Adaptive Parameter Selection for Strategy Adaptation in Differential Evolution

Wenyin Gong
School of Computer Science
China University of Geosciences
Wuhan, 430074 P.R. China
cug11100304@yahoo.com.cn

Zhihua Cai
School of Computer Science
China University of Geosciences
Wuhan, 430074 P.R. China
zhcai@cug.edu.cn

ABSTRACT

In order to automatically select the most suitable strategy for a specific problem without any prior knowledge, in this paper, we present an adaptive parameter selection technique for strategy adaptation in differential evolution (DE). First, a simple strategy adaptation mechanism is employed to implement the adaptive strategy selection in DE. Then, the probability- matching-based adaptive parameter selection method is proposed to select the best parameter of the strategy adaptation mechanism; in this way, it can accelerate the strategy adaptation mechanism to choose the most suitable strategy while solving a problem. To evaluate the performance of our approach, thirteen widely used benchmark functions are chosen as the test suite. The performance of our approach is compared with other DE variants, including two recently proposed DE with strategy adaptation. The results indicate that our approach is highly competitive to the compared algorithms. In addition, compared with the classical DE algorithms with single strategy, our method obtains better results in terms of the quality of the final solutions and the convergence speed.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords

Differential evolution, adaptive parameter selection, strategy adaptation, global optimization

1. INTRODUCTION

Differential evolution (DE), proposed by Storn and Price in 1995 [14, 15], is an efficient and effective population-based evolutionary algorithm for the global optimization. Among its advantages are its simple structure, ease of use, speed, and robustness, which allows it on many real-world applications, such as data mining, IIR design, neural network training, pattern recognition, digital filter design, engineering design, etc. [10]. More details of DE applications can be found at a recent survey of DE in [3].

In the seminal DE algorithm [14, 15], a single mutation strategy was used for the generation of new solutions; later on, Price and Storn suggested ten other different strategies [10, 16]. Although augmenting the robustness of the underlying algorithm, these many available strategies led the user to the problem of defining which of them would be most suitable for the problem at hand – a difficult and crucial task for the performance of DE [12, 11].

In [5], we proposed several simple strategy adaptation mechanisms (SaMs) for strategy adaptation in DE. However, the final performance of SaJADE with the first SaM may be influenced by the order of the strategies in the strategy pool [5]. To remedy this drawback, in this paper, the probability- matching-based adaptive parameter selection (APS) method is proposed to select the best parameter of the strategy adaptation mechanism; in this way, it can accelerate the strategy adaptation mechanism to choose the most suitable strategy while solving a problem. To verify the performance of our approach, 13 benchmark functions are selected as the test suit; and the performance of our approach is compared with the classical DE algorithm and some state-of-the-art DE variants.

The rest of this paper is organized as follows. Section 2 briefly introduces the related work of this paper. In Section 3, we describe our proposed method in detail, followed by the experimental results and discussions in Section 4. Finally, Section 5 is devoted to conclusions and future work.

2. RELATED WORK

Without loss of generality, in this work, we consider the following numerical optimization problem:

$$\text{Minimize} \quad f(\mathbf{x}), \qquad \mathbf{x} \in S, \tag{1}$$

where $S \subseteq \mathbb{R}^D$ is a compact set, $\mathbf{x} = [x_1, x_2, \cdots, x_D]^T$, and D is the dimension, *i.e.*, the number of decision variables. Generally, for each variable x_j, it satisfies a boundary constraint, such that:

$$L_j \leq x_j \leq U_j, j = 1, 2, \cdots, D. \tag{2}$$

where L_j and U_j are respectively the lower bound and upper bound of x_j.

2.1 Differential Evolution

DE [15] is a simple evolutionary algorithm (EA) for global numerical optimization. It creates new candidate solutions by combining the parent individual and several other individuals of the same population. A candidate replaces the parent only if it has an equal or better fitness value. The pseudo-code of the original DE algorithm is shown in Algorithm 1, where D is the number of decision variables; NP is the population size; F is the mutation scaling factor; CR is the crossover rate; $x_{i,j}$ is the j-th variable of the solution $\mathbf{x}_i$; $\mathbf{u}_i$ is the offspring. The function $\mathrm{rndint}(1, D)$ returns a uniformly distributed random integer number between 1 and D, while $\mathrm{rndreal}_j[0, 1)$ gives a uniformly distributed random real number in $[0, 1)$, generated anew for each value of j. Many mutation strategies to create a candidate are available; in Algorithm 1, the use of the classic "DE/rand/1/bin" strategy is illustrated (see line 9).

Algorithm 1 The DE algorithm with DE/rand/1/bin strategy

1: Generate the initial population
2: Evaluate the fitness for each individual
3: **while** the halting criterion is not satisfied **do**
4:   **for** $i = 1$ to NP **do**
5:     Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq i$
6:     j_{rand} = rndint$(1, D)$
7:     **for** $j = 1$ to D **do**
8:       **if** $\mathrm{rndreal}_j[0, 1) < CR$ **or** j is equal to j_{rand} **then**
9:         $u_{i,j} = x_{r_1,j} + F \cdot \left(x_{r_2,j} - x_{r_3,j}\right)$
10:       **else**
11:         $u_{i,j} = x_{i,j}$
12:       **end if**
13:     **end for**
14:   **end for**
15:   **for** $i = 1$ to NP **do**
16:     Evaluate the offspring $\mathbf{u}_i$
17:     **if** $f(\mathbf{u}_i)$ is better than **or** equal to $f(\mathbf{x}_i)$ **then**
18:       Replace $\mathbf{x}_i$ with $\mathbf{u}_i$
19:     **end if**
20:   **end for**
21: **end while**

From Algorithm 1, we can see that there are only three control parameters in DE. These are NP, F and CR. As for the terminal conditions, we can either fix the maximum number of fitness function evaluations (*Max_NFFEs*) or define a desired solution value to be reached (*VTR*).

