall performance of DEIE on the CEC 2013 [42], CEC 2014 [43], and CEC2017 sets [44], several top-ranked DE variants is used as the competitor algorithms. In this experiment, the termination criterion are set to MaxFEs $= 10,000 \times D$ as suggested in [44].

First, DEIE is compared with four advanced DE variants on the CEC 2013 set, namely, SHADE [46], ZEPDE [9], IDE [5], and SinDE [47]. Table 1 reports the mean and Std values of 30-D functions. DEIE produces good results on 11 out of 28 functions compared to all competitors, performs significantly better on 14, 17, 13, and 16 out of 28 functions, and exhibits similar performance on 4, 3, 3, and 3 functions, respectively. SHADE, ZEPDE, IDE, and SinDE show remarkably better performance than DEIE on 10, 8, 12, and 9 functions, respectively. The last row of Table 1 gives the analysis of the optimization performance obtained by Wilcoxon′s test. DEIE outperforms ZEPDE and IDE (p-value < 0.05). SHADE, SinDE and DEIE performs at the same level of optimization, but DEIE achieved best 14 and 13 cases than SHADE and SinDE.

Second, DEIE is compared with other four advanced DE variants on the CEC 2014 set as shown in Table 2, namely, L-SHADE [48], MC-SHADE [49], iLSHADE [50], and ADDE [10]. DEIE obtains good results on 11 out of 30 functions compared to all competitors. The results reveal that DEIE may tend to perform better on unimodal and multimodal functions. DEIE significantly outperforms others on 14, 21, 12, and 16 functions, respectively. DEIE gets equally good performance on 8, 4, 8, and 7 functions, respectively, compared to other algorithms. However, L-SHADE, MC-SHADE, iLSHADE, and ADDE are obviously better than DEIE on 8, 8, 5, and 10 functions, respectively. Clearly, DEIE is significantly better than MC-SHADE. Although the significance test shows that DEIE has no significant advantage over L-SHADE, iLSHADE and ADDE, it can obtain the optimal results in 14, 21, 12 and 16 cases, respectively.

TABLE 3. Results of mean and standard deviation of the function error obtained by LSHADE-cnEpSin, IDEbestNsize, jSO, EDEV and DEIE for CEC 2017 benchmark set at $D = 30$.

