Targeted Mutation: A Novel Mutation Strategy for Differential Evolution

Weijie Zheng*, Haohuan Fu*, Guangwen Yang*†

*Ministry of Education Key Laboratory for Earth System Modeling, Center for Earth System Science, Tsinghua University
†Department of Computer Science and Technology, Tsinghua University
Beijing, China
zhengwj13@mails.tsinghua.edu.cn, {haohuan, ygw}@mail.tsinghua.edu.cn

Abstract—Differential Evolution (DE) has been shown as an effective, efficient and robust evolutionary computing algorithm. The main force to generate promising offspring is the mutation operator. Usually, two randomly selected vectors are used to generate the differential vector, which maintains the large diversity of mutant directions and ensures the possibility to find global optima. However, strong randomness also leads to the ineffective searching and slow convergence speed. A proper degree of certainty in differential vector will help the population evolve efficiently. This paper proposes a novel mutation strategy called Targeted Mutation that takes the determined target vector as the starting point of the differential vector and maintains the randomness of the ending point, which makes a better trade-off between the certainty and randomness in the differential vector. Besides, Targeted Mutation adopts the best vector as the base vector. The extensive experiments of comparison with two popular mutation operators on 20 benchmark functions demonstrate the competitive performance of our proposed targeted mutation scheme. Our method achieves better or equivalent performance over 70% of total benchmarks against the other two methods. 17 out of 20 function results can get further improved when roughly tuning parameters on each function, showing the potential ability to get even better results. In addition, an integrated evaluation scoring scheme is designed to provide a more concrete demonstration of the overall performance of different approaches, and our method gains the highest score.

Keywords—Differential evolution (DE), mutation operator, evolutionary computing, numerical optimization

I. INTRODUCTION

Evolutionary algorithms (EAs), inspired by the natural selection and Darwinian principle, have been utilized to solve complex search and optimization problems. In the EA community, differential evolution (DE) is a simple but efficient algorithm that was first introduced by Storn and Price [1]. It has been shown to outperform some other EAs, such as the genetic algorithm (GA) and the particle swarm optimization (PSO), over several benchmarks [2]. According to the DE survey [3], till now, DE has been successfully applied to various areas of science and engineering, such as optimal power flow in electrical power systems, system identification in control systems, and pattern recognition.

Similar to other EAs, DE is a population-based stochastic searching technique. It employs mutation, crossover (recombination) and selection operators in each generation to drive the population towards the status with better fitness, and eventually towards the global optimum. The substantial characteristic of DE is the differential mutation that considers the difference of two individuals (called the differential vector) as the direction to generate the offspring of the target individual. The DE variant can utilize one or more differential vectors to determine the mutant direction, and generate the mutant vector by adding the direction to the base vector. In addition to the different mutant vector generation strategies, the control parameters (the population size NP, the mutation scaling factor $F$ and the crossover rate $Cr$) also have a huge impact on the performance of the algorithm.

Generally, all population-based searching algorithms, no exception for DE, undergo a long period of computing time due to their stochastic nature, and have the potential to improve the accuracy further. Many literatures have put their efforts to improve the performance of DE, which can be divided into two categories.

One category put the efforts on new operators, such as mutation, crossover or selection operators. For example, Fan and Lampinen [4] proposed a trigonometric mutation operator that considered the center as well as the fitness of the three randomly selected vectors. Das et al. [5] introduced the DEGL method which combined the local neighborhood with the global mutation. Based on the clustering characteristic of DE algorithm, Epitropakis et al. [6] put forward the proximity-based mutation operator that substituted the random selection of the parents during the mutation and proposed a probabilistic selection due to the distance from the target individual. Gong and Cai [7] believed that the species with different fitness had different opportunities to propagate offspring, and put forward the rank-DE in which the base vector and terminal point of the differential vector were selected from the population based on their fitness ranking during the mutation. Guo [8] utilized the eigenvectors of covariance matrix of the individual solutions and proposed a rotation-invariant crossover operator.

The other category designed adaptive (self-adaptive) schemes to take the best of the parameters and strategies. For instance, jDE [9] encoded $F$ and $Cr$ into the individual vector and let them join the evolution. To take the strength of various mutation strategies, CoDE [10] combined three mutation strategies and three parameter settings and get them adapted in the process. In SaDE [11], both mutation
strategy and associated control parameters were gradually self-adapted by learning from the experiences.

