Potential-Based Differential Evolution Algorithm With Joint Adaptation of Parameters and Strategies

LI TIAN, ZHICHAO LI, AND XUEFENG YAN, (Member, IEEE)

Key Laboratory of Advanced Control and Optimization for Chemical Processes of Ministry of Education, East China University of Science and Technology, Shanghai 200237, China

Corresponding author: Xuefeng Yan (xfyan@ecust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 21878081, and in part by the Fundamental Research Funds for the Central Universities of China under Grant 222201917006.

ABSTRACT In the differential evolution (DE) algorithm, numerous studies have independently performed strategy adaptation and parameter adaptation. However, the strategy and parameters are interrelated in their impact on algorithm performance. It is well known that different problems and evolutionary stages require different appropriate parameters and strategies, but the fact that the same is true for different individuals is ignored. Few studies have focused on the difference in fitness values between two successive generations, which contains substantial evolution information. This study proposes a potential-based DE algorithm with joint adaptation of parameters and strategies (JAPSPDE). In JAPSPDE, a new population classification scheme, a new classification evolution mechanism, and a new joint adaptation mechanism are proposed to circumvent the three abovementioned issues. In the population classification scheme, individuals are divided into potential and unpotential individuals according to the improvement in fitness values between two generations. A classification evolution mechanism is applied by evolving potential individuals and unpotential individuals in two ways. In addition, a three-dimensional probability array is constructed to achieve joint adaptation of parameters and strategies. Finally, after properly combining the above algorithmic components, JAPSPDE can find the most appropriate combination of control parameters and mutation strategies for specific problems, stages, and individuals. The performance of JAPSPDE is evaluated in comparison with five well-known DE algorithms on BBOB2012 and CEC2014 and with six up-to-date evolution algorithms on CEC2014. The comparison results demonstrate the competitive performance of JAPSPDE.

INDEX TERMS Differential algorithm, individual potential, joint adaptation of parameters and strategies.

I. INTRODUCTION

The differential evolution (DE) algorithm was proposed by Rainer Storn and Kenneth Price in 1995 [1] to solve Chebyshev polynomials. Because of its simplicity, robustness, and validity, applications of DE include various domains, such as feature selection [2], [3], image segmentation [4], energy storage [5], economic emission dispatch [6], color image encryption [7], fault identification [8], automatic generation control [9], chemical processing, artificial intelligence, and pattern recognition.

In DE, three elements influence the optimization performance: mutation factor $F$, crossover rate $CR$, and mutation strategy. However, these three elements are problem dependent. Different problems and even different stages of the same problem require different appropriate parameters and strategies. Finding the best combination of mutation strategies and parameter values for a specific problem and specific stage is time consuming. Many researchers have proposed adaptive DE variants that automatically adjust the parameters and choose mutation strategies during the evolution process. However, two problems exist in these studies.

First, all individuals are treated equally in these DE variants, without considering the difficulty of achieving improvements in fitness value for different individuals or the
differences between individuals. However, differences exist between individuals, and algorithms may achieve better performance by assigning different evolutionary approaches to different individuals. Certain algorithms take the differences in individual fitness values into account and adopt different mutation strategies for different individuals, such as IDE [10]. IDE classifies individuals according to their fitness value and evolves them in different ways. In IDE, individuals are classified as superior and inferior individuals based on their fitness values. Superior individuals adopt two suitable mutation strategies, and inferior individuals adopt two other suitable mutation strategies. However, this approach uses fixed strategies for two types of individuals, which is not a strict adaptive method and does not apply the idea that individuals automatically choose appropriate strategies.

Second, in these versions of the DE algorithm, adaptations of the three evolutionary elements (control parameters \( F \) and \( CR \) and mutation strategies) are performed independently during evolution. Other algorithms combine the two control parameters and adaptively adjust them, such as EPSDE [11] and ZEPDE [12]. However, during the evolution process, parameters and strategies may also be relevant. For example, for strategy A, it may be suitable for a smaller \( F \) value and subsequently a larger \( CR \) value, or it may be possible for strategy A to be suitable for larger \( F \) values in conjunction with smaller \( CR \) values. The three factors interact with each other. However, previous studies do not consider the interrelationship of the three factors.

To solve the aforementioned problems, a potential-based DE algorithm with joint adaptation of parameters and strategies (JAPSPDE) is proposed. In JAPSPDE, a new population classification scheme is designed by classifying individuals into potential and unpotential individuals according to the improvement of the fitness values between two generations. On this basis, a classification evolution mechanism is proposed. Potential individuals and unpotential individuals evolve in two ways. The experience learned from the generation of potential individuals is used to guide future individuals. The generation process of potential individuals is separated into two cases in which they are sourced from previous potential or unpotential individuals. The study results of the two cases are applied to lead the evolution of current potential and unpotential individuals. In addition, a new joint adaptation mechanism is implemented by adapting \( F \), \( CR \) and the mutation strategies in a joint manner. Firstly, a variety of different combinations of parameters and mutation strategies are preset. In each generation, the selection probability of each combination is determined by its contribution to the improvement in fitness values when generating potential individuals. This mechanism enables the algorithm to find a promising combination during the evolution process. Moreover, two historical memories that record historical successful probability combinations are introduced to enhance the robustness of the proposed algorithm. Finally, by combining the above algorithmic components, JAPSPDE can find the most appropriate combination of control parameters and mutation strategies for specific problems, stages, and individuals. The performance of this approach is verified via comparisons with several well-known DE variants and evolution algorithms (EAs) on two benchmark sets. The main contributions of this paper can be summarized as the following several points.

1) Potential and unpotential individuals are defined based on the improvement of fitness values. Selecting appropriate control parameters and mutation strategies for individuals with different characteristics is conducive to obtaining better optimization performance.

2) A new joint adaptation mechanism is implemented by adapting \( F \), \( CR \) and the mutation strategies in a joint manner. Different combinations of mutation strategies and control parameters are preset, and the probability that each combination will be selected for potential and unpotential individuals in the next evolution is updated according to the degree of improvement of each combination on the fitness values.

3) Historical memories that record historical successful combinations are introduced to enhance the robustness of the proposed algorithm.

The remaining article is organized as follows. The original DE is briefly described in Section II. Section III reviews the related works. Section IV elaborates on the basic rationale of JAPSPDE, and Section V tests its performance with extensive experiments. Conclusions are given in Section VI.

II. BASIC DIFFERENTIAL EVOLUTION

In the DE algorithm, \( NP \) individuals (\( NP \): population size) with \( D \) dimensions (\( D \): number of decision variables), \( x_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,D}) \) (\( i = 1, 2, \ldots, NP \), \( j = 1, 2, \ldots, D \) search in a parallel manner within the search space. In each evolution of the generation, each individual vector is used once as the target vector. The basic steps of the detailed DE algorithm are described below [1], [13], [14].

A. INITIALIZATION

First, a random initial population is generated and is required to cover the entire searching region, and it is typically generated using a uniformly distributed random function.

In the iterative process of the DE algorithm, the following three steps are repeated: mutation, crossover and selection.

B. MUTATION

After initialization, the mutation operation is performed on every target vector \( x_i^G \) to generate a mutant vector \( v_i^G \). For example, in the most commonly used mutation strategy (DE/rand/1), a mutant vector is generated by adding a random vector to a weighted differential vector, which represents another two different random vector subtractions:

\[
v_i^G = x_i^G + F \cdot (x_{r_2}^G - x_{r_3}^G),
\]

where \( r_1, r_2, r_3 \in \{1, 2, \cdots, NP\} \) are different from each other and are also different from the index \( i \). \( F \) is the scaling factor that controls the amplification of the differential vector.
For variables that exceed the limit, the following operation should be performed:

\[
v_{ij}^G = \begin{cases} 
    \min \{x_{ij}^{\text{upper}}, 2x_{ij}^{\text{lower}} - v_{ij}^G\}, & \text{if } v_{ij}^G < x_{ij}^{\text{lower}} \\
    \max \{x_{ij}^{\text{lower}}, 2x_{ij}^{\text{upper}} - v_{ij}^G\}, & \text{if } v_{ij}^G > x_{ij}^{\text{upper}}
\end{cases}
\]

(2)

where \(x_{ij}^{\text{lower}}\) and \(x_{ij}^{\text{upper}}\) represent the maximum and minimum of the \(j\)th dimension, respectively.

C. Crossover

The crossover operation is applied to \(x_i^G\) and \(v_i^G\) to generate a trial vector \(u_i^G\).