2.2 DE with Strategy Adaptation

As mentioned above, there are many mutation strategies in DE; however, the choice of the best strategy for a problem is difficult. In order to alleviate this pitfall, some approaches have already been proposed to do so, being very briefly reminded in the following. Xie and Zhang [20] presented a swarm algorithm framework, where a neural network is adopted to update the weights of the DE strategies adaptively. Qin *et al.* [12, 11] proposed a variant of DE, named SaDE, that updates the weights of each strategy in the search based on their previous success rate. In [1, 21], strategy adaptation techniques similar to SaDE are used to enhance DE's performance. In [6], we proposed the use of a strategy adaptation method for DE, based on the *Probability Matching* technique being fed by relative fitness improvements; while in Gong *et al.* [5] a different family of strategy adaptation techniques was presented. Mallipeddi *et al.* [7] presented a DE variant with ensemble of parameters and mutation strategies, called EPSDE, in which the strategy of each target vector is initialized randomly and, during the evolution process, if the generated offspring is better than its target vector, the strategy of the target vector is stored in the next generation; otherwise, the strategy of the target vector is selected randomly from the pool or from the previous successful strategies stored with equal probability. In [9], Pan *et al.* presented an improved DE, referred to as SspDE, in which a strategy list, a scaling factor list, and a crossover rate list are encoded into the individual, being constantly updated during the evolution in a self-adaptive manner. Wang *et al.* [19] proposed a composite DE (CoDE). In CoDE, each strategy generated its trial vector with a parameter setting randomly selected from the parameter candidate pool.

3. OUR APPROACH

From the literature review in Section 2.2, we can see that although there are some DE variants with strategy adaptation, the study on adaptive strategy selection in DE is still scarce. In addition, in [5], we proposed several simple strategy adaptation mechanisms (SaMs) for strategy adaptation in DE. However, the final performance of SaJADE with the first SaM may be influenced by the order of the strategies in the strategy pool [5]. Based on these considerations, in this work, we propose an adaptive parameter selection (APS) method to choose the most suitable parameter for the first method proposed in [5]. In this way, the improved technique is able to accelerate the strategy adaptation mechanism to choose the most suitable strategy while solving a problem and alleviate the order-influence in the original method. Our proposed approach is referred to as APS-SADE, *i.e.*, APS for the Strategy Adaptation in DE.

3.1 Strategy Adaptation Mechanism

In this work, the strategy adaptation mechanism proposed in [5] is employed. For the i-th individual X_i, a strategy parameter, $\eta_i \in [0, 1)$, is used to control the selection of the strategy. Suppose that we have K strategies in the strategy pool, for the i-th target vector its mutation strategy ($S_i = \{1, 2, \cdots, K\}$) is obtained as:

$$S_i = \lfloor \eta_i \times K \rfloor + 1 \tag{3}$$

At each generation g, for the i-th solution the strategy parameter η_i is independently generated in the following manner:

$$\eta_i = \begin{cases} \mathrm{rndn}_i(\mu_s, 1/6), & \text{if } g = 1 \\ \mathrm{rndn}_i(\mu_s, 0.1), & \text{otherwise} \end{cases} \tag{4}$$

where $\mathrm{rndn}_i(\mu_s, 0.1)$ indicates a normal distribution of mean μ_s and standard deviation 0.1. If $\eta_i \notin [0, 1)$, then it is truncated to $[0, 1)$.

Denote H_s as the set of all successful strategy parameters η_i's at generation g. The mean μ_s is initialized to be 0.5 and then updated at the end of each generation as follows:

$$\mu_s = (1 - c) \times \mu_s + c \times \mathrm{mean}_A(H_s) \tag{5}$$

where c is a positive constant in $[0, 1]$ and $\mathrm{mean}_A(\cdot)$ is the usual arithmetic mean operation. $c = 0.1$ is used in this work as originally used in [5].

3.2 Adaptive Parameter Selection

As mentioned above, the performance of the SaM described in Section 3.1 may be influenced by the strategy order in the pool. That is, its performance could be influenced by the initial value of μ_s. To remedy this drawback, in this section, we propose an probability-matching (PM) based APS method to adaptively choose the most suitable μ_s for a specific problem. The PM-based APS method is inspired by our previous work for the strategy adaptation in DE [6].

Let A_{μ_s} be a parameter pool to store the initial μ_s values. Suppose we have $K > 1$ values in the pool $A_{\mu_s} = \{a_1, \cdots, a_K\}$ and a probability vector $\mathbf{P}(t) = \{p_1(t), \cdots, p_K(t)\}$ $(\forall t : 0 \leq$

$p_i(t) \leq 1; \sum_{i=1}^{K} p_i(t) = 1$). In this work, the PM technique is used to adaptively update the probability $p_a(t)$ of each value a based on its reward. Denote $r_a(t)$ as the reward[1] that a value a receives after its application at time t. $q_a(t)$ is the known quality of a value a, that is updated as follows [18]:

$$q_a(t+1) = q_a(t) + \alpha[r_a(t) - q_a(t)], \tag{6}$$

where $\alpha \in (0, 1]$ is the adaptation rate. The PM method updates the probability $p_a(t)$ as follows [4, 18]:

$$p_a(t+1) = p_{min} + (1 - K \cdot p_{min}) \frac{q_a(t+1)}{\sum_{i=1}^{K} q_i(t+1)}. \tag{7}$$

where $p_{min} \in (0, 1)$ is the minimal probability value of each value, used to ensure that no operator gets lost [18]. In this work, $A_{\mu_s} = \{0.1, 0.5, 0.9\}$, and $K = 3$. Note that at each generation, each μ_s value in the parameter pool A_{μ_s} is updated by Eqn. (5).

The idea behind the APS for μ_s is that for different problems the PM-based APS method adaptively updates the probability of each value in the evolution process; and the most suitable value will get the highest probability, and hence, it will be selected with higher probability in the following generation. Then, combined with the SaM, the most suitable strategy for the problem at hand will be selected. Therefore, the APS method is able to accelerate the SaM to choose the most suitable strategy while solving a problem, and finally, enhances the performance of the SaM in DE.

3.3 Strategy Pool

In order to select different mutation strategies to form the strategy pool, in this work, we have chosen four strategies that were also used in SaDE [11]. The four strategies are described as follows. All of them are controlled by the DE crossover rate CR unless a change is mentioned.

1) "DE/rand/1":

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot \big(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\big) \tag{8}$$

2) "DE/rand-to-best/2":

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot \big(\mathbf{x}_{best} - \mathbf{x}_{r1}\big) + F \cdot \big(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\big) + F \cdot \big(\mathbf{x}_{r_4} - \mathbf{x}_{r_5}\big) \tag{9}$$

3) "DE/rand/2":

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot \big(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\big) + F \cdot \big(\mathbf{x}_{r_4} - \mathbf{x}_{r_5}\big) \tag{10}$$

4) "DE/current-to-rand/1":

$$\mathbf{v}_i = \mathbf{x}_i + F \cdot \big(\mathbf{x}_{r_1} - \mathbf{x}_i\big) + F \cdot \big(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\big) \tag{11}$$

where $\mathbf{x}_{best}$ is the best individual in the current generation, $r_1, r_2, r_3, r_4, r_5 \in \{1, \cdots, NP\}$, and $r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq i$.

