Differential Evolution with a Two-Stage Optimization Mechanism for Numerical Optimization

Zhi-Zhong Liu, Yong Wang, *Member, IEEE,* Shengxiang Yang, *Senior Member, IEEE*, and Zixing Cai, *Senior Member, IEEE*

***Abstract*—Differential Evolution (DE) is a popular paradigm of evolutionary algorithms, which has been successfully applied to solve different kinds of optimization problems. To design an effective DE, it is necessary to consider different requirements of the exploration and exploitation at different evolutionary stages. Motivated by this consideration, a new DE with a two-stage optimization mechanism, called TSDE, has been proposed in this paper. In TSDE, based on the number of fitness evaluations, the whole evolutionary process is divided into two stages, namely the former stage and the latter stage. TSDE focuses on improving the search ability in the former stage and emphasizes the convergence in the latter stage. Hence, different trial vector generation strategies have been utilized at different stages. TSDE has been tested on 25 benchmark test functions from IEEE CEC2005 and 30 benchmark test functions from IEEE CEC2014. The experimental results suggest that TSDE performs better than four other state-of-the-art DE variants.**

***Keywords*—Differential evolution, parameter candidate pool, strategy candidate pool, two-stage optimization.**

I. Introduction

DIFFERENTIAL evolution (DE), proposed by Storn and Price in 1995 [1] [2], is a simple yet efficient evolutionary algorithm (EA). Due to its simple structure and ease of implementation, DE has obtained many successful applications in numerous areas during the past twenty years, such as engineering optimal design, data mining, power flow optimization, and so on [3] [4] [5]. At present, many researchers have paid their attention on the theoretical analysis of DE and have also proposed a lot of DE variants [6]. The area of DE has quickly become a hotspot in the community of evolutionary computation.

A classical DE contains three basic operators: mutation, crossover and selection. Via these operators, DE evolves the population toward the optimal solution. A combination of the mutation operator with the crossover operator is called the trial vector generation strategy. Currently, many trial vector generation strategies have been proposed. However, the effect of the trial vector generation strategies on DE's performance has not been investigated in depth. In general, greedy trial vector generation strategies (such as DE/best/1/bin and DE/current-to-best/1/bin) which utilize the information of the best individual might make DE less robust yet more efficient in terms of convergence. On the contrary, if the best individual's information is not utilized, DE might have slower convergence speed but a bigger opportunity to avoid the premature convergence. Some researchers have recognized that it is difficult for a fixed trial vector generation strategy to meet the different requirements of the exploration and exploitation during the evolution [7]. To design an effective DE, there is an agreement: at the early stage of evolution, more emphases should be put on the exploration, and at the later stage of evolution, the fast convergence becomes more attractive.

In view of the above consideration, a novel DE, referred as TSDE, is proposed in this paper. In TSDE, the whole evolutionary process is divided into two stages based on the number of fitness evaluations: the former stage and the latter stage. In the former stage, some reliable trial vector generation strategies which have good search performance are chosen to generate the trial vector. While in the latter stage, some greedy trial vector generation strategies are selected to produce the offspring, with the aim of accelerating the convergence. TSDE had been tested on 25 benchmark test functions from IEEE CEC2005 [8] and 30 benchmark test functions from IEEE CEC2014 [9]. The experimental results suggest that the performance of TSDE is better than that of four other state-of-the-art DE variants.

The rest of this paper is organized as follows. Section II introduces DE and the related work. Section III presents the details of TSDE. The experimental results are reported in Section IV. Section V concludes this paper.

II. Differential Evolution and The Related Work

A. Differential Evolution (DE)

As a population-based heuristic search algorithm, DE evolves with a population of *NP* candidate solutions. Each solution (also called a target vector) is denoted as $\vec{x}_i^g = (x_{i,1}^g, x_{i,2}^g, ..., x_{i,D}^g),\ \ i \in \{1,2,...NP\}$, where g denotes the generation number, D denotes the dimension of the search space, and *NP* is the population size. At the beginning of the evolution, the jth variable of the ith target vector can be initialized as:

$$x_{i,j}^0 = L_j + rand(0,1) * (U_j - L_j) \quad (1)$$

where $rand(0,1)$ denotes a uniform random number with the range [0,1], and L_j and U_j denote the lower and upper bounds of the jth variable, respectively. Next, DE implements mutation, crossover, and selection step by step.

Z.-Z. Liu and Z. Cai are with the School of Information Science and Engineering, Central South University, Changsha 410083, China. (e-mail: zhizhongliu@csu.edu.cn; zxcai@csu.edu.cn)

Y. Wang (Corresponding Author) is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and also with the School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK. (e-mail: ywang@csu.edu.cn)

S. Yang is with the School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK. (e-mail: syang@dmu.ac.uk)

Mutation: At generation g, a mutant vector $\vec{v}_i^{\,g}$ is generated by the mutation operator for each target vector $\vec{x}_i^{\,g}$. Five widely used mutation operators are listed as follows:

- DE/rand/1

$$\vec{v}_i^{\,g} = \vec{x}_{r1}^{\,g} + F*(\vec{x}_{r2}^{\,g} - \vec{x}_{r3}^{\,g}) \quad (2)$$

- DE/rand/2

$$\vec{v}_i^{\,g} = \vec{x}_{r1}^{\,g} + F*(\vec{x}_{r2}^{\,g} - \vec{x}_{r3}^{\,g}) + F*(\vec{x}_{r4}^{\,g} - \vec{x}_{r5}^{\,g}) \quad (3)$$

- DE/current-to-best/1

$$\vec{v}_i^{\,g} = \vec{x}_i^{\,g} + F*(\vec{x}_{best}^{\,g} - \vec{x}_i^{\,g}) + F*(\vec{x}_{r1}^{\,g} - \vec{x}_{r2}^{\,g}) \quad (4)$$

- DE/best/1

$$\vec{v}_i^{\,g} = \vec{x}_{best}^{\,g} + F*(\vec{x}_{r1}^{\,g} - \vec{x}_{r2}^{\,g}) \quad (5)$$

- DE/current-to-rand/1

$$\vec{v}_i^{\,g} = \vec{x}_i^{\,g} + F*(\vec{x}_{r1}^{\,g} - \vec{x}_i^{\,g}) + F*(\vec{x}_{r2}^{\,g} - \vec{x}_{r3}^{\,g}) \quad (6)$$

In the above equations, $r1$, $r2$, $r3$, $r4$, and $r5$ are mutually different integers randomly chosen from [1, NP] and also different from i, $\vec{x}_{best}^{\,g}$ denotes the best individual in the current population, and F is the scaling factor.