Fun	D	LSHADE-cnEpSin Mean(Std Dev)	IDEbestNsize Mean(Std Dev)	jSO Mean(Std Dev)	EDEV Mean(Std Dev)	DEIE Mean(Std Dev)
F_1	30	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	8.36E−16(3.38E−15)$^{+}$	**0.00E+00(0.00E+00)**
F_3	30	**0.00E+00(0.00E+00)**$^{\approx}$	4.05E+00(3.07E+00)$^{+}$	**0.00E+00(0.00E+00)**$^{\approx}$	4.24E−11(3.03E−10)$^{+}$	**0.00E+00(0.00E+00)**
F_4	30	4.23E+01(3.07E+00)$^{+}$	2.42E+00(3.62E+00)$^{+}$	5.90E+01(7.78E−01)$^{+}$	1.02E+00(1.75E+00)$^{+}$	**3.30E−01(1.22E−01)**
F_5	30	1.23E+01(2.34E+00)$^{+}$	2.27E+01(5.03E+00)$^{+}$	**8.60E+00(2.10E+00)**$^{-}$	3.05E+01(7.40E+00)$^{+}$	1.18E+01(6.51E+00)
F_6	30	**0.00E+00(0.00E+00)**$^{-}$	**0.00E+00(0.00E+00)**$^{-}$	6.00E−09(2.71E−08)$^{+}$	1.09E−13(2.23E−14)$^{-}$	2.88E−13(2.46E−13)
F_7	30	4.33E+01(2.17E+00)$^{+}$	5.15E+01(4.81E+00)$^{+}$	**3.90E+01(1.46E+00)**$^{-}$	6.42E+01(5.25E+00)$^{+}$	4.23E+01(6.81E−01)
F_8	30	1.29E+01(2.86E+00)$^{-}$	2.36E+01(4.79E+00)$^{+}$	**9.10E+00(1.84E+00)**$^{-}$	2.76E+01(7.96E+00)$^{+}$	1.65E+01(1.06E+00)
F_9	30	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	**0.00E+00(0.00E+00)**$^{\approx}$	1.69E−01(3.49E−01)$^{+}$	**0.00E+00(0.00E+00)**
F_{10}	30	**1.39E+03(2.10E+02)**$^{-}$	2.02E+03(3.85E+02)$^{+}$	1.50E+03(2.77E+02)$^{+}$	2.16E+03(4.88E+02)$^{+}$	1.43E+03(2.69E+02)
F_{11}	30	1.35E+01(1.94E+01)$^{+}$	6.44E+00(2.78E+00)$^{-}$	**3.00E+00(2.65E+00)**$^{-}$	1.88E+01(7.54E+00)$^{+}$	8.80E+00(7.28E−02)
F_{12}	30	3.72E+02(2.01E+02)$^{-}$	3.45E+03(2.40E+03)$^{+}$	**1.70E+02(1.02E+02)**$^{-}$	5.72E+03(5.78E+03)$^{+}$	2.90E+03(9.12E+02)
F_{13}	30	1.73E+01(1.02E+01)$^{-}$	3.09E+01(1.12E+01)$^{-}$	**1.50E+01(4.83E+00)**$^{-}$	4.93E+01(8.06E+01)$^{-}$	5.80E+01(1.02E+01)
F_{14}	30	2.16E+01(2.26E+00)$^{-}$	2.33E+01(9.69E+00)$^{+}$	2.20E+01(1.25E+00)$^{+}$	**1.49E+01(1.08E+01)**$^{-}$	2.19E+01(6.17E+00)
F_{15}	30	3.24E+00(1.98E+00)$^{-}$	7.58E+00(2.19E+00)$^{+}$	**1.10E+00(6.91E−01)**$^{-}$	1.17E+01(1.02E+01)$^{+}$	5.23E+00(3.22E−01)
F_{16}	30	**2.29E+01(3.07E+01)**$^{-}$	1.79E+02(1.23E+02)$^{+}$	7.90E+01(8.48E+01)$^{-}$	4.99E+02(1.70E+02)$^{+}$	1.68E+02(1.78E+02)
F_{17}	30	**2.86E+01(5.56E+00)**$^{-}$	4.14E+01(1.21E+01)$^{+}$	3.30E+01(8.08E+00)$^{+}$	4.69E+01(2.93E+01)$^{+}$	3.23E+01(7.50E+00)
F_{18}	30	2.11E+01(7.52E−01)$^{-}$	3.21E+01(6.75E+00)$^{-}$	**2.00E+01(2.87E+00)**$^{-}$	5.25E+01(1.34E+02)$^{-}$	3.72E+02(1.39E+02)
F_{19}	30	5.83E+00(1.92E+00)$^{+}$	9.18E+00(2.29E+00)$^{+}$	4.50E+00(1.73E+00)$^{+}$	7.71E+00(4.99E+00)$^{+}$	**4.12E+00(1.00E+00)**
F_{20}	30	3.03E+01(7.35E+00)$^{+}$	4.05E+01(2.24E+01)$^{+}$	2.90E+01(5.85E+00)$^{+}$	3.31E+01(4.91E+01)$^{+}$	**5.01E+00(2.41E+00)**
F_{21}	30	2.12E+02(2.56E+00)$^{-}$	2.25E+02(4.59E+00)$^{+}$	**2.10E+02(1.96E+00)**$^{-}$	2.36E+02(7.86E+00)$^{+}$	2.17E+02(2.58E+00)
F_{22}	30	**1.00E+02(1.00E−03)**$^{\approx}$	**1.00E+02(0.00E+00)**$^{\approx}$	**1.00E+02(0.00E+00)**$^{\approx}$	2.79E+02(6.20E+02)$^{+}$	**1.00E+02(0.00E+00)**
F_{23}	30	3.56E+02(3.73E+00)$^{+}$	3.68E+02(6.74E+00)$^{+}$	**3.50E+02(3.30E+00)**$^{-}$	3.83E+02(8.17E+00)$^{+}$	3.55E+02(3.35E+00)
F_{24}	30	**4.28E+02(2.95E+00)**$^{-}$	4.37E+02(5.33E+00)$^{-}$	4.30E+02(2.47E+00)$^{-}$	4.53E+02(7.43E+00)$^{-}$	5.73E+02(9.77E+00)
F_{25}	30	3.87E+02(8.90E−03)$^{\approx}$	3.87E+02(1.23E−01)$^{\approx}$	3.90E+02(7.68E−03)$^{+}$	**3.83E+02(6.63E+00)**$^{-}$	3.87E+02(7.31E−02)
F_{26}	30	9.49E+02(4.60E+01)$^{-}$	1.05E+03(2.95E+02)$^{+}$	**9.20E+02(4.30E+01)**$^{-}$	1.31E+03(9.85E+01)$^{+}$	1.01E+03(3.78E+02)
F_{27}	30	5.04E+02(6.70E+00)$^{+}$	4.96E+02(8.85E+00)$^{+}$	5.00E+02(7.00E+00)$^{+}$	5.00E+02(1.38E−04)$^{+}$	**4.86E+02(7.19E+00)**
F_{28}	30	3.15E+02(3.86E+01)$^{+}$	3.17E+02(3.90E+01)$^{+}$	3.10E+02(3.03E+01)$^{+}$	4.44E+02(7.95E+01)$^{+}$	**3.00E+02(5.22E−05)**
F_{29}	30	4.35E+02(7.36E+00)$^{+}$	4.55E+02(2.28E+01)$^{+}$	4.30E+02(1.36E+01)$^{+}$	**4.09E+02(6.59E+01)**$^{-}$	4.20E+02(1.86E+02)
F_{30}	30	**1.98E+03(4.17E+01)**$^{-}$	2.30E+03(1.84E+02)$^{-}$	2.00E+03(1.90E+01)$^{-}$	2.23E+02(2.19E+01)$^{-}$	2.56E+03(1.97E+02)
$+/\approx/-$		10/5/14	19/4/6	11/4/14	21/0/8	
p-value		0.1531	0.0480	0.1578	0.0252	

