A Unified Differential Evolution Algorithm for Global Optimization

Ji Qiang and Chad Mitchell

***Abstract*—In this paper, we propose a new unified differential evolution (uDE) algorithm for single objective global optimization. Instead of selecting among multiple mutation strategies as in the conventional differential evolution algorithm, this algorithm employs a single equation as the mutation strategy. It has the virtue of mathematical simplicity and also provides users the flexbility for broader exploration of different mutation strategies. Numerical tests using twelve basic unimodal and multimodal functions show promising performance of the proposed algorithm in comparison to convential differential evolution algorithms.**

***Index Terms*—differential evolution (DE), evolutionary optimization.**

I. Introduction

Differential evolution is a simple yet efficient population-based, stochastic, evolutionary algorithm. It was first introduced by Storn and Price in 1995 as a global optimization algorithm to optimize real parameter, real valued functions and has received a lot of interest since then [1], [2], [3], [4], [6], [7], [8]. In a number of comparison studies, the differential evolution algorithm performed more efficiently than many stochastic optimization methods such as simulated annealing, controlled random search, evolutionary programming, the particle swarm method, and genetic algorithms [2], [9], [10], [11]. It has been successfully used in a variety of applications and demonstrated its effectiveness.

The differential evolution algorithm uses the scaled differences of parent solutions as a mutation operator to generate next-generation candidates for global optimization. In the paper of Storn and Price, five different mutation strategies were proposed [12]. Several additional variants of these mutation strategies were later proposed to improve the properties of the mutation operation, e.g. to make it rotationally invariant [13]. In this paper, we propose a unified differential evolution algorithm for global optimization. This algorithm integrates the various commonly-used mutation strategies into a single expression. It is mathematically simpler than the conventional algorithm with its multiple mutation strategies, and also provides users the flexibility to explore new combinations of different mutation strategies during optimization.

The rest of the paper is organized as follows: in Section 2, the standard differential evolution algorithm with multiple mutation strategies is reviewed. In Section 3, the unified differential evolution agorithm is discussed. In Section 4, numerical benchmarks with conventional mutation strategies are presented. A discussion and summary are given in Section 5.

The authors are with the Accelerator and Fusion Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA (e-mail: jqiang@lbl.gov; chadmitchell@lbl.gov).

Manuscript received xx.

II. Standard differential evolution algorithm

The differential evolution algorithm starts with a population initialization. A group of NP solutions in the control parameter space is randomly generated to form the initial population. This initial population can be generated by sampling from a uniform distribution within the parameter space if no prior information about the optimal solution is available, or by sampling from a known distribution (e.g., Gaussian) if some prior information is available.

After initialization, the differential evolution algorithm updates the population from one generation to the next generation until reaching a convergence condition or until the maximum number of function evaluations is reached. At each generation, the update step consists of three operations: mutation, crossover, and selection. The mutation and the crossover operations produce new candidates for the next generation population and the selection operation is used to select from among these candidates the appropriate solutions to be included in the next generation.

A. Mutation Strategies

During the mutation operation stage, for each population member (target vector) $\vec{x}_i$, $i = 1, 2, 3, \cdots, NP$ at generation G, a new mutant vector $\vec{v}_i$ is generated by following a mutation strategy. Some commonly used conventional mutation strategies are [1], [2], [7]:

$$\text{DE/rand/1}: \quad \vec{v}_i = \vec{x}_{r_1} + F_{xc}(\vec{x}_{r_2} - \vec{x}_{r_3}) \tag{1}$$

$$\text{DE/rand/2}: \quad \vec{v}_i = \vec{x}_{r_1} + F_{xc}(\vec{x}_{r_2} - \vec{x}_{r_3}) + F_{xc}(\vec{x}_{r_4} - \vec{x}_{r_5}) \tag{2}$$

$$\text{DE/best/1}: \quad \vec{v}_i = \vec{x}_b + F_{xc}(\vec{x}_{r_1} - \vec{x}_{r_2}) \tag{3}$$

$$\text{DE/best/2}: \quad \vec{v}_i = \vec{x}_b + F_{xc}(\vec{x}_{r_1} - \vec{x}_{r_2}) + F_{xc}(\vec{x}_{r_3} - \vec{x}_{r_4}) \tag{4}$$

$$\text{DE/current-to-best/1}: \quad \vec{v}_i = \vec{x}_i + F_{cr}(\vec{x}_b - \vec{x}_i) + F_{xc}(\vec{x}_{r_1} - \vec{x}_{r_2}) \tag{5}$$

$$\text{DE/current-to-best/2}: \quad \vec{v}_i = \vec{x}_i + F_{cr}(\vec{x}_b - \vec{x}_i) + F_{xc}(\vec{x}_{r_1} - \vec{x}_{r_2}) + F_{xc}\ (\vec{x}_{r_3} - \vec{x}_{r_4}) \tag{6}$$

$$\text{DE/current-to-rand/1}: \quad \vec{v}_i = \vec{x}_i + F_{cr}(\vec{x}_{r_1} - \vec{x}_i) + F_{xc}\ (\vec{x}_{r_2} - \vec{x}_{r_3}) \tag{7}$$