4. EXPERIMENTAL RESULTS AND ANALYSIS

In order to evaluate the performance of our approach, 13 benchmark functions are chosen from the literature as the test suite. These functions ($f_{01} - f_{13}$) are selected from [22]. Due to the tight space limitation, we omit to describe them here. More details can be found in [22].

[1] The reward $r_a(t)$ is calculated as the normalized average reward proposed in [6]. Due to the tight space limitation, we omit to describe it. More details can be found in [6].

4.1 Parameter Settings

For all experiments, we use the following parameters unless a change is mentioned.

- Dimension of each function: $D = 30$;
- Population size: $NP = 100$;
- $CR = 0.9, F = 0.5, K = 3, p_{min} = 0.05$, and $\alpha = 0.3$;
- Initially for each probability $p_i = 1/K, i = 1, \cdots, K$;
- VTR: For functions $f_{01} - f_{06}$ and $f_{08} - f_{13}$, VTR = 10^{-8}; for functions f_{07}, VTR = 10^{-2};
- For all functions: Max_NFFEs=$300,000$.

Moreover, in our experiments, each function is optimized over 50 independent runs. To avoid any initialization bias, we also use the same set of initial random populations to evaluate the different algorithms, as done in [8].

4.2 Performance Criteria

Four performance criteria are selected from the literature [17] to evaluate the performance of the algorithms. These criteria are described as follows.

- **Error**: The error of a solution $\mathbf{x}$ is defined as $f(\mathbf{x}) - f(\mathbf{x}^*)$, where $\mathbf{x}^*$ is the global minimum of the function. The minimum error is recorded when the Max_NFFEs is reached in 50 runs. The average and standard deviation of the error values are calculated as well.
- **NFFEs**: The NFFEs is also recorded when the VTR is reached. The average and standard deviation of the NFFEs values are calculated.
- **Success rate (S_r)**: The successful run of an algorithm indicates that the algorithm can result in a function value no worse than the VTR before the Max_NFFEs condition terminates the trial. The successful rate S_r is calculated as the number of successful runs divided by the total number of runs.
- **Convergence graphs**: The convergence graphs show the median error performance of the best solution over the total runs, in the respective experiments.

4.3 On the Adaptation of APS-SADE

In order to analyze the adaptation of our proposed APS-SADE, in this section, the performance of APS-SADE is compared with the original DE algorithm with single strategy in the strategy pool as mentioned in Section 3.3. The results are shown in Tables 1 and 2. All results are averaged over 50 independent runs. The best and the second best results are highlighted in **grey boldface** and **boldface**, respectively. In Table 1, the paired Wilcoxon signed-rank test at $\alpha = 0.05$ is adopted to compare the significance between two algorithms. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test, which can be used as an alternative to the paired t-test when the results cannot be assumed to be normally distributed [13]. In the last row of Table 1, according to the Wilcoxon's test, the results are summarized as "$w/t/l$", which means that APS-SADE wins in w functions, ties in t functions, and loses in l functions, compared with its competitors. The evolution trend of some selected functions[2] is plotted in Figures 1 and 2.

[2] Due to the tight space limitation, we only present the figures of some presentative functions. Interested reader can contact the authors to require the figures of all 13 functions.

Table 1: Comparison on the Error Values Between the Classical DE with Single Strategy and APS-SADE for All Functions at $D = 30$.

F	strategy1	strategy2	strategy3	strategy4	APS-SADE
f_{01}	6.41E-32 ± 8.33E-32†	**1.15E-54 ± 1.59E-54**†	8.21E-01 ± 2.41E-01†	3.08E+00 ± 3.98E+00†	**3.74E-99 ± 4.76E-99**
f_{02}	6.50E-16 ± 4.67E-16†	**2.48E-25 ± 1.20E-25**†	3.13E+00 ± 7.29E-01†	2.62E-02 ± 3.01E-02†	**1.46E-47 ± 1.02E-47**
f_{03}	2.49E-05 ± 2.07E-05†	**3.11E-10 ± 2.58E-10**†	5.09E+03 ± 1.15E+03†	2.84E+01 ± 1.91E+01†	**1.74E-21 ± 3.41E-21**
f_{04}	8.82E-02 ± 2.19E-01†	**1.06E-11 ± 6.33E-12**‡	1.57E+01 ± 2.06E+00†	2.94E+00 ± 1.09E+00†	**2.04E-08 ± 6.64E-08**
f_{05}	1.43E+00 ± 1.01E+00†	**2.39E-01 ± 9.56E-01**†	3.82E+02 ± 1.23E+02†	9.97E+01 ± 6.72E+01†	**7.97E-02 ± 5.64E-01**
f_{06}	**0.00E+00 ± 0.00E+00**	**0.00E+00 ± 0.00E+00**	2.74E+00 ± 1.10E+00†	3.04E+00 ± 2.94E+00†	**0.00E+00 ± 0.00E+00**
f_{07}	4.71E-03 ± 1.21E-03†	3.02E-03 ± 8.79E-04†	7.45E-02 ± 1.85E-02†	**1.01E-03 ± 4.53E-04**	**9.78E-04 ± 2.90E-04**
f_{08}	**6.59E+03 ± 7.04E+02**‡	7.44E+03 ± 3.18E+02†	7.42E+03 ± 2.48E+02†	7.88E+03 ± 2.60E+02†	**7.22E+03 ± 3.11E+02**
f_{09}	1.41E+02 ± 2.06E+01	1.69E+02 ± 9.16E+00†	2.19E+02 ± 1.22E+01†	**1.31E+02 ± 8.18E+00**‡	**1.40E+02 ± 1.03E+01**
f_{10}	**4.14E-15 ± 0.00E+00**	4.57E-15 ± 1.17E-15	1.05E+00 ± 2.61E-01†	2.79E-01 ± 3.01E-01†	**4.07E-15 ± 5.02E-16**
f_{11}	**0.00E+00 ± 0.00E+00**	1.92E-03 ± 4.54E-03†	8.95E-01 ± 4.46E-02†	5.98E-01 ± 3.83E-01†	**3.45E-04 ± 1.73E-03**
f_{12}	**1.93E-32 ± 6.70E-33**†	**1.57E-32 ± 0.00E+00**	3.33E+00 ± 9.20E-01†	4.34E-03 ± 1.29E-02†	**1.57E-32 ± 0.00E+00**
f_{13}	**1.44E-30 ± 1.80E-30**†	**1.35E-32 ± 0.00E+00**	2.20E+01 ± 6.89E+00†	1.63E-02 ± 4.14E-02†	**1.35E-32 ± 0.00E+00**
$w/t/l$	8/4/1	8/4/1	13/0/0	11/1/1	−

† indicates APS-SADE is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that APS-SADE is significantly worse than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 2: Comparison on the NFFEs Values Between the Classical DE with Single Strategy and APS-SADE for the Successful Functions at $D = 30$.