The crossover operation exchanges information between \(x_i^G\) and \(v_i^G\), usually by adopting binomial (bin) or exponential (exp) crossover. The binomial crossover operation is described as follows:

\[
u_{ij}^G = \begin{cases} 
    v_{ij}^G, & \text{if } \text{rand}(j) \leq \text{CR or } j = j_{\text{rand}} \\
    x_{ij}^G, & \text{otherwise}
\end{cases}
\]

(3)

where \(\text{CR}\) is the crossover rate, \(\text{rand}(j)\) is a random number in \([0, 1]\). \(j_{\text{rand}} \in \{1, \ldots, D\}\) ensures that the trial individual is not the same as the target individual.

D. Selection

Finally, the better of \(x_i^G\) and \(u_i^G\) is retained in the \(G + 1\) generation. Taking minimization optimization as an example,

\[
x_i^{G+1} = \begin{cases} 
    u_i^G, & \text{if } f(u_i^G) < f(x_i^G) \\
    x_i^G, & \text{if } f(x_i^G) \leq f(u_i^G)
\end{cases}
\]

(4)

In each generation, steps \(B\) to \(D\) are performed until the termination condition is reached.

III. Related Work

In the DE algorithm, the mutation strategy and control parameters are the critical factors for algorithm performance. Numerous research studies have been performed on setting the control parameters and choosing mutation strategies for DE.

Certain works in the literature [15], [16] give empirical guidelines on the settings for \(F\) and \(\text{CR}\) and the selection of the mutation strategy. However, these values are often inconsistent and contradictory because they are obtained based on specific problems. However, the characteristics of most actual problems are unknown, and the process for searching for solutions is changing, which means that the fixed parameter values and mutation strategy cannot satisfy a certain problem and each phase of the evolution process. Therefore, various adaptive parameter control methods and strategy selection schemes have been proposed to adjust the parameters and strategies automatically during the evolution process. For example, Brest et al. [17] proposed a self-adaptive DE known as jDE in which the control parameters are encoded to individuals and determined with uniform distributions. SaDE, as proposed by Qin et al. [18], adopts four mutation strategies for which the selection probabilities are calculated by their own success rates in a learning period. In SaDE, \(F\) is adapted with a fixed normal distribution, and \(\text{CR}\) with respect to each strategy is adapted with a variable normal distribution in which the location parameter is the median of the successful \(\text{CR}\) values in a learning period with respect to the strategy. JADE [19] designs a new mutation strategy, which is the only strategy used in the algorithm. Similar to SaDE, JADE uses two distributions to generate \(F\) and \(\text{CR}\) (a Cauchy distribution for \(F\) and a normal distribution for \(\text{CR}\)), but both location parameters are updated in JADE as the weighted mean between the old value and the mean (the Lehmer mean for \(F\) and arithmetic mean for \(\text{CR}\)) of the successful values at the previous generation. To improve the robustness of JADE, the SHADE approach proposed by Tanabe and Fukunaga [20] introduces historical memory to store the mean of the successful parameters of each generation. The authors proposed forward L-SHADE [21] to improve the performance of SHADE by adding a population reduction scheme. Different from JADE, SHADE, and L-SHADE with a fixed strategy, another improved version of SHADE, known as SA-SHADE [22], uses five mutation strategies; thus, it adds a memory of successful mutation strategies, and the memory is reset after certain generations. The memory is initialized with random strategies and updated with the most successful strategy in each generation. For each individual, a random strategy is chosen from the memory. Wu et al. [23] proposed MPEDE, which randomly divides the population into three equally small-sized indicator subpopulations and one large-sized reward subpopulation at every generation. MPEDE uses three mutation strategies, which are assigned to three indicator subpopulations, and the best-performing strategy is assigned to the reward subpopulation in a certain generations. The parameter control method of MPEDE is the same as that of JADE. IMPEDE, proposed by Tong et al. [24], improves MPEDE by replacing one mutation strategy and adopting the weighted Lehmer mean strategy in the parameter adaptation process to balance exploration and exploitation. EPSDE, proposed by Mallipeddi et al. [11], has three pools with three mutation strategies, six \(F\) values, and nine \(\text{CR}\) values. A strategy, an \(F\) value, and a \(\text{CR}\) value randomly chosen from three pools are assigned to each initial individual. The successful combination is retained for the next generation and stored in a successful combination pool. The failed individuals are assigned a new strategy and parameters from three pools or the successful combination pool. CoDE, as proposed by Wang et al. [25], applies three sets of combined parameter values and three strategies. For each individual, each strategy and a random parameter setting are used to implement a crossover operation. The selection operator is implemented among the three trial vectors and the target vector. Tang et al. [10] proposed IDE, the parameters and strategy of which are all adapted based on individual difference in fitness values. In IDE, \(F\) and \(\text{CR}\) are generated by two normal distributions with location parameters based on the rank of each individual. Superior and inferior
individuals in the former phase and latter phase adopt four strategies. Tian et al. [26] introduced an improved version of IDE known as IDEI by developing a combined mutation strategy that eliminates the need to separate the evolution process into two phases, freeing the algorithm from dividing the population into two categories. Similar to IDE, TSDE [27], proposed by Liu et al., establishes two different strategy pools containing two or three strategies with different characteristics for the former and the latter stages. In addition, TSDE adopts three parameter settings as in CoDE. For each individual, an arbitrary strategy in the corresponding strategy pool, according to its evolution stage, and a random parameter setting are selected to generate the trial vector. Fan and Yan [12] proposed ZEPDE, which divides the zone of parameters into four equal regions. For each region, two control parameters are determined with a Cauchy distribution and a normal distribution, the local parameters of which are based on the weighted mean of the successful parameters in this region. In ZEPDE, the selection probability in the next generation of each mutation strategy is based on the difference between the trial vector and the worst vector fitness values. PKDE, proposed by Fan et al. [28], adapts $F$ with a Cauchy distribution and a local parameter that is gradually decreased during the evolution and adapts $CR$ with a normal distribution and a local parameter that is gradually increased during the evolution to balance convergence and diversity. Mohamed and Suganthan [29] proposed EFADE, which balances convergence and diversity by introducing a triangular strategy and two novel parameter adaptation schemes. Fan et al. [30] proposed a search space reduction mechanism for the DE algorithm by sequentially finding the trust half area of each variable. The selected DE variant is used within the entire trust region found. Although Wu et al. also proposed to optimizing strategies and parameters of differential evolution as a whole (ACODE) [31]. The idea of JASPDE is different from ACODE’s idea. ACODE uses an ant colony optimization algorithm to adaptively configure mutation strategies, crossover strategies, scaling factors, and crossover factors, which selects mutation strategy, scaling factor value, cross strategy and crossover rate for each individual one by one according to the weight of each component. In the proposed JASPDE, different combinations of mutation strategies and control parameters are set in advance, and update the probability that each combination will be selected in the next evolution according to the degree of improvement of each combination on the fitness values. In addition, ACODE does not divide the population and treat all individuals equally. However, the proposed JASPDE takes into account the differences between individuals, and divides the individuals in the population into potential and unpotential individuals to evolve separately.

IV. PROPOSED ALGORITHM

In this section, the potential-based DE algorithm with joint adaptation of parameters and strategies (JASPDE) is introduced in seven parts. In IV-A, we elaborate the general idea of JASPDE, IV-B introduces initialization, IV-C explains joint adaptation, IV-D presents the parameters and strategy generations, IV-E demonstrates the joint probability update, IV-F demonstrates the historical memory update, and IV-G ends by summarizing the complete processes of JASPDE.

A. GENERAL IDEA OF JASPDE

We propose JASPDE based on the following considerations.

First, the adjustment of parameters and strategies is based on individual differences. We know that different problems and evolutionary stages require different control parameters and mutation strategies. Various studies have been proposed to set parameters and select strategies according to the problem characteristics and evolutionary stages [10], [26], [32]. However, the observation that different individuals also require different parameters and strategies has been ignored. Individual characteristics are often not used in parameter setting and strategy selection.

Individuals with different characteristics need different evolutionary approaches. Only a few algorithms, such as IDE, which assigns different strategies to good and bad individuals classified based on fitness values, have acknowledged this fact. Although certain researchers have observed the individual differences in fitness values, no studies have noted the individual differences in the improvement of fitness values – individual potential.