Last, DEIE is compared with other three advanced DE variants on the CEC 2017 set as shown in Table 3, namely, LSHADE-cnEpSin [51], IDEbestNsize [52], jSO [53], and EDEV [54]. In the 30 CEC2017 benchmark functions, f2 has deleted in the updated version [55]. DEIE yields good results on 9 functions compared to all competitors. It follows that DEIE may prefer unimodal and composition functions. DEIE performs better than the competitors on 10, 19, 11, and 21 functions, respectively. DEIE gets equally good performance on 5, 4, 4 and 0 functions, respectively. LSHADE-cnEpSin, IDEbestNsize, and jSO outperform DEIE on 14, 6, 14, and 8 functions, respectively. DEIE outperforms IDEbestNsize and EDEV. LSHADE-cnEpSin and jSO obtain best performance in 14 and 15 cases, but there is no significant difference compared with DEIE. In conclusion, DEIE achieves better or at least comparable performance compared to these top-ranked DE algorithms.

B. EFFECTS OF DEIE COMPONENTS

The effectiveness of the proposed DEIE maybe depends on the evolutionary stage division based on entropy, the basic mutation strategy DE/rand/1 and DE/best/1 are further used in exploration and exploitation stages, respectively. To discuss the validity of the proposed DEIE, experiments are conducted on benchmark CEC2017 set to identify the effect of each component. The results are presented in Table 4, where DEIE-rand and DEIE-best represent DEIE using only DE/rand/1 strategy and DEIE utilizing only DE/best/1 strategy, respectively. As shown in Table 4, DEIE significantly outperforms the other DEIE variants on the majority of functions.

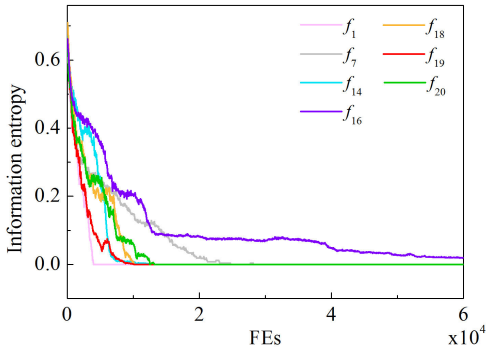


FIGURE 3. Curves of information entropy.

Fig.3 shows the information entropy curves for seven classic functions(these formulas are shown in the supplementary file Table S-I). The trend of rapid decline indicates that the

TABLE 4. Results of mean and standard deviation of the function error obtained by DEIE-rand, DEIE-best and DEIE for CEC 2017 benchmark set at $D = 30$.

