Differential Evolution Using Mutation Strategy With Adaptive Greediness Degree Control

Wei-Jie Yu [a], Jing-Jing Li [b] (Corresponding Author), Jun Zhang [c] and Meng Wan [d]
[a] Department of Psychology, Sun Yat-sen University
[a] Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education
[a] Engineering Research Center of Supercomputing Engineering Software, Ministry of Education
[b] School of Computer Science, South China Normal University
[c] School of Advanced Computing, Sun Yat-sen University
[d] Center for Science and Technology Development, Ministry of Education
jingjing.li1124@gmail.com

ABSTRACT
Differential evolution (DE) has been demonstrated to be one of the most promising evolutionary algorithms (EAs) for global numerical optimization. DE mainly differs from other EAs in that it employs difference of the parameter vectors in mutation operator to search the objective function landscape. Therefore, the performance of a DE algorithm largely depends on the design of its mutation strategy. In this paper, we propose a new kind of DE mutation strategies whose greediness degree can be adaptively adjusted. The proposed mutation strategies utilize the information of top *t* solutions in the current population. Such a greedy strategy is beneficial to fast convergence performance. In order to adapt the degree of greediness to fit for different optimization scenarios, the parameter *t* is adjusted in each generation of the algorithm by an adaptive control scheme. This way, the convergence performance and the robustness of the algorithm can be enhanced at the same time. To evaluate the effectiveness of the proposed adaptive greedy mutation strategies, the approach is applied to original DE algorithms, as well as DE algorithms with parameter adaptation. Experimental results indicate that the proposed adaptive greedy mutation strategies yield significant performance improvement for most of cases studied.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Heuristic methods*

General Terms
Algorithms

GECCO '14, July 12–16, 2014, Vancouver, BC, Canada

http://dx.doi.org/10.1145/2576768.2598236

Keywords
Adaptive control, differential evolution (DE), evolutionary algorithm (EA), global optimization, mutation strategy

1. INTRODUCTION
Differential evolution (DE), which was first proposed by Storn and Price in 1995 [1][2], is a simple and powerful evolutionary algorithm (EA) for global numerical optimization. During the last few years, DE has been successfully applied to a variety of real-world problems from diverse domains of science and engineering [3]−[6]. Similar to other EAs, DE employs three general evolutionary operators, i.e. mutation, crossover, and selection in each generation to evolve the population. In these three operators, the mutation operator which utilizes difference of parameter vectors to explore the search space mainly makes DE differs from other EAs. Therefore, how to design efficient mutation strategies for DE so as to improve the performance of the algorithm has become a significant and promising research topic in DE.

The five original and most frequently used DE mutation strategies in the literature were proposed by Storn and Price [3][4]. Among them, two mutation strategies, namely, DE/rand/1 and DE/rand/2 are performed in a random manner. For example, in the DE/rand/1 mutation, all the three parent vectors are selected randomly from the current population. Such mutation strategies with high degree of randomness are good at exploring and able to locate different promising regions of the search space. However, they may be poor at exploitation of the solutions and cause slow convergence of the algorithm [7]. On the contrary, the other three mutation strategies, namely, DE/best/1, DE/best/2, and DE/current-to-best/1 incorporate the best solution information. The best parameter vector with the best fitness value in the population is selected as one of the parent vectors involved in mutation. Such greedy mutation strategies can help improving the convergence performance. Nevertheless, due to the exploitative tendency, the population may lose its diversity too early and reduce its exploration ability to search new region of the search space. This way, individuals of the population are more likely to be trapped in some local optima, especially when solving complex multimodal problems.

As the above analysis shows, different DE strategies have different performance characteristics and are suitable for solving different optimization problems [8]−[10]. In addition, to optimize a specific problem, different phases of the optimization process also require different mutation strategies. Therefore, many researchers employed the concept of combing different mutation strategies in an ensemble and proposed enhanced DE algorithms such as SaDE [11], EPSDE [12], SaJADE [13], and CoDE [14], etc. The mutation strategies in the ensemble approach have distinct characteristics, so that they can be adaptively selected to exhibit distinct performance for different optimization phases when dealing with a particular problem.

In this paper, instead of utilizing the ensemble approach, we propose a new kind of mutation strategies whose greediness degree can be adaptively adjusted to fit for different optimization scenarios. Different from the original DE algorithm, in our approach, one of the parents involved in the mutation is randomly selected from the top t solutions in the current population. This greedy strategy benefits from fast convergence performance, and the new introduced parameter t controls the greediness degree. To adapt the greediness degree to the requirements of different problem scenarios, we design a simple yet efficient adaptive control scheme for adjusting the value of parameter t. Consequently, the proposed mutation strategies can provide fast convergence performance while maintaining the reliability of the algorithm at a high level. Moreover, the approach can be easily applied to other advanced DE variants with minimal changes.

To validate the effectiveness of our proposed adaptive greedy mutation strategies on DE, our approach is compared with the original DE algorithm with different mutation strategies. In addition, the proposed adaptive greedy mutation strategies are also applied to other DE variant with parameter adaptation. Experimental results show that our proposed adaptive greedy mutation strategies are able to enhance the performance of the original DE and the advanced DE with parameter adaptation for most of the cases studied.

The rest of this paper is organized as follows. Section 2 reviews the DE algorithm and the related works on variants of DE mutation strategies. Section 3 describes the proposed adaptive greedy mutation strategies in detail. In Section 4, experiments are carried out on benchmark functions to verify the influence of our proposed approach. Finally, Section 5 draws the conclusions.