The improvement in fitness values refers to the improvement of fitness values between two generations compared with the parent individual, i.e., $\Delta f^G(x_i) = f(x_i^{G-1}) - f(x_i^G)$, which is characterized as individual potential. In our algorithm, all individuals are sorted in descending order by $\Delta f$. The individuals in the first $ps \times 100\%$ with a larger $\Delta f$ can be treated as potential individuals, while the remainder are treated as unpotential individuals. In each generation, the population is divided into two types based on their improvement of fitness values, potential (abbreviated as $P$) and unpotential (abbreviated as $UP$) individuals. Different evolutionary approaches are applied for these two types of individuals.

Second, to find the evolutionary approach (parameter setting and strategy) that can deliver a larger improvement in fitness values for each individual, we study the production process of potential individuals and learn from these previous successful experiences in generations of potential individuals to guide the evolution of future generations. The sources of potential individuals at the current generation ($P^G$) fall into two categories: potential parent individuals in the previous generation ($P^{G-1}$) and unpotential parent individuals in the previous generation ($UP^{G-1}$). We abbreviate these two cases as $P - P$ and $UP - P$ in the following.

Based on the above two points, we study the evolutionary approaches of $P - P$ and $UP - P$ to guide the evolution of future $P$ and $UP$ individuals, respectively.

In addition, an evolutionary approach that produces larger improvement in fitness values can be considered a more
appropriate evolutionary approach, and its selection probability is likely to be increased.

Third, we adapt $F$, $CR$, and the mutation strategy in a joint manner. In previous studies, adaptations of the control parameters and mutation strategy are often performed independently, with separate adaptive strategies, adaptive $F$, and adaptive $CR$. Certain algorithms combine the two control parameters, such as CoDE [25], EPSDE [11], and ZEPDE [12]. However, in the evolutionary process, the parameters and strategies are also related. For example, for the same strategy, it may be suitable to combine different parameter combinations for different individuals at different stages. For instance, sometimes a strategy requires a large $F$ and a large $CR$ combination, other times a strategy requires a small $F$ and a small $CR$ combination, a large $F$ and a small $CR$ combination or a small $F$ and a large $CR$ combination. The three elements interact with each other. However, previous studies have neglected the relationship among these three factors (control parameter $F$, control parameter $CR$, and mutation strategy). Considering the interrelationship among the three factors, our algorithm uses a joint adaptation method for $F$, $CR$, and mutation strategies to obtain the best combination of parameters and strategies for each individual in each generation.

Different from the fixed parameter value combinations of CoDE and EPSDE, and different from the only parameter zone combinations of ZEPDE, we divide the parameter interval into several segments and subsequently combine them with mutation strategies. For example, the $F$ interval is divided into $A$ segments, the $CR$ interval is divided into $B$ segments, and $C$ strategies are adopted; thus, there are $m = A^2B^2C$ combinations of the $F$ interval, $CR$ interval, and strategy. We use an artificial three-dimensional probability distribution array with $m = A^2B^2C$ elements to represent the joint probability distribution of $m$ combinations. The values of $A$, $B$, and $C$ can be determined by the users. For example, when $A = B = C = 3$, the three-dimensional probability distribution array with $m = 27$ elements can be shown as in Fig. 1. As Fig. 1 shows, the three dimensions are $F$ intervals, $CR$ intervals, and strategies. Each cell in the array represents the selection probability of the corresponding combination.

Based on previous discussions and the second point, a combination (an evolutionary approach) that produces a larger improvement in fitness values is considered a more appropriate combination (a more appropriate evolutionary approach), and we increase its selection probability, whether for $F$ or $UP$. The larger the improvement in fitness values produced by the combination is, the more obvious the superiority of the combination, and the greater its selection probability.

Fourth, to improve the robustness of JAPSPDE, we apply two historical memories, $Memory_{F-jp}$ and $Memory_{UP-jp}$, to store the joint probabilities of $m$ combinations in each generation for case $P-P$ and case $UP-P$, respectively. The size of both memories is equal to $H$. Unlike the memories that store values in SHADE, our memories store vectors/arrays. In SHADE, the value directly selected from the memory is the location parameter of the distribution. In our algorithm, a roulette wheel is applied to randomly select a combination according to the probability distribution array first selected from the memory. The strategy and the location parameters of the distributions are determined by the combination, which is described in detail in section IV-D.

**B. Initialization**

We initialize the joint probability of potential and unpotential individuals $jp_p \in \mathbb{R}^{27}$ and $jp_{UP} \in \mathbb{R}^{27}$ with $1/27$ and set all values in $Memory_{F-jp} \in \mathbb{R}^{27 \times H}$ and $Memory_{UP-jp} \in \mathbb{R}^{27 \times H}$ to $jp_p$ and $jp_{UP}$, respectively, with $ps = 0.35$ and $NP_{init} = 150$. In this paper, we adopt the LPSR population reduction scheme used in L-SHADE.

In this paper, the range of $F$ is $[0.4, 1]$. We divide the suggested range of $F$ into three equal segments of $[0.4, 0.6)$, $[0.6, 0.8)$, and $[0.8, 1)$. The range of $CR$ is $[0, 1]$. We divide the range of $CR$ into three equal segments of $[0, 1/3)$, $[1/3, 2/3)$, and $[2/3, 1]$. Three mutation strategies with different searching characteristics are applied. Thus, we combine the $F$ interval, $CR$ interval and mutation strategy into a 3D probability distribution array with $(3^3 \times 3) = 27$ elements. The 27 combinations are shown in Table 1. For parameter $F$, “$1$”, “$2$”, and “$3$” represent the intervals $[0.4, 0.6)$, $[0.6, 0.8)$, and $[0.8, 1)$, respectively. For parameter $CR$, “$1$”, “$2$”, and “$3$” represent the intervals $[0, 1/3)$, $[1/3, 2/3)$, and $[2/3, 1)$, respectively. For mutation strategy $st$: “$1$”, “$2$”, and “$3$” represent strategies “DE/rand-to-pbest/bin/1”, “DE/current/bin/1”, and “DE/current-to-pbest/bin/1 with the archive”, respectively.

The three mutation strategies adopted in our algorithm are DE/rand-to-p best/bin/1, DE/current-to-p best/bin/1 with the archive, and DE/current/bin/1. We adopt three mutation strategies with different characteristics. The first strategy has a good exploration ability, the second has a good exploitation ability, and the third balances exploration and exploitation.

The initial population is randomly generated.
C. JOINT ADAPTATION

In the first three generations, for each individual, two control parameters are randomly generated in their own ranges. Each strategy in the strategy pool is allocated to individuals with equal probability.

From the fourth generation on, joint adaptation is adopted. At first, all combinations are assigned to all individuals with equal probability.

In each generation, for every individual, two control parameters are randomly generated in their own ranges. Each strategy in the strategy pool is allocated to individuals with equal probability. For each individual, two control parameters are likely to be located near the selected location parameters.

The specific process is described as follows. The combination index is

\[ C_i = index. \]  

The \( F \) interval number is

\[ f_i = (C_i - 1) \text{div} (9) + 1. \]

Thus, the lower limit and upper limit of the \( F \) interval are

\[ F_{\text{low}, i} = 0.2 \times (f_i - 1) + 0.4, \]

\[ F_{\text{up}, i} = 0.2 \times f_i + 0.4. \]

The location parameter of the Cauchy distribution is randomly generated from this interval

\[ M_{F,i} = \text{rand}(F_{\text{low}, i}, F_{\text{up}, i}). \]

The Cauchy distribution is adopted to generate \( F \),

\[ F_i^G = \text{randCauchy}(M_{F,i}, 0.1). \]

If \( CR\text{flag}(i) = 1 \), then \( CR_i^G = 0 \); otherwise, the \( CR \) interval number is \( cr_i = ((C_i - 1) \mod (9)) \text{div}(3) + 1 \).

In the second case, the lower limit and upper limit of the \( CR \) interval are

\[ CR_{\text{low}, i} = (cr_i - 1)/3, \]

\[ CR_{\text{up}, i} = cr_i/3. \]

The location parameter of the normal distribution is randomly generated from this interval

\[ M_{CR,i} = \text{rand}(CR_{\text{low}, i}, CR_{\text{up}, i}). \]

The normal distribution is adopted to generate \( CR \),

\[ CR_i^G = \text{randnormal}(M_{CR,i}, 0.1). \]

The mutation strategy number is determined by

\[ st_i^G = (C_i - 1) \mod (3) + 1. \]

If \( st \) equals 1, then strategy “DE/rand-to-pbest/bin/1” is applied. If \( st \) equals 2, then strategy “DE/current/bin/1” is applied. If \( st \) equals 3, then strategy “DE/current-to-pbest/bin/1 with the archive” is applied.