Except [6] and [7] adopting different probability schemes, other methods usually utilized two randomly selected vectors to generate the differential vector, which keeps the large diversity of directions and ensures the possibility to find the global optima. However, this action also leads to the ineffective searching and slow convergence speed. Adding some certainty to the generation of the differential vector will help the population evolve more effectively and achieve better results within the limited time, but it should also be noted that too much certainty will be greedy and can cause stagnation. A better trade-off between the randomness and certainty in the differential vector is required to achieve the better trade-off between the efficiency and accuracy.

In this paper, a novel special DE mutation strategy, called Targeted Mutation (TM), is proposed. TM takes the determined target vector into consideration, and adopts the target vector as the starting point of differential vector and keeps the randomness of the ending point. To strengthen the performance and further reduce the diversity, TM chooses the best vector of the current population as the base vector. Extensive experiments have been conducted to compare it with two popular mutation strategies [1] on 20 benchmark functions.

The contributions of this paper can be summarized as follows:

- This paper proposes a novel mutation strategy, called Targeted Mutation, that takes the determined target vector as the starting point of the differential vector and maintains the randomness of the ending point, and adopts the best vector as the base vector.
- This mutation achieves better or equivalent performance over 70% of total benchmarks against the other two popular mutation strategies.
- An integrated evaluation scoring scheme is designed to rank the overall performance of different methods across various benchmarks, and the proposed method gets the highest score.

Organization of the rest paper is as follows. The brief description of the basic Differential Evolution is shown in Section II. Section III introduces the proposed mutation strategy. Extensive experiments and analysis are conducted in Section IV. Section V discusses the designed integrated evaluation scoring scheme and ranks the three methods. Finally, Section VI concludes the paper and discusses the future work.

II. DIFFERENTIAL EVOLUTION

Without loss of generality, this paper discusses the minimization problems. That is, for the objective function \( f(x) \), the goal is to find \( x^* \in \mathbb{R}^D \) satisfying \( f(x^*) \leq f(x) \), \( \forall x \in \mathbb{R}^D \). In practice, we restrict the \( \mathbb{R}^D \) to a compact set, such as \( \Omega = \prod_{j=1}^{D} [L_j, U_j] \).

DE is a population-based iterative search method for the continuous global optimization problems. The \( g \)-th iteration is noted as the \( g \)-th generation. The individuals (solutions) of the population in \( g \)-th generation are represented by \( x^g_i = (x^g_{i,1}, x^g_{i,2}, ..., x^g_{i,D}), i = 1, 2, ..., NP, \) where \( NP \) is the population size.

**Initialization.** When \( g = 0 \), the initial population \( x^0_i = (x^0_{i,1}, x^0_{i,2}, ..., x^0_{i,D}), i = 1, 2, ..., NP \) are generated randomly in the pre-defined boundary \( [L, U] \), where \( L = \{L_1, L_2, ..., L_D\} \), \( U = \{U_1, U_2, ..., U_D\} \).

**Mutation.** For each individual \( x^g_i \) in the population, called target vector, a mutant vector \( v^g_i \) is generated from the base vector and the scaled differential vector(s). That is

\[
v^g_i = x^g_{\text{base}} + F(x^g_{r_1} - x^g_{r_2}) + F(x^g_{r_3} - x^g_{r_4})
\]

where \( F \) is the scaling factor, and \( x^g_{r_1} - x^g_{r_2} \) and \( x^g_{r_3} - x^g_{r_4} \) are called the differential vectors, and \( r_1, r_2, r_3, r_4 \) are distinct integers randomly selected from 1 to \( NP \) and all different from \( i \). According to different choices of the base vector and different numbers of the differential vectors, the general DE variants can be denoted as \( \text{DE/base}(\text{number of differential vectors}) \). For example, the two most frequently used mutation schemes [1] (all integers in the form of \( r_s \) are mutually different and different from \( i \)) can be summarized as

1) \( \text{DE/rand/1} \)

\[
v^g_i = x^g_{r_2} + F(x^g_{r_2} - x^g_{r_1})
\]

2) \( \text{DE/best/1} \)

\[
v^g_i = x^g_{\text{best}} + F(x^g_{r_1} - x^g_{r_2})
\]