2. DE ALGORITHM AND RELATED WORKS

2.1 Differential Evolution (DE) Algorithm

DE is a population-based stochastic algorithm designed for global numerical optimization. Similar to other EAs, DE searches for a global optimum in the feasible solution space with a population of parameter vectors $\{\mathbf{x}_{i,g} = [x_{i,1,g}, x_{i,2,g}, ..., x_{i,D,g}], i = 1,2,..., NP\}$, where g denotes the current generation, D is the dimension of the search space, and NP is the population size. In generation g=0, the jth component of the ith vector can be initialized as

$$x_{i,j,0} = x_{\min,j} + \text{rand}(0,1) \cdot (x_{\max,j} - x_{\min,j}) \quad (1)$$

where rand(0,1) is a uniform random number on the interval [0,1], and $x_{\min,j}$, $x_{\max,j}$ are the prescribed minimum and maximum bounds of the jth dimension, respectively. After initialization, DE enters an evolutionary process which includes mutation, crossover, and selection operations.

Mutation: In each generation g, the mutation operation is applied to each individual $\mathbf{x}_{i,g}$ (also called target vector) to create its corresponding mutant vector $\mathbf{v}_{i,g}$. The six most frequently used mutation strategies are listed as follows.

- DE/rand/1:

$$\mathbf{v}_{i,g} = \mathbf{x}_{r1,g} + F \cdot (\mathbf{x}_{r2,g} - \mathbf{x}_{r3,g}) \quad (2)$$

- DE/best/1:

$$\mathbf{v}_{i,g} = \mathbf{x}_{\text{best},g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) \quad (3)$$

- DE/rand/2:

$$\mathbf{v}_{i,g} = \mathbf{x}_{r1,g} + F \cdot (\mathbf{x}_{r2,g} - \mathbf{x}_{r3,g}) + F \cdot (\mathbf{x}_{r4,g} - \mathbf{x}_{r5,g}) \quad (4)$$

- DE/best/2:

$$\mathbf{v}_{i,g} = \mathbf{x}_{\text{best},g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) + F \cdot (\mathbf{x}_{r3,g} - \mathbf{x}_{r4,g}) \quad (5)$$

- DE/current-to-rand/1:

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r2,g} - \mathbf{x}_{r3,g}) \quad (6)$$

- DE/current-to-best/1:

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F \cdot (\mathbf{x}_{\text{best},g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) \quad (7)$$

It can be seen that the mutant vector $\mathbf{v}_{i,g}$ is generated by combing a base vector with one or two scaled difference vectors. In the above equations, the indices r_1, r_2, r_3, r_4, and r_5 are distinct integers randomly selected from the range [1, NP], and all are different from the index i. $\mathbf{x}_{\text{best},g}$ is the vector with the best fitness value in the current generation. The factor F is a positive control parameter for amplifying the difference vectors.

Crossover: In order to enhance population diversity, a crossover operation exchanges some components of the mutant vector $\mathbf{v}_{i,g}$ with the target vector $\mathbf{x}_{i,g}$ to generate a trial vector $\mathbf{u}_{i,g}$. The process can be expressed as

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

where rand(0,1) is a uniformly distributed random number as before. j_{rand} is an integer randomly generated from the range $[1, D]$, which is used to ensure the trial vector has at least one component different from the target vector. The crossover probability CR is another control parameter that determines the fraction of vector components inherited from the mutant vector.

Selection: To decide whether the target or the trial vector can survive to the next generation, the selection operation is finally performed. For a minimization problem, the vector with the lower

fitness value enters the next generation, which can be expressed as follows:

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g}, & \text{if } f(\mathbf{u}_{i,g}) \le f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g}, & \text{otherwise} \end{cases} \quad (8)$$

where $f(\mathbf{x})$ is the objective function for the minimization problem.

The above three operators are repeated until the algorithm meets a termination criteria, such as a maximum iteration number or a maximum number of fitness evaluations.

2.2 Variants of DE Mutation Strategies

In addition to the six most frequently used mutation strategies mentioned in Section 2.1, a lot of other variants have been proposed to further improve the performance of DE. Fan and Lampinen [15] proposed a trigonometric mutation to increase the convergence speed of DE. Kaelo and Ali [16] proposed a hybrid mutation operator which uses the attraction-repulsion technique of an electromagnetism-like algorithm. In order to balance the exploration and exploitation capabilities of DE, many researchers designed variants of some greedy mutation strategies [17]−[19]. These variants utilize the information of multiple good solutions instead of the single best solution in the entire population. However, the greediness degree of these variants remains unchanged during the evolution.

Besides developing new DE mutation strategies, some researchers investigated DE mutation frameworks that can be applied to different DE variants. For example, Epitropakis *et al*. [20] proposed a mutation framework in which the probability of selecting an individual to become a parent is inversely proportional to its distance from the individual undergoing mutation. Gong and Cai [21] proposed a kind of ranking-based mutation framework, where some of the parents in the mutation operator are proportionally selected according to their rankings in the current population. Cai and Wang [22] proposed a DE mutation framework that exploits the neighborhood and direction information of the population.

Unlike the above methods, some DE variants such as SaDE [11], EPSDE [12], SaJADE [13], and CoDE [14] employed an ensemble of different mutation strategies. In the ensemble approach, a pool of mutation strategies competes to produce successful offspring population. The mutation strategies present in a pool have diverse characteristics, so that they can exhibit distinct performance characteristics during different phases of the evolution, when optimizing a specific problem [5].

3. DE MUTATION STRATEGIES WITH ADAPTIVE GREEDINESS DEGREE CONTROL

In this section, we propose the adaptive greedy mutation strategies, where one of the parent vectors in the mutation is randomly selected from the top t solutions in the current population. Next, we discuss which parent vector in the mutation should be selected from the top t solutions and how to adaptively adjust the value of parameter t.

3.1 Vector Selection

Before performing mutation in each generation, we first find out the current top t solutions based on the fitness values. Then, the *base* vector in the mutation strategy is randomly selected from the current top t solutions, while the other vectors in the mutation are selected randomly as the original DE algorithm. Note that if the base vector in a mutation strategy is the target vector (e.g. DE/current-to-best/1), instead of the base vector, the *terminal* point of the difference vector is randomly chosen as one of the current top t solution.