Crossover: The crossover operator is implemented on each pair of $\vec{x}_i^{\,g}$ and $\vec{v}_i^{\,g}$ to generate a trial vector $\vec{u}_i^{\,g} = (u_{i,1}^g, u_{i,2}^g, \ldots, u_{i,D}^g)$:

$$u_{i,j}^g = \begin{cases} v_{i,j}^g, & if\ rand_j(0,1) \le CR\ \ or\ \ j = j_{rand} \\ x_{i,j}^g, & otherwise \end{cases} \quad (7)$$

where j_{rand} is an integer randomly selected from the range [1, D], $rand_j(0,1)$ is a uniformly distributed random number between 0 and 1, and CR is the crossover control parameter.

Selection: DE adopts a one-to-one greedy way to select the better one from $\vec{x}_i^{\,g}$ and $\vec{u}_i^{\,g}$ for the next generation. For a minimization problem, the selection operator can be formulated as follows:

$$\vec{x}_i^{\,g+1} = \begin{cases} \vec{u}_i^{\,g}, & if\ f(\vec{u}_i^{\,g}) \le f(\vec{x}_i^{\,g}) \\ \vec{x}_i^{\,g}, & otherwise \end{cases} \quad (8)$$

where $f(\cdot)$ is the objective function.

B. The Related work

Since DE's inception, it has attracted considerable attention. Many researchers focused on improving the performance of DE by the three control parameters: the scaling factor—F, the crossover control parameter—CR, and the population size—NP. Note that, there is no fixed control parameter setting which can meet all the requirements for different problems or even for a single problem throughout the evolution. Tuning these control parameters to proper values is a very time-consuming process. To avoid tedious optimization trials for adjusting these control parameters, many parameter adaptation techniques have been developed in recent years. On the basis of the classification of Angeline [10] and Eiben *et al.* [11], these parameter adaptation techniques can be categorized into three classes: deterministic parameter control, adaptive parameter control, and self-adaptive parameter control. In deterministic parameter control, no feedback information of the evolutionary process is applied to change the control parameters and all these control parameters are updated by some deterministic rules. One example is L-SHADE [12] in which the population size linearly decreases over the course of search. Adaptive parameter control utilizes the feedback information to adjust the control parameters. Some famous DE variants (such as SaDE [7], jDE [13] and JADE [14]) can be categorized into this class. By exploiting the idea of "the evolution of evolution", in self-adaptive parameter control, the control parameters also undergo the mutation, crossover and selection, and evolve to their proper values in a manner similar to the individuals in the population. SPDE [15] and DESAP [16] belong to this class. In addition to these parameter adaptation techniques, Wang *et al*. [17] presented a composite DE called CoDE. By constructing a parameter candidate pool which contains some representative control parameter settings, every trial vector generation strategy in CoDE randomly selects one control parameter setting from the parameter candidate pool. In order to balance the convergence and the diversity of the population, all these control parameters settings are carefully selected and they can complement to each other. The parameter candidate pool proposed in [17] is also adopted in TSDE.

Another active research direction is to improve DE's trial vector generation strategies. Iorio and Li [18] designed a rotation-invariant operator named DE/current-to-rand/1 for the rotated problems. Fan and Lampinen [19] developed a trigonometric mutation operator, which exhibits good local search ability and convergence performance. In JADE [14], a generalization of DE/current-to-best/1, i.e., DE/current-to-*p*best/1, is proposed to generate the offspring. By making use of the information of multiple best solutions, DE/current-to-*p*best/1 aims at striking a balance between the exploration and exploitation. Some researchers also investigated the ensemble of different trial vector generation strategies, e.g., SaDE [7] and CoDE [17]. In both SaDE and CoDE, a strategy candidate pool is established. In SaDE, the strategy candidate pool involves four different trial vector generation strategies: DE/rand/1/bin, DE/rand/2/bin, DE/current-to-best/1/bin, and DE/current-to-rand/1. Each strategy in SaDE has a certain probability to be chosen for each individual, and the selected ratio is self-adapted during the evolution. The strategy candidate pool in CoDE consists of DE/rand/1/bin, DE/rand/2/bin, and DE/current-to-rand/1. Different from SaDE, for each target vector, CoDE generates three trial vectors by using the three trial vector generation strategies in the strategy candidate pool, and only the best one from the target vector and these three trial vectors can survive into the next generation. Both SaDE and CoDE achieve competitive performance. The success of SaDE and CoDE indicates that the ensemble method is a promising direction to improve DE's performance and TSDE proposed in this paper can also be classified into this category. However, unlike CoDE and SaDE, TSDE establishes different strategy candidate pools at different evolutionary stages.

III. TSDE

Each trial vector generation strategy has its own advantages. Note, however, that single trial vector generation strategy might not be able to solve various kinds of problems. As pointed out previously, the ensemble method, which utilizes the advantages of different trial vector generation strategies, seems to be a promising way to improve DE's performance. It is noteworthy that in the current ensemble methods, a fixed strategy candidate pool has been constructed. Therefore, the capability of these methods to meet the requirements of the exploration and exploitation during the evolution is still limited. On the basis of the above consideration, in the proposed TSDE, the whole evolutionary process is divided into two different stages: the former stage and the latter stage. Moreover, different strategy candidate pools are constructed to generate the trial vectors at different stages. Specifically, TSDE focuses on improving the exploration ability in the former stage and emphasizes the convergence in the latter stage. In addition, the parameter candidate pool introduced in [17] is utilized in TSDE. The details of TSDE are described as follows.