**Crossover.** It is also called recombination. This operator generates the trial/offspring vector \( u^g_i \) based on the target vector \( x^g_i \) and the mutant vector \( v^g_i \). The typical one is the binomial crossover:

\[
u^g_{i,j} = \begin{cases} v^g_{i,j}, & \text{if } \text{rand}(0, 1) \leq C_r \text{ or } j = j_{\text{rand}} \\ x^g_{i,j}, & \text{otherwise} \end{cases}
\]

where \( i = 1, 2, ..., NP; j = 1, 2, ..., D; C_r \in [0, 1] \) is called the crossover rate, and \( j_{\text{rand}} \) is an integer randomly selected from 1 to \( D \), to make sure that at least one component of the trial vector inherits from the mutant vector.

**Selection.** The fitness values \( f(\cdot) \) of the target vector \( x^g_i \) and its offspring vector \( u^g_i \) are compared, and the one with better fitness value is to be selected as an individual of the next generation. This operator is expressed as follows:

\[
x^{g+1}_i = \begin{cases} u^g_i, & \text{if } f(u^g_i) \leq f(x^g_i) \\ x^g_i, & \text{otherwise} \end{cases}
\]
III. PROPOSED METHOD

In the entire process of DE, mutation, crossover, and selection operators play different roles to achieve a better solution. In mutation, the random change happens on the current population so that every candidate in the solution space will have the opportunity to enter the next generation and get itself inherited. Different mutation strategies focus on different aspects. DE/rand/1 maximizes the randomness and treats every vector equally, but DE/best/1 considers the peculiarity of the current best vector. DE/best/1 believes that the promising vector is likely to surround the best vector, then the best vector is chosen as the base vector. The significance of mutation is to widen the diversity of the candidates and enhance the searching ability. The differential vector determines the direction of searching, and the scaled factor $F$ controls the step length of the search and prevents DE from stagnation. For crossover, the mutant vector is recomposed with the target vector. Then, besides the mutant vector, there are $2D - 2$ possible trial vectors, and the diversity is further increased. Finally, the selection operator is employed to preserve the most promising vector entering the next generation, and maintain the stable population size. The selection operator ensures the non-degeneration property of the evolution process.

From the above discussion, the main engine of DE that pulls the population to improvement is the mutation operator. The base vector, differential vector, and the scaled factor determine a certain mutant vector. In the view of global searching, the existing methods always kept the randomness of both vectors that generate the differential vector, which maintains the large diversity of directions and ensures the possibility to find global optima. However, this action also leads to the ineffective searching and slow convergence speed. Adding some certainty to the randomness will help the population evolve effectively and obtain better results in the limited time, but it also should be noticed that too much or even mere certainty will be too greedy and can cause stagnation. The better trade-off between the randomness and certainty in the differential vector is required to achieve better trade-off between the efficiency and accuracy.

This paper reduces the randomness of the differential vector by taking the determined target vector as the starting point of the differential vector, and maintains the randomness of the ending point of the differential vector to avoid being too greedy. For the base vector, this paper utilizes the best vector to strengthen the performance and further reduce the diversity of the possible candidates. Then, this paper proposed an novel mutation, named Targeted Mutation (TM), shown in the following equation, where $v_i^g$ represents the mutant vector, $r$ is randomly generated from 1, 2, ..., $NP$ but different from current index $i$.

$$v_i^g = x_{best}^g + F(x_r^g - x_i^g)$$ (6)

This DE variant with targeted mutation strategy, called tDE, adopts the binary crossover, and maintains the popular selection scheme. The pseudocode of the complete algorithm is shown in the Algorithm 1.