By incorporating the vector selection scheme into the six original mutation strategies mentioned in Section 2.1, we develop three novel adaptive greedy variants as follows:

- DE/*at*best/1:

$$\mathbf{v}_{i,g} = \mathbf{x}^{t}_{\text{best},g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) \quad (9)$$

- DE/*at*best/2:

$$\mathbf{v}_{i,g} = \mathbf{x}^{t}_{\text{best},g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) + F \cdot (\mathbf{x}_{r3,g} - \mathbf{x}_{r4,g}) \quad (10)$$

- DE/current-to-*at*best/1:

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F \cdot (\mathbf{x}^{t}_{\text{best},g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) \quad (11)$$

where $\mathbf{x}^{t}_{\text{best},g}$ is randomly chosen as one of the current top t solutions, t = 1, 2, …, NP. Among them, DE/*at*best/1 is derived from DE/rand/1 and DE/best/1, DE/*at*best/2 is derived from DE/rand/2 and DE/best/2, and DE/current-to-*at*best/1 is derived from DE/current-to-rand/1 and DE/current-to-best/1. Note that tuning the parameter t can control the greediness degree of the search process. In the following subsection, we describe how to adjust the value of parameter t.

3.2 Adaptive Control Scheme for Parameter t

To adapt the greediness degree to the requirements of different problem scenarios, we design a simple yet efficient scheme to adaptively adjust the value of parameter t. Each individual in the initial population is associated with an integer value of t which is randomly selected from the range [1, NP]. In the mutation process, each target vector produces mutant vector using its own value of parameter t. Then the crossover operator is applied to each pair of mutant vector and target vector to generate a trial vector. If the generated trial vector produced is better than the target vector, the value of parameter t is retained with trial vector which becomes the target vector in the next generation. Otherwise, if the trial vector is worse than the target vector, the target vector is randomly reinitialized with a new value of parameter t. This adaptive control scheme allows better values of parameter t to be more likely to survive and offspring would be produced by these better parameter values in the future generations.

3.3 DE With Adaptive Greedy Mutation Strategy

Combining the proposed adaptive greedy mutation strategy with DE, the DE algorithm using mutation strategy with adaptive greediness degree control (AGDE) is developed. Figure 1 illustrates the flowchart of the AGDE algorithm. It can be seen

that AGDE maintains the simple structure of the original DE algorithm. In addition, since information of good solutions is utilized in the mutation, the algorithm benefits from fast convergence. The robustness of the AGDE can be further enhanced by the adaptive greediness control scheme. Moreover, the adaptive greedy mutation strategies are also able to integrate into other DE variants with minimal changes.

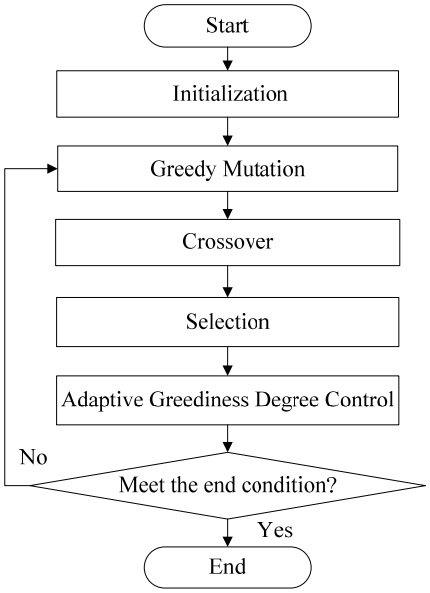

Figure 1. Flowchart of the AGDE algorithm.

4. EXPERIMENTAL RESULTS

4.1 Benchmark Functions

In this section, experiments are carried out to validate the performance of the proposed AGDE algorithms. We employ 25 benchmark functions from CEC 2005 special session on real-parameter optimization [23] as the test suite. Based on their characteristics, these functions can be divided into the four classes: 1) unimodal functions (F_1–F_5); 2) basic multimodal functions (F_6–F_{12}); 3) expanded multimodal functions (F_{13}–F_{14}); and 4) hybrid composition functions (F_{15}–F_{25}). More details of this benchmark suite can be found in [23].

4.2 Experimental Setup

In the experiments, we first test the influence of our approach on different mutation strategies in original DE. The performance of three AGDEs proposed in Section 3 (i.e. DE/*at*best/1, DE/*at*best/2 and DE/current-to-*at*best/1) is compared with their respective original algorithms. Then, we validate the effectiveness of the proposed mutation strategies on an advanced DE variant, namely, jDE [24]. jDE uses a self-adaptive approach to adapt the parameter values of F and CR, so that it can obtain promising results among various mutation strategies. To make a fair comparison, we employ the parameter settings for all of the compared DE algorithms as follows:

1) Function dimension: $D = 30$;

2) Population size: $NP = 100$ [7][10][24][25];

3) Function evaluations (FEs): FEs = 300 000;

4) Number of runs (NumR): NumR = 25.

To measure the performance of the algorithms, we calculate the average and standard deviation of the function error value $f(\mathbf{x}) - f(\mathbf{x}^*)$, where $\mathbf{x}$ is the best solution of each run and $\mathbf{x}^*$ is the global optimum of the benchmark function. For clarity, the results of the best algorithms are highlighted in **boldface** for each benchmark function. Moreover, we apply the paired Wilcoxon signed-rank test [26] to evaluate the statistical significant differences between two algorithms. Bases on the test, we mark the results with "+", "=", and "–", which indicate that the performance of our proposed algorithm is significantly better than, equal to, and worse than the corresponding algorithm, respectively.

4.3 Effects on Original DE With Different Mutation strategies

In this section, we evaluate the effectiveness of our proposed different mutation strategies in the original DE. Six mutation strategies [see (2)–(7)] are compared with their corresponding adaptive greedy variants [see (9)–(11)] in the experiment. As suggested in most of the literatures [2][24][27][28], the original DE algorithm set the values of F and CR to 0.5 and 0.9, respectively. The experimental results of DE/*at*best/1, DE/*at*best/2, and DE/current-to-*at*best/1 compared with their respective original algorithms are shown in Table 1, Table 2, and Table 3, respectively. From these results, we can find that, for the majority of the test functions, the AGDE algorithms achieve significantly better error values compared with their corresponding original DE algorithms.