A. Trial Vector Generation Strategies

To meet DE's different requirements of the exploration and exploitation at different stages, in TSDE, we roughly divide the whole evolutionary process into two stages based on the number of the fitness evaluations (FEs), namely the former stage and the latter stage. Both the former stage and the later stage play very important roles on DE's performance and it is really difficult to identify which stage is more important. So we simply consider that these two stages are of equal importance. Let *Max_FEs* be the maximum number of FEs, the number of FEs less than *Max_FEs*/2 means that DE is in the former stage. Otherwise, DE is in the latter stage. At different stages, different strategy candidate pools are constructed to meet the design requirements.

In the former stage, the main aim is to improve DE's exploration ability and avoid the premature convergence. Among all DE's trial vector generation strategies, DE/rand/1/bin is the most popular one in the literature. Moreover, it is capable of improving the search ability of the population. Thus, DE/rand/1/bin is chosen into the first strategy candidate pool. Compared with DE/rand/1/bin, DE/rand/2/bin contains two difference vectors and has better exploration ability. Again, DE/rand/2/bin is included in the first strategy candidate pool. In both DE/rand/1/bin and DE/rand/2/bin, the base vector in the mutation operator is randomly selected from the population, which implies the search in these two trial vector generation strategies has no bias toward any special direction. Therefore, they have good robustness. In particular, they are suitable for solving multimodal problems. However, it should be noted that the performance of DE/rand/1/bin and DE/rand/2/bin drastically deteriorates on the rotated problems [20]. It is because the binomial crossover is dependent on the coordinate system and not a rotation-invariant process. To address this drawback, we also put DE/current-to-rand/1 which utilizes the arithmetic crossover into the first strategy candidate pool. In summary, there are three trial vector generation strategies in the first strategy candidate pool:

1) DE/rand/1/bin;
2) DE/rand/2/bin;
3) DE/current-to-rand/1.

On the contrary, in the latter stage, the fast convergence of DE becomes more attractive. In view of the above consideration, DE/current-to-best/1/bin is chosen into the second strategy candidate pool. DE/current-to-best/1/bin utilizes the information of the best individual in the current generation to guide the evolution, and therefore, it has fast convergence speed. Due to the fact that TSDE should have the capability to solve the rotated problems, DE/current-to-rand/1 is also involved in the second strategy candidate pool. Finally, the second strategy candidate pool contains the following two trial vector generation strategies:

1) DE/current-to-best/1/bin;
2) DE/current-to-rand/1.

After the strategy candidate pools have been established, another interesting issue is how to select the trial vector generation strategy from them for generating the trial vector. In TSDE, for each target vector, a trial vector generation strategy is randomly selected from the first or second strategy candidate pool, according to the evolutionary stage.

B. Control Parameter Settings

In TSDE, we utilize the parameter candidate pool proposed in [17] and all control parameter settings in the pool have the equal probability to be selected for generating the trial vector. There are three reasons:

- The relationship between the control parameters and trial vector generation strategies is really complicated and not fully investigated, especially when the trial vector generation strategies change over the course of evolution. Hence it is very difficult to adapt the control parameters to their appropriate values.
- The effectiveness of the parameter candidate pool has been demonstrated in [17]. In [17], it is also concluded that the adaptive selection of the control parameter setting performs worse than the random selection of the control parameter settings.
- The usage of the control parameter settings in [17] is simpler than that of the adaptive/self-adaptive parameter control.

The parameter candidate pool contains:

1) [F= 1.0, CR=0.1];
2) [F= 1.0, CR=0.9];
3) [F= 0.8, CR=0.2].

These three control parameter settings are widely used in many DE variants and their properties have been fully investigated. [F=1.0, CR=0.1] is suitable for solving the separable problems, [F=1.0, CR=0.9] is used to encourage the

Input: *NP*: the population size.
Max_FEs: the maximum number of fitness evaluations.
The first strategy candidate pool: DE/rand/1/bin, DE/rand/2/bin, and DE/current-to-rand/1.
The second strategy candidate pool: DE/current-to-best/1/bin and DE/current-to-rand/1.
The parameter candidate pool: [F=1, CR=0.1], [F=1, CR=0.9], and [F= 0.8, CR= 0.2].

g=0;
Generate the initial population $P^0 = \{\vec{x}_1^0, \vec{x}_2^0, ..., \vec{x}_{NP}^0\}$ according to Eq. (1);
Evaluate the objective function value of each target vector in P^0 ;
$FEs = NP$;
While $FEs \le Max\_FEs$ **do**
  $P^{g+1} = \emptyset$;
  For i = 1:NP
    If $FEs \le Max\_FEs / 2$
      Randomly select one trial vector generation strategy from the first strategy candidate pool, randomly select one control parameter setting from the parameter candidate pool, and use the selected trial vector generation strategy and control parameter setting to generate one trial vector $\vec{u}_i^g$;
    Else
      Randomly select one trial vector generation strategy from the second strategy candidate pool, randomly select one control parameter setting from the parameter candidate pool, and use the selected trial vector generation strategy and control parameter setting to generate one trial vector $\vec{u}_i^g$;
    End If
    Evaluate the objective function value of $\vec{u}_i^g$;
    $P^{g+1} = P^{g+1} \bigcup select(\vec{x}_i^g, \vec{u}_i^g)$;
    FEs=FEs+1;
  End For
  g=g+1;
End While
Output: the individual with the smallest objective function value in the population.

Fig.1. The pseudocode of TSDE

global exploration, and [F=0.8, CR=0.2] is applied to enhance DE's convergence performance.