Fun	D	DEIE-rand Mean(Std Dev)	DEIE-best Mean(Std Dev)	DEIE Mean(Std Dev)
F_1	30	4.95E−11(4.93E−11)$^+$	4.78E−14(2.71E−14)$^+$	**0.00E+00(0.00E+00)**
F_3	30	1.91E−01(2.19E−01)$^+$	2.89E−13(2.34E−13)$^+$	**0.00E+00(0.00E+00)**
F_4	30	8.31E+01(6.90E+00)$^+$	8.41E+01(4.03E+00)$^+$	**3.30E−01(1.22E−01)**
F_5	30	1.66E+02(1.22E+01)$^+$	1.46E+02(2.72E+01)$^+$	**1.18E+01(6.51E+00)**
F_6	30	**1.14E−13(0.00E+00)**$^-$	4.19E−04(6.91E−04)$^+$	2.88E−13(2.46E−13)
F_7	30	2.04E+02(9.07E+00)$^+$	1.84E+02(2.17E+01)$^+$	**4.23E+01(6.81E+01)**
F_8	30	1.67E+02(9.70E+00)$^+$	1.31E+02(4.75E+01)$^+$	**1.65E+01(1.06E+00)**
F_9	30	**0.00E+00(0.00E+00)**$^\approx$	3.18E−13(1.37E−13)$^+$	**0.00E+00(0.00E+00)**
F_{10}	30	6.41E+03(2.88E+02)$^+$	6.33E+03(4.20E+02)$^+$	**1.43E+03(2.69E+02)**
F_{11}	30	6.90E+01(2.52E+01)$^+$	1.50E+01(1.59E+01)$^+$	**8.80E+00(7.28E−02)**
F_{12}	30	1.34E+04(6.89E+03)$^+$	1.89E+04(5.16E+04)$^+$	**2.90E+03(9.12E+02)**
F_{13}	30	1.16E+04(7.28E+03)$^+$	6.41E+03(6.84E+03)$^+$	**5.80E+01(1.02E+01)**
F_{14}	30	7.66E+01(7.10E+00)$^+$	3.50E+01(1.06E+01)$^+$	**2.19E+01(6.17E+00)**
F_{15}	30	6.56E+01(6.10E+01)$^+$	2.28E+01(2.49E+01)$^+$	**5.23E+00(3.22E−01)**
F_{16}	30	5.47E+02(2.45E+02)$^+$	6.21E+02(2.24E+02)$^+$	**1.68E+02(1.78E+02)**
F_{17}	30	7.97E+01(5.14E+01)$^+$	6.13E+01(4.92E+01)$^+$	**3.23E+01(7.50E+00)**
F_{18}	30	**3.17E+02(2.61E+02)**$^-$	3.45E+03(1.60E+04)$^+$	3.72E+02(1.39E+02)
F_{19}	30	3.64E+01(2.86E+01)$^+$	1.48E+01(5.77E+00)$^+$	**4.12E+00(1.00E+00)**
F_{20}	30	9.07E+01(7.39E+01)$^+$	1.07E+02(8.66E+01)$^+$	**5.01E+00(2.41E+00)**
F_{21}	30	3.63E+02(8.65E+00)$^+$	3.23E+02(5.04E+01)$^+$	**2.17E+02(2.58E+00)**
F_{22}	30	**1.00E+02(0.00E+00)**$^\approx$	**1.00E+02(8.18E−01)**$^\approx$	**1.00E+02(0.00E+00)**
F_{23}	30	5.10E+02(5.40E+01)$^+$	4.27E+02(5.83E+01)$^+$	**3.55E+02(3.35E+00)**
F_{24}	30	5.96E+02(9.45E+00)$^+$	**5.02E+02(6.34E+01)**$^-$	5.73E+02(9.77E+00)
F_{25}	30	**3.87E+02(4.62E−02)**$^\approx$	**3.87E+02(6.71E−02)**$^\approx$	**3.87E+02(7.31E−02)**
F_{26}	30	2.26E+03(7.63E+02)$^+$	1.17E+03(4.12E+02)$^+$	**1.01E+03(3.78E+02)**
F_{27}	30	4.98E+02(2.21E+01)$^+$	5.02E+02(8.14E+00)$^+$	**4.86E+02(7.19E+00)**
F_{28}	30	3.10E+02(3.19E+01)$^+$	3.24E+02(4.37E+01)$^+$	**3.00E+02(5.22E−05)**
F_{29}	30	6.90E+02(1.42E+02)$^+$	4.86E+02(6.03E+01)$^+$	**4.20E+02(1.86E+02)**
F_{30}	30	4.33E+03(1.29E+03)$^+$	2.70E+03(9.93E+02)$^+$	**2.56E+03(1.97E+02)**
$+/\approx/-$		24/3/2	26/2/1	

TABLE 5. Ranking of Friedman's test for different values of K.

	K=2	K=3	K=5	K=7	K=9
Ranking	5.64	4.98	5.93	5.88	4.57
	K=11	K=13	K=15	K=17	K=19
Ranking	5.48	5.6	5.33	5.71	5.88

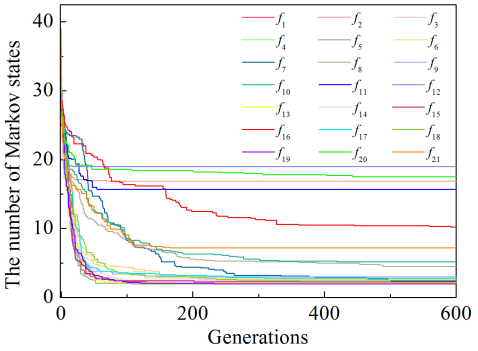


FIGURE 4. Curves of parameter K.

population rapidly locates several promising states from multiple states. Subsequently, the value of information entropy is zero because the population is concentrated on a certain state. This result is consistent with the convergence of the function with good results in Table S-III (supplementary file). The function with the worse results, such as f_9 function, has an information entropy curve that is always greater than zero although it declines at the beginning. The curve of f_{16} function indicates that the population continues to vacillate between at least two states. Thus, no optimal solution can be found.

C. PARAMETER STUDY

In the proposed DEIE, the parameters, i.e., *Gen*, K, NP, F and CR need to be discussed. The experiments of this part are conducted on 21 classic functions (see supplementary file Table S-I).

1) LEARNING PERIOD *Gen* AND *K*

Considering that the K Markov state is formed at the end of the learning process, the learning period *Gen* must be optimized to consider the learning effect of the Markov states and the overall optimization of the DEIE. Large *Gen* can ensure accurate learning of the number of Markov states but will consume computational resources when the algorithm explores the optimal solution. Conversely, small *Gen* will result in coarse division. First, it is investigated that the relationship between the maximum learning period *Gen* and the number of Markov states K. Fig.4 shows the phenomenon that the number of Markov states K decreases as the iteration progresses, whereas K for most functions becomes stable after 100 iterations. To avoid the possibility of diminishing the optimization effect caused by large number of iterations, the maximum learning period *Gen* is set to a fixed number of 100.