$$\text{DE/current-to-rand/2}: \ \vec{v}_i = \vec{x}_i + F_{cr}(\vec{x}_{r_1} - \vec{x}_i) + F_{xc}\ (\vec{x}_{r_2} - \vec{x}_{r_3}) + F_{xc}\ (\vec{x}_{r_4} - \vec{x}_{r_5}) \quad (8)$$

$$\text{DE/rand-to-best/1}: \ \vec{v}_i = \vec{x}_{r_1} + F_{cr}(\vec{x}_b - \vec{x}_i) + F_{xc}\ (\vec{x}_{r_2} - \vec{x}_{r_3}) \quad (9)$$

$$\text{DE/rand-to-best/2}: \ \vec{v}_i = \vec{x}_{r_1} + F_{cr}(\vec{x}_b - \vec{x}_i) + F_{xc}(\vec{x}_{r_2} - \vec{x}_{r_3}) + F_{xc}(\vec{x}_{r_4} - \vec{x}_{r_5}) \quad (10)$$

where the integers r_1, r_2, r_3, r_4 and r_5 are chosen randomly from the interval $[1, NP]$ and are different from the current index i, F_{xc} is a real scaling factor that controls the amplification of the differential variation, $\vec{x}_b$ is the best solution among the NP population members at the generation G, and F_{cr} is a weight for the combination between the original target vector and the best parent vector or the random parent vector. The strategy DE/rand/1 is the most widely used mutation strategy proposed in the original paper of Storn and Price. It has stronger exploration capability but may converge more slowly than the strategies that use the best solution from the parent generation. The strategy DE/rand/2 uses two difference vectors and may result in better perturbation than the strategies that use one difference vector [14]. The strategies DE/best/1 and DE/best/2 take advantage of the best solution found in the parent population and have a faster convergence towards the optimal solution [15]. However, they may become stuck at a local minimum point during multimodal function optimization. The DE/current-to-best/1 and DE/current-to-best/2 strategies provide a compromise between exploitation of the best solution and exploration of the parameter space. The DE/current-to-rand/1 and DE/current-to-rand/2 mutation strategies are rotation-invariant strategies [13]. The DE/rand-to-best/ strategies are similar to the DE/current-to-best/ strategies, but larger diversity of the mutant vector is attained by using a randomly selected parent vector instead of the current target parent vector.

B. Crossover

A crossover operation between the new generated mutant vector $\vec{v}_i$ and the target vector $\vec{x}_i$ is used to further increase the diversity of the new candidate solution. This operation combines the two vectors into a new trial vector $\vec{U}_i, i = 1, 2, 3, \cdots, NP$, where the components of the trial vector are obtained from the components of $\vec{v}_i$ or $\vec{x}_i$ according to a crossover probability CR. In the binomial crossover scheme, for a D dimensional control parameter space, the new trial vector $\vec{U}_i$, $i = 1, 2, \cdots, NP$ is generated using the following rule:

$$\vec{U}_i = (u_{i1}, u_{i2}, \cdots, u_{iD}) \quad (11)$$

$$u_{ij} = \begin{cases} v_{ij}, & \text{if } \text{rand}_j \leq CR \ \text{ or } j = \text{mbr}_i \\ x_{ij}, & \text{otherwise} \end{cases} \quad (12)$$

where rand_j is a randomly chosen real number in the interval $[0, 1]$, and the index mbr_i is a randomly chosen integer in the range $[1, D]$. This ensures that the new trial vector contains at least one component from the new mutant vector.

C. Selection

The new generated trial solution $\vec{U}_i$ is checked against the boundary in the control parameter space. If the solution is out of the boundary, a new trial solution is generated from a random sampling within the boundary.

The selection operation in DE is based on a one-to-one comparison. The new trial solution $\vec{U}_i$ is checked against the original target parent solution $\vec{x}_i$. If the new trial solution produces a better objective function value, it will be put into the next generation $(G+1)$ population. Otherwise, the original parent is kept in the next generation population.

The above procedure is repeated for all NP parents to generate the next generation population. Many generations are used to attain the final global optimal solution.

III. The unified differential evolution algorithm

Ten different mutation strategies have been proposed for the standard differential evolution algorithm (Eqs. 1-10). While DE/rand/1/bin has been widely used, DE/best/1/bin was proposed to have better performance in a number of optimization test examples [15]. Meanwhile, DE/best/2/bin was suggested as a highly beneficial method in the ICEC'96 contest [16]. In this paper, we propose a unified mutation strategy for the differential evolution algorithm. The unified mutation strategy can be written as:

$$\vec{v}_i = \vec{x}_i + F_1(\vec{x}_b - \vec{x}_i) + F_2(\vec{x}_{r_1} - \vec{x}_i) + F_3(\vec{x}_{r_2} - \vec{x}_{r_3}) + F_4(\vec{x}_{r_4} - \vec{x}_{r_5}) \quad (13)$$