C. The Framework of TSDE

The framework of TSDE is presented in Fig. 1. Initially, *NP* target vectors are produced according to Eq. (1). Subsequently, we judge which stage TSDE belongs to. For each target vector, one trial vector generation strategy is randomly chosen from the corresponding strategy candidate pool and one control parameter setting is randomly chosen from the parameter candidate pool. By combining the trial vector generation strategy with the control parameter setting, a trial vector is produced. Afterward, the trial vector is compared with its target vector and the better one survives into the next generation. The above process is repeated until the maximum number of FEs is reached.

IV. Experimental Study

TSDE has been tested on 25 benchmark functions from IEEE CEC2005 [8] (denoted as F_1-F_{25}) and 30 benchmark test functions from IEEE CEC2014 [9] (denoted as cf_1-cf_{30}). In our experiments, the dimension of each test function was set to 30, the population size of TSDE was set to 30, 25 independent runs and 51 independent runs were implemented for each test function in IEEE CEC2005 and in IEEE CEC2014, respectively, and the *Max_FEs* was set to 300,000.

In order to assess an algorithm's performance, the average and standard deviation of the function error value $(f(\vec{x}^{best}) - f(\vec{x}^*))$ are recorded, where $\vec{x}^{best}$ denotes the best result found in a run and $\vec{x}^*$ is the global optimum. Note that the average function error value smaller than 10^{-8} was taken as zero in this paper.

A. Comparison with Other DE variants

In this subsection, TSDE was compared with four famous DE variants: JADE [14], CoDE [17], jDE [13], and SaDE [7].

Firstly, the 25 benchmark test functions from IEEE CEC2005 have been chosen for experimental studies. The experimental results are summarized in Table I and the last three rows of Tables I conclude the comparison results. It is clear that TSDE performs the best among these five DE variants. Specifically, we can give the following comments:

1) Unimodal Functions F_1-F_5: Unimodal functions are often used to test the convergence performance of an algorithm. From the results, it is clear that JADE and TSDE are the best and the second best for these five unimodal functions. The fact that JADE outperforms TSDE may be because JADE employs a greedy mutation operator (i.e., DE/current-to-*p*best/1) during the whole evolutionary process while TSDE only utilizes the information of the best individual in the latter stage. Even though TSDE and CoDE adopt the same strategy candidate pool at the early stage of the evolution, TSDE emphasizes the convergence in the latter stage by employing the greedy trial vector generation strategy, i.e., DE/current-to-best/1/bin. As a result, TSDE achieves better performance on these unimodal functions than CoDE. TSDE also performs better than jDE. This could be attributed to the fact that jDE does not adopt any greedy trial vector generation strategies during the evolution. Although SaDE utilizes one greedy trial vector generation strategy, its performance is still worse than TSDE.
2) Basic Multimodal Functions F_6-F_{12}: TSDE achieves remarkable better performance than the other four competitors on these seven test functions. TSDE performs better than JADE on five test functions (i.e., F_6-F_8 and F_{11}-F_{12}) and only loses on F_{10}. Compared with CoDE, jDE, and SaDE, TSDE outperforms on two, six, and six test functions, respectively. CoDE, jDE, and SaDE cannot surpass TSDE on any test function. The superior performance of TSDE suggests that TSDE achieves a better balance between the exploration and exploitation during the whole evolutionary process.

TABLE I
EXPERIMENTAL RESULTS OF JADE, CODE, JDE, SADE, AND TSDE OVER 25 INDEPENDENT RUNS ON 25 TEST FUNCTIONS WITH 30D FROM IEEE CEC2005 USING 300,000 FES. "MEAN ERROR" AND "STD DEV" INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE FUNCTION ERROR VALUES OBTAINED IN 25 RUNS, RESPECTIVELY. WILCOXON'S RANK SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN TSDE AND EACH OF JADE, CODE, JDE, AND SADE.