1) For the unimodal functions F_1–F_5, it can be found that all the three AGDEs obtain significant better results than their corresponding original DEs on most of these functions. The only exception is that the performance of DE/*at*best/2 is deteriorated on 4 out of the 5 functions when compared with DE/best/2.

2) For the basic multimodal functions F_6–F_{12}, regardless of the mutation strategy used, the adaptive greedy DE variants consistently outperform the original DEs in most of the cases (23 out of 42). In the remaining 19 cases, AGDEs provide 13 similar results and 6 worse results compared with their corresponding original DEs, respectively.

3) For the expanded multimodal functions F_{13}–F_{14}, the adaptive greedy approach does not yield great performance improvement. The AGDEs exhibits improvement performance in 4 out of 12 cases, while in 4 cases the proposed scheme deteriorates the performance.

4) For the hybrid composition functions F_{15}–F_{25}, the AGDEs exhibit significant improvement on most of these functions, although they are very difficult to solve for most optimization algorithms. In all the 66 cases, AGDEs are significantly better than original DEs in 48 cases and worse than them in only one case. In particular, DE/*at*best/1 and DE/current-to-*at*best/1 are significantly better than DE/best/1 and DE/current-to-best/1 on all of the 11 functions, respectively.

Table 1. Effects on DE/rand/1 and DE/best/1 in Original DE

Fun.	DE/rand/1 Mean (Std.)	DE/best/1 Mean (Std.)	DE/*a*best/1 Mean (Std.)
F_1	**0.00E+00 (0.00E+00)** =	5.21E−13 (3.67E−12) +	**0.00E+00 (0.00E+00)**
F_2	2.81E−05 (1.87E−05) +	2.26E−12 (1.66E−12) +	**7.96E−14 (2.78E−14)**
F_3	3.84E+05 (2.58E+05) +	**1.72E+04 (1.02E+04)** −	8.35E+04 (4.98E+04)
F_4	1.74E−02 (2.07E−02) +	3.78E+02 (7.13E+02) +	**2.33E−09 (8.99E−09)**
F_5	5.78E+01 (4.58E+01) +	2.10E+03 (7.31E+02) +	**4.78E+01 (6.53E+01)**
F_6	**8.19E−02 (1.80E−01)** +	1.44E+00 (1.86E+00) +	4.78E−01 (1.29E+00)
F_7	**1.08E−03 (2.96E−03)** =	2.09E−02 (2.21E−02) +	1.15E−02 (1.17E−02)
F_8	2.10E+01 (4.90E−02) =	2.10E+01 (3.17E−02) =	**2.09E+01 (6.27E−02)**
F_9	1.31E+02 (2.37E+01) +	1.08E+02 (2.51E+01) +	**1.69E+01 (4.07E+00)**
F_{10}	1.81E+02 (1.05E+01) +	**1.48E+02 (3.52E+01)** =	**1.48E+02 (2.97E+01)**
F_{11}	3.94E+01 (1.01E+00) +	**2.21E+01 (1.93E+00)** −	3.83E+01 (5.99E+00)
F_{12}	**1.29E+03 (1.67E+03)** =	2.89E+03 (3.82E+03) =	2.17E+03 (2.54E+03)
F_{13}	1.53E+01 (1.01E+00) +	**6.92E+00 (2.34E+00)** =	8.62E+00 (4.31E+00)
F_{14}	1.33E+01 (1.40E−01) =	**1.20E+01 (6.53E−01)** −	1.32E+01 (2.00E−01)
F_{15}	4.08E+02 (4.45E+01) =	4.73E+02 (8.70E+01) +	**4.04E+02 (4.45E+01)**
F_{16}	2.12E+02 (9.97E+00) +	3.28E+02 (1.29E+02) +	**1.26E+02 (1.05E+02)**
F_{17}	2.25E+02 (1.36E+01) =	3.02E+02 (1.22E+02) +	**2.20E+02 (4.39E+01)**
F_{18}	9.00E+02 (3.83E+01) +	9.67E+02 (4.66E+01) +	**8.85E+02 (4.28E+01)**
F_{19}	9.04E+02 (1.33E+00) +	9.65E+02 (2.64E+01) +	**8.77E+02 (4.81E+01)**
F_{20}	9.04E+02 (1.07E+00) +	9.63E+02 (3.73E+01) +	**8.94E+02 (3.50E+01)**
F_{21}	**5.00E+02 (1.67E−13)** =	7.56E+02 (3.03E+02) +	**5.00E+02 (1.56E−13)**
F_{22}	9.08E+02 (1.05E+01) +	1.05E+03 (4.32E+01) +	**8.99E+02 (1.26E+01)**
F_{23}	**5.34E+02 (3.20E−04)** =	1.04E+03 (1.90E+02) +	**5.34E+02 (3.56E−04)**
F_{24}	**2.00E+02 (2.84E−14)** =	8.54E+02 (4.92E+02) +	**2.00E+02 (2.84E−14)**
F_{25}	**2.09E+02 (1.25E−01)** =	7.89E+02 (5.52E+02) +	**2.09E+02 (1.61E−02)**
+/=/−	14/11/0	18/4/3	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 2. Effects on DE/rand/2 and DE/best/2 in Original DE