To further reveal the impact of *Gen*, the effect of the number of Markov states K on the optimization results is investigated, where K varies from 0 to 20 under $Gen = 100$ according to Fig.4. And the other parameters are identical to those used in Section V-A. Table S-IX (see supplementary file) presents the mean and Std values under different K. For clarity, Friedman's test is used, and Table 5 reports the influence of different values of K against the performance of DEIE. It reveals that no significant differences exist under different values of K in DEIE. Therefore, $Gen = 100$ is a suitable choice.

2) POPULATION SIZE *NP*

In order to investigate the impact of NP, six frequently used settings, i.e., 30, 40, 50, 60, 80, and 100, are employed in DEIE. The rest parameter settings are the same as that described at the start of Section V. Table S-X (see supplementary file) presents the mean and Std values under different NP on 30-D classical benchmark. Clearly, DEIE with $NP = 50$ obtains better performance compared to DEIE using other NP settings. Furthermore, the Friedman rankings are summarized in Table 6. It indicates that $NP = 50$ achieves the best results on 12 out of 21 functions, and obtains the best ranking. In this way, $NP = 50$ is more suitable for DEIE.

TABLE 6. Ranking of Friedman's test for different values of *NP*.

	NP=30	NP=40	NP=50	NP=60	NP=80	NP=100
Ranking	4.45	3.12	2.00	2.93	4.00	4.05

TABLE 7. Results of the Wilcoxon's test obtained by DEIE and DEIE_fixed on classical benchmark functions at $D = 30$.

DEIE v.s.	+	≈	−	p-value
DEIE_fixed	14	4	3	2.34E−43

3) SCALING FACTOR *F* AND CROSSOVER RATE *CR*

The initial values for F and CR are set to 0.5, and 0.5 is reassigned to F and CR when the range of them are exceeded. The rest parameter settings are the same as DEIE. The results of Wilcoxon's test obtained by DEIE_fixed (without adaptive parameters) and DEIE on 30-D classical benchmark are shown in Table 7. Clearly, DEIE with adaptive parameters obtains better performance on 14 functions compared to DEIE. The mean and Std values of DEIE_fixed are shown in Table S-XI (see supplementary file).From these data, we can conclude that parameter selection based on current stage division can improve the performance of DEIE. Therefore, parameter adaptive mechanism may be another promising direction for us.

D. REAL-WORLD APPLICATION

In this part, DEIE is utilized to solve a real-life problem, namely, the protein structure prediction (PSP) problem that is essential in bioinformatics. Proteins are an important component of all cells and tissues in the human body, and their function is directly determined by their three-dimensional (3D) native structure. For example, the protein 4UEX is a structure of human saposin which is an important auxiliary factor for acid hydrolytic enzyme to degrade complex glycosphingolipids. Serious metabolic diseases may be generated by deficiencies of saposin or hydrolytic enzyme [56]. High-throughput, high-precision protein structure predicting technology will strongly promote the development of life science, greatly accelerate the development of cancer, viral antibiotics, targeted drugs and new proteases.

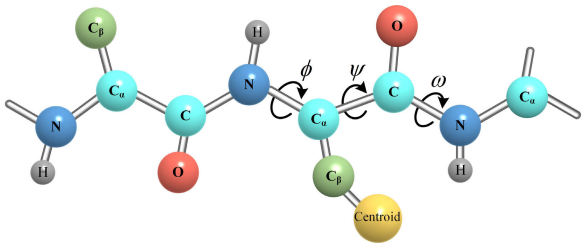


FIGURE 5. Coarse-grained representation of the protein structure.

The protein folding thermodynamic hypothesis point out that its native structure is the conformation with minimal potential energy. The features of PSP problem are inaccurate energy function and expensive function evaluation cost. The state-of-the-art methods in this field, such as Rosetta [57] and I-TASSER [58], employ multistage Monte Carlo and its variant algorithms with fixed computational cost for each stage. There are some problems need to be considered: (1) For small proteins, less computational cost possibly could be sufficient to generate native-like protein conformations. Meanwhile, a higher computational cost is needed for large proteins due to vast conformation space. Fixed cost may lead to waste of small protein exploration, but insufficient exploration of large protein. (2) Without an effective strategy, Monte Carlo adopted widely in this field is prone to premature convergence. The proposed DEIE is effective for tackling PSP problem: (a) the dynamic stage division technique is effective and adaptive for different length of proteins; (b) entropy, describing disorder or uncertainty of a system, may be more applicable for solving imprecise function models from macroscopic perspectives; (c) DEIE has good scalability, and the mutation strategy can be flexibly adjusted according to different proteins.