D. PARAMETERS AND STRATEGY GENERATION

Once the combination is determined, the \( F \), \( CR \), and strategy are determined as follows.

As the combination of \( x_i^G \) is determined, its \( F \) interval, \( CR \) interval, and strategy are determined. We use a Cauchy distribution to generate \( F \) and a normal distribution to generate \( CR \). A random value from the selected \( F \) interval is assigned to the location parameter of the Cauchy distribution. A random value from the selected \( CR \) interval is assigned to the location parameter of the normal distribution. Therefore, the newly generated control parameters are likely to be located near the selected location parameters.

TABLE 1. 27 combinations of \( F \), \( CR \), and strategy.

| combination | \( F \) | \( CR \) | \( st \) |
|-------------|--------|--------|--------|
| 1           | 1      | 1      | 1      |
| 2           | 1      | 1      | 2      |
| 3           | 1      | 1      | 3      |
| 4           | 1      | 2      | 1      |
| 5           | 1      | 2      | 2      |
| 6           | 1      | 2      | 3      |
| 7           | 1      | 3      | 1      |
| 8           | 1      | 3      | 2      |
| 9           | 1      | 3      | 3      |
| 10          | 2      | 1      | 1      |
| 11          | 2      | 1      | 2      |
| 12          | 2      | 1      | 3      |
| 13          | 2      | 2      | 1      |
| 14          | 2      | 2      | 2      |
| 15          | 2      | 2      | 3      |
| 16          | 2      | 3      | 1      |
| 17          | 2      | 3      | 2      |
| 18          | 2      | 3      | 3      |
| 19          | 3      | 1      | 1      |
| 20          | 3      | 1      | 2      |
| 21          | 3      | 1      | 3      |
| 22          | 3      | 2      | 1      |
| 23          | 3      | 2      | 2      |
| 24          | 3      | 2      | 3      |
| 25          | 3      | 3      | 1      |
| 26          | 3      | 3      | 2      |
| 27          | 3      | 3      | 3      |

E. JOINT PROBABILITY UPDATE

For each individual, we calculate its improvement of fitness values for two consecutive generations,

\[ \Delta f_i^G(i) = f(x_i^G-1) - f(x_i^G), \]

\[ \Delta f^{G-1}(i) = f(x_i^{G-2}) - f(x_i^{G-1}). \]

We judge its category (\( P^G \) or \( UP^G \)), and its parent’s category (\( P^{G-1} \) or \( UP^{G-1} \)). If \( \Delta f_i^G(i) \) is ranked \( ps \times 100\% \) and \( \Delta f_i^{G-1}(i) \neq 0 \), then the target individual is a potential individual in the current generation (\( P^G \)), and if not, then the target individual is an unpotential individual in the current generation (\( UP^G \)). If \( \Delta f_i^{G-1}(i) \) is ranked \( ps \times 100\% \) and
\( \Delta f^{G-1}(i) \neq 0 \), then the target individual is a potential individual in the previous generation \( (P^{G-1}) \), and if not, then the target individual is an unpotential individual in the previous generation \( (UP^{G-1}) \).

If \( x_i^G \) is potential in the \( G \)th generation, we record its used combination number \( k \) and check whether its parent individual \( x_{i-1}^{G-1} \) is potential or unpotential. If \( x_{i-1}^{G-1} \) belongs to \( P^{G-1} \), then it is a \( P-P \) case. The number of the corresponding combination for the \( P-P \) case increases by 1. If \( x_{i-1}^{G-1} \) belongs to \( UP^{G-1} \), then it is a \( UP-P \) case. The number of the corresponding combination for the \( UP-P \) case increases by 1.

The pseudocode of the joint probability update process is presented in Algorithm 1.

**Algorithm 1 Joint Probability Update Progress**

\[
\begin{align*}
    N_{P-P} & = \text{zeros}(27, 1); N_{UP-P} = \text{zeros}(27, 1); \Delta f_{P-P} = \text{zeros}(27, 1); \\
    \Delta f_{UP-P} & = \text{zeros}(27, 1); CR_P = []; CR_{UP} = [].
\end{align*}
\]

for \( i = 1 \) to \( NP_{\text{now}} \) do

\[
\begin{align*}
    & \text{calculate } \Delta f^G(i) \text{ and } \Delta f^{G-1}(i); \\
    & \text{if } \text{ith individual is potential in } G \text{th generation:} \\
    & \quad k = C_i; \% \text{ the combination number that adopted by ith individual} \\
    & \text{if } \text{ith individual is potential in } (G-1) \text{th generation:} \\
    & \quad CR_P.append(CR_i); \\
    & \quad N_{P-P,k}++; \Delta f_{P-P,k} = \Delta f_{P-P,k} + \Delta f(i); \\
    & \text{else:} \\
    & \quad CR_{UP}.append(CR_i); \\
    & \quad N_{UP-P,k}++; \Delta f_{UP-P,k} = \Delta f_{UP-P,k} + \Delta f(i); \\
    & \text{end if}
\end{align*}
\]

end if

end for

for \( l = 1 \) to 27 do

\[
\begin{align*}
    Aver_{\text{impro}}(AVI) & = \Delta f_{P-P,l}/(N_{P-P,l} + 1E - 200); \%
\end{align*}
\]

\[
\text{AVI}_{UP-P,l} = \Delta f_{UP-P,l}/(N_{UP-P,l} + 1E - 200)
\]

end for

for \( l = 1 \) to 27 do

\[
\begin{align*}
    jp_{P,l} & = AVI_{P-P,l}/\text{sum(AVI}_{P-P}); \\
    jp_{UP,l} & = AVI_{UP-P,l}/\text{sum(AVI}_{UP-P});
\end{align*}
\]

end for

**G. COMPLETE PROCEDURE OF THE PROPOSED JAPSPDE**

The complete pseudocode of JAPSPDE is presented in Algorithm 3.

**Algorithm 3 JAPSPDE**

\[
\begin{align*}
\text{Initialization phase:} \quad & \text{Initialize } jp_P \in R^{27} \text{ and } jp_{UP} \in R^{27} \text{ with } 1/27; \\
& \text{Set all values in } Memory_{P-jp} \in R^{27\times H} \text{ and } Memory_{UP-jp} \in R^{27\times H} \text{ to } jp_P \text{ and } jp_{UP}, \text{ respectively;} \\
& \text{Set all values in } Memory_{P-\text{CRflag}} \in R^H \text{ and } Memory_{UP-\text{CRflag}} \in R^H \text{ to } 0; \\
& \text{FEs} = 0; G = 1; A = \phi; NP_{\text{init}} = 150; NP^G = NP_{\text{init}}; \\
& \text{Index counter } k_1 = 1, k_2 = 1; CR_P = \phi, CR_{UP} = \phi; \\
& \text{Initialize population randomly;}
\end{align*}
\]

\[
\text{Main loop:} \quad \text{while } FEs < FEs_{\text{max}} \text{ (the termination criterion is not met) do}
\]

\[
\begin{align*}
NP_{\text{now}} & = NP^G; \\
& \text{for } i = 1 \text{ to } NP_{\text{now}} \text{ do}
\end{align*}
\]

\[
\begin{align*}
    & r_1 = \text{randint}(1, H) \\
    & \text{if } \text{ith individual is potential:}
\end{align*}
\]

\[
\begin{align*}
    & \text{if } Memory_{P-\text{CRflag}}(r_1) = 1, \text{ then } CR_{\text{flag}}(i) = 1; \\
    & \text{end if}
\end{align*}
\]

\[
\begin{align*}
    & p_1 = Memory_{P-jp}(.; r_1); \\
    & \text{else:}
\end{align*}
\]

\[
\begin{align*}
    & \text{if } Memory_{UP-\text{CRflag}}(r_1) = 1, \text{ then } CR_{\text{flag}}(i) = 1; \\
    & \text{end if}
\end{align*}
\]

\[
\begin{align*}
    & p_1 = Memory_{UP-jp}(.; r_1); \\
    & \text{end if}
\end{align*}
\]

\[
\begin{align*}
    & r = \text{rand}; cump = 0; index = 0;
\end{align*}
\]
while $cump < r$ do
  index += +;
  $cump = cump + p_i$(index);
end while