Fun.	DE/rand/2 Mean (Std.)	DE/best/2 Mean (Std.)	DE/*a*best/2 Mean (Std.)
F_1	7.88E−01 (3.04E−01) +	6.59E−14 (1.85E−14) +	**2.96E−14 (2.83E−14)**
F_2	7.59E+03 (2.05E+03) +	**2.64E−13 (8.34E−14)** −	2.72E+00 (9.40E−01)
F_3	5.32E+07 (1.35E+07) +	**1.46E+05 (7.24E+04)** −	9.23E+05 (2.25E+05)
F_4	1.58E+04 (2.45E+03) +	**1.31E−04 (3.89E−04)** −	4.97E+01 (1.16E+01)
F_5	8.13E+03 (6.36E+02) +	**1.07E+02 (2.03E+02)** −	8.54E+02 (1.18E+02)
F_6	4.89E+03 (1.74E+03) +	**4.78E−01 (1.30E+00)** −	6.01E+00 (1.37E+01)
F_7	6.97E+00 (2.74E+00) +	1.11E−02 (9.20E−03) +	**2.87E−06 (1.47E−06)**
F_8	**2.09E+01 (5.63E−02)** =	2.10E+01 (4.76E−02) +	**2.09E+01 (5.23E−02)**
F_9	2.16E+02 (1.08E+01) +	**1.82E+02 (1.51E+01)** =	1.92E+02 (8.87E+00)
F_{10}	2.41E+02 (1.06E+01) +	**1.97E+02 (1.28E+01)** =	2.04E+02 (9.12E+00)
F_{11}	3.96E+01 (1.08E+00) +	3.98E+01 (7.92E−01) +	**3.94E+01 (7.92E−01)**
F_{12}	5.25E+05 (6.62E+04) +	**1.28E+03 (2.36E+03)** −	2.26E+05 (1.46E+05)
F_{13}	2.06E+01 (8.38E−01) +	**1.55E+01 (1.41E−01)** −	1.71E+01 (1.17E−01)
F_{14}	1.35E+01 (1.33E−01) =	**1.33E+01 (2.16E−01)** =	**1.33E+01 (2.99E−01)**
F_{15}	4.06E+02 (6.84E+00) =	3.89E+02 (1.02E+02) =	**3.88E+02 (4.13E+01)**
F_{16}	2.69E+02 (1.27E+01) +	3.18E+02 (9.97E+01) +	**2.28E+02 (1.36E+01)**
F_{17}	3.02E+02 (1.65E+01) +	3.44E+02 (1.14E+02) +	**2.57E+02 (1.25E+01)**
F_{18}	9.40E+02 (4.00E+00) +	9.08E+02 (3.03E+00) =	**9.07E+02 (2.73E−01)**
F_{19}	9.40E+02 (3.03E+00) +	9.08E+02 (3.49E+01) −	**9.07E+02 (5.20E−01)**
F_{20}	9.39E+02 (3.92E+01) +	8.99E+02 (3.52E+01) =	**9.07E+02 (7.35E−01)**
F_{21}	**5.00E+02 (1.11E−01)** +	5.48E+02 (1.10E+02) +	**5.00E+02 (1.72E−13)**
F_{22}	1.02E+03 (1.36E+01) +	9.31E+02 (1.88E+01) =	**9.30E+02 (7.19E+00)**
F_{23}	5.35E+02 (4.68E−01) =	6.17E+02 (1.62E+02) +	**5.34E+02 (3.62E−04)**
F_{24}	**2.00E+02 (1.31E−01)** +	2.00E+02 (9.26E−13) =	**2.00E+02 (2.84E−14)**
F_{25}	2.29E+02 (3.60E+00) +	2.47E+02 (1.60E+02) +	**2.09E+02 (1.58E−01)**
+/=/−	21/4/0	9/8/8	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 3. Effects on DE/current-to-rand/1 and DE/current-to-best/1 in Original DE

Fun.	DE/current-to-rand/1 Mean (Std.)	DE/current-to-best/1 Mean (Std.)	DE/current-to-*a*best/1 Mean (Std.)
F_1	6.57E+03 (1.31E+03) +	1.89E+03 (1.05E+03) +	**1.71E+03 (6.08E+02)**
F_2	6.91E+03 (1.57E+03) +	4.60E+03 (2.36E+03) +	**3.28E+03 (1.26E+03)**
F_3	3.81E+06 (2.06E+06) +	6.19E+06 (4.51E+06) +	**2.67E+06 (1.30E+06)**
F_4	3.09E+03 (1.49E+03) +	6.30E+02 (6.48E+02) +	**2.51E+02 (3.04E+02)**
F_5	5.84E+03 (1.16E+03) +	8.52E+03 (1.53E+03) +	**4.79E+03 (1.06E+03)**
F_6	1.10E+09 (4.76E+08) +	**1.43E+08 (1.31E+08)** −	2.39E+08 (2.14E+08)
F_7	3.35E+03 (3.67E+02) +	2.42E+03 (5.77E+02) +	**1.67E+03 (3.45E+02)**
F_8	**2.09E+01 (8.18E−02)** =	2.10E+01 (4.11E−02) =	**2.09E+01 (4.60E−02)**
F_9	1.54E+02 (1.11E+01) +	**8.43E+01 (2.25E+01)** =	8.54E+01 (5.56E+01)
F_{10}	1.78E+02 (1.35E+01) +	**1.19E+02 (2.72E+01)** −	1.57E+02 (2.82E+01)
F_{11}	3.91E+01 (9.69E−01) +	**1.47E+01 (1.79E+00)** −	3.32E+01 (9.62E+00)
F_{12}	2.64E+04 (1.37E+04) +	3.43E+04 (2.11E+04) +	**1.53E+04 (9.15E+03)**
F_{13}	1.38E+01 (8.82E−01) +	**4.90E+00 (3.19E+00)** −	1.22E+01 (1.07E+00)
F_{14}	1.27E+01 (3.09E−01) +	**1.20E+01 (3.72E−01)** −	1.24E+01 (2.84E−01)
F_{15}	4.78E+02 (8.69E+01) +	4.92E+02 (9.00E+01) +	**4.18E+02 (1.17E+02)**
F_{16}	1.86E+02 (1.13E+01) +	2.46E+02 (1.64E+02) +	**1.54E+02 (1.35E+02)**
F_{17}	**2.07E+02 (2.21E+01)** =	2.78E+02 (1.49E+02) +	2.10E+02 (7.17E+01)
F_{18}	**9.23E+02 (4.51E+01)** =	9.92E+02 (3.36E+01) +	**9.23E+02 (5.12E+01)**
F_{19}	9.13E+02 (4.44E+01) =	1.00E+03 (2.93E+01) +	**9.04E+02 (5.67E+01)**
F_{20}	**9.03E+02 (4.60E+01)** =	9.98E+02 (2.17E+01) +	9.25E+02 (4.75E+01)
F_{21}	1.03E+03 (1.32E+02) +	1.07E+03 (2.02E+02) +	**8.87E+02 (2.26E+02)**
F_{22}	9.38E+02 (1.48E+01) +	1.02E+03 (3.93E+01) +	**9.27E+02 (1.10E+01)**
F_{23}	1.00E+03 (1.49E+02) +	1.13E+03 (1.12E+02) +	**9.58E+02 (1.73E+02)**
F_{24}	6.96E+02 (9.76E+01) +	9.34E+02 (2.59E+02) +	**3.51E+02 (1.27E+02)**
F_{25}	1.09E+03 (2.82E+02) +	1.44E+03 (5.46E+01) +	**7.97E+02 (4.59E+02)**
+/=/−	20/5/0	18/2/5	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