F	strategy1	strategy2	strategy3	strategy4	APS-SADE
f_{01}	1.05E+05 ± 1.98E+03 (1.00)	**6.40E+04 ± 1.04E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**3.68E+04 ± 5.94E+02** (1.00)
f_{02}	1.76E+05 ± 3.45E+03 (1.00)	**1.15E+05 ± 1.90E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**6.24E+04 ± 8.72E+02** (1.00)
f_{03}	NA ± NA (0.00)	**2.66E+05 ± 6.51E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**1.43E+05 ± 6.21E+03** (1.00)
f_{04}	NA ± NA (0.00)	**2.29E+05 ± 5.33E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**2.59E+05 ± 2.66E+04** (0.80)
f_{05}	NA ± NA (0.00)	**1.95E+05 ± 5.55E+03** (0.94)	NA ± NA (0.00)	NA ± NA (0.00)	**1.77E+05 ± 7.16E+03** (0.98)
f_{06}	3.83E+04 ± 1.55E+03 (1.00)	2.48E+04 ± 9.64E+02 (1.00)	NA ± NA (0.00)	**1.01E+04 ± 8.34E+02** (0.20)	**1.36E+04 ± 5.34E+02** (1.00)
f_{07}	1.45E+05 ± 3.45E+04 (1.00)	8.53E+04 ± 2.30E+04 (1.00)	NA ± NA (0.00)	**1.91E+04 ± 5.50E+03** (1.00)	**2.93E+04 ± 6.66E+03** (1.00)
f_{10}	1.63E+05 ± 3.04E+03 (1.00)	**1.01E+05 ± 1.29E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**5.70E+04 ± 7.45E+02** (1.00)
f_{11}	1.09E+05 ± 3.20E+03 (1.00)	**6.73E+04 ± 2.35E+03** (0.84)	NA ± NA (0.00)	NA ± NA (0.00)	**3.81E+04 ± 8.11E+02** (0.96)
f_{12}	9.62E+04 ± 2.78E+03 (1.00)	**6.16E+04 ± 1.73E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**3.24E+04 ± 8.25E+02** (1.00)
f_{13}	1.14E+05 ± 3.32E+03 (1.00)	**7.61E+04 ± 2.27E+03** (1.00)	NA ± NA (0.00)	NA ± NA (0.00)	**3.88E+04 ± 1.28E+03** (1.00)
$\sum S_r$	8.00	10.78	0.00	1.20	10.74

According to results shown in Table 1, w.r.t. the error values, it can be seen that APS-SADE obtains the best overall results compared with the original DE algorithm with single strategy. APS-SADE significantly outperforms DE with "DE/rand/1", "DE/rand-to-best/2", "DE/rand/2", and "DE/current-to-rand/1" on $8, 8, 13$, and 11 functions, respectively. On 9 functions, APS-SADE is able to provide the best error values. On the rest 4 functions (f_{04}, f_{08}, f_{09}, and f_{11}), our approach is the second best one.

In terms of the NFFEs values on the successful functions shown in Table 2, we can see that on 8 out of 11 functions, APS-SADE obtains the best NFFEs values. Additionally, on the rest 3 functions, APS-SADE gets the second best results. DE with "DE/rand-to-best/2" strategy obtains the highest overall $S_r = 10.78$ values, followed by our approach $S_r = 10.74$.

On analyzing the adaptation of APS-SADE, the figures of the convergence graph, the evolution trend of each μ_s in the pool, and the evolution trend of each μ_s probability for the selected function are plotted in Figures 1 and 2. When showing the evolution trend of μ_s values and probabilities in the pool, the mean curves and error bars are plotted in Figures 1 and 2. The error bars are the standard deviations of the values over 50 independent runs. They can clearly show the evolution trend of μ_s values and probabilities. For the clarity, there are only 20 error bars plotted for each figure. This is a sample average standard deviation over all runs. According to Figures 1 and 2 we can see that:

- With respect to the convergence speed, on the majority of the functions APS-SADE is capable of providing the fastest convergence speed, followed by DE with "DE/rand-to-best/2", DE with "DE/rand/1", DE with "DE/current-to-rand/1", and DE with "DE/rand/2". By carefully examining the convergence graphs, we can see that at the beginning of the evolution process DE with "DE/current-to-rand/1" obtains the fastest convergence speed. However, it stagnates after some generations. The reason is that in "DE/current-to-rand/1" the current solution $\mathbf{x}_i$ is the base vector, the solution generated by this strategy performs the *local search* around $\mathbf{x}_i$. Therefore, it converges fastest at the beginning of the evolution process, but stagnates after generations.

- Since DE with "DE/current-to-rand/1" strategy has the local search ability, it can obtain the highest reward at a certain stage. Thus, "DE/current-to-rand/1", *i.e.*the 4-th strategy, needs to be selected with the highest probability compared with other three strategies in APS-SADE. This phenomenon can be observed on the majority of the test functions ($f_{01} - f_{04}, f_{06}$, and $f_{09} - f_{13}$) in our approach. For example, as shown in Figures 1 and 2, on functions f_{01}, f_{11}, f_{13} we can see that the third parameter $\mu 3$ in A_{μ_s} converges around 0.9, and it gets the highest probability. This means that $\mu_s = 0.9$ will be selected with the highest probability to generated the mutant vector in the following generation in Eqn. (4). Based on the normal distribution, η_i is in $(0.6, 1.2)$ with 95%. After truncating in $(0.6, 1.0)$, based on Eqn. (3) it can be seen that the fourth strategy will be selected more frequently. Furthermore, this phenomenon indicates that when the minimal probability $p_{min} > 0$, the strategies in the strategy pool are able to implement the *cooperation* in APS-SADE. For example, "DE/rand/2" strategy is able to provide higher per-