Algorithm 1 DE with targeted mutation strategy

Step 0: Load the parameters defined by the user
($NP$: population size, $F$: scaled factor, $Cr$: crossover rate, $FES_{max}$: maximum number of function evaluations)

Step 1: Initialize the population
(Set $g = 0$, and randomly generate $NP$ vectors $x_1^0, x_2^0, ..., x_{NP}^0$ within the predefined boundary $[L, U]$, and calculate the corresponding object function value. $FES = NP$)

Step 2: Evolution
while $FES < FES_{max}$ do

Step 2.1: Mutation
(For every target vector $x_i^g$, $i = 1, 2, ..., NP$, generate a mutation vector $v_i^g$)
for $i = 1 : NP$ do
$v_i^g = x_{best}^g + F(x_r^g - x_i^g)$
end for

Step 2.2: Crossover
(Generate a trial vector according to the mutation vector and the target vector)
for $i = 1 : NP$ do
for $j = 1 : D$ do
if $rand(0, 1) \leq Cr$ or $j = j_{rand}$ then
$u_{i,j}^g = v_{i,j}^g$
else
$u_{i,j}^g = x_{i,j}^g$
end if
end for
end for

Step 2.3 Selection
(Select the better one from the target vector and corresponding trial vector to enter the next generation)
for $i = 1 : NP$ do
if $f(u_i^g) \leq f(x_i^g)$ then
$x_i^{g+1} = u_i^g$
else
$x_i^{g+1} = x_i^g$
end if
end for

Step 2.4 Increment
$g = g + 1$, $FES = FES + NP$
end while

In the general mutation strategy, since the two vectors that generate differential vector are randomly chosen and must be mutually different and different from target vector, there are $(NP - 1) \times (NP - 2)$ possible candidate differential vectors (directions) for each target vector. While in this method, only one vector is randomly chosen to generate the differential vector. Since it must be different from the target vector, there are only $NP - 1$ possible directions for a determined target vector. Obviously, the diversity is hugely reduced. In theory, the decreased diversity of candidate direction will hurt the performance. However, the experiments in Section IV compare tDE with other two strategies, and the results show that the performance of tDE is not damaged, instead
IV. EXPERIMENTS

A. Settings

In order to give a convincing demonstration of the TM strategy, a suite with 20 benchmark functions [12] has been employed, as shown below.

- F1: Shifted Sphere Function
- F2: Shifted Schwefel’s Problem 1.2
- F3: Shifted Rotated High Conditioned Elliptic Function
- F4: Shifted Schwefel’s Problem 1.2 with Noise in Fitness
- F5: Schwefel’s Problem 2.6 with Global Optimum on Bounds
- F6: Shifted Rosenbrock’s Function
- F7: Shifted Rotated Griewank’s Function without Bounds
- F8: Shifted Rotated Ackley’s Function with Global Optimum on Bounds
- F9: Shifted Rastrigin’s Function
- F10: Shifted Rotated Rastrigin’s Function
- F11: Shifted Rotated Weierstrass Function
- F12: Schwefel’s Problem 2.13
- F13: Expanded Extended Griewank’s plus Rosenbrock’s Function (F8F2)
- F14: Shifted Rotated Expanded Scaffer’s F6
- F15: Hybrid Composition Function
- F16: Rotated Hybrid Composition Function
- F17: Rotated Hybrid Composition Function with Noise in Fitness
- F18: Rotated Hybrid Composition Function with the Global Optimum on the Bounds
- F19: Rotated Hybrid Composition Function
- F20: Non-Continuous Rotated Hybrid Composition Function

These functions are all single objective optimization problems, and involve both unimodal functions ($F_1 - F_5$) and multimodal functions ($F_6 - F_{20}$). Avoiding the optimum solution located at zeros and increasing the generalization ability, this suite considers the shift on the optimal solution which is zero in the original problem, and the rotation of the argument. Moreover, separable and non-separable, optimal solution on the boundary, and noise function are also considered. All experiments are implemented by MATLAB2010b on a personal computer with the 3.20GHz CPU, 8G memory, and Windows 7 operating system.

For convenience, the rest of the paper uses tDE to denote the DE variant with TM strategy.

For each algorithm, 50 independent runs are carried out to get a believable result. The dimension $D$ is set to be 30 for all functions, and the corresponding population size $NP$ is set to be 100 [9][12]. In each run, the program ends when the times of the function evaluations reach the predefined $maxEFS = 10000 \times D$. The minimum of each function is zero, then the final value of the fitness function is actually the fitness error value $|f(x^*) - f(x)|$. For comparison, the
initial generation is randomly generated by the same seed, and we use the same control parameter setting (showing below) as [13] for DE/rand/1 and DE/best/1, and also for tDE.