A coarse-grained representation of the protein structure used in Rosetta [59] is adopted in DEIE as shown in Fig.5. Therefore, the dihedral Angles of residues are used to encode the proteins. According to the latest prevalent trends, complex energy constructed from physicochemical knowledge and spatial geometry knowledge is used to guide protein folding, which is expressed as follows:

$$f = E_{\text{Rosetta}} + E_{\text{distance}} \tag{18}$$

where f represent the energy function combined by protein physicochemical model and knowledge model. E_{Rosetta} is Rosetta score3 physicochemical energy. E_{distance} is geometric model based on residue distance. Detailed description can be found in [57].

On the basis of stage division, the implementation of operators in DEIE can be redesigned flexibly according to current application scenarios. (1) Initialization. The initial population is generated through random dihedral angle perturbation. (2) Mutation operation. PSP-specific versions of DE/rand/1 and DE/best/1 mutation strategy are designed to accommodate different stages, the details are shown in the Supplementary materials. (3) Crossover operation. The trial conformation is generated by exchange residue information between the target and mutate conformation. (4) Selection operation. Following the Metropolis criterion, the satisfying offsprings can be survived into next generation.

In this experiment, DEIE is compared with four algorithms over 50 nonredundant proteins with various lengths of the amino acid sequence. Two current representative predicting methods are Rosetta-d (a distance-assisted fragment assembly method) and L-BFGSfold (a distance geometry optimization method). Two DEIE variants are DEIE_v1 (DEIE with only PSP-specific versions of DE/rand/1 mutation strategy) and DEIE_v2 (DEIE with only PSP-specific versions of DE/best/1 mutation strategy). Detailed description of

TABLE 8. Average results obtained by Rosetta-d, LBFGSfold, DEIE_v1, DEIE_v2 and DEIE on 50 proteins.

	Rosetta-d	LBFGSfold	DEIE_v1	DEIE_v2	DEIE
RMSD	5.93	4.95	11.20	5.43	4.47
TMscore	0.564	0.670	0.360	0.662	0.696
p-value	2.23E-07	2.38E-09	8.03E-10	8.73E-10	NA

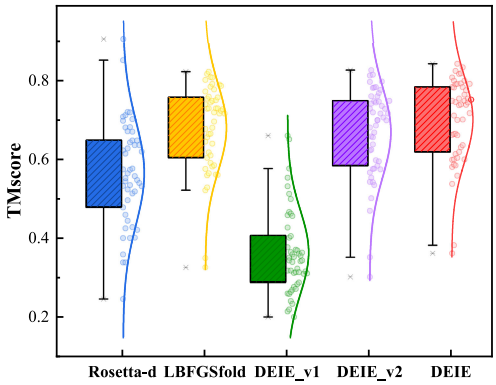


FIGURE 6. Boxplot for the TM-scores of the all 50 proteins predicted by Rosetta-d, LBFGSfold, DEIE_v1, DEIE_v2, and DEIE.

experiment setting and compared methods can be found in supplementary materials.

Comparison of predicted results generated by Rosetta-d, LBFGSfold, DEIE_v1, DEIE_v2 and DEIE are listed in Table 8 on all 50 benchmark proteins, and the detailed results of each protein are presented in Table S-XII of Supplementary materials. Two well-known structural quality measures are used in assessing the similarity of the predicted conformation and a reference conformation, generally the native structure. One is the root mean square deviation (RMSD), where smaller RMSD means the smaller deviation and the better model accuracy. The other one is the template modeling score (TMscore), which ranges in [0,1] and higher value reflects better folding accuracy. Meanwhile, a TMscore ≥ 0.5 represents correctly folded models. Compared with two state-of-the-art methods, the average RMSD of DEIE (4.47Å) is reduced by 24.63% compared to Rosetta-d (5.93Å) and 9.74% compared to LBFGSfold (4.95Å). The average TMscore by DEIE (0.696) is 23.45% higher than that of Rosetta-d (0.564) and 3.83% higher than that of LBFGSfold (0.670). Compared with two DEIE variants, the average RMSD of DEIE (4.47Å) is lower than that of DEIE_v1 (11.20Å) and DEIE_v2 (5.43Å), the average TM-score of DEIE (0.696) is higher than that of DEIE_v1(0.360) and DEIE_v2(0.662). All the results in Table 8 show that the prediction accuracy of DEIE is significantly better than each of compared methods (with P-values of <0.05). The adaptive switching mechanism and the cooperation of these two strategies are effective.

Fig.6 intuitively reflects the comparison of DEIE with other methods on 50 proteins. Compared with Rosetta-d, LBFGSfold, DEIE_v1, and DEIE_v2, DEIE achieves a lower

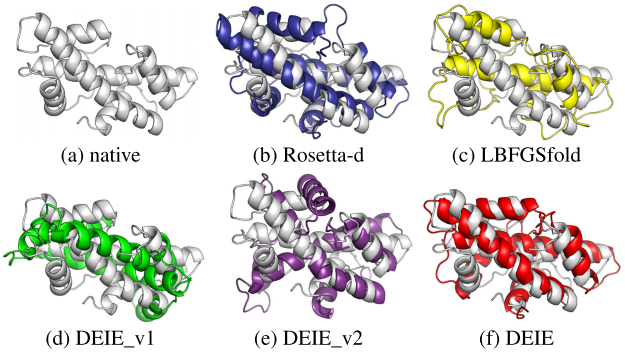


FIGURE 7. Representative example of 1F1E_A 3D structure obtained by Rosetta-d (blue), LBFGSfold (yellow), DE_v1 (green), DE_v2 (purple), and DEIE (red) with the native(grey).