Test Functions with 30D from IEEE CEC2005		JADE Mean Error±Std Dev	CoDE Mean Error±Std Dev	jDE Mean Error±Std Dev	SaDE Mean Error±Std Dev	TSDE Mean Error±Std Dev
Unimodal Functions	F_1	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
	F_2	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	1.11E-06±1.96E-06−	8.26E-06±1.65E-05−	0.00E+00±0.00E+00
	F_3	8.42E+03±7.26E+03+	1.05E+05±6.25E+04−	1.98E+05±1.10E+05−	4.27E+05±2.08E+05−	6.22E+04±3.76E+04
	F_4	0.00E+00±0.00E+00+	5.81E-03±1.38E-02−	4.40E-02±1.26E-01−	1.77E+02±2.67E+02−	9.14E-04±2.04E-03
	F_5	8.59E-08±5.23E-07+	3.31E+02±3.44+02−	5.11E+02±4.40+02−	3.25E+03±5.90+02−	2.63E+02±3.18+02
Basic Multimodal Functions	F_6	1.02E+01±2.96E+01−	1.60E-01±7.85E-01≈	2.35E+01±2.50E+01−	5.31E+01±3.25E+01−	1.60E-01±7.82E-01
	F_7	8.07E-03±7.42E-03−	7.46E-03±8.55E-03−	1.18E-02±7.78E-03−	1.57E-02±1.38E-02−	6.20E-03±7.21E-03
	F_8	2.09E+01±1.68E-01−	2.01E+01±1.41E-01≈	2.09E+01±4.86E-02−	2.09E+01±4.95E-02−	2.01E+01±1.53E-01
	F_9	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	2.39E-01±4.33E-01−	0.00E+00±0.00E+00
	F_{10}	2.41E+01±4.61E+00+	4.15E+01±1.16E+01−	5.54E+01±8.46E+00−	4.72E+01±1.01E+01−	3.74E+01±1.13E+01
	F_{11}	2.53E+01±1.65E+00−	1.18E+01±3.40E+00≈	2.79E+01±1.61E+00−	1.65E+01±2.42E+00−	1.17E+01±2.73E+00
	F_{12}	6.15E+03±4.79E+03−	3.05E+03±3.80E+03≈	8.63E+03±8.31E+03−	3.02E+03±2.33E+03≈	3.22E+03±3.74E+03
Expanded Multimodal Functions	F_{13}	1.49E+00±1.09E-01+	1.57E+00±3.27E-01+	1.66E+00±1.35E-01+	3.94E+00±2.81E-01−	1.80E+00±4.12E-01
	F_{14}	1.23E+01±3.11E-01≈	1.23E+01±4.81E-01≈	1.30E+01±2.00E-01−	1.26E+01±2.83E-01−	1.23E+01±3.13E-01
Hybrid Composition Functions	F_{15}	3.51E+02±1.28E+02≈	3.88E+02±6.85E+01≈	3.77E+02±8.02E+01≈	3.76E+02±7.83E+01≈	3.90E+02±7.13E+01
	F_{16}	1.01E+02±1.24E+02−	7.73E+01±5.13E+01≈	7.94E+01±2.96E+01≈	8.57E+01±6.94E+01−	7.73E+01±6.41E+01
	F_{17}	1.47E+02±1.33E+02−	6.67E+01±2.12E+01≈	1.37E+02±3.80E+01−	7.83E+01±3.76E+01−	6.39E+01±1.94E+01
	F_{18}	9.04E+02±1.03E+00≈	9.04E+02±1.04E+00≈	9.04E+02±1.08E+00≈	8.68E+02±6.23E+02≈	9.04E+02±7.06E-01
	F_{19}	9.04E+02±8.40E-01≈	9.04E+02±9.42E-01≈	9.04E+02±1.11E+00≈	8.74E+02±6.22E+01≈	9.04E+02±9.21E-01
	F_{20}	9.04E+02±8.47E-01≈	9.04E+02±9.01E-01≈	9.04E+02±1.10E+00≈	8.78E+02±6.23E+01≈	9.04E+02±1.32E+00
	F_{21}	5.00E+02±4.67E-13≈	5.00E+02±4.88E-13≈	5.00E+02±4.80E-13≈	5.52E+02±1.82E+02−	5.00E+02±2.14E-13
	F_{22}	8.66E+02±1.91E+01≈	8.63E+02±2.43E+01≈	8.75E+02±1.91E+01−	9.36E+02±1.83E+01−	8.65E+02±1.74E+01
	F_{23}	5.50E+02±8.05E+01−	5.34E+02±4.12E-04≈	5.34E+02±2.77E-04≈	5.34E+02±3.57E-03≈	5.34E+02±3.84E-04
	F_{24}	2.00E+02±2.85E-14≈	2.00E+02±2.85E-14≈	2.00E+02±2.85E-14≈	2.00E+02±6.20E-13≈	2.00E+02±2.89E-14
	F_{25}	2.11E+02±7.92E-01≈	2.11E+02±9.02E-01≈	2.11E+02±7.32E-01≈	2.14E+02±2.00E+00−	2.11E+02±1.08E+00
+		5	1	1	0	
−		8	5	13	17	
≈		12	19	11	8	

"+", "—", and "≈" denote that the performance of the corresponding DE variant is better than, worse than, and similar to that of TSDE, respectively.

3) Expanded Multimodal Functions F_{13}-F_{14}: For F_{13}, TSDE performs worse than JADE, CoDE, and jDE, and only surpasses SaDE. It might be because DE's behavior is very complicated, and our mechanism for dividing the evolutionary process is not suitable for this test function. With regard to F_{14}, TSDE performs similar to JADE and CoDE, and beats jDE and SaDE.
4) Hybrid Composition Functions F_{15}-F_{25}: These 11 test functions are very complex. The experimental results indicate that TSDE performs better than or similar to the other four DE variants on these test functions. TSDE outperforms JADE, jDE, and SaDE on three, two, and five test functions, respectively. However, JADE, jDE and SaDE cannot beat TSDE on any test function. In addition, TSDE and CoDE have the same performance on these test functions.

Subsequently, the 30 benchmark test functions from IEEE CEC2014 are used to produce the experimental results, which have been summarized in Tables II. Clearly, TSDE still performs the best among the five DE variants. Compared with JADE, TSDE achieves better performance on 13 test functions and worse performance on eight test functions. TSDE performs worse than CoDE on five test functions and better than CoDE on 13 test functions. TSDE beats jDE on 18 test functions and loses on only four test functions. Surprisingly, TSDE performs better than SaDE on 24 test functions and SaDE surpasses TSDE on only three test functions.

In summary, it is evident that the overall performance of TSDE is better than that of the four competitors. The superior performance of TSDE might be attributed to the ensemble of multiple trial vector generation strategies, by considering different requirements of the exploration and exploitation at different evolutionary stages.

B. The Effectiveness of the Two-Stage Optimization Mechanism in TSDE

The aim of this subsection is to verify the effectiveness of

TABLE II
EXPERIMENTAL RESULTS OF JADE, CODE, JDE, SADE, AND TSDE OVER 51 INDEPENDENT RUNS ON 30 TEST FUNCTIONS WITH 30D FROM IEEE CEC2014 USING 300,000 FES. "MEAN ERROR" AND "STD DEV" INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE FUNCTION ERROR VALUES OBTAINED IN 51 RUNS, RESPECTIVELY. WILCOXON'S RANK SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN TSDE AND EACH OF JADE, CODE, JDE, AND SADE.