where F_1, F_2, F_3 and F_4 are four parameters that can be adpatively adjusted during the optimization process. From the above equation, one can see that for $F_1 = 0$, $F_2 = 1$, and $F_4 = 0$, this equation reduces to DE/rand/1; for $F_1 = 0$, $F_2 = 1$, and $F_3 = F_4$, it reduces to DE/rand/2; for $F_1 = 1$, $F_2 = 0$, and $F_4 = 0$, it reduces to DE/best/1; for $F_1 = 1$, $F_2 = 0$, and $F_3 = F_4$, it reduces to DE/best/2; for $F_2 = 0$ and $F_4 = 0$, it reduces to DE/current-to-best/1; for $F_2 = 0$ and $F_3 = F_4$, it reduces to DE/current-to-best/2; for $F_1 = 0$, and $F_4 = 0$, it reduces to DE/current-to-rand/1; for $F_1 = 0$, and $F_3 = F_4$, it reduces to DE/current-to-rand/2; for $F_2 = 1$, and $F_4 = 0$, it reduces to DE/rand-to-best/1; for $F_2 = 1$, and $F_3 = F_4$, it reduces to DE/rand-to-best/2. Using the single equation (13), ten mutation strategies in the standard differential evolution algoritm can be written as a single mutation strategy. Meanwhile, by using a different set of parameters F_1, F_2, F_3, F_4, a combination of different strategies can be achieved. For example, from our experience, we found that using $F_1 = 0.25$, $F_2 = 0.25$, $F_3 = 0.2$, $F_4 = 0.2$, and $CR = 0.8$ in uDE can give reasonable performance in a number of test studies. If these parameters can be adaptively adjusted during the optimization evolution, then multiple mutation strategies and their combinations can be used during different stages of optimization. Thus, the unified strategy has the virtue of mathematical simplicity and also provides the user with flexibility for broader exploration of different mutation strategies.

In the following, we also propose a simple adaptive method for uDE (called uDEadapt) so that the user does not have

to specify the five parameters (F_1, F_2, F_3, F_4, CR) prior to optimization. In this adaptive uDE scheme, after the population initialization, the five parameters are randomly generated following a uniform distribution in the interval $[0, 1]$, and the mutation strategy (13) is used to produce new trial solutions for the next generation. If the best solution in the new generation is better than the best solution of the parent generation, this group of control parameters is put into a successful pool and these parameters are reused to produce another generation of trial solutions. If the best solution of the current parent generation is not improved, these five control parameters will be discarded. A new set of control parameters are then generated from either: 1) a uniform sampling within $[0, 1]$ with probability 0.5, or 2) a random choice from the parameters stored in the successful pool, provided the pool is not empty. This process is repeated for many generations until obtaining the maximum allowed number of function evaluations or convergence to the optimal solution.

IV. NUMERICAL BENCHMARK

A. Test functions

Twelve well-known test functions that have been widely used in studies of global optimization in evolutionary computation are used in this paper for numerical benchmarking [10], [11], [15], [17], [18]. These functions are given below:

(1) Sphere function

$$F_{\mathrm{sph}}(\vec{x}) = \sum_{i=1}^{N} x_i^2; \quad -100 \le x_i \le 100;$$

(2) Schwefel's problem 1.2

$$F_{\mathrm{sch2}}(\vec{x}) = \sum_{j=1}^{N} \left(\sum_{i=1}^{j} x_i \right)^2; \quad -100 \le x_i \le 100;$$

(3) Quartic function with noise

$$F_{\mathrm{qrt}}(\vec{x}) = \sum_{i=1}^{N} i x_i^4 + \mathrm{rand}[0, 1); \quad -1.28 \le x_i \le 1.28;$$

(4) Rosenbrock's function

$$F_{\mathrm{ros}}(\vec{x}) = \sum_{i=1}^{N-1} \left(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \right);$$
$$-100 \le x_i \le 100;$$

(5) Ackley's function

$$F_{\mathrm{ack}}(\vec{x}) = 20 + \exp(1) - 20 \exp\left(-0.2 \sqrt{\frac{1}{N} \sum_{i=1}^{N} x_i^2} \right)$$
$$- \exp\left(\frac{1}{N} \sum_{i=1}^{N} \cos(2\pi x_i) \right); \quad -32 \le x_i \le 32;$$

(6) Griewank's function

$$F_{\mathrm{grw}}(\vec{x}) = \sum_{i=1}^{N} \frac{x_i^2}{4000} - \prod_{i=1}^{N} \cos \frac{x_i}{\sqrt{i}} + 1;$$
$$-600 \le x_i \le 600;$$

(7) Rastrigin's function

$$F_{\mathrm{ras}}(\vec{x}) = 10N + \sum_{i=1}^{N} \left(x_i^2 - 10 \cos(2\pi x_i) \right);$$
$$-5 \le x_i \le 5;$$

(8) Schwefel's function

$$F_{\mathrm{sch}}(\vec{x}) = 418.9829N - \sum_{i=1}^{N} (x_i \sin(\sqrt{|x_i|}));$$
$$-500 \le x_i \le 500;$$