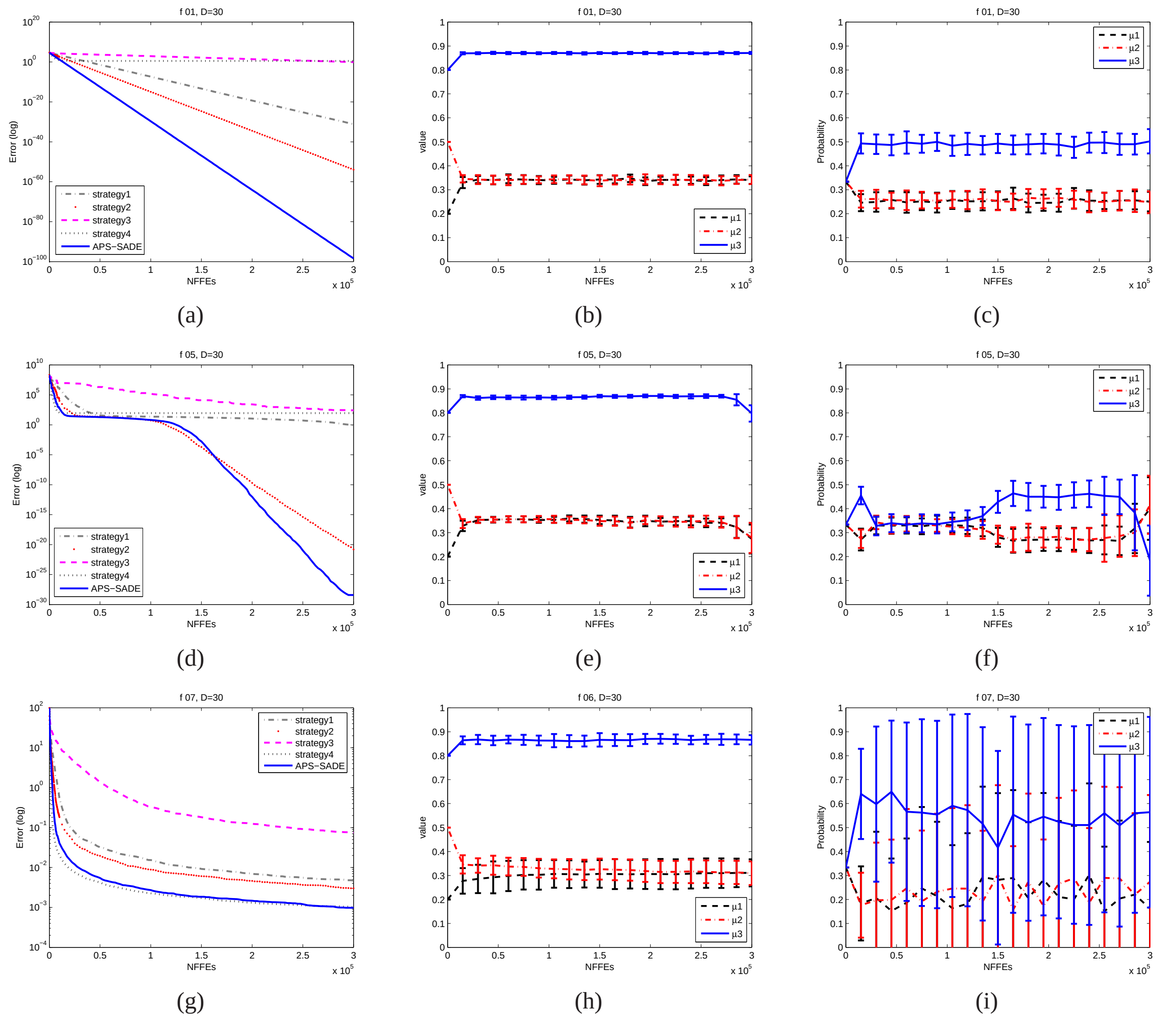

Figure 1: Adaptation analysis of APS-SADE on the selected functions. (a, b, c) f_{01}; (d, e, f) f_{05}; (g, h, i) f_{07}. Hereinafter, for each function, the left figure is the convergence graph of each algorithm; the middle one is the evolution trend of the μ_s values in the pool; and the right one is the evolution trend of the probability of each μ_s value.

turbation for the population, and hence, it can prevent the stagnation caused by "DE/current-to-rand/1". On the other hand, the local search ability of "DE/current-to-rand/1" can accelerate the convergence rate. This is why on the most of functions APS-SADE is able to obtain the best results compared to DE with single strategy as shown in Tables 1 and 2.

- For function f_{05}, the Rosenbrock's function, from Figure 1 (d - f) we can see that $\mu 3$ in the parameter pool still converges around 0.9. However, only at the early stage and the late stage $\mu 3$ obtains the highest probability. At the middle stage, all the three μ_s get the similar probability around 0.33. The reason might be that each strategy in the strategy pool is able to provide the similar reward at the middle stage in APS-SADE.

- For function f_{07}, it is a noisy function. From Figure 1 (g - i) it can be observed that the 4-th strategy is able to obtain the best results, and hence, $\mu 3$ converges around 0.9 and obtains the highest probability. However, due to the influence of the random fluctuation, the standard deviations of the probability are large as shown in the error bars in Figure 1 (i).

- For function f_{08}, it is a multimodal function with many local optima. According to Figure 2 (a - c) we can see that the first strategy "DE/rand/1" obtains the best results. Therefore, $\mu 1$ converges around 0.2, and it obtains the highest average probability. The standard deviations of the probability are large caused by the fluctuated reward obtained by each strategy.

Based on the above analysis, we can conclude that the PM-based APS for strategy adaptation in DE is able to adaptively choose the most suitable strategy for a problem at hand.

4.4 Comparison on Different Strategy Adaptation Methods

In order to show the enhanced performance of our proposed ASP-SADE method, in this section, APS-SADE is compared with some DE variants with different strategy adaptation techniques proposed recently. Three DE variants are selected: SaDE proposed in [11],