RMSD on 31 of 50 proteins, accounting for 62%, and a higher TM-score on 42 of 50 proteins, accounting for 84%. Fig.7 shows one case which the superimpositions of two target model predicted by all algorithms and the experimental structure. On protein 1F1E_A, the native structure, Rosetta-d, LBFGSfold, DEIE_v1, DEIE_v2 and DEIE structure are marked in cyan, pink, blue, orange, and green, respectively. The accuracy of prediction is represented by the similarity between the predicted and natural structures. As shown in the Fig.7, DEIE achieves more accurate results than others.

VI. CONCLUSION

This paper presents a DEIE algorithm that balances the exploitation and exploration ability of DE by using suitable mutation strategy in different stages. In DEIE, the information entropy metric is proposed to determine the current stage of evolution after combining the number of Markov states with the transition matrix of Markov state model. The information entropy metric uses the historical evolutionary information across generations, which reveal the trend of the movement of individuals in the search space. It is reasonable to estimate the extent that the population explores the solution space and then divided evolutionary process into two stages. Consequently, the stage-specific mutation strategy is allocated to the population individuals by using the information entropy. Experimental results described in Section V verify that DEIE performs better than or at least competitive with diverse state-of-the-art DE variants on CEC2013, 2014, and 2017 benchmark sets. Moreover, DEIE also achieves the promising performance on real-world PSP problem. The sensitivity of DEIE to parameters is also studied. The main work of the manuscript focuses on dynamic division of the evolutionary stages based on entropy and stage-specific mutation strategy adaptation. To avoid confusion about the effectiveness of the dynamic stage division, only the simple basic mutation strategies are adopted to highlight main ideas of the manuscript. This may be the reason why DEIE cannot perform so well on some benchmark functions compared to some top-ranked algorithms. The new optimized mutation strategies based on stage division and performance

optimization based on benchmark function may be another promising direction for us, which may enhance the performance of the algorithm. In addition, DEIE is more suitable for large and complex PSP practical application because it maybe tackles the multistage problems of general concern in this field.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

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

[3] 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.

[4] M. Yang, C. Li, Z. Cai, and J. Guan, "Differential evolution with auto-enhanced population diversity," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 302–315, Feb. 2015.

[5] 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.

[6] Z. Z. Liu, Y. Wang, S. Yang, and Z. Cai, "Differential evolution with a two-stage optimization mechanism for numerical optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Vancouver, BC, Canada, Jul. 2016, pp. 861–872.

[7] 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.

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

[9] 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.

[10] Z.-H. Zhan, Z.-J. Wang, H. Jin, and J. Zhang, "Adaptive distributed differential evolution," *IEEE Trans. Cybern.*, vol. 50, no. 11, pp. 4633–4647, Nov. 2020.

[11] Y. Li and G. H. Li, "Differential evolutionary algorithm with an evolutionary state estimation method and a two-level selection mechanism," *Soft Comput.*, vol. 24, pp. 11561–11581, Dec. 2020.

[12] 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.

[13] E. Naghib and A. Nobakhti, "Entropic differential evolution—eDE," in *Proc. Amer. Control Conf.*, Boston, MA, USA, Jul. 2016, pp. 2440–2446.

[14] L. H. Wu, Y. N. Wang, X. F. Yuan, and S. W. Zhou, "Environmental/economic power dispatch problem using multi-objective differential evolution algorithm," *Electr. Power Syst. Res.*, vol. 80, no. 9, pp. 1171–1181, Sep. 2010.

[15] H. Zhang, D. Yue, X. Xie, S. Hu, and S. Weng, "Multi-elite guide hybrid differential evolution with simulated annealing technique for dynamic economic emission dispatch," *Appl. Soft Comput.*, vol. 34, pp. 312–323, Sep. 2015.

[16] H. L. Chen, Y. C. Liang, and J. D. Padilla, "Using discrete Differential Evolution and entropy to solve the MRCPSP," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Donostia, Spain, Jun. 2017, pp. 2437–2442.

[17] M. Ali, C. W. Ahn, and M. Pant, "Multi-level image thresholding by synergetic differential evolution," *Appl. Soft Comput.*, vol. 17, pp. 1–11, Apr. 2014.

[18] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.

[19] E.-N. Dragoi and V. Dafinescu, "Parameter control and hybridization techniques in differential evolution: A survey," *Artif. Intell. Rev.*, vol. 45, no. 4, pp. 447–470, Apr. 2016.