(9) Salomon's function

$$F_{\mathrm{sal}}(\vec{x}) = -\cos\left(2\pi \sqrt{\sum_{i=1}^{N} x_i^2} \right)$$
$$+ 0.1 \sqrt{\sum_{i=1}^{N} x_i^2} + 1; \quad -100 \le x_i \le 100;$$

(10) Whitely's function

$$F_{\mathrm{wht}}(\vec{x}) = \sum_{j=1}^{N} \sum_{i=1}^{N} \left(\frac{y_{i,j}^2}{4000} - \cos(y_{i,j}) + 1 \right);$$
where $y_{i,j} = 100(x_j - x_i^2)^2 + (1 - x_i)^2$;
$$-100 \le x_i \le 100;$$

(11) Weierstrass's function

$$F_{\mathrm{wst}}(\vec{x}) = \sum_{i=1}^{N} w(x_i, 0.5, 3, 20) - N w(0, 0.5, 3, 20);$$
where
$$w(x_i, a, b, m) = \sum_{k=0}^{m} a^k \cos(2\pi b^k (x_i + 0.5));$$
$$-0.5 \le x_i \le 0.5;$$

(12) Generalized penalized function

$$F_{\mathrm{pn1}}(\vec{x}) = \frac{\pi}{N} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{N-1} (y_i - 1)^2 [1 \right.$$
$$\left. + 10 \sin^2(\pi y_{i+1})] + (y_N - 1)^2 \right\}$$
$$+ \sum_{i=1}^{N} u(x_i, 10, 100, 4);$$
where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a_i \\ 0, & -a \le x_i \le a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases}$$
$$-50 \le x_i \le 50;$$

The sphere function is a continuous, unimodal and separable function. The Schewefel's problem 1.2 is a non-separable unimodal function. The noisy quartic function is a unimodal non-separable function with random noise in the objective value. The Rosenbrock's function with dimension greater

than three is a multimodal and non-separable problem. The global minimum lies inside a parabolic shaped flat valley. The Ackely's function is also a multimodal non-separable problem and has many local minima and a narrow global minimum. The Rastrigin's function is a complex multimodal separable problem with many local minima. The Griewank's function is a multimodal non-separable function. The Salomon's function and the Whitely's function are non-separable and highly multmodal with many local minima. The Weierstrass function is a multimodal, nonseparable continuous function that is differentiable only on a set of points. The generalized penalized funtion is a multimodal nonseparable irregular and discontinuous function. The number of local minima from test functions (5) to (12) increases quickly with the problem dimension. The exact global minimum for all of these problems is achieved for an objective function value of zero.

B. Benchmark results

To test the proposed uDE algorithms, we carried out numerical optimization using the 12 benchmark objective functions listed in the above subsection with dimensions $N = 10$, 30, and 50 respectively. We compare the proposed uDE algorithms with two widely used conventional DE algorithms, DE/rand/1/bin ($F = 0.9$, $CR = 0.9$) [2], [19], DE/rand/1/bin ($F = 0.5$, $CR = 0.9$) [10], [11], [15], and DE/best/1/bin ($F = 0.6$, $CR = 0.3$) [15]. The parameters used in uDE are given in Section III ($F_1 = 0.25$, $F_2 = 0.25$, $F_3 = 0.2$, $F_4 = 0.2$, and $CR = 0.8$). The choice of these parameters is based on the consideration of a balance between the exploitation of the best solution found in the current population, the exploration of the random solution, and the current target solution. The scale factors for the two difference vector and the crossover probability are chosen to increase the speed of convergence. The simple adaptive uDE (uDEadapt) algorithm is also used in the benchmark tests. The maximum number of function evalutions is set as $10,000N$. The population size (NP) for the 10, 30, and 50 dimensional problems is set as 50, 60, and 100, respectively. Each optimization is performed for 25 different random seeds. The average objective function value and its standard deviation at the end of the maximum number of function evaluations is reported in Table I for each of the 10 dimensional objective functions, in Table II for the 30 dimensional functions, and in Table III for the 50 dimensional functions. The minimum average objective value for each problem is shown in bold font. It is seen that among the conventional differential evolution algorithms, the algorithm DE/best/1/bin performs better than DE/rand/1/bin with either $F = 0.9, CR = 0.9$ or $F = 0.5, CR = 0.9$. The performance of the conventional differential evolution algorithm DE/rand/1/bin is quite sensitive to the choice of the scaling parameter F. The use of $F = 0.5$ for DE/rand/1/bin shows better performance than the case with $F = 0.9$. In the numerical tests with ten dimensional objective functions, the conventional DEs perform better than the uDEs in eight out of 12 test examples. There are still two examples in which the uDEs outperform the convential DE algorithms. In the tests with 30 dimensional objective functions, the conventional DEs win six out of the 12 test examples. The uDEs win five out of the 12 test examples. In the tests with 50 dimensional objective functions, the conventional DEs win four out of 12 test examples, and the unified DEs win seven out the 12 test examples. From these test problems, it appears that the uDEs perform better with increasing problem dimension. This might be due to the balance of exploitation and exploration in the uDEs with combined mutation strategies.