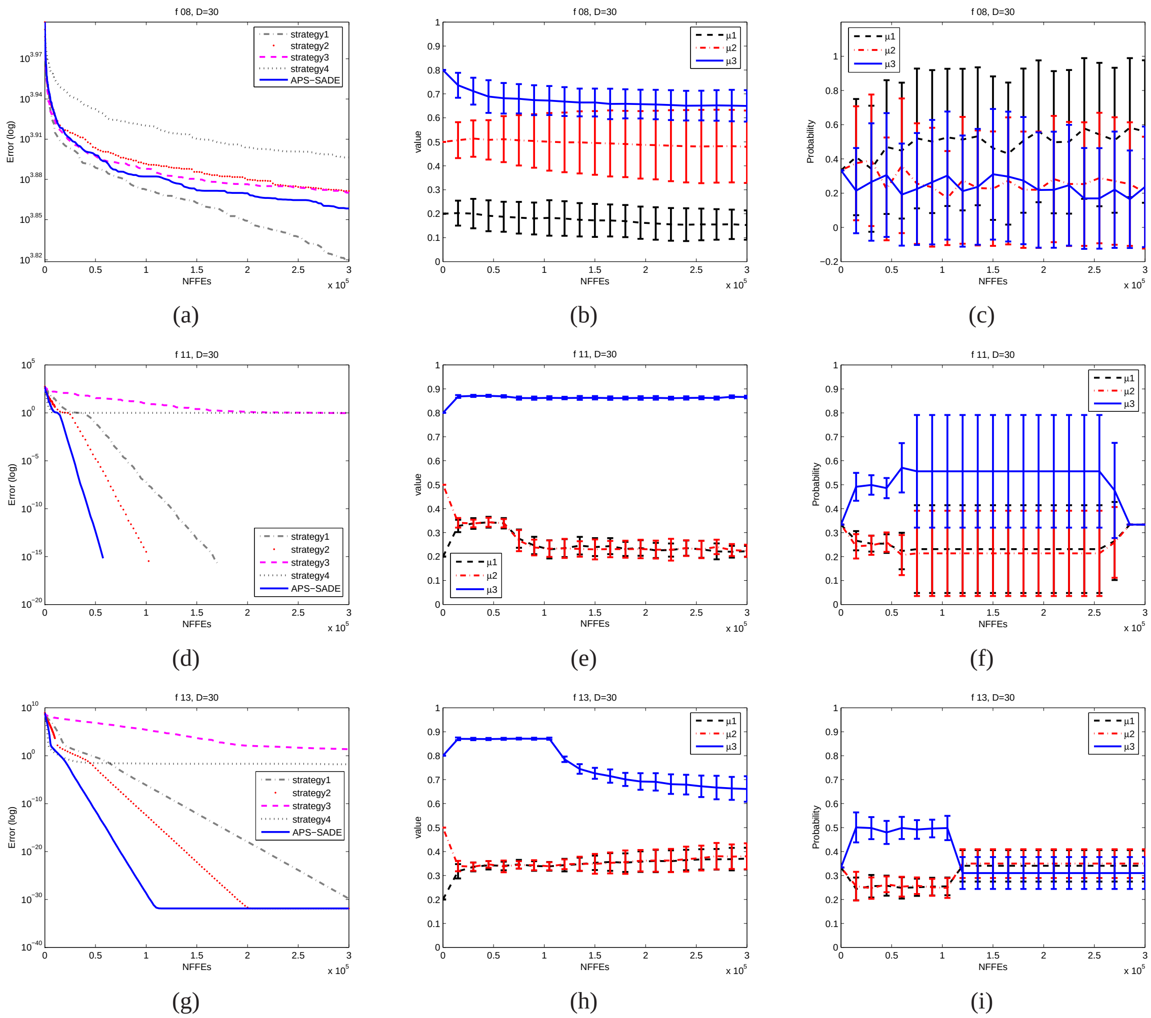

Figure 2: Adaptation analysis of APS-SADE on the selected functions. (a, b, c) f_{08}; (d, e, f) f_{11}; (g, h, i) f_{13}.

AdapSS-DE proposed in [6], and SADE using the SaM described in Section 3.1 with initial $\mu_s = 0.5$. In addition, the Uniform-SADE, SADE with uniformly selected initial μ_s value in $A_{\mu_s} = \{0.1, 0.5, 0.9\}$, is also compared with APS-SADE. Note that all above-mentioned 5 DE variants with $CR = 0.9$ and $F = 0.5$. The results are shown in Table 3. All results are averaged over 50 independent runs. The best and the second best results are highlighted in **grey boldface** and **boldface**, respectively. Similar to the methods used in [23, 5], the *intermediate* results are also reported for the functions where several algorithms can obtain the global optimum of these functions. In these cases, the Wilcoxon signed-rank test is only compared with the intermediate results.

From Table 3 we can see that APS-SADE significantly outperforms SaDE, SADE, and Uniform-SADE on $12, 10$, and 9 functions, respectively. Both APS-SADE and AdapSS-DE obtain similar results, and APS-SADE is slightly better than AdapSS-JADE. APS-SADE is significantly better than AdapSS-DE on 4 functions, while it loses on 2 functions. Compared between SADE and Uniform-SADE, we can obtain that Uniform-SADE is better than SADE on the majority of the functions. It means that the fixed initial μ_s value is affected by the order of the strategies in the strategy pool. In addition, from Table 3 we can also conclude that the proposed ASP-SADE method is able to alleviate the drawback of the original SADE method and accelerate the the original SaM to choose the most suitable strategy while solving a problem.

4.5 Effect of Control Parameter Adaptation

In the previous experimental sections, the control parameters CR and F of APS-SADE are fixed as described in Section 4.1. In this section, in order to study the effect of the adaptive control parameters proposed in the DE literature, we employ two techniques proposed in [2] and [11] in APS-SADE. The two variants are referred to as APS-SADE1 (APS-SADE with parameter adaptation proposed in [2]) and APS-SADE2 (APS-SADE with parameter adaptation proposed in [11]). APS-SADE1 is compared with jDE [2], and APS-SADE2 is compared with SaDE [11]. Note that in SaDE and APS-SADE2 the fourth strategy “DE/current-to-rand/1” is not controlled by the crossover operation as originally used in [11]. For APS-SADE1, jDE, APS-SADE2, and SaDE, other parameters are kept unchanged as shown in Section 4.1. The error values are shown in Table 4. All results are averaged over 50 independent runs. The best and results are highlighted in **boldface**. The *intermediate* results are also reported for the functions where several algorithms can obtain the global optimum of these

Table 3: Comparison on the Error Values of DE Variants with Different Strategy Adaptation Methods for All Functions at $D = 30$.