- Mutation scaled factor $F = 0.5$
- Crossover rate $Cr = 0.5$

B. Comparison and Analysis

In order to demonstrate the performance of the TM, the experiments are conducted on the 20 benchmark functions with DE/rand/1, DE/best/1, and tDE. Table I shows the mean value and standard deviation value of the 50 independent runs on each algorithm. Wilcoxon’s rank sum test [6][8][10], a nonparametric statistic test, at the 5% significance level is conducted between tDE and DE/rand/1 (DE/best/1), in order to judge whether the tDE can get significantly superior results or not. The null hypothesis is that there is no difference between the two test algorithms. When the null hypothesis is rejected and the tDE significantly outperforms the other in the perspective of the statistics, “+” is marked in the case, and “−” is marked if the null hypothesis is rejected but the tDE performs significantly poorer than the other one. When the null hypothesis cannot be rejected, “=” is marked, that is, there is no significant difference between the two test algorithms.

1) Comparison with DE/rand/1: From Table I, in the total of 20 functions, tDE achieves better performance on 13 functions, and poorer performance only on 5 functions, and 75% of the total benchmarks better than or equivalent to DE/rand/1. tDE shows overwhelming superiority, especially on the basic multimodal functions. It is obvious that on the basic multimodal functions, tDE uses the best vector as the base and reduces the searching of useless direction, thus accelerating the evolution process so that tDE can achieve better results in the limited number of function evaluations. From Table I, it seems that tDE is not much comparative to DE/rand/1 among the hybrid composition functions. However, when we look into more measurement indices, some other phenomenon appears. Table II shows the detailed information about the every 50 independent runs on the hybrid composition functions. From Table II, we can notice that the best values that tDE can reach are approximately equal or even much better than that of DE/rand/1. For other three functions, tDE achieves the equal or a slightly better value. It shows that tDE can reach a better solution, hence when the overall performance are comparative, choosing tDE means a potential better result.

2) Comparison with DE/best/1: From Table I, tDE and DE/best/1 get the comparative performance. There are 7 functions better than DE/best/1 and totally 14 functions (70% of the total) better than or equivalent to DE/best/1. But the attention should be paid to the hybrid composition functions. The superior advantage of tDE is shown among these relatively complex problems. Both methods utilize the greedy strategy that the area around the best vector is more worth searching, but the inherent difference is the searching direction. The reason that tDE can reach better solutions is that the tDE’s reducing the randomness and promoting the likelihood of the effective directions. Some other attention should also be paid. For the separable functions F1 and F9, tDE surpasses the DE/best/1. For the pair of F16 and F17, the only difference is the noise on the fitness or not. The disturbance of tDE is much lower than DE/best/1, which may indicate the better anti-noise capacity of tDE.

From Table I, in the total of 20 functions, tDE achieves better performance on 13 functions, and poorer performance only on 5 functions, and 75% of the total benchmarks better than or equivalent to DE/best/1. From Table II, every best value of tDE in each function is better than or equivalent to DE/best/1, the worst case of tDE is not worse than that of DE/best/1, and the median also shows a better performance than DE/best/1. All these indices demonstrate the positive effect of tDE’s reducing diversity.

3) Overall Comparison: From Table I, there are 15 out of 20 (75%) functions that the tDE is better than or equivalent to DE/rand/1, and 14 functions (70%) that tDE is better than or equivalent to DE/best/1. From the minimum mean error among these three methods, which is in bold in the Table I, 8 out of 20 minimum mean is achieved by tDE, 7 for DE/rand/1 and 4 for DE/best/1, and they achieve the same mean value on F7. From Table II, the best values that tDE can reach show its strength.

In order to get the visualization of these three methods, the mean error values after $0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0) \times \max F ES$ number of function values are plotted in Fig. 1 to show the convergence of each method. Four functions are chosen from four different communities, that is, unimodal, basic multimodal, expanded multimodal and hybrid composition communities. From Fig. 1(a) – (c), tDE and DE/best/1 outperform the DE/rand/1. On F9 and F14 shown in Fig. 1(b) and 1(c), tDE achieves the best result and the fastest convergence speed over others. In Fig. 1(d), tDE and DE/rand/1 surpass the DE/best/1. Although DE/rand/1 maintains the advantage at first, the persistent convergence of tDE leads to the equal result in the end, and may achieve better performance over DE/rand/1 if more function evaluations are allowed.