In Figs. 1 and 2, we show the evolution of the objective function value of the 12 test functions for the algorithms shown in Table III with dimension $N = 50$. At each generation, the objective function value has been averaged over 25 random seeds. It is seen that the unified differential algorithms perform quite well in most test examples to reach converged solutions. The adaptive uDE algorithm (uDEadapt) does not have the fast convergence rate of uDE, but proves more robust in finding the optimal solutions in Rosenbrock's function and in Whitely's function by adjusting the five control parameters during the optimization.

V. Discussion and Summary

In this paper, we proposed a unified mutation strategy for the differential evolution algorithm. In comparison to the standard differential evolution algorithm, this method has the advantages of both mathematical simplicity and flexibility for exploring broader mutation strategies. The disadvantage of this algorithm is that it involves more control parameters. Instead of three control parameters, F_{cr}, F_{xc} and CR, as in the conventional DE, the unified DE has five control parameters, F_1, F_2, F_3, F_4 and CR. The performance of the algorithm will depend on the choice of these parameters. In this study, we suggested a simple adaptive scheme to select a set of control parameters for the whole population. In previous studies, a number of adaptive methods have been proposed to determine the control parameters and the combination of mutation strategy and control parameters [10], [11], [14], [20], [22], [23], [24]. These more sophisticated adaptive methods involve using a set of control parameters for each individual solution in the population. In future work, we will explore how these more advanced adaptive methods can be used to improve the performance of the proposed unified differential evolution algorithm.

Acknowledgment

The work was supported by the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used computer resources at the National Energy Research Scientific Computing Center and at the National Center for Computational Sciences.

References

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

[2] R. Storn and K. Price, "Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization 1a1:341-359, (1997).

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

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT DE STRATEGIES FOR $N = 10$.

Function	rand/1/bin (0.9,0.9)		rand/1/bin (0.5,0.9)		best/1/bin (0.6,0.3)		uDE		uDEadapt	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_{sph}	2.54E-13	2.35E-13	2.88E-83	5.42E-83	**3.73E-94**	1.14E-93	3.07E-79	1.44E-78	4.62E-69	2.26E-68
F_{sch2}	7.62E-06	5.12E-06	**1.13E-53**	1.42E-53	4.79E-13	5.40E-13	3.74E-52	9.40E-52	1.66E-26	8.14E-26
F_{qrt}	8.15E-03	2.79E-03	8.15E-04	3.64E-04	6.80E-04	2.25E-04	**5.96E-05**	3.15E-05	3.78E-04	2.26E-04
F_{ros}	**1.08E-03**	8.00E-04	3.08E+00	1.24E+00	1.78E+00	2.90E+00	5.66E+00	3.84E+00	9.61E-01	2.21E+00
F_{ack}	3.26E-07	1.62E-07	3.00E-15	6.96E-16	2.97E-15	6.96E-16	2.54E-15	1.30E-15	**-4.44E-16**	0.00E+00
F_{grw}	2.10E-01	1.46E-01	1.56E-02	1.25E-02	**0.00E+00**	0.00E+00	8.88E-18	4.35E-17	1.11E-04	5.42E-04
F_{ras}	8.88E+00	6.11E+00	7.64E-01	8.57E-01	**3.19E-46**	1.07E-45	2.83E-06	1.37E-05	5.91E-25	2.19E-24
F_{sch}	2.58E+01	9.18E+01	**1.27E-04**	3.15E-12	6.12E-04	5.59E-04	4.23E+02	4.42E+02	4.64E-02	5.28E-02
F_{sal}	1.07E-01	1.05E-02	**9.99E-02**	1.86E-09	**9.99E-02**	3.87E-10	**9.99E-02**	0.00E+00	**9.99E-02**	2.95E-09
F_{wht}	3.88E+01	1.70E+01	1.26E+01	1.36E+01	**2.97E-01**	6.45E-01	1.84E+00	7.17E-01	2.78E+00	2.75E+00
F_{wst}	2.52E-03	7.86E-04	**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_{pn1}	2.04E-13	1.89E-13	**4.71E-32**	0.00E+00	7.63E-08	5.46E-08	2.69E-06	2.53E-06	6.15E-08	8.75E-08

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT DE STRATEGIES FOR $N = 30$.