F	NFFEs	SaDE	AdapSS-DE	SADE	Uniform-SADE	APS-SADE
f_{01}	300,000	3.02E-80 ± 3.26E-80†	**3.76E-99 ± 1.66E-98**	4.09E-47 ± 4.50E-47†	3.95E-85 ± 3.45E-85†	**3.74E-99 ± 4.76E-99**
f_{02}	300,000	5.41E-38 ± 3.45E-38†	**2.57E-48 ± 1.87E-48**‡	8.82E-22 ± 3.54E-22†	3.27E-40 ± 1.90E-40†	**1.46E-47 ± 1.02E-47**
f_{03}	300,000	2.78E-17 ± 3.98E-17†	**2.70E-20 ± 7.94E-20**†	2.28E-08 ± 1.76E-08†	2.01E-19 ± 4.21E-19†	**1.74E-21 ± 3.41E-21**
f_{04}	300,000	6.62E-06 ± 1.43E-05†	1.10E-05 ± 2.40E-05†	**8.32E-10 ± 4.50E-10**‡	**1.10E-09 ± 2.65E-09**‡	2.04E-08 ± 6.64E-08
f_{05}	300,000	7.97E-02 ± 5.64E-01†	7.97E-02 ± 5.64E-01†	**4.57E-16 ± 6.21E-16**†	7.97E-02 ± 5.64E-01	7.97E-02 ± 5.64E-01
f_{06}	10,000	9.37E+01 ± 2.12E+01†	**1.19E+01 ± 3.00E+00**‡	8.99E+02 ± 1.51E+02†	5.75E+01 ± 1.32E+01†	**1.75E+01 ± 4.27E+00**
	300,000	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_{07}	300,000	1.53E-03 ± 4.68E-04†	**1.01E-03 ± 2.96E-04**	3.34E-03 ± 8.32E-04†	1.30E-03 ± 3.51E-04†	**9.78E-04 ± 2.90E-04**
f_{08}	300,000	7.35E+03 ± 2.74E+02†	**7.09E+03 ± 3.68E+02**	7.27E+03 ± 3.52E+02	7.25E+03 ± 3.54E+02	**7.22E+03 ± 3.11E+02**
f_{09}	300,000	1.50E+02 ± 8.60E+00†	**1.41E+02 ± 1.03E+01**	1.66E+02 ± 1.02E+01†	1.49E+02 ± 7.80E+00†	**1.40E+02 ± 1.03E+01**
f_{10}	50,000	7.47E-06 ± 1.91E-06†	**1.84E-07 ± 4.94E-08**	4.92E-03 ± 1.01E-03†	2.80E-06 ± 5.80E-07†	**1.62E-07 ± 4.77E-08**
	300,000	**4.07E-15 ± 5.02E-16**	4.14E-15 ± 0.00E+00	4.14E-15 ± 0.00E+00	4.14E-15 ± 0.00E+00	**4.07E-15 ± 5.02E-16**
f_{11}	300,000	**1.48E-04 ± 1.05E-03**	**1.48E-04 ± 1.05E-03**	3.45E-04 ± 1.73E-03	**0.00E+00 ± 0.00E+00**	3.45E-04 ± 1.73E-03
f_{12}	50,000	3.33E-11 ± 1.79E-11†	**6.44E-15 ± 6.34E-15**	6.27E-05 ± 5.16E-05†	4.82E-12 ± 2.95E-12†	**9.80E-15 ± 6.78E-15**
	300,000	1.57E-32 ± 0.00E+00	1.57E-32 ± 0.00E+00	1.57E-32 ± 0.00E+00	1.57E-32 ± 0.00E+00	1.57E-32 ± 0.00E+00
f_{13}	50,000	5.19E-09 ± 4.65E-09†	**7.55E-10 ± 5.27E-09**†	5.67E-02 ± 4.44E-02†	8.43E-10 ± 6.48E-10†	**3.10E-12 ± 8.51E-12**
	300,000	1.35E-32 ± 0.00E+00	1.35E-32 ± 0.00E+00	1.35E-32 ± 0.00E+00	1.35E-32 ± 0.00E+00	1.35E-32 ± 0.00E+00
$w/t/l$		12/1/0	4/7/2	10/2/1	9/3/1	–

† indicates APS-SADE is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that APS-SADE is significantly worse than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.

functions. In these cases, the Wilcoxon signed-rank test is only compared with the intermediate results.

Compared between APS-SADE1 and jDE, it can be seen that APS-SADE1 is significantly better than jDE on 10 out of 13 functions. On the rest 3 functions (f_{04}, f_{08}, f_{09}), jDE significantly outperforms APS-SADE1.

Compared APS-SADE2 with SaDE, the similar results can be observed like APS-SADE1 vs. jDE. On 10 functions, APS-SADE2 is significantly better than SaDE, while only on 2 functions (f_{03} and f_{04}) APS-SADE2 is worse than SaDE. On function f_{08}, there is no significant difference between APS-SADE2 and SaDE with respect to the intermediate error values.

According to the analysis on the results shown in Table 4, we can conclude that the enhanced performance of APS-SADE is not influenced by the control parameter adaptation techniques proposed in jDE and SaDE. Thus, our proposed APS technique for strategy adaptation can be similarly useful for performance enhancement of other adaptive DE variants.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a probability-matching-based adaptive parameter selection (APS) technique for the strategy adaptation in DE. The APS technique is used to alleviate the drawback in our previous proposed strategy adaptation mechanism [5]. The main contribution of this paper is the APS technique for adaptively choose the most suitable parameter in the parameter pool while solving a problem. By integrating APS and SaM into DE, APS-SADE is proposed to adaptively select the most suitable strategy for a specific problem. Based on the experimental results, we can conclude that:

- APS-SADE is able enhance the original DE algorithm with single strategy. Moreover, the analysis of strategy adaptation indicates that our proposed method is able to adaptively select the most suitable strategy for a specific problem without any prior knowledge. In addition, in APS-SADE the cooperation of different strategies in the strategy pool is implemented.

- The proposed APS method for strategy adaptation alleviates the order-influence of strategies in SaM proposed in [5] and accelerates the original SaM to select the most suitable strategy for the problem at hand. Therefore, APS-SADE obtains better results compared with SADE and Uniform-SADE.

- The enhanced performance of APS-SADE is not influenced by the control parameter adaptation techniques.

Although the APS method is used for the strategy adaptation in this work, it is a general method for parameter adaptation. This method could also be used for other parameter adaptation in evolutionary algorithms, such as the adaptation of CR and F in DE, the adaptation of μ_{CR} and μ_F in JADE [23], etc. In our future work, we will verify this expectation. In addition, we will test our approach in more test problems, such as the CEC-2005 bechmark functions [17], and compare it with other DE variants reviewed in [3].

Acknowledgment

This work was partly supported by the Foundation of State Key Lab of Software Engineering under Grant No. SKLSE2010-08-13, the National Natural Science Foundation of China under Grant No. 61075063, and the Research Fund for the Doctoral Program of Higher Education under Grant No. 20090145110007.

6. REFERENCES

[1] J. Brest, B. Bošković, S. Greiner, V. Žumer, and M. S. Maučec. Performance comparison of self-adaptive and adaptive differential evolution algorithms. *Soft Comput.*, 11(7):617–629, May 2007.

[2] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Trans. on Evol. Comput.*, 10(6):646–657, Dec 2006.

[3] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Trans. on Evol. Comput.*, 2011. To appear.

[4] D. E. Goldberg. Probability matching, the magnitude of reinforcement, and classifier system bidding. *Mach. Learn.*, 5(4):407–425, 1990.

Table 4: The Influence of Parameter Adaptation on APS-SADE. The Error Values Are Only Reported Herein.