Overall, based on the results and analysis, it can be seen that our proposed adaptive greedy approach is able to improve the performance of the original algorithm with different mutation strategies. In the next subsection, we will test the influence of the adaptive greedy mutation on advanced DE variant with parameter adaptation.

4.4 Effects on jDE With Different Mutation strategies

In order to study the effect of our approach more comprehensively, we further incorporate the adaptive greedy mutation strategies into an advanced DE variant, namely, jDE, which uses a self-adaptive approach to adapt the values of F and CR throughout the evolutionary process. The experimental results are shown in Tables 4–6. It can be found that, for the majority of the test functions, the adaptive greedy jDEs provide significantly better results compared with their corresponding jDEs. For example, jDE/*a*best/1 significantly improves the performance of jDE/best/1 on 19 out of 25 functions but only loses on 3 functions. jDE/current-to-*a*best/1 significantly outperforms jDE/current-to-rand/1 on 15 functions, whereas worse than it on only one function.

In general, from results shown in Tables 4–6, we can conclude that our proposed adaptive greedy mutation strategies are also capable of improving the performance of the advanced DE variant with parameter adaptation.

Table 4. Effects on DE/rand/1 and DE/best/1 in jDE

Fun.	jDE/rand/1 Mean (Std.)	jDE/best/1 Mean (Std.)	jDE/*a*rbest/1 Mean (Std.)
F_1	**0.00E+00 (0.00E+00)** =	1.05E−13 (6.87E−14) +	**0.00E+00 (0.00E+00)**
F_2	3.29E−06 (3.59E−06) +	1.28E−12 (1.24E−12) +	**2.59E−13 (1.49E−13)**
F_3	1.91E+05 (7.94E+04) +	**9.74E+03 (7.18E+03)** −	5.11E+04 (3.25E+04)
F_4	1.67E−01 (3.49E−01) +	4.37E+02 (5.60E+02) +	**1.51E−03 (3.82E−03)**
F_5	1.10E+03 (4.65E+02) −	3.05E+03 (5.58E+02) +	**1.64E+03 (4.09E+02)**
F_6	2.01E+01 (2.36E+01) +	**1.12E+00 (1.79E+00)** =	1.59E+00 (1.95E+00)
F_7	**1.22E−02 (9.57E−03)** =	2.32E−02 (2.22E−02) +	1.54E−02 (1.33E−02)
F_8	**2.09E+01 (4.91E−02)** =	2.10E+01 (2.82E−02) =	**2.09E+01 (3.36E−02)**
F_9	**0.00E+00 (0.00E+00)** =	6.90E+01 (1.70E+01) +	**0.00E+00 (0.00E+00)**
F_{10}	5.60E+01 (7.35E+00) +	9.28E+01 (1.67E+01) +	**3.49E+01 (1.11E+01)**
F_{11}	2.86E+01 (2.11E+00) +	**2.41E+01 (2.78E+00)** −	2.67E+01 (2.97E+00)
F_{12}	1.51E+04 (7.10E+03) +	1.59E+03 (2.15E+03) +	**1.06E+03 (1.58E+03)**
F_{13}	1.71E+00 (1.78E−01) +	3.44E+00 (1.26E+00) +	**1.65E+00 (1.07E−01)**
F_{14}	1.30E+01 (1.82E−01) =	**1.23E+01 (2.98E−01)** −	1.29E+01 (2.22E−01)
F_{15}	**3.47E+02 (8.92E+01)** =	3.97E+02 (6.40E+01) =	3.60E+02 (8.49E+01)
F_{16}	8.90E+01 (6.78E+01) +	1.85E+02 (8.94E+01) +	**7.97E+01 (7.19E+01)**
F_{17}	1.33E+02 (1.70E+01) +	2.53E+02 (1.28E+02) +	**8.67E+01 (6.85E+01)**
F_{18}	**9.06E+02 (1.82E+00)** =	9.52E+02 (2.16E+01) +	9.08E+02 (2.24E+01)
F_{19}	**9.07E+02 (1.76E+00)** =	9.47E+02 (2.22E+01) +	9.12E+02 (3.77E+00)
F_{20}	**9.02E+02 (2.01E+01)** =	9.56E+02 (2.21E+01) +	9.08E+02 (2.23E+01)
F_{21}	**5.00E+02 (1.33E−13)** =	6.83E+02 (2.69E+02) +	**5.00E+02 (1.64E−13)**
F_{22}	**9.05E+02 (8.35E+00)** =	9.70E+02 (3.76E+01) +	**9.05E+02 (1.11E+01)**
F_{23}	**5.34E+02 (3.01E−04)** =	7.47E+02 (2.44E+02) +	**5.34E+02 (5.04E−01)**
F_{24}	**2.00E+02 (2.84E−14)** =	3.27E+02 (3.42E+02) +	**2.00E+02 (2.84E−14)**
F_{25}	2.10E+02 (3.62E−01) +	2.95E+02 (2.81E+02) +	**2.09E+02 (3.47E−02)**
+/=/−	11/13/1	19/3/3	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 5. Effects on DE/rand/2 and DE/best/2 in jDE