Function	rand/1/bin (0.9,0.9)		rand/1/bin (0.5,0.9)		best/1/bin (0.6,0.3)		uDE		uDEadapt	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_{sph}	1.19E+00	8.96E-01	**1.35E-71**	1.84E-71	2.29E-50	3.85E-50	1.19E-35	1.56E-35	1.17E-25	5.27E-25
F_{sch2}	4.10E+03	1.93E+03	**4.95E-12**	6.41E-12	1.84E+00	6.71E-01	1.17E-09	1.28E-09	2.58E-05	4.43E-05
F_{qrt}	6.73E-02	1.82E-02	2.25E-03	4.84E-04	2.55E-03	7.63E-04	7.96E-05	2.80E-05	**5.20E-04**	2.50E-04
F_{ros}	2.44E+03	1.92E+03	1.28E+01	9.19E+00	3.11E+00	5.35E+00	3.57E+00	7.19E+00	**2.09E-14**	6.77E-14
F_{ack}	6.32E-01	3.79E-01	3.68E-15	1.30E-15	4.53E-15	1.74E-15	**3.11E-15**	0.00E+00	5.54E-13	2.13E-12
F_{grw}	8.31E-01	1.38E-01	5.92E-04	2.01E-03	**0.00E+00**	0.00E+00	8.88E-18	3.01E-17	1.33E-17	3.61E-17
F_{ras}	1.27E+02	3.72E+01	1.22E+01	4.11E+00	**2.58E-22**	4.43E-22	4.26E-16	7.59E-16	7.63E-03	3.64E-02
F_{sch}	5.79E+03	1.40E+03	7.16E+02	7.96E+02	**3.82E-04**	7.28E-12	1.48E+03	2.45E+03	2.43E-01	7.37E-01
F_{sal}	1.57E+00	2.41E-01	1.82E-01	3.66E-02	1.64E-01	4.80E-02	**9.99E-02**	0.00E+00	**9.99E-02**	1.73E-10
F_{wht}	7.04E+06	3.17E+07	2.71E+02	1.42E+02	1.32E+01	2.33E+01	1.96E+00	4.61E+00	**8.40E-30**	3.35E-29
F_{wst}	2.12E+00	7.44E-01	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	3.79E-10	1.84E-09
F_{pn1}	2.29E-01	2.89E-01	4.15E-03	2.03E-02	**1.57E-32**	4.86E-40	3.28E-23	1.12E-22	1.64E-17	4.07E-17

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT DE STRATEGIES FOR $N = 50$.

Function	rand/1/bin (0.9,0.9)		rand/1/bin (0.5,0.9)		best/1/bin (0.6,0.3)		uDE		uDEadapt	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_{sph}	2.85E+03	9.21E+02	4.84E-35	8.77E-35	**7.90E-38**	4.48E-38	3.32E-34	3.58E-34	5.11E-23	2.45E-22
F_{sch2}	7.35E+04	5.83E+03	1.28E+00	4.69E-01	6.76E+02	2.50E+02	**2.06E-06**	3.47E-06	1.06E-03	1.19E-03
F_{qrt}	1.43E+00	8.18E-01	6.34E-03	1.18E-03	3.93E-03	9.42E-04	**9.07E-05**	3.70E-05	4.89E-04	2.14E-04
F_{ros}	1.36E+08	6.28E+07	2.69E+01	1.59E+01	4.16E+00	2.45E+00	2.81E+00	7.33E+00	**6.17E-16**	3.01E-15
F_{ack}	9.29E+00	8.04E-01	6.66E-15	0.00E+00	6.66E-15	0.00E+00	**3.11E-15**	0.00E+00	3.12E-14	1.17E-13
F_{grw}	2.66E+01	8.29E+00	3.11E-16	7.02E-17	**0.00E+00**	0.00E+00	1.87E-16	9.27E-17	1.51E-16	7.61E-17
F_{ras}	4.53E+02	4.00E+01	1.91E+02	5.33E+01	1.09E-11	5.33E-11	**2.77E-15**	1.13E-15	2.75E+00	6.03E+00
F_{sch}	1.37E+04	5.61E+02	1.22E+04	1.63E+03	**6.36E-04**	0.00E+00	3.21E+03	4.58E+03	1.36E+02	2.99E+02
F_{sal}	8.01E+00	7.13E-01	2.00E-01	1.97E-08	2.00E-01	1.21E-08	**9.99E-02**	1.86E-09	**9.99E-02**	1.64E-10
F_{wht}	1.23E+15	1.38E+15	1.39E+03	3.36E+02	2.05E+02	1.66E+02	1.26E+00	4.48E+00	**4.21E-36**	1.32E-35
F_{wst}	3.13E+01	3.09E+00	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	2.22E-09	1.08E-08
F_{pn1}	4.46E+05	5.99E+05	2.49E-03	1.22E-02	**8.12E-31**	5.94E-31	7.76E-27	2.62E-26	2.63E-20	1.28E-19

[4] U. K. Chakraborty (ed.), *Advances in Differential Evolution*, Springer, Berlin, 2008.

[5] A. Qing, *Differential Evolution: Fundamentals and Applications in Electrical Engineering*, John Wiley, New York, 2009.

[6] J. Zhang and A. C. Sanderson, *Adaptive Differential Evolution: A Robust Approach to Multimodal Problem Optimization*, Springer, Berlin, Germany, 2009.

[7] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," Artif. Intell. Rev. 33, p. 61, 2010.

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

[9] M. M. Ali and A. Trn, "Population set based global optimization algorithms: Some modifications and numerical studies," Comput. Oper. Res., vol. 31, no. 10, pp. 1703-1725, 2004.