$C_i = index$; % combination number

$f_i = (C_i - 1) \text{div}(9) + 1; F_{low,i} = 0.2 \times (f_i - 1) + 0.4; F_{up,i} = 0.2 \times f_i + 0.4$;

$M_F,i = rand(F_{low,i}, F_{up,i})$;

$F_G = rand_{\text{cauchy}}(M_F,i,0.1)$; % $F$

if $CRflag(i) = 1$, then
  $CR^G = 0$;
else
  $cr_i = ((C_i - 1) \text{mod}(9)) \text{div}(3) + 1$;
  $CR_{low,i} = (cr_i - 1)/3$;
  $CR_{up,i} = cr_i/3$;
  $M_{CR,i} = rand(CR_{low,i}, CR_{up,i})$;
  $CR^G = rand_{\text{normal}}(M_{CR,i},0.1)$;
end if % $CR$

$sft^G = (C_i - 1) \text{mod}(3) + 1$; % mutation strategy

// mutation
if $sft^G = 1$, then
  $x^G_i = \frac{x^{G,j}}{R_i} + \frac{x^{G,pbest}}{R_i} + F_G \cdot (x^{G,j} - x^{G,pbest})$;
else $sft^G = 2$, then
  $x^G_i = x^{G,j} + F_G \cdot (x^{G,pbest} - x^{G,j}) + F_G \cdot (x^{G,j} - x^{G,pbest})$;
else
  $x^G_i = x^{G,j} + F_G \cdot (x^{G,pbest} - x^{G,j}) + F_G \cdot (x^{G,j} - x^{G,pbest})$;
end if // crossover
for $j = 1$ to $D$
  $rand = \text{randint}(1,D)$;
if $rand(0,1) < CR^G$ or $j == rand$, then
  $u^G_i = x^G_i$;
else
  $u^G_i = x^{G,j}$
end if
end for

// selection
for $i = 1$ to $NP_{now}$ do
  if $f(u^G_i) < f(x^G_i)$, then
    $x^{G+1}_i = u^G_i$;
    $f(x^{G+1}_i) = f(u^G_i)$;
  else
    $x^{G+1}_i = x^G_i$;
    $f(x^{G+1}_i) = f(x^G_i)$;
end if
if $f(u^G_i) < f(x^G_i)$, then
  $x^G_i \rightarrow A$;
end if

// update $p_p$ and $p_{UP}$ as in Algorithm 1
// update $Memory_{p_p}, Memory_{UP-p_p}, Memory_{p_p-CRflag}$ and $Memory_{UP-UP}$ as in Algorithm 2.
$G += +$;
// optional LPSR strategy

Calculate $NP^G$, $NP_{now} = NP^G$;
if $NP_{now} < NP^{G-1}$, then
  sort individuals in $X^G$ base on their fitness values and delete lowest $(NP^{G-1} - NP_{now})$ members;
  resize the archive size;
end if
end while

H. TIME COMPLEXITY OF JAPSPDE

The time complexity of JAPSPDE mainly comes from two aspects: one is the mutation, crossover and selection operations of DE algorithm, and the other is the determination of the joint probabilities and the adaptive adjustment of the population’s mutation strategies and control parameters.

In the differential evolution algorithm, the time complexity mainly comes from the number of iterations, the population size $NP$ and the number of decision variables $D$. Then the time complexity of each iteration is $O(NP^G \times D)$.

The time complexity of the second part mainly comes from updating the joint probabilities for potential individuals and unpotential individuals respectively. Then the time complexity of each iteration is $O(NP^G \times 27)$.

Therefore, the time complexity of JAPSPDE is as follows,

$$
O \left( \max \left( NP^1 \times D + \ldots NP^{G_{max}} \times D \right) \right),
$$

$$
= O \left( \max \left( NP^1 \times 27 + \ldots NP^{G_{max}} \times 27 \right) \right),
$$

$$
= O \left( \max \left( \left( NP^1 + \ldots + NP^{G_{max}} \right) \times 27 \right) \right).
$$ (18)

V. EXPERIMENTAL RESULTS

The validity of the proposed algorithm is tested on two benchmark sets: CEC2014 [33] and BBOB2012 [34]. JAPSPDE is compared with five well-known DE algorithms: CoDE [25], EPSDE [11], JADE [19], jDE [17], and SaDE [18]. Additionally, JAPSPDE is also compared with six state-of-date EAs: IDEI [26], ETT-ODE [35], ZEPDE [12], SHADE [20], M_PSO_MA [36] and QIAEA [37].

Section V-A introduces the experimental settings and the statistical analysis method used. In Section V-B, JAPSPDE is compared with five well-known DE algorithms on BB0B2012. In Section V-C, JAPSPDE is compared with five well-known DE algorithms on CEC2014. In Section V-D, JAPSPDE is compared with six state-of-the-art EAs performed on CEC2014.

A. EXPERIMENT SETTINGS AND STATISTICAL ANALYSIS METHODS

The experimental results are obtained by 30 independent runs. The maximum number of function evaluations (MaxFEs) is set as 10000*D. The accuracy of the optimization result is 1E-8, i.e., when it is smaller than 1E-8, it can be
treated as 0. All experiments are executed under the Windows 10 operating system with MATLAB R2017b.

To perform an effective analysis of the experimental results, two nonparametric statistical methods and three post hoc procedures are selected. The significance level is set to 0.05. The two nonparametric statistical methods are Wilcoxon’s rank sum test and Friedman’s test. The three post hoc procedures are Bonferroni–Dunn’s test, Bonferroni–Holm’s procedure, and Hochberg’s procedure.

For Wilcoxon’s rank sum test, “+”, “−”, and “≈” indicate that the effect of JAPSPDE is obviously superior to, inferior to, or similar to that of the comparison algorithm, respectively. For Friedman’s test, the algorithms with smaller rankings have better performance. For the three post hoc procedures, a significant difference exists between the control and compared algorithms when the \( p \)-value is less than 0.05.

### B. PERFORMANCE COMPARISON WITH FIVE WELL-KNOWN DES ON BBOB2012

In this experiment, we use 24 BBOB2012 functions with 30D to compare the effect of JAPSPDE with five well-known differential DE algorithms (i.e., jDE, JADE, SaDE, CoDE, and EPSDE).

We note that the experimental data of the five DE algorithms are obtained from [38]. The experimental results (the mean error) of the six algorithms on BBOB2012 with 30D are shown in supplementary Table 13, in which the best solutions of each function are given in bold font.

From Table 13, we observe that for five separable functions (F1\(_{\text{BBOB2012}}\) – F5\(_{\text{BBOB2012}}\)), JADE performs the best because of the use of a greedy strategy, but JAPSPDE also performs better than the other five opponents. For nine unimodal functions (F6\(_{\text{BBOB2012}}\) – F14\(_{\text{BBOB2012}}\)), it is obvious that JAPSPDE is the algorithm with the best performance, and it obtains the best solution on F5, F6, F7, F8, F11, F12, F13, and F14. For ten multimodal functions (F15\(_{\text{BBOB2012}}\) – F24\(_{\text{BBOB2012}}\)), CoDE is the best-performing algorithm due to its excellent exploration capabilities, and JAPSPDE is also superior to the other five algorithms.

As Fig. 2 shows, the winning times of the six algorithms are 4, 5, 9, 11, 6, and 14, respectively. JAPSPDE obtains the greatest number of best results. In addition, Table 2 indicates that JAPSPDE obtains the best ranking among the compared algorithms. Table 3 gives the \( p \)-values calculated by three post hoc procedures and indicates that JAPSPDE is significantly better than SaDE, a adaptation method of the \( F \), \( CR \), and strategy.

### C. PERFORMANCE COMPARISON WITH FIVE WELL-KNOWN DES ON CEC2014

In this experiment, we use 30 CEC2014 test functions to compare the performance of JAPSPDE with the performance of five well-known DE algorithm variants. For persuasive comparison, we used the same parameters as in the original papers.

Supplementary Tables 14 and 15 show the optimization results of the six algorithms on CEC2014 with 30D and 50D, respectively. At the same time, the comparison results calculated by Wilcoxon’s rank sum test of each comparison algorithm and the proposed algorithm for each test function are also shown in Tables 14 and 15, respectively.

