2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

An Improved JADE algorithm for Global Optimization

Ming Yang, Zhihua Cai, Changhe Li and Jing Guan*

***Abstract*— In differential evolution (DE), the optimal value of the control parameters is problem-dependent. Many improved DE algorithms have been proposed with the aim of improving the effectiveness for solving general problems. As a very known adaptive DE algorithm, JADE adjusts the crossover probability CR of each individual by a norm distribution, in which the value of standard deviation is fixed, based on its historical record of success. The fixed and small standard deviation results in that the generated CR may not suitable for solving a problem. This paper proposed an improvement for the adaptation of CR, in which the standard deviation is adaptive. The diversity of values of CR was improved. This improvement was incorporated into the JADE algorithm and tested on a set of 25 scalable benchmark functions. The results showed that the adaptation of CR improved the performance of the JADE algorithm, particularly in comparisons with several other peer algorithms on high-dimensional functions.**

I. Introduction

DIFFERENTIAL EVOLUTION (DE), introduced by Price and Storn [1], is a simple yet powerful evolutionary algorithm (EA) for global optimization problems. Nowadays DE has become one of the most frequently used EAs for solving global optimization problems [2], mainly because it has good convergence property and is principally easy to understand. DE has been modified and extended with several new versions [3], [4]. Its effectiveness and efficiency have been successfully demonstrated in many real-life application fields.

DE creates a new candidate solution by combining the information of a parent individual and several other individuals of the population. There are many different trial vector generation strategies for DE, each of which seems to be suitable for some particular tasks or for solving a certain type of problems [5], [6]. There are three control parameters in DE: the amplification factor of the difference vector—F, the crossover control parameter—CR, and the population size—NP. All these three parameters significantly affect the performance of DE algorithms [7]–[9]. The optimal choice of the three control parameters in DE often depends on the problems to be solved. For a specific problem, one may need to spend a huge amount of time to try and fine-tune the corresponding parameters. To address this issue, several adaptive and self-adaptive DE algorithms regarding F and CR were developed to solve general problems efficiently [3], [4], [10].

Liu and Lampinen proposed a fuzzy adaptive differential evolution (FADE) [11], using fuzzy logic controllers to adapt the control parameters F and CR for the mutation and crossover operations. Qin and Suganthan proposed a self-adaptive DE (SaDE), where the choice of learning strategy and the two control parameters F and CR does not require predefining [12]. In SaDE suitable learning strategy and parameter settings are gradually self-adapted according to population learning experience. Teo proposed a DE algorithm with a self-adaptive population size NP (abbreviated as DESAP [13]–[15]), based on the self-adaptive Pareto DE proposed by Abbas [16]. DESAP makes F, CR and NP evolvable with individuals to adapt mutation, crossover parameters, and population size by normal random numbers. Brest et al. [17] proposed a new adaptive DE, called jDE, using a self-adapting mechanism for the control parameters F and CR associated with each individual. jDE was further extended by adapting two mutation strategies [18] and the new version was named jDE-2. Another new adaptive DE, called JADE, was proposed by Zhang et al. [19], in which the parameters adaptation was implemented by evolving mutation factors and crossover probabilities based on their historical records of success.

In JADE algorithm [19], the crossover probability CR_i of each individual is independently generated according to a normal distribution of a fixed standard deviation, whose value is equal to 0.1. This small value of fixed standard deviation results in that the generated normal distributed numbers can not distribute in the whole range [0,1], which was discussed in Sect. II-B. Therefore, the generated CR may not suitable for solving a problem. In this paper, we studied the adaptations of F and CR of the JADE algorithm, and proposed an improvement for the adaptation of CR.

This paper is structured as follows. Section II contains the main contribution of this paper including the introduction of the adaptation of CR in the JADE algorithm and our improvement for it. Section III presents and discusses the experimental results on the tested benchmark functions. Section IV concludes the paper with several final remarks.

II. The Improved JADE algorithm

In this section, we introduce the adaptive schemes for F and CR of the JADE algorithm [19], and propose an improvment for the adaptation of CR for JADE.

A. Adaptations of F and CR in JADE

DE employs the mutation operation to produce a mutant vector $\mathbf{v}_{i,G}$ with respect to an individual $\mathbf{x}_{i,G}$, so-called

Ming Yang, Zhihua Cai and Changhe Li are with the School of Computer Science, China University of Geosciences, Wuhan, 430074, China (e-mail: yangming0702@gmail.com, zhcai@cug.edu.cn, changhe.lw@gmail.com).

Jing Guan is with the China Ship Development and Design Center, Wuhan, 430064, China (email: g_jing0414@163.com).

This work was supported by the fund of the National Natural Science Foundation of China (No.61305086, No.61203306 and No.61305079) and the Fundamental Research Funds for the Central Universities (No.CUG120114).