[20] W. Gong and Z. Cai, "Differential evolution with ranking-based mutation operators," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2066–2081, Dec. 2013.

[21] J. 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.

[22] Y. Li, H. Guo, X. Liu, Y. Li, W. Pan, B. Gong, and S. Pang, "New mutation strategies of differential evolution based on clearing niche mechanism," *Soft Comput.*, vol. 21, no. 20, pp. 5939–5974, Oct. 2017.

[23] X. Qiu, J. 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.

[24] W. Gong, Á. 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, 2011.

[25] 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.

[26] R. Mallipeddi and M. Lee, "An evolving surrogate model-based differential evolution algorithm," *Appl. Soft Comput.*, vol. 34, pp. 770–787, Sep. 2015.

[27] W. Gong, Z. Cai, G. 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.

[28] B. Xu, X. Chen, and L. Tao, "Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization," *Inf. Sci.*, vol. 435, pp. 240–262, Apr. 2018.

[29] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, Jun. 2011.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] M. Z. Ali, N. H. Awad, and P. N. Suganthan, "Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization," *Appl. Soft Comput.*, vol. 33, pp. 304–327, Aug. 2015.

[35] L. Cui, G. Li, Q. Lin, J. Chen, and N. Lu, "Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations," *Comput. Oper. Res.*, vol. 67, pp. 155–173, Mar. 2016.

[36] G. Wu, R. Mallipeddi, P. N. Suganthanc, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Inf. Sci.*, vol. 329, pp. 329–345, Feb. 2016.

[37] 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, Nov. 2014.

[38] M. Z. Ali, N. H. Awad, and P. N. Suganthan, "A mixed strategy for evolutionary programming based on local fitness landscape," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[39] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 246–263, Apr. 2015.

[40] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Portland, OR, USA, Jul. 2004, pp. 1382–1389.

[41] W. Gong, Z. Cai, and D. Liang, "Adaptive ranking mutation operator based differential evolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 716–727, Apr. 2015.

[42] 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., Zhengzhou Univ., Singapore, Tech. Rep. 201212, 2013.

[43] 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., Singapore, Tech. Rep. 201311, Dec. 2013.

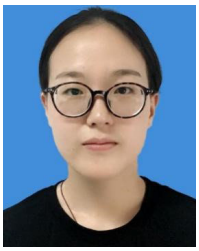
[44] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective bound constrained real-parameter numerical optimization," School Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Tech. Rep. 201611, Nov. 2016.
[45] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky, "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems," *J. Global Optim.*, vol. 31, no. 4, pp. 635–672, 2005.
[46] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, Jun. 2013, pp. 71–78.
[47] 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.
[48] 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, 2014, pp. 1658–1665.
[49] 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, Canada, Jul. 2016, pp. 4797–4803.
[50] 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, Jul. 2016, pp. 1188–1195.
[51] 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, Spain, Oct. 2017, pp. 372–379.
[52] P. Bujok and J. Tvrdík, "Enhanced individual-dependent differential evolution with population size adaptation," in *Proc. IEEE Congr. Evol. Comput.*, Donostia, Spain, 2017, pp. 1358–1365.
[53] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Donostia, Spain, Jun. 2017, pp. 1311–1318.
[54] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants," *Inf. Sci.*, vol. 423, pp. 172–186, Jan. 2018.
[55] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization," School Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Tech. Rep. 201709, Sep. 2017.
[56] C. H. Hill, R. J. Read, and J. E. Deane, "Structure of human saposin a at lysosomal pH," *Acta Crystallogr. F, Struct. Biol. Commun.*, vol. 71, no. 7, pp. 895–900, 2015.
[57] C. A. Rohl, C. E. M. Strauss, K. M. S. Misura, and D. Baker, "Protein structure prediction using Rosetta," *Methods Enzymol.*, vol. 383, pp. 66–93, Jan. 2004.
[58] A. Roy, A. Kucukural, and Y. Zhang, "I-TASSER: A unified platform for automated protein structure and function prediction," *Nature Protocols*, vol. 5, no. 4, pp. 725–738, Apr. 2010.
[59] S. Kmiecik, D. Gront, M. Kolinski, L. Wieteska, A. E. Dawid, and A. Kolinski, "Coarse-grained protein models and their applications," *Chem Rev.*, vol. 116, no. 14, pp. 7898–7936, 2016.

LIUJING WANG is currently pursuing the Ph.D. degree in control science and engineering with the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

Her research interests include intelligent information processing, optimization theory and algorithm design, and bioinformatics.

XIAOGEN 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 Postdoctoral Fellow with the Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA. His research interests include intelligent information processing, optimization theory and algorithm design, and bioinformatics.



TENGYU XIE received the master's degree from the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

Her research interests include intelligent information processing, optimization theory and algorithm design, 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 research interests include intelligent information processing, optimization theory and algorithm design, and bioinformatics.

GUIJUN 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.