Table 14 indicates that JAPSPDE performs the best among all the comparison algorithms on CEC2014 with 30D. For unimodal functions, except for the case in which JADE performs better on F1, the experimental results of JAPSPDE are almost superior to all comparison algorithms on all functions. For simple multimodal functions, CoDE and JADE perform better than JAPSPDE. However, the proposed algorithm has better results than jDE, SaDE, and EPSDE. Because CoDE and JADE balance convergence and diversity well, their performance on simple multimodal functions is better than that of other algorithms. For hybrid functions, CoDE does not outperform JAPSPDE on any function except F19. In addition, on other hybrid functions, the optimization results of JAPSPDE are no worse than the results of the other algorithms on any function. For composition functions, the optimization results of JAPSPDE do not exceed those of JAPSPDE in any function.

![FIGURE 2. Number of the best results obtained by each algorithm on BBOB2012 with 30D.](image-url)
JAPSPDE and EPSDE performance comparison, JAPSPDE is superior to EPSDE on 2 functions, and EPSDE is better than JAPSPDE on 4 functions.

Similar to the results presented in Table 14, those presented in Table 15 indicate that JAPSPDE achieves the best average performance on CEC2014 with 50D. For unimodal functions, except for the case in which JADE performs better on F1, the experimental results of JAPSPDE are not inferior to those of any algorithm on any function. For simple multimodal functions, CoDE and JADE perform better because of their good balancing ability. However, JAPSPDE has better results than jDE, SaDE, and EPSDE. For hybrid functions, CoDE does not outperform JAPSPDE on any function except F19. In addition, the optimization results of JAPSPDE are almost always the best among all the algorithms on all hybrid functions. For composition functions, the optimization results of jDE, SaDE, JADE, and CoDE do not exceed those of JAPSPDE in any function except F19. In addition, the optimization results of JAPSPDE are almost always the best among all the algorithms on all hybrid functions.

Table 4 presents contrasting outcomes calculated by Wilcoxon’s rank sum test, which indicate that JAPSPDE outperforms the compared algorithms on most CEC2014 functions. More specifically, with respect to 30D functions, JAPSPDE is significantly superior to jDE, SaDE, JADE, EPSDE, and CoDE in 17, 24, 13, 17, and 10 functions out of 30 functions, respectively, and for the 50D functions, JAPSPDE significantly outperforms the five DE algorithms on 19, 25, 14, 21, and 14 functions, respectively. Compared with the 30-dimensional problem, JAPSPDE has more obvious advantages in the 50-dimensional problems.
Table 5 lists the rankings calculated by Friedman’s test of the six algorithms on CEC2014 with 30 dimensions and 50 dimensions, indicating that JAPSPDE is the best algorithm for both the 30D and 50D problems. Tables 6 and 7 present the p-values obtained by three post hoc procedures on CEC2014 with 30 dimensions and 50 dimensions, respectively. From Table 6, we note that JAPSPDE significantly outperforms jDE, SaDE, and EPSDE in the 30D problems. Table 7 indicates that JAPSPDE significantly outperforms EPSDE and SaDE in the 50D problems.

In summary, the experimental results of the six algorithms on the 30-dimensional and 50-dimensional CEC2014 benchmark set indicate that JAPSPDE performs better than or at least competitively with the other five well-known DEs.

### D. PERFORMANCE COMPARISON WITH SIX LATEST EAS ON CEC2014

In addition, six up-to-date EAs (i.e., IDEI, ETI-ODE, ZEPDE, SHADE, M_PSO_MA and QIAEA) are chosen for comparison.
for comparison with our algorithm on the 30-dimensional CEC2014 benchmark set.

We note that the experimental data of IDEI, ETI-ODE, ZEPDE, SHADE, M_PSO_MA and QIAEA are obtained from the original articles. The experimental results (the mean error) on CEC2014 with 30D are shown in supplementary Table 16, where the best solutions of each function are given in bold font.

Table 8 shows the rankings of the seven algorithms according to Friedman's test. From Table 8, we observe that JAPSPDE is the best-performing algorithm among these comparison algorithms.
TABLE 15. Results of JAPSPDE and five well-known DEs on CEC2014 with 50D.

| fun         | CoDE      | EPSDE     | JADE      | jDE        | SaDE       | JAPSPDE     |
|-------------|-----------|-----------|-----------|------------|------------|-------------|
| F1_{CEC2014}| 2.16E+05  | 2.18E+06  | 1.79E+04  | 4.71E+05   | 8.78E+05   | 3.58E+04    |
| F2_{CEC2014}| 9.03E+01  | 2.21E+08  | 1.10E+14  | 7.77E+09   | 3.96E+03   | 5.97E+14    |
| F3_{CEC2014}| 1.73E+01  | 3.62E+04  | 4.13E+03  | 7.43E+09   | 4.71E+03   | 2.96E+13    |
| F4_{CEC2014}| 3.15E+01  | 3.52E+03  | 1.62E+01  | 8.01E+01   | 8.04E+01   | 9.81E+00    |
| F5_{CEC2014}| 2.01E+01  | 2.06E+01  | 2.04E+01  | 2.04E+01   | 2.07E+01   | 2.06E+01    |
| F6_{CEC2014}| 7.94E+00  | 4.57E+01  | 1.65E+01  | 2.59E+01   | 9.44E+01   | 7.07E+06    |
| F7_{CEC2014}| 3.20E+03  | 1.03E+02  | 2.22E+03  | 2.31E+13   | 1.81E+02   | 0.00E+00    |
| F8_{CEC2014}| 2.98E+01  | 3.79E+01  | 3.79E+15  | 9.09E+14   | 2.45E+06   | 1.18E+14    |
| F9_{CEC2014}| 7.29E+00  | 5.33E+01  | 5.11E+01  | 9.86E+01   | 9.67E+01   | 7.64E+01    |
| F10_{CEC2014}| 3.19E+00  | 8.30E+02  | 1.29E+01  | 1.25E+03   | 1.40E+01   | 1.83E+01    |
| F11_{CEC2014}| 4.37E+00  | 9.37E+03  | 3.84E+03  | 1.31E+04   | 6.14E+01   | 6.32E+01    |
| F12_{CEC2014}| 9.25E+00  | 8.42E+01  | 3.05E+01  | 1.50E+03   | 1.98E+01   | 8.08E+01    |
| F13_{CEC2014}| 3.41E+00  | 3.61E+01  | 3.12E+01  | 3.85E+01   | 4.63E+01   | 3.03E+01    |
| F14_{CEC2014}| 2.79E+00  | 3.30E+01  | 3.29E+01  | 3.23E+01   | 3.09E+01   | 3.08E+01    |
| F15_{CEC2014}| 6.75E+00  | 1.77E+00  | 7.27E+00  | 1.16E+01   | 1.17E+01   | 1.19E+01    |
| F16_{CEC2014}| 1.83E+00  | 2.09E+01  | 1.77E+02  | 1.83E+01   | 2.02E+01   | 1.93E+01    |
| F17_{CEC2014}| 1.52E+02  | 2.36E+02  | 2.36E+02  | 2.20E+04   | 7.92E+04   | 2.32E+03    |
| F18_{CEC2014}| 3.15E+02  | 3.46E+03  | 3.46E+03  | 3.62E+03   | 5.35E+02   | 8.42E+01    |
| F19_{CEC2014}| 6.80E+00  | 2.41E+01  | 1.19E+01  | 1.27E+01   | 1.82E+01   | 1.08E+01    |
| F20_{CEC2014}| 1.58E+02  | 4.84E+02  | 7.90E+00  | 5.74E+01   | 1.11E+03   | 4.79E+01    |
| F21_{CEC2014}| 5.48E+03  | 8.25E+04  | 1.24E+04  | 1.07E+04   | 1.02E+05   | 7.20E+02    |
| F22_{CEC2014}| 6.40E+02  | 7.43E+02  | 7.52E+02  | 5.82E+02   | 4.36E+02   | 2.88E+02    |
| F23_{CEC2014}| 3.44E+02  | 3.37E+02  | 3.44E+02  | 3.44E+02   | 3.44E+02   | 3.44E+02    |
| F24_{CEC2014}| 2.72E+02  | 2.73E+02  | 2.75E+02  | 2.69E+02   | 2.77E+02   | 2.73E+02    |
| F25_{CEC2014}| 2.08E+02  | 2.02E+02  | 2.18E+02  | 2.07E+02   | 2.15E+02   | 2.06E+02    |
| F26_{CEC2014}| 1.10E+02  | 1.00E+02  | 1.04E+02  | 1.00E+02   | 1.00E+02   | 1.00E+02    |
| F27_{CEC2014}| 5.55E+02  | 1.55E+03  | 4.41E+02  | 4.41E+02   | 8.58E+02   | 3.41E+02    |
| F28_{CEC2014}| 1.19E+03  | 1.13E+03  | 1.10E+03  | 1.44E+03   | 1.14E+03   | 1.14E+03    |
| F29_{CEC2014}| 9.33E+02  | 2.28E+02  | 8.83E+02  | 1.00E+03   | 1.54E+03   | 8.02E+02    |
| F30_{CEC2014}| 8.77E+03  | 1.09E+03  | 9.60E+03  | 8.61E+03   | 1.19E+04   | 8.96E+03    |

Note: The CEC2014 benchmark set contains three unimodal functions (F1_{CEC2014} – F3_{CEC2014}), thirteen simple multimodal functions (F4_{CEC2014} – F16_{CEC2014}), six hybrid functions (F17_{CEC2014} – F22_{CEC2014}), and eight composition functions (F23_{CEC2014} – F30_{CEC2014}). Tables 9-12 show the Friedman’s rankings of the seven algorithms, from which we note that JAPSPDE performs better on hybrid and composition functions.