[10] J. Brest, S. Greiner, B. Bovskovic, M. Mernik, and V. ? Zumer, "Self adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," IEEE Transactions on Evolutionary Computation, vol. 10, no. 6, pp. 646-657, 2006.

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

[12] Storn R (1996b) On the usage of differential evolution for function optimization. In: Proceedings of the IEEE biennial conference of the North American fuzzy information processing society, pp 519-523.

[13] 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.

[14] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization,"

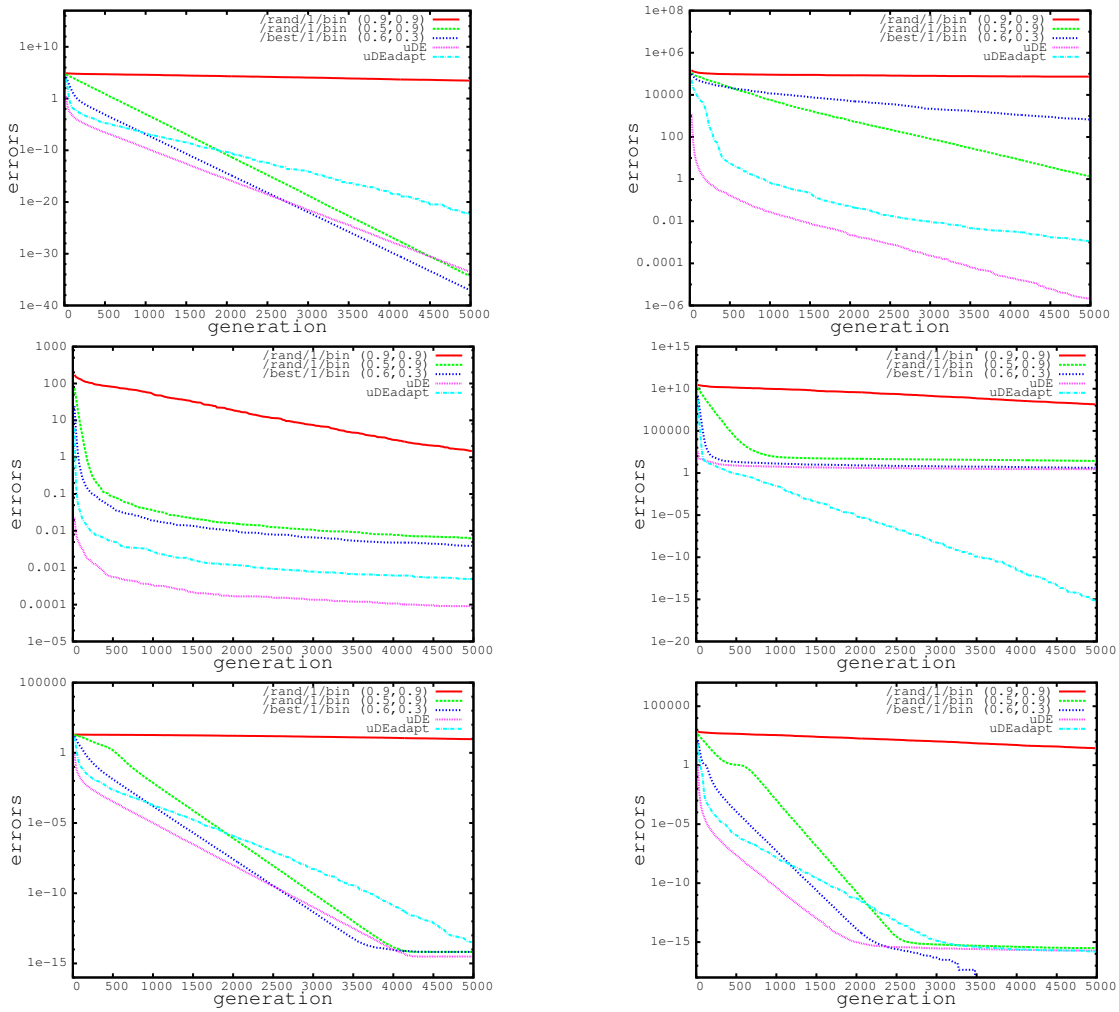

Fig. 1. Evolution of the average error in the first six test functions: top left is the sphere function, top right is the Schwefel's problem 1.2, middle left is the noisy quartic function, middle right is the Rosenbrock's function, bottom left is the Ackley's function, and bottom right is the Griewank's function.

IEEE Transactions on Evolutionary Computation, vol. 13, no. 2, pp. 398-417, 2009.

[15] E. Mezura-Montes, J. Velazquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in Proc. Genet. Evol. Comput. Conf., 2006, pp. 485- 492.

[16] K. Price and R. Storn, "Minimizeing the real functions of the ICEC'96 contest by differential evolution," IEEE International Conference on Evolutionary Computation, 1996, pp.842-844.