Test Functions with 30D from IEEE CEC2014		JADE Mean Error±Std Dev	CoDE Mean Error±Std Dev	jDE Mean Error±Std Dev	SaDE Mean Error±Std Dev	TSDE Mean Error±Std Dev
Unimodal Functions	cf_1	6.09E+02±1.18E+03+	2.50E+04±1.75E+04−	7.35E+04±6.12E+04−	3.60E+05±2.74E+05−	1.65E+04±1.35E+04
	cf_2	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
	cf_3	9.86E-04±5.95E-03−	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	1.92E+01±5.60E+01−	0.00E+00±0.00E+00
Simple Multimodal Functions	cf_4	0.00E+00±0.00E+00+	3.51E-04±6.86E-04−	5.09E+00±1.48E+01−	4.13E+01±3.65E+01−	5.42E-05±1.21E-04
	cf_5	2.03E+01±3.23E-02−	2.00E+01±6.27E-02≈	2.03E+01±3.80E-02−	2.05E+01±4.94E-02−	2.00E+01±5.32E-02
	cf_6	9.15E+00±2.21E+00−	2.14E+00±1.77E+00≈	3.39E+00±3.97E+00−	4.86E+00±2.15E+00−	2.16E+00±1.99E+00
	cf_7	0.00E+00±0.00E+00≈	3.86E-04±1.99E-03−	0.00E+00±0.00E+00≈	1.12E-02±1.50E-02−	0.00E+00±0.00E+00
	cf_8	0.00E+00±0.00E+00≈	1.95E-02±1.39E-01−	0.00E+00±0.00E+00≈	5.85E-02±2.36E-01−	0.00E+00±0.00E+00
	cf_9	2.62E+01±4.96E+00+	3.74E+01±1.10E+01+	4.40E+01±5.33E+00−	3.72E+01±8.61E+00+	3.97E+01±9.25E+00
	cf_{10}	8.16E-03±1.18E-02+	4.69E-01±4.36E-01+	1.22E-03±4.94E-03+	2.82E-01±4.35E-01+	1.33E+00±1.34E+00
	cf_{11}	1.67E+03±2.13E+02≈	1.86E+03±5.55E+02−	2.41E+03±3.11E+02−	3.25E+03±5.37E+02−	1.72E+03±4.41E+02
	cf_{12}	2.67E-01±3.57E-02−	6.49E-02±3.57E-02+	4.56E-01±6.46E-02−	7.95E-01±9.96E-02−	7.87E-02±4.42E-02
	cf_{13}	2.20E-01±3.25E-02+	2.31E-01±5.21E-02≈	3.04E-01±3.54E-02−	2.66E-01±4.05E-02−	2.37E-01±5.87E-02
	cf_{14}	2.41E-01±3.18E-02−	2.42E-01±3.72E-02−	2.83E-01±2.95E-02−	2.35E-01±3.70E-02−	2.28E-01±3.64E-02
	cf_{15}	3.20E+00±4.55E-01≈	3.06E+00±8.77E-01≈	5.89E+00±7.23E-01−	4.10E+00±1.40E+00−	3.26E+00±7.37E-01
	cf_{16}	9.30E+00±4.61E-01≈	9.23E+00±9.21E-01+	9.85E+00±3.81E-01−	1.10E+01±2.64E-01−	9.44E+00±7.57E-01
Hybrid Functions	cf_{17}	1.91E+04±1.08E+05−	1.38E+03±1.39E+03−	1.13E+03±9.03E+02−	1.40E+04±1.36E+04−	9.16E+02±7.87E+02
	cf_{18}	1.14E+02±1.97E+02−	1.50E+01±7.46E+00−	1.66E+01±6.53E+00−	3.52E+02±4.95E+02−	1.37E+01±6.12E+00
	cf_{19}	4.48E+00±7.56E-01−	2.82E+00±5.06E-01−	4.36E+00±5.94E-01−	6.31E+00±1.15E+01−	2.76E+00±5.40E-01
	cf_{20}	3.11E+03±3.01E+03−	1.18E+01±5.15E+00−	1.16E+01±3.51E+00−	1.39E+02±2.02E+02−	1.01E+01±3.82E+00
	cf_{21}	1.33E+04±4.12E+04−	2.05E+02±1.40E+02−	2.74E+02±1.71E+02−	4.46E+03±7.23E+03−	1.94E+02±1.65E+02
	cf_{22}	1.44E+02±7.74E+01+	1.74E+02±1.03E+02≈	1.08E+02±7.15E+01+	1.54E+02±5.78E+01+	1.79E+02±1.01E+02
Composition Functions	cf_{23}	3.15E+02±4.01E-13≈	3.15E+02±3.73E-13≈	3.15E+02±4.01E-13≈	3.15E+02±2.24E-13≈	3.15E+02±4.12E-13
	cf_{24}	2.25E+02±3.60E+00≈	2.24E+02±2.05E+00≈	2.25E+02±2.56E+00≈	2.26E+02±2.79E+00≈	2.24E+02±1.17E+00
	cf_{25}	2.03E+02±1.13E+00≈	2.03E+02±7.37E-01≈	2.03E+02±5.31E-01≈	2.08E+02±2.54E+00−	2.03E+02±5.48E-01
	cf_{26}	1.02E+02±1.39E+01−	1.02E+02±1.39E+01−	1.00E+02±4.02E-02≈	1.11E+02±3.24E+01−	1.00E+02±4.87E-02
	cf_{27}	3.35E+02±4.68E+01+	3.79E+02±3.84E+01≈	3.62E+02±4.69E+01+	4.20E+02±4.42E+01−	3.73E+02±4.14E+01
	cf_{28}	7.96E+02±4.63E+01+	8.39E+02±3.31E+01≈	7.99E+02±2.68E+01+	8.93E+02±3.46E+01−	8.34E+02±2.91E+01
	cf_{29}	8.28E+02±3.27E+02−	7.57E+02±1.49E+02−	8.13E+02±7.12E+01−	1.10E+03±2.16E+02−	7.20E+02±1.16E+02
	cf_{30}	1.66E+03±7.61E+02−	9.06E+02±3.61E+02+	1.40E+03±5.06E+02−	1.48E+03±5.40E+02−	9.49E+02±3.79E+02
+		8	5	4	3	
−		13	13	18	24	
≈		9	12	8	3	