C. Potential Improvement

Due to the time and space limitation, instead of testing all pairs of $F$ and $Cr$, the sketchy experiments are conducted on either changing $F$ or $Cr$ and keeping the other as 0.5 to discover the potential ability of tDE. Each pair of control parameters also runs on each function 50 times independently. Table III shows the minimum mean error and standard deviation value and its corresponding parameter pair for each function.

Comparing Table III and Table I to get the sense on the searching ability of tDE, there are 17 better results than that
Table II: Detailed experimental results of DE/rand/1, DE/best/1, and tDE over 50 independent runs on hybrid composition functions of 30 variables with 300 000 FES

<table>
<thead>
<tr>
<th>Function</th>
<th>Method</th>
<th>Best</th>
<th>Worst</th>
<th>Median</th>
<th>Mean</th>
<th>Std</th>
</tr>
</thead>
<tbody>
<tr>
<td>F15</td>
<td>DE/rand/1</td>
<td>2.00001739491E+02</td>
<td>400</td>
<td>3.00000000000E+02</td>
<td>3.10273379738E+02</td>
<td>67.65350586517</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>90.92764698846</td>
<td>5.55617712266E+02</td>
<td>3.96530982564E+02</td>
<td>3.67386390258E+02</td>
<td>1.00506221714E+02</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>30.82277364920</td>
<td>5.08714313962E+02</td>
<td>400</td>
<td>3.42551829060E+02</td>
<td>1.28327143140E+02</td>
</tr>
<tr>
<td>F16</td>
<td>DE/rand/1</td>
<td>5.55617712266E+02</td>
<td>2.53721796541E+02</td>
<td>1.9794871422E+02</td>
<td>2.23502760846E+02</td>
<td>1.46148607635E+02</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>5.01615042667E+02</td>
<td>2.092576592E+02</td>
<td>1.79549871422E+02</td>
<td>2.01669968319E+02</td>
<td>1.50578370244E+02</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>5.01045696062E+02</td>
<td>400</td>
<td>3.42551829060E+02</td>
<td>3.67386390258E+02</td>
<td>1.00506221714E+02</td>
</tr>
<tr>
<td>F17</td>
<td>DE/rand/1</td>
<td>5.55617712266E+02</td>
<td>2.76880324062E+02</td>
<td>2.3639037372E+02</td>
<td>2.35078370244E+02</td>
<td>1.50578370244E+02</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>5.52568896970E+02</td>
<td>2.66180343461E+02</td>
<td>2.23502760846E+02</td>
<td>2.01669968319E+02</td>
<td>1.46148607635E+02</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>5.0681366699E+02</td>
<td>1.97373579770E+02</td>
<td>3.42551829060E+02</td>
<td>3.67386390258E+02</td>
<td>1.00506221714E+02</td>
</tr>
<tr>
<td>F18</td>
<td>DE/rand/1</td>
<td>9.0664980943E+02</td>
<td>9.05809828012E+02</td>
<td>9.0581284628E+02</td>
<td>9.0584128092E+02</td>
<td>0.269942345169</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>9.43923254047E+02</td>
<td>9.09533754922E+02</td>
<td>9.1304188477E+02</td>
<td>9.1884258011</td>
<td>3.89778368230</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>9.23998076445E+02</td>
<td>9.11201336065E+02</td>
<td>9.1175065498E+02</td>
<td>9.1175065498E+02</td>
<td>0.269942345169</td>
</tr>
<tr>
<td>F19</td>
<td>DE/rand/1</td>
<td>5.000000467999E+02</td>
<td>5.000000467999E+02</td>
<td>5.000000467999E+02</td>
<td>5.000000467999E+02</td>
<td>2.87102621016E-13</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>1.15643822205E+03</td>
<td>5.000000467999E+02</td>
<td>5.31580269312E+02</td>
<td>5.16502180023E+02</td>
<td>1.36254431151E+02</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>8.13241807786E+02</td>
<td>5.000000467999E+02</td>
<td>5.2462879211E+02</td>
<td>5.2462879211E+02</td>
<td>8.31360562571</td>
</tr>
<tr>
<td>F20</td>
<td>DE/rand/1</td>
<td>5.3416407736E+02</td>
<td>5.3416486735E+02</td>
<td>5.3416407736E+02</td>
<td>5.3416407736E+02</td>
<td>1.11671861013E-04</td>
</tr>
<tr>
<td></td>
<td>DE/best/1</td>
<td>5.3416407736E+02</td>
<td>5.81756743615E+02</td>
<td>6.36781623314E+02</td>
<td>6.36781623314E+02</td>
<td>1.36254431151E+02</td>
</tr>
<tr>
<td></td>
<td>tDE</td>
<td>5.3416407736E+02</td>
<td>5.37951668952E+02</td>
<td>5.99350875233E+02</td>
<td>5.99350875233E+02</td>
<td>1.4836230487E+02</td>
</tr>
</tbody>
</table>