E. EFFECTIVENESS OF JAPSPDE

A 30-D F10_{CEC2014} is adopted to verify the effectiveness of the proposed adaptive mechanism. Fig. 3 shows the
TABLE 16. Results of PDE and six up-to-date DEs on CEC2014 with 30D.

| fun     | IDEI     | ETI-ODE | ZEPDE   | SHADE   | M_PSO_MA | QIAEA | JAPSPDE |
|---------|----------|---------|---------|---------|-----------|-------|---------|
| F1_CEC2014 | 9.57E+03 | 1.31E+05 | 1.24E+04 | 3.04E+02 | 3.09E+03 | 9.97E+05 | 1.72E+03 |
| F2_CEC2014 | 0.00E+00 | 1.00E+03 | 2.84E-14 | 0.00E+00 | 0.00E+00 | 4.20E+02 | 0.00E+00 |
| F3_CEC2014 | 0.00E+00 | 0.00E+00 | 6.82E-14 | 0.00E+00 | 0.00E+00 | 2.92E+03 | 0.00E+00 |
| F4_CEC2014 | 4.47E-01 | 4.98E+00 | 1.89E-04 | 0.00E+00 | 5.29E+00 | 8.86E+02 | 1.31E-13 |
| F5_CEC2014 | 2.09E+01 | 2.04E+01 | 2.02E+01 | 2.01E+01 | 2.00E+01 | 1.02E+03 | 2.04E+01 |
| F6_CEC2014 | 5.65E-01 | 6.09E-01 | 2.50E+00 | 1.00E+00 | 1.05E+01 | 1.20E+03 | 6.49E-00 |
| F7_CEC2014 | 0.00E+00 | 9.66E-04 | 1.25E-13 | 0.00E+00 | 1.22E-02 | 1.40E+03 | 0.00E+00 |
| F8_CEC2014 | 3.18E-01 | 3.37E+01 | 0.00E+00 | 0.00E+00 | 1.03E+00 | 1.62E+03 | 0.00E+00 |
| F9_CEC2014 | 2.15E+01 | 2.27E+01 | 3.52E+01 | 1.60E+01 | 7.63E+01 | 1.84E+03 | 3.28E+01 |
| F10_CEC2014 | 1.57E+02 | 1.25E+03 | 1.24E+00 | 5.41E-03 | 4.54E+02 | 6.45E+03 | 1.38E-00 |
| F11_CEC2014 | 3.06E+03 | 1.86E+03 | 2.11E+03 | 1.47E+03 | 2.59E+03 | 6.41E+03 | 2.58E-03 |
| F12_CEC2014 | 2.42E+00 | 2.14E-01 | 1.75E-01 | 1.65E-01 | 8.06E-02 | 2.40E+03 | 6.19E-01 |
| F13_CEC2014 | 2.06E-01 | 2.32E-01 | 1.77E-01 | 2.11E-01 | 3.29E-01 | 2.60E+03 | 2.25E-01 |
| F14_CEC2014 | 2.70E-01 | 2.41E-01 | 2.17E-01 | 2.39E-01 | 2.16E-01 | 2.80E+03 | 2.67E-01 |
| F15_CEC2014 | 3.21E+00 | 3.56E+00 | 3.04E+00 | 2.52E+00 | 9.90E+00 | 3.01E+03 | 5.19E+00 |
| F16_CEC2014 | 1.07E+01 | 9.96E+00 | 1.00E+00 | 9.13E+00 | 1.03E+00 | 3.21E+03 | 1.03E+01 |
| F17_CEC2014 | 9.13E+02 | 5.68E+02 | 7.68E+02 | 1.06E+03 | 1.75E+03 | 2.83E+05 | 3.37E+02 |
| F18_CEC2014 | 1.77E+01 | 9.23E+00 | 2.89E+01 | 6.52E+02 | 2.14E+03 | 3.91E+03 | 1.48E+01 |
| F19_CEC2014 | 2.47E+00 | 2.38E+00 | 2.54E+00 | 4.65E+00 | 7.35E+00 | 3.80E+03 | 3.30E+00 |
| F20_CEC2014 | 7.91E+00 | 9.50E+00 | 8.97E+00 | 1.46E+01 | 5.56E+01 | 5.85E+03 | 8.93E+00 |
| F21_CEC2014 | 1.79E+02 | 2.89E+02 | 1.95E+02 | 2.90E+02 | 1.89E+03 | 4.88E+04 | 1.89E+02 |
| F22_CEC2014 | 1.23E+02 | 2.27E+02 | 2.39E+01 | 1.04E+02 | 2.80E+02 | 4.61E+03 | 5.33E+01 |
| F23_CEC2014 | 3.15E+02 | 3.15E+02 | 3.15E+02 | 3.15E+02 | 3.15E+02 | 4.92E+03 | 3.15E+02 |
| F24_CEC2014 | 2.24E+02 | 2.16E+02 | 2.23E+02 | 2.25E+02 | 2.24E+02 | 5.02E+03 | 2.23E+02 |
| F25_CEC2014 | 2.03E+02 | 2.03E+02 | 2.04E+02 | 2.03E+02 | 2.04E+02 | 5.21E+03 | 2.03E+02 |
| F26_CEC2014 | 1.00E+02 | 1.00E+02 | 1.00E+02 | 1.00E+02 | 1.00E+02 | 5.03E+03 | 1.00E+02 |
| F27_CEC2014 | 3.66E+02 | 3.93E+02 | 3.31E+02 | 3.15E+02 | 6.70E+02 | 5.74E+03 | 3.00E+02 |
| F28_CEC2014 | 7.97E+02 | 8.03E+02 | 8.00E+02 | 8.32E+02 | 1.13E+03 | 6.44E+03 | 7.92E+02 |
| F29_CEC2014 | 7.15E+02 | 6.99E+02 | 7.36E+02 | 7.16E+02 | 1.73E+03 | 7.27E+03 | 7.30E+02 |
| F30_CEC2014 | 8.96E+02 | 5.90E+02 | 8.55E+02 | 1.75E+03 | 2.52E+03 | 7.53E+03 | 6.81E+02 |

probability change curves of the dominant combinations for potential individuals and unpotential individuals in the evolution process. It can be seen from Fig. 3 (a) that in the early and middle evolution stages, the 20th and 21st combinations play a leading role in the evolution for potential individuals. In the later evolution stage, the 27th combination played a leading role. Therefore, JAPSPDE has a better exploration ability in the early stage and a better exploitation ability in
the later stage. For unpotential individuals, multiple combinations play a leading role in the early and middle evolution stages, while the 7th combination plays a leading role in the later evolution stage. It can be concluded that all individuals cannot be treated equally. Therefore, dividing individuals into potential and unpotential individuals and evolve them separately is necessary.

VI. CONCLUSION

In the DE algorithm, searching for the best combination of control parameters and mutation strategy during evolution is a vital but difficult and time-consuming process. To settle this issue, this study proposes a potential-based DE algorithm with joint adaptation of parameters and strategies (JAPSPDE). In JAPSPDE, individuals are classified into potential and unpotential individuals and are treated differently. To achieve joint adaptation of parameters and strategies, the ranges of the control parameters are divided into several segments, and several mutation strategies combine them into several combinations. The combination that produces more improvement in fitness values has a better chance of being selected in future evolution, which increases the algorithm’s exploitation ability. To increase the robustness of the proposed algorithm, two historical memories that record historical successful combination are applied. Therefore, JAPSPDE can balance exploitation ability and exploration ability.