[17] P. N. Suganthan, N. Hansen, J. J. Lang, 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] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," IEEE Transactions on Evolutionary Computation, vol. 12, no. 1, pp. 107-125, 2008.

[19] J. Ronkkonen, S. Kukkonen, and K. Price, in Proceedings IEEE Congress on Evolutionary Computation, CEC 2005, vol. 1, p. 506-513, Edinburgh, UK (2005).

[20] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," Soft Computing, A Fusion of Foundations, Methodologies and Applications, vol. 9, no. 6, pp. 448-642, 2005.

[21] Q. K. Pan, P. N. Suganthan, L. Wang, L. Gao, and R. Mallipeddi, "A differential evolution algorithm with self-adapting strategy and control parameters," Computers & Operations Research, vol. 38, no. 1, pp. 394-408, 2011.

[22] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," Applied Soft Computing 11, p. 1679, 2011.

[23] Y. Wang, Z. Cai, Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," IEEE transcations on Evolutionary Computation, vol. 15, no.1, p. 55, 2011.

[24] 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, vol. 41, no. 2, pp. 397-413, 2011.

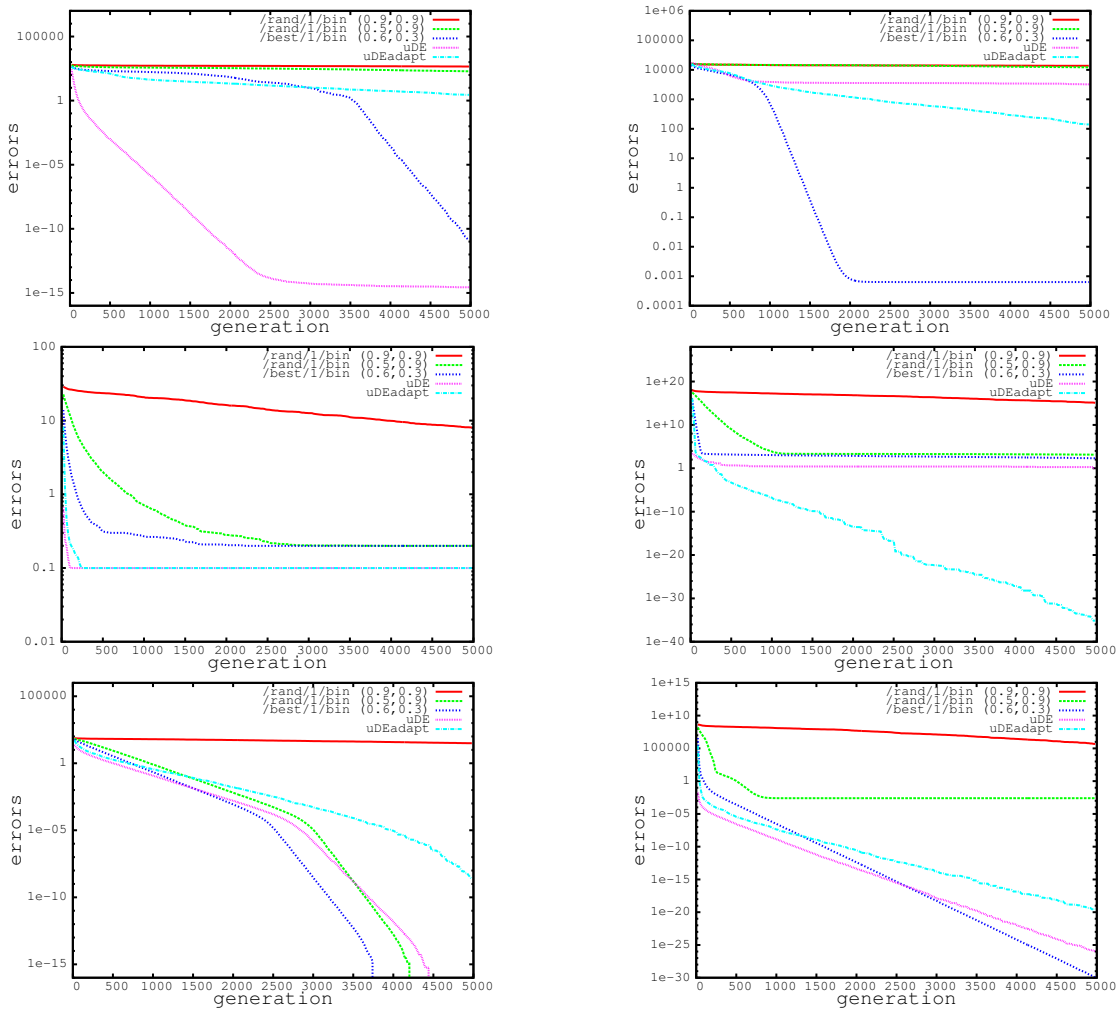

Fig. 2. Evolution of the average error in the first six test functions: top left is the Rastrigin's function, top right is the Schwefel's function, middle left is the Salomon's function, middle right is the Whitely's function, bottom left is the Weierstrass's function, and bottom right is the generalized penalized function.