Fun.	jDE/rand/2 Mean (Std.)	jDE/best/2 Mean (Std.)	jDE/*a*rbest/2 Mean (Std.)
F_1	**0.00E+00 (0.00E+00)** =	6.14E−14 (1.54E−14) +	**0.00E+00 (0.00E+00)**
F_2	2.23E−03 (4.77E−03) +	5.07E−13 (1.65E−13) =	**3.55E−13 (3.05E−13)**
F_3	2.73E+05 (1.63E+05) +	**1.22E+04 (8.56E+03)** −	6.23E+04 (3.43E+04)
F_4	6.71E+00 (2.04E+01) +	3.34E+00 (7.48E+00) +	**1.77E−04 (3.11E−04)**
F_5	7.25E+02 (3.81E+02) =	2.13E+03 (6.71E+02) +	8.91E+02 (4.12E+02)
F_6	1.54E+01 (1.19E+01) +	1.92E+00 (2.00E+00) +	**4.99E−01 (1.08E+00)**
F_7	**4.14E−03 (4.76E−03)** −	1.27E−02 (1.34E−02) =	1.30E−02 (9.34E−03)
F_8	**2.09E+01 (4.46E−02)** =	**2.09E+01 (5.15E−02)** =	2.09E+01 (4.15E−02)
F_9	**0.00E+00 (0.00E+00)** =	1.12E+01 (9.61E+00) +	**0.00E+00 (0.00E+00)**
F_{10}	6.62E+01 (9.32E+00) +	7.16E+01 (1.58E+01) +	**4.61E+01 (9.12E+00)**
F_{11}	2.83E+01 (1.91E+00) =	**2.52E+01 (3.00E+00)** −	2.82E+01 (1.41E+00)
F_{12}	2.27E+04 (4.94E+03) +	**2.70E+03 (4.24E+03)** −	1.25E+04 (8.99E+03)
F_{13}	1.81E+00 (1.55E−01) +	1.87E+00 (2.17E−01) +	**1.74E+00 (1.60E−01)**
F_{14}	1.31E+01 (2.26E−01) =	**1.28E+01 (2.90E−01)** =	1.30E+01 (2.30E−01)
F_{15}	**2.70E+02 (1.76E+02)** −	3.81E+02 (8.03E+02) =	3.84E+02 (5.43E+01)
F_{16}	9.67E+01 (1.92E+01) +	1.44E+02 (8.78E+01) +	**7.65E+01 (2.21E+01)**
F_{17}	1.57E+02 (1.56E+01) +	1.58E+02 (1.11E+02) +	**1.23E+02 (2.25E+01)**
F_{18}	**9.08E+02 (1.51E+00)** =	9.16E+02 (4.05E+01) +	**9.08E+02 (1.91E+00)**
F_{19}	**9.07E+02 (1.40E+00) =**	9.11E+02 (1.73E+01) +	9.08E+02 (1.84E+00)
F_{20}	**9.07E+02 (1.24E+00) =**	9.21E+02 (2.93E+01) +	9.08E+02 (2.49E+00)
F_{21}	**5.00E+02 (1.07E−13)** =	6.30E+02 (2.47E+02) +	**5.00E+02 (1.65E−13)**
F_{22}	9.23E+02 (8.36E+00) +	9.46E+02 (2.28E+01) +	**9.03E+02 (9.93E+00)**
F_{23}	**5.34E+02 (2.59E−04)** =	7.23E+02 (2.66E+02) +	**5.34E+02 (2.78E−04)**
F_{24}	**2.00E+02 (6.25E−13)** =	**2.00E+02 (9.17E−13)** =	**2.00E+02 (2.84E−14)**
F_{25}	**2.09E+02 (2.73E−01)** +	**2.11E+02 (1.20E+00)** =	**2.09E+02 (2.31E−01)**
+/=/−	11/12/2	15/7/3	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 6. Effects on DE/current-to-rand/1 and DE/current-to-best/1 in jDE

Fun.	jDE/current-to-rand/1 Mean (Std.)	jDE/current-to-best/1 Mean (Std.)	jDE/current-to-*a*rbest/1 Mean (Std.)
F_1	1.14E−14 (2.27E−14) +	5.23E−14 (1.85E−14) +	**0.00E+00 (0.00E+00)**
F_2	1.50E+01 (3.82E+01) +	**1.96E−12 (4.77E−12)** −	8.83E−02 (4.18E−01)
F_3	4.64E+05 (2.62E+05) +	**3.99E+04 (7.85E+04)** −	1.63E+05 (9.16E+04)
F_4	9.05E+01 (7.87E+01) +	**8.58E−01 (8.12E−01)** −	7.57E+00 (2.29E+01)
F_5	2.81E+03 (3.89E+02) +	2.30E+03 (7.24E+02) =	**2.29E+03 (5.41E+02)**
F_6	3.34E+01 (2.38E+01) +	1.31E+01 (2.89E+01) =	**1.30E+00 (1.50E+00)**
F_7	1.29E−02 (1.10E−02) +	2.21E−02 (2.78E−02) +	**8.36E−03 (1.16E−02)**
F_8	**2.09E+01 (5.19E−02)** =	**2.09E+01 (8.74E−02)** =	**2.09E+01 (4.12E−02)**
F_9	1.50E−12 (6.12E−12) +	5.00E−14 (1.85E−14) +	**0.00E+00 (0.00E+00)**
F_{10}	3.76E+01 (6.61E+00) +	4.80E+01 (1.08E+01) +	**3.51E+01 (6.57E+00)**
F_{11}	**2.50E+01 (1.37E+00)** =	2.53E+01 (1.14E+00) +	**2.50E+01 (1.71E+00)**
F_{12}	9.40E+03 (3.03E+03) +	**2.49E+03 (4.19E+03)** −	5.06E+03 (2.76E+03)
F_{13}	**1.66E+00 (1.82E−01)** =	1.69E+00 (1.79E−01) =	1.70E+00 (1.59E−01)
F_{14}	1.28E+01 (1.99E−01) +	1.27E+01 (2.37E−01) +	**1.26E+01 (2.83E−01)**
F_{15}	**2.15E+02 (9.32E+01)** −	3.52E+02 (1.42E+02) +	3.04E+02 (1.36E+01)
F_{16}	**6.57E+01 (1.14E+01)** +	1.74E+02 (1.51E+02) +	7.00E+01 (6.79E+01)
F_{17}	1.04E+02 (2.28E+01) +	2.02E+02 (1.52E+02) +	**9.31E+01 (2.95E+01)**
F_{18}	**8.75E+02 (5.65E+01)** =	8.87E+02 (5.51E+01) =	8.91E+02 (5.10E+01)
F_{19}	**8.71E+02 (5.83E+01)** =	9.19E+02 (2.53E+01) +	8.84E+02 (5.29E+00)
F_{20}	**8.86E+02 (534E+01)** =	8.89E+02 (5.58E+01) =	8.90E+02 (5.08E+01)
F_{21}	**5.00E+02 (1.51E−13)** =	6.29E+02 (2.43E+02) +	**5.00E+02 (1.61E−13)**
F_{22}	**9.26E+02 (1.01E+01)** =	**9.27E+02 (2.04E+01)** +	**9.21E+02 (1.15E+01)**
F_{23}	5.35E+02 (1.93E+00) +	6.30E+02 (2.13E+02) +	**5.34E+02 (6.50E−01)**
F_{24}	**2.00E+02 (1.03E−12)** =	**2.00E+02 (1.35E−12)** =	**2.00E+02 (2.84E−14)**
F_{25}	2.12E+02 (7.86E−01) +	2.61E+02 (1.93E+02) +	**2.11E+02 (6.81E−01)**
+/=/−	15/9/1	14/7/4	−