"+", "−", and "≈" denote that the performance of the corresponding DE variant is better than, worse than, and similar to that of TSDE, respectively.

dividing the whole evolutionary process into two stages and establishing two strategy candidate pools. We consider three variants of TSDE, i.e., Former-TSDE, Latter-TSDE, and Reverse-TSDE. In Former-TSDE and Latter-TSDE, the evolutionary process is not divided into two stages. In Former-TSDE, only the first strategy candidate pool is utilized to generate the trial vector throughout the evolution, while in the Latter-TSDE, only the second strategy candidate pool is applied to produce the trial vector over the course of search. Note, however, that in Reverse-TSDE, the whole evolution is also divided into two different stages. The difference between Reverse-TSDE and TSDE is that Reverse-TSDE employs the second strategy candidate pool in the former stage, and utilizes the first strategy candidate pool in the latter stage. The control parameter settings in these three variants were kept the same as in TSDE to make the comparison fair.

We tested these three TSDE's variants on 25 benchmark test functions with 30D from IEEE CEC2005. Table III summarizes the experimental results of TSDE and its three variants.

From Table III, it is evident that TSDE significantly outperforms its three variants. More specifically, TSDE performs better than Former-TSDE on 11 test functions and is worse than Former-TSDE on only one test function. For unimodal functions, TSDE outperforms Former-TSDE on three test functions and Former-TSDE cannot beat TSDE on any test function. It may be because TSDE's second strategy

TABLE III
EXPERIMENTAL RESULTS OF FORMER-TSDE, LATTER-TSDE, REVERSE-TSDE, AND TSDE OVER 25 INDEPENDENT RUNS ON 25 TEST FUNCTIONS WITH 30D FROM IEEE CEC2005 USING 300,000 FES. "MEAN ERROR" AND "STD DEV" INDICATE THE AVERAGE AND STANDARD DEVIATION OF THE FUNCTION ERROR VALUES OBTAINED IN 25 RUNS, RESPECTIVELY. WILCOXON'S RANK SUM TEST AT A 0.05 SIGNIFICANCE LEVEL IS PERFORMED BETWEEN TSDE AND EACH OF FORMER-TSDE, LATTER-TSDE, AND REVERSE-TSDE.

Test Functions with 30D from IEEE CEC2005		Former-TSDE Mean Error±Std Dev	Latter-TSDE Mean Error±Std Dev	Reverse-TSDE Mean Error±Std Dev	TSDE Mean Error±Std Dev
Unimodal Functions	F_1	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
	F_2	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
	F_3	1.04E+05±7.16E+04−	4.21E+04±2.48E+04+	4.67E+04±2.69E+04+	6.22E+04±3.76E+04
	F_4	2.31E-03±2.88E-03−	4.35E-04±1.32E-03+	1.31E-03±2.33E-03−	9.14E-04±2.04E-03
	F_5	3.16E+02±3.26E+02−	2.76E+02±3.69+02−	1.41E+02±1.73+02+	2.63E+02±3.18+02
Basic Multimodal Functions	F_6	3.18E-01±1.10E+00−	6.37E-01±1.49E+00−	6.37E-01±1.49E+00−	1.60E-01±7.82E-01
	F_7	8.60E-03±8.04E-03−	9.06E-03±9.08E-03−	1.34E-02±1.10E-02−	6.20E-03±7.21E-03
	F_8	2.01E+01±1.60E-01≈	2.01E+01±1.18E-01≈	2.01E+01±1.38E-01≈	2.01E+01±1.53E-01
	F_9	3.97E-02±1.98E-01−	4.59E+00±2.40E+00−	5.37E+00±3.03E+00−	0.00E+00±0.00E+00
	F_{10}	3.90E+01±1.28E+01−	4.88E+01±1.12E+01−	4.95E+01±9.33E+00−	3.74E+01±1.13E+01
	F_{11}	1.23E+01±2.67E+00−	1.43E+01±3.82E+00−	1.49E+01±4.27E+00−	1.17E+01±2.73E+00
	F_{12}	5.62E+03±6.94E+03−	2.70E+03±3.23E+03+	3.27E+03±4.20E+03≈	3.22E+03±3.74E+03
Expanded Multimodal Functions	F_{13}	1.49E+00±3.39E-01+	2.28E+00±8.07E-01−	2.27E+00±4.72E-01−	1.80E+00±4.12E-01
	F_{14}	1.22E+01±4.21E-01≈	1.25E+01±5.77E-01≈	1.23E+01±4.87E-01≈	1.23E+01±3.13E-01
Hybrid Composition Functions	F_{15}	4.00E+02±8.63E+01≈	3.85E+02±1.05E+02≈	3.56E+02±1.17E+02≈	3.90E+02±7.13E+01
	F_{16}	8.46E+01±6.98E+01−	9.98E+01±7.40E+01−	1.25E+02±1.08E+02−	7.73E+01±6.41E+01
	F_{17}	9.09E+01±7.80E+01−	1.11E+02±9.71E+01−	8.13E+01±2.34E+01−	6.39E+01±1.94E+01
	F_{18}	9.04E+02±1.10E+00≈	9.05E+02±1.17E+00≈	9.05E+02±1.18E+00≈	9.04E+02±7.06E-01
	F_{19}	9.04E+02±7.86E-01≈	9.04E+02±1.60E+00≈	9.05E+02±1.24E+00≈	9.04E+02±9.21E-01
	F_{20}	9.04E+02±1.01E+00≈	9.05E+02±1.61E+00≈	9.05E+02±1.65E+00≈	9.04E+02±1.32E+00
	F_{21}	5.00E+02±1.39E-13≈	5.00E+02±2.08E-13≈	5.12E+02±6.00E+01−	5.00E+02±2.14E-13
	F_{22}	8.73E+02±2.30E+01≈	8.76E+02±2.39E+01≈	8.80E+02±2.76E+01≈	8.65E+02±1.74E+01
	F_{23}	5.34E+02±4.30E-04≈	5.50E+02±8.05E+02−	5.34E+02±4.05E-04≈	5.34E+02±3.84E-04
	F_{24}	2.00E+02±2.90E-14≈	2.00E+02±6.26E-13≈	2.00E+02±6.20E-13≈	2.00E+02±2.89E-14
	F_{25}	2.11E+02±9.82E-01≈	2.11E+02±1.08E+00≈	2.11E+02±1.36E+00≈	2.11E+02±1.08E+00
+		1	3	2	
−		11	10	10	
≈		13	12	13	