F	NFFEs	jDE	APS-SADE1	SaDE	APS-SADE2
f_{01}	300,000	1.64E-61 $\pm$ 2.28E-61†	**9.87E-103 $\pm$ 3.04E-102**	2.77E-78 $\pm$ 7.11E-78†	**1.57E-92 $\pm$ 2.46E-92**
f_{02}	300,000	1.96E-36 $\pm$ 1.60E-36†	**3.38E-53 $\pm$ 4.73E-53**	7.18E-39 $\pm$ 5.19E-39†	**4.16E-46 $\pm$ 3.16E-46**
f_{03}	300,000	2.14E-06 $\pm$ 2.09E-06†	**4.31E-10 $\pm$ 1.34E-09**	**1.22E-06 $\pm$ 1.73E-06‡**	3.94E-03 $\pm$ 1.10E-02
f_{04}	300,000	**5.38E-09 $\pm$ 4.38E-09‡**	3.07E-08 $\pm$ 1.34E-07	**7.41E-17 $\pm$ 4.99E-17‡**	4.21E-09 $\pm$ 3.04E-09
f_{05}	300,000	8.79E+00 $\pm$ 1.84E+00†	**2.99E+00 $\pm$ 1.46E+00**	1.05E+00 $\pm$ 3.07E+00†	**8.03E-01 $\pm$ 3.26E+00**
f_{06}	10,000	5.96E+02 $\pm$ 1.28E+02†	**3.30E+01 $\pm$ 1.18E+01**	4.43E+01 $\pm$ 9.37E+00†	**1.44E+01 $\pm$ 2.74E+00**
	300,000	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00
f_{07}	300,000	3.50E-03 $\pm$ 7.54E-04†	**1.02E-03 $\pm$ 3.64E-04**	1.67E-03 $\pm$ 5.18E-04†	**1.37E-03 $\pm$ 3.47E-04**
f_{08}	100,000	**1.27E-10 $\pm$ 1.65E-10‡**	2.18E-10 $\pm$ 5.80E-10	**1.11E-05 $\pm$ 1.93E-05**	1.56E-05 $\pm$ 1.73E-05
	300,000	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00
f_{09}	100,000	**4.58E-04 $\pm$ 2.12E-03‡**	9.62E-02 $\pm$ 2.56E-01	1.76E+01 $\pm$ 2.29E+00†	**4.40E+00 $\pm$ 2.58E+00**
	300,000	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00
f_{10}	50,000	2.28E-04 $\pm$ 6.50E-05†	**1.17E-07 $\pm$ 6.86E-08**	2.34E-06 $\pm$ 4.83E-07†	**1.62E-07 $\pm$ 4.02E-08**
	300,000	4.14E-15 $\pm$ 0.00E+00	**2.79E-15 $\pm$ 1.74E-15**	4.14E-15 $\pm$ 0.00E+00	4.14E-15 $\pm$ 0.00E+00
f_{11}	50,000	8.57E-06 $\pm$ 2.50E-05†	**1.17E-07 $\pm$ 6.86E-08**	3.27E-10 $\pm$ 2.44E-10†	**1.08E-12 $\pm$ 1.21E-12**
	300,000	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00	0.00E+00 $\pm$ 0.00E+00
f_{12}	50,000	6.96E-08 $\pm$ 5.00E-08†	**7.65E-15 $\pm$ 7.93E-15**	3.17E-12 $\pm$ 1.73E-12†	**7.74E-15 $\pm$ 4.40E-15**
	300,000	1.57E-32 $\pm$ 0.00E+00	1.57E-32 $\pm$ 0.00E+00	1.57E-32 $\pm$ 0.00E+00	1.57E-32 $\pm$ 0.00E+00
f_{13}	50,000	2.14E-05 $\pm$ 2.31E-05†	**3.82E-12 $\pm$ 7.14E-12**	1.02E-09 $\pm$ 9.87E-10†	**1.61E-12 $\pm$ 5.97E-13**
	300,000	1.35E-32 $\pm$ 0.00E+00	1.35E-32 $\pm$ 0.00E+00	1.35E-32 $\pm$ 0.00E+00	1.35E-32 $\pm$ 0.00E+00
$w/t/l$		10/0/3	–	10/1/2	–

† indicates APS-SADE is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that APS-SADE is significantly worse than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.

[5] W. Gong, Z. Cai, C. X. Ling, and H. Li. Enhanced differential evolution with adaptive strategies for numerical optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Part B – Cybernetics*, 2010. To appear.

[6] W. Gong, A. Fialho, and Z. Cai. Adaptive strategy selection in differential evolution. In J. Branke, editor, *Genetic and Evolutionary Computation Conference (GECCO 2010)*, pages 409–416. ACM Press, July 2010.

[7] R. Mallipeddi, P. Suganthan, Q. Pan, and M. Tasgetiren. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, 2010. In press.

[8] N. Noman and H. Iba. Accelerating differential evolution using an adaptive local search. *IEEE Trans. on Evol. Comput.*, 12(1):107–125, Feb 2008.

[9] Q.-K. Pan, P. Suganthan, L. Wang, L. Gao, and R. Mallipeddi. A differential evolution algorithm with self-adapting strategy and control parameters. *Computers and Operations Research*, 38(1):394 – 408, Jan. 2011.

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

[11] A. K. Qin, V. L. Huang, and P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans. on Evol. Comput.*, 13(2):398–417, Apr 2009.

[12] A. K. Qin and P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *IEEE Congr. on Evol. Comput.*, pages 1785–1791, 2005.

[13] C. Shaw, K. Williams, and R. Assassa. Patients' views of a new nurse-led continence service. *Journal of Clinical Nursing*, 9(4):574–584, 2003.

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

[15] R. Storn and K. Price. Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optim.*, 11(4):341–359, Dec 1997.

[16] R. Storn and K. Price. Home page of differential evolution, 2010.

[17] 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 CEC2005 special session on real-parameter optimization, 2005.

[18] D. Thierens. An adaptive pursuit strategy for allocating operator probabilities. In *Proc. Genetic Evol. Comput. Conf.*, pages 1539–1546, 2005.

[19] Y. Wang, Z. Cai, and Q. Zhang. Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Trans. on Evol. Comput.*, 2010. To appear.

[20] X.-F. Xie and W.-J. Zhang. SWAF: Swarm algorithm framework for numerical optimization. In *Proc. Genetic Evol. Comput. Conf., Part I*, volume 3102 of *LNCS*, pages 238–250, 2004.

[21] Z. Yang, K. Tang, and X. Yao. Self-adaptive differential evolution with neighborhood search. In *Proc. IEEE Congress on Evol. Comput.*, pages 1110–1116, 2008.

[22] X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster. *IEEE Trans. on Evol. Comput.*, 3(2):82–102, Jul 1999.

[23] J. Zhang and A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. on Evol. Comput.*, 13(5):945–958, Oct 2009.