"+", "=", and "−" indicate that the performance of our proposed algorithm is respectively better than, equal to, and worse than the corresponding algorithm according to the Wilcoxon signed-rank test at $\alpha = 0.05$.

5. CONCLUSION

In this paper, adaptive greedy mutation strategies have been proposed for the DE algorithm. The proposed strategies utilize the information of top t solutions in the population. To adapt the greediness degree to fit for different optimization scenarios, an adaptive control scheme has been proposed for adjusting the value of parameter t. The adaptive greedy mutation strategies have been applied to the original DE and jDE. Experiments have been conducted on 25 benchmark functions from CEC 2005. The results demonstrate the effectiveness of the adaptive greedy mutation strategies to improve the overall performance.

The applications of the proposed strategies to other state-of-the-art DE variants will be studied in future work. The extension of the proposed strategies to multi-objective and dynamic optimization will be another future direction.

6. ACKNOWLEDGMENTS

This work was supported in part by the National High-Technology Research and Development Program (863 Program) of China No.2013AA01A212, in part by the NSFC for Distinguished Young Scholars 61125205, in part by the NSFC No. 61332002, No.61300044 and No. 61170220.

7. REFERENCES

[1] R. M. Storn and K. V. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over

continuous spaces," TR-95-012, 1995 [Online]. Available: http://icsi.berkeley.edu/~storn/litera.html.

[2] R. M. Storn and K. V. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, pp. 341–359, 1997.

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

[4] K. V. Price, "An introduction to differential evolution," in *New Ideas in Optimization*, D. Corne, M. Dorigo, and V. Glover, Eds. London, U.K.: McGraw-Hill, 1999, pp. 79–108.

[5] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Trans Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[6] F. Neri and V. Tirronen, "Recent advances in differential evolution: A survey and experimental analysis," *Artif. Intell. Rev.*, vol. 33, no. 1/2, pp. 61–106, Feb. 2010.

[7] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 107–125, Feb. 2008.

[8] D. K. Tasoulis, V. P. Plagianakos, and M. N. Vrahatis, "Clustering in evolutionary algorithms to efficiently compute simultaneously local and global minima," in *Proc. IEEE CEC*, vol. 2. 2005, pp. 1847–1854.

[9] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Balancing the exploration and exploitation capabilities of the differential evolution algorithm," in *Proc. IEEE CEC*, 2008, pp. 2686–2693.

[10] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximitybased mutation operators," *IEEE Trans Evol. Comput*., vol. 15, no. 1, pp. 99–119, Feb. 2011.

[11] 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.

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

[13] W. Gong, Z. Cai, C. X. Ling, and H. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 2, pp. 397–413, Apr. 2011.

[14] 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.

[15] H. Y. Fan and J. Lampinen, "A trigonometric mutation operator to differential evolution," *J. Global Optim*., vol. 27, no. 1, pp. 105–129, 2003.

[16] P. Kaelo and M. M. Ali, "Differential evolution algorithms using hybrid mutation," *Comput. Optimization Appl*., vol. 37, pp. 231–246, Jun. 2007.

[17] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput*., vol. 13, no. 3, pp. 526–553, Jun. 2009.

[18] J. Q. 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.

[19] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern*., vol. 42, no. 2, pp. 397–413, Apr. 2012.

[20] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximitybased mutation operators," *IEEE Trans Evol. Comput*., vol. 15, no. 1, pp. 99–119, Feb. 2011.

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

[22] Y. Cai and J. Wang, "Differential evolution with neighborhood and direction information for numerical optimization," *IEEE Trans. Cybern*., vol. 43, no. 6, pp. 2202–2215, Dec. 2013.

[23] 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, KanGAL Rep. No. 2005005, May 2005, IIT Kanpur, India.

[24] 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.

[25] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition based differential evolution," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 64–79, Feb. 2008.

[26] J. Derrac, S. Garc´ıa, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[27] J. Rönkkönen, S. Kukkonen, and K. V. Price, "Real-parameter optimization with differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, 2005, pp. 506–513.

[28] U. K. Chakraborty, *Advances in Differential Evolution*. Berlin, Germany: Springer, 2008.