Figure 1: Convergence curves of three different mutation strategies
(a) on F3, (b) on F9, (c) on F14, (d) on F17.
is defined as the mean error, then the ranking score of method \( i \) is calculated as

\[
R_i = \sum_{j=1}^{20} r_{i_{0},j}
\]

(8)

According to (7), if the \( e_{i_{0},j} \) is the best result, that is the lowest value of the mean errors among three methods, then \( r_{i_{0},j} = 1 \), and if \( e_{i_{0},j} \) is the worst, \( r_{i_{0},j} = 0 \).

The final ranking score of the method \( i_{0} \) is calculated as

\[
R_{i_{0}} = \sum_{j=1}^{20} r_{i_{0},j}
\]

(8)

Through the above setting, the method with higher final score is regarded to have a better performance. Fig. 2 plots the scores of these three methods on each function, and the final scores of the three methods are drawn in Fig. 3. From Fig. 2, tDE has achieved over 0.8 score on 13 functions, while that for DE/rand/1 and DE/best/1 is 8 and 9 respectively. Also from Fig. 2, tDE performs relatively well on average, comparing to the DE/rand/1’s bad performance on expanded multimodal functions (F13 and F14), and comparing to DE/best/1’s bad performance on hybrid composition functions (F15-F20). That is, in general, tDE performs better than at least one other method. From Fig. 3, we can see tDE achieves the highest score 14.617, while DE/rand/1 gets 9.799 and DE/best/1 gets 10.416.

In conclusion, tDE owns the competitive performance against the DE/rand/1 and DE/best/1. Therefore, this way to reduce the diversity of the candidate directions does not damage the performance but achieves some improvement. Then tDE can also be employed as one of the basic mutation strategies.

V. CONCLUSION AND FUTURE WORK

Generally, DE utilizes two random selected vectors to generate the differential vector, thus ensures a wide range of candidate directions. However, this action also leads to the ineffective searching and slow convergence speed. Adding some certainty to the randomness will help the population evolve more effectively and achieve better results within the limited time, but adding too much certainty will be greedy and can cause stagnation. This paper proposed a novel mutation strategy called Targeted Mutation that takes the determined target vector as the starting point of the
differential vector and maintains the randomness of the ending point, which makes a good trade-off between the randomness and certainty in the differential vector. The structure of this DE variant with Targeted Mutation, called tDE, is much alike to the basic popular mutation operators, and is simple and easy to implement.

The extensive experiments in this paper were conducted on the 20 global real-parameter optimization benchmark functions. We compared tDE with two DE variants using basic and widely-used mutation strategies, i.e., DE/rand/1 and DE/best/1. Over 70% of the function results that tDE got were better than or equivalent to the other two methods according to the Wilcoxon’s rank sum test. 17 of 20 function results can be further improved if the parameters can be tuning for each function separately. For the overall ranking, tDE got the highest score. Therefore, tDE is expected to be another basic mutation strategy of the DE family.

In the future, some other attempts to reduce the diversity of the differential vector but improve the efficiency or accuracy are well encouraged. In addition, in terms of the potential improvement when tuning control parameters, more specific discussions on the parameters as well as some proper adaptive or self-adaptive scheme are expected to be discovered to maximize the performance of the proposed basic mutation strategy. Moreover, because the tDE is a basic mutation strategy, some complex combinations with other methods or strategies deserve the research efforts.

ACKNOWLEDGMENT
This work is supported in part by National Natural Science Foundation of China (Grant No. 61303003, 41374113), and by National High-tech R&D (863) Program of China (Grant No. 2013AA01A208).

REFERENCES

Figure 2: Three methods’ scores on each function