The performance of JAPSPDE is evaluated by comparison with five well-known DE variants on BBOB2012 and CEC2014 and by comparison with six up-to-date evolutionary algorithms on CEC2014. The comparison results prove the competitive performance of JAPSPDE.

For future work, two issues deserve further study. The number of combinations can be adjusted, and the application of JAPSPDE to practical tasks is also an interesting subject.

APPENDIX

See Tables 13–16.

REFERENCES

[1] R. Storn and K. V. Price, “Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces,” Int. Comput. Sci. Inst., Berkeley, CA, USA, Tech. Rep. TR-95-012, 1995.

[2] M. A. Elaziz, A. A. Ewees, R. A. Ibrahim, and S. Lu, “Opposition-based moth-flame optimization improved by differential evolution for feature selection,” Math. Comput. Simul., vol. 168, pp. 48–75, Feb. 2020.

[3] Y. Zhang, D.-W. Gong, X.-Z. Gao, T. Tian, and X.-Y. Sun, “Binary differential evolution with self-learning for multi-objective feature selection,” Inf. Sci., vol. 507, pp. 67–85, Jan. 2020.

[4] O. Tarkhanov and H. Shen, “An adaptive differential evolution algorithm to optimal multi-level thresholding for MRI brain image segmentation,” Expert Syst. Appl., vol. 138, Dec. 2019, Art. no. 112820.

[5] A. Bhuvanesh, S. T. J. Christia, S. Kannan, M. K. Pandian, and K. Gangatharan, “Application of differential evolution algorithm and its variants for solving energy storage technologies integrated generation expansion planning,” Iranian J. Sci. Technol., Trans. Electr. Eng., vol. 43, no. 4, pp. 883–896, Dec. 2019.

[6] X. Shen, D. Zou, N. Duan, and Q. Zhang, “An efficient fitness-based differential evolution algorithm and a constraint handling technique for dynamic economic emission dispatch,” Energy, vol. 186, Nov. 2019, Art. no. 115801.

[7] M. Kaur, V. Kang, and L. Li, “Color image encryption approach based on memetic differential evolution,” Neural Comput. Appl., vol. 31, no. 11, pp. 7975–7987, Nov. 2019.

[8] J. J. Guedes, M. F. Castoldi, A. Goedtel, C. M. Agulhari, and D. S. Sanches, “Differential evolution applied to line-connected induction motors stator fault identification,” Soft Comput., vol. 23, no. 21, pp. 11217–11226, Nov. 2019.

[9] S. A. Ajithapriyadarsini, P. M. Mary, and M. W. Inthuvarajan, “Automatic generation control of a multi-area power system with renewable energy source under deregulated environment: Adaptive fuzzy logic-based differential evolution (DE) algorithm,” Soft Comput., vol. 23, no. 22, pp. 12087–12101, Nov. 2019.

[10] 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.

[11] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, “Differential evolution algorithm with ensemble of parameters and mutation strategies,” Appl. Soft Comput., vol. 11, no. 2, pp. 1679–1696, Mar. 2011.

[12] 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.

[13] R. Storn and K. V. 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.

[14] K. V. Price, “Differential evolution: A fast and simple numerical optimizer,” in Proc. North Amer. Fuzzy Inf. Process. Jun. 1996, pp. 524–527.

[15] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. C. Coello, “A comparative study of differential evolution variants for global optimization,” in Proc. 8th Annu. Conf. Genetic Evol. Comput. (GECCO), Seattle, WA, USA, Jul. 2006, pp. 485–492.

[16] R. Gamarra, S. D. Müller, and A. Koumoutsakos, “A parameter study for differential evolution,” in Proc. Wseas Int. Conf. Adv. Intell. Syst., Fuzzy Syst., Evol. Comput., 2002, pp. 293–298.

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

[18] 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.

[19] 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.

[20] R. Tanabe and A. Fukunaga, “Success-history based parameter adaptation for differential evolution,” in Proc. IEEE Conge. Evol. Comput., Jun. 2013, pp. 71–78.

[21] R. Tanabe and A. S. Fukunaga, “Improving the search performance of SHADE using linear population size reduction,” in Proc. IEEE Congr. Evol. Comput., Jul. 2014, pp. 1655–1665.

[22] D. Dawar and S. A. Ludwig, “Effect of strategy adaptation on differential evolution in presence and absence of parameter adaptation: An investigation,” J. Artif. Intell. Soft Comput. Res., vol. 8, no. 3, pp. 211–235, Jul. 2018.

[23] G. Wu, R. Mallipeddi, P. N. Suganthan, R. Wang, and H. Chen, “Differential evolution with multi-population based ensemble of mutation strategies,” Inf. Sci., vol. 329, pp. 329–345, Feb. 2016.
[24] L. Tong, M. Dong, and C. Jing, “An improved multi-population ensemble differential evolution,” *Neurocomputing*, vol. 290, pp. 130–147, May 2018.

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

[26] M. Tian, X. Gao, and C. Dai, “Differential evolution with improved individual-based parameter setting and selection strategy,” *Appl. Soft Comput.*, vol. 56, pp. 286–297, Jul. 2017.

[27] 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)*, Jul. 2016, pp. 3170–3177.

[28] Q. Fan, X. Yan, and Y. Xue, “Prior knowledge guided differential evolution,” *Soft Comput.*, vol. 21, no. 22, pp. 1–18, 2017.

[29] A. W. Mohamed and P. N. Suganthan, “Real-parameter unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation,” *Soft Comput.*, vol. 22, no. 10, pp. 3215–3235, May 2018.

[30] Q. Fan, X. Yan, Y. Zhang, and C. Zhu, “A variable search space strategy based on sequential trust region determination technique,” *IEEE Trans. Cybern.*, early access, May 16, 2019, doi: 10.1109/TCYB.2019.2914060.

[31] G. Wu, W. Peng, X. Hu, R. Wang, and H. Chen, “Configuring differential evolution adaptively via path search in a directed acyclic graph for data clustering,” *Swarm Evol. Comput.*, vol. 55, Jun. 2020, Art. no. 100690.

[32] 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.

[33] J. J. Liang, B.-Y. Qu, and P. N. Suganthan, “Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization,” Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou, China, Tech. Rep. 201311, 2013.

[34] N. Hansen, A. Auger, S. Finck, and R. Ros, “Real-parameter black-box optimization benchmarking: Experimental setup,” INRIA Futurs, Équipe TAO, Univ. Paris-Sud, Orsay, France, Tech. Rep. RR-6828, 2013.

[35] W. Du, S. Y. S. Leung, Y. Tang, and A. V. Vasilakos, “Differential evolution with event-triggered impulsive control,” *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 244–257, Jan. 2017.

[36] G. Singh and K. Deep, “Effectiveness of new multiple-PSO based membrane optimization algorithms on CEC 2014 benchmarks and Iris classification,” *Natural Comput.*, vol. 16, no. 3, pp. 473–496, 2017.

[37] Y. Yang, X. Zong, D. Yao, and S. Li, “Improved Alopex-based evolutionary algorithm (AEA) by quadratic interpolation and its application to kinetic parameter estimations,” *Appl. Soft Comput.*, vol. 51, pp. 23–38, Feb. 2017.

[38] Q. Fan, X. Yan, and Y. Zhang, “Auto-selection mechanism of differential evolution algorithm variants and its application,” *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 636–653, Oct. 2018.

---

**L. Tian et al.** received the B.S. degree in automation from the East China University of Science and Technology, Shanghai, China, in 2014, where she is currently pursuing the Ph.D. degree with the Key Laboratory of Advanced Control and Optimization for Chemical Processes of the Ministry of Education. Her current research interests include the differential evolution algorithm, multiobjective optimization, and their real-world applications.

**Z. Li** received the B.S. degree in automation from the East China University of Science and Technology, Shanghai, China, in 2016, where he is currently pursuing the Ph.D. degree with the Key Laboratory of Advanced Control and Optimization for Chemical Processes of the Ministry of Education. His research interests include the differential evolution algorithm, multivariate statistical process monitoring, data-driven fault diagnosis, data mining, and machine learning.

**X. Yan** (Member, IEEE) received the B.S. degree in biochemical engineering and the Ph.D. degree in control theory engineering from Zhejiang University, Hangzhou, China, in 1995 and 2002, respectively. He is currently a Professor at the East China University of Science and Technology, Shanghai, China. His research interests include complex chemical process modeling, optimizing and controlling, process monitoring, fault diagnosis, and intelligent information processing.

---

* * *