"+", "—", and "≈" denote that the performance of the corresponding DE variant is better than, worse than, and similar to that of TSDE, respectively.

candidate pool in the latter stage can significantly enhance the convergence performance. In addition, TSDE also provides overall better performance than Former-TSDE on the other complex test functions (i.e., basic multimodal functions, expanded multimodal functions, and hybrid composition functions). TSDE performs better than Former-TSDE on eight test functions and Former-TSDE cannot beat ISDE on any test functions except for F13. This comparison also indicates that the fast convergence in the latter stage is also necessary for solving the complex test functions.

Compared with Latter-TSDE, TSDE performs worse than Latter-TSDE on unimodal functions and better than Latter-TSDE on the other complex test functions (i.e., basic multimodal functions, expanded multimodal functions, and hybrid composition functions). For these complex test functions, Latter-TSDE wins TSDE on only one test function and is worse than TSDE on nine test functions. Hence, we conclude that the first strategy candidate pool can be used to improve the robustness and the reliability when solving the complex test functions.

TSDE also performs a little bit worse than Reverse-TSDE on unimodal functions, however TSDE outperforms Reverse-TSDE on nine complex test functions (i.e., basic multimodal functions, expanded multimodal functions, and hybrid composition functions). The above results demonstrate that the sequence of using these two strategy candidate pools does not have a significant effect on the unimodal functions, while is very important for the complex test functions.

From the comparison between TSDE and its three variants, it can be concluded that DE has different requirements of the exploration and exploitation at different evolutionary stages, and that the former stage and the latter stage of TSDE are capable of strengthening the exploration of DE and enhancing the convergence performance of DE, respectively.

V. CONCLUSION

In this paper, a new DE with a two-stage optimization mechanism, called TSDE, has been proposed. In TSDE, the whole evolutionary process is divided into two stages: the former stage and the latter stage. In the former stage, TSDE

focuses on improving the exploration ability to avoid the premature convergence, while in the latter stage, TSDE prefers to the convergence performance. Hence, different strategy candidate pools have been established at different evolutionary stages. In addition, the parameter candidate pool is also applied to TSDE for the control parameter setting.

TSDE has been tested on 25 benchmark test functions from IEEE CEC2005 and 30 benchmark test functions from IEEE CEC2014, and the experimental results suggest that TSDE performs better than the other four state-of-the-art DE variants. The comparison between TSDE and its three variants demonstrates the effectiveness of the two-stage optimization mechanism.

The Matlab source code can be downloaded from Y. Wang's homepage: http://ist.csu.edu.cn/YongWang.htm

ACKNOWLEDGMENTS

This work was supported in part by the Innovation-driven Plan in Central South University (No. 2015CXS012 and No. 2015CX007), in part by the National Natural Science Foundation of China under Grant 61273314, in part by the EU Horizon 2020 Marie Sklodowska-Curie Individual Fellowships (Project ID: 661327), in part by the Engineering and Physical Sciences Research Council of UK under Grant EP/K001310/1, in part by the Hunan Provincial Natural Science Fund for Distinguished Young Scholars (Grant No. 2016JJ1018), and in part by the Program for New Century Excellent Talents in University under Grant NCET-13-0596.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces," Berkeley, CA, Tech. Rep. TR-95-012, 1995.

[2] R. Storn and K. V. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997.

[3] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer-Verlag, 2005.

[4] B. V. Babu and S. A. Munawar, "Differential evolution strategies for optimal design of shell-and-tube heat exchangers," *Chemical Engineering Science*, vol. 62, no. 14, pp. 3720-3739, 2007.

[5] B. Alatas, A. Erhan, and K. Ali, "MODENAR: Multi-objective differential evolution algorithm for mining numeric association rules," *Applied Soft Computing*, vol. 8, no. 1, pp. 646-656, 2008.

[6] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation,* vol. 15, no. 1, pp. 4-31, 2011.

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

[8] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technol. Univ., Singapore, Tech. Rep. KanGAL #2005005, May 2005, IIT Kanpur, India.

[9] 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," Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore, December 2013.

[10] P. J. Angeline, "Adaptive and self-adaptive evolutionary computations," in *Computational Intelligence: A Dynamic Systems Perspective.* 1995, pp.152-163.

[11] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 124-141, 1999.

[12] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput.*, pp. 1658-1665, 2014.

[13] 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 Transactions on Evolutionary Computation,* vol. 10, no. 6, pp. 646-657, 2006.

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

[15] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm," in *Proc. IEEE Congr. Evol. Comput.*, vol.1. pp. 831-836, 2002.

[16] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Computing,* vol. 10, no. 8, pp. 673-686, 2006.

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

[18] A. W. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Proc. Adv. AI*, 2004, pp. 861-872.

[19] H. Y. Fan and J. Lampinen, "A trigonometric mutation operator to differential evolution," *Journal of Global Optimization*, vol. 27, no. 1, pp. 105-129, 2003.

[20] Y. Wang, H.-X. Li, T. Huang, and L. Li, "Differential evolution based on covariance matrix learning and bimodal distribution parameter setting," *Applied Soft Computing*, vol. 18, pp. 232-247, 2014.