* Author for correspondence.

target vector, at each generation G. For each D-dimensional target vector $\mathbf{x}_{i,G}$, its associated mutant vector $\mathbf{v}_{i,G} = \{v_{i,1,G}, v_{i,2,G}, \ldots, v_{i,D,G}\}$ can be generated via a certain mutation strategy.

$x_{1,1,G}$	$x_{1,2,G}$	...	$x_{1,D,G}$	$F_{1,G}$	$CR_{1,G}$
$x_{2,1,G}$	$x_{2,2,G}$	...	$x_{2,D,G}$	$F_{2,G}$	$CR_{2,G}$
...	...	...	...	...	...
$x_{NP,1,G}$	$x_{NP,2,G}$	...	$x_{NP,D,G}$	$F_{NP,G}$	$CR_{NP,G}$

Fig. 1. Individual encoding of the JADE algorithm, where NP is the population size.

In the JADE algorithm, each individual in the population is extended with parameter values. Fig. 1 shows the individual encoding of the JADE algorithm. The control parameters F and CR are adjusted based on their historical records of success. Both of them are adjusted at the individual level. Potential good values of these encoded control parameters lead to good individuals which, in turn, are more likely to survive and produce offspring and, hence, these good parameter values are propagated.

At each generation, the crossover probability CR_i of each individual $\mathbf{x}_i$ is independently generated according to a normal distribution of mean μ_{CR} and standard deviation 0.1

$$CR_i = \text{randn}_i(\mu_{CR}, 0.1) \tag{1}$$

and then truncated to [0,1]. The mean μ_{CR} is initialized to be 0.5 and then updated at the end of each generation as

$$\mu_{CR} = (1-c) \cdot \mu_{CR} + c \cdot \text{mean}_A(S_{CR}) \tag{2}$$

where S_{CR} is the set of all successful crossover probabilities CR_i at a generation, c is a positive constant between 0 and 1 and mean_A is the usual arithmetic mean.

Similarly, at each generation, the mutation factor F_i of each individual $\mathbf{x}_i$ is independently generated according to a Cauchy distribution with location parameter μ_F and scale parameter 0.1

$$F_i = \text{randc}_i(\mu_F, 0.1) \tag{3}$$

and then truncated to be 1 if $F_i \geq 1$ or regenerated if $F_i \leq 0$. The location parameter μ_F of the Cauchy distribution is initialized to be 0.5 and then updated at the end of each generation as

$$\mu_F = (1-c) \cdot \mu_F + c \cdot \text{mean}_L(S_F) \tag{4}$$

where S_F is the set of all successful mutation factors in a generation, mean_L is the Lehmer mean

$$\text{mean}_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \tag{5}$$

B. Improved Adaptation of CR in JADE

In the JADE algorithm [19], the crossover probability CR_i of each individual is independently generated according to a normal distribution with a fixed standard deviation, whose value is equal to 0.1.

Fig. 2 illustrates the probability density function for the normal distribution of mean μ and standard deviation σ. In the figure, it can seen that 68% of values drawn from a normal distribution are within one standard deviation σ away from the mean μ. Similarly, about 95% of the values lie within two standard deviations from the mean, and about 99.7%, nearly all, of the values lie within three standard deviations from the mean. This fact is known as the 68-95-99.7 rule, or the 3-$sigma$ rule.

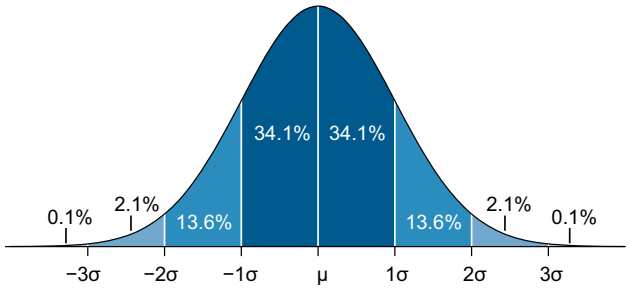


Fig. 2. The graph of probability density function for the normal distribution of mean μ and standard deviation σ.

Nearly all normal distributed numbers are within $[\mu - 3 \cdot \sigma, \mu + 3 \cdot \sigma]$. Therefore, for the JADE algorithm [19], nearly all crossover probabilities CR_i are generated within the range $[\mu_{CR} - 0.3, \mu_{CR} + 0.3]$ (see Eq. (1)). For example, at the beginning of evolution, μ_{CR}=0.5. The crossover probabilities CR_i are generated within the range $[0.2, 0.8]$, which can not cover the whole range $[0, 1]$. If the best crossover probability lies in $[0, 0.2]$ or in $[0.8, 1]$, this adaptation of CR_i can not generate suitable values. For adapting the values of F_i, although the scale parameter of Cauchy distribution is the fixed value 0.1, the Cauchy distributed numbers can cross the range $[0, 1]$.

To improve the adaptation of CR_i, we propose a method in which the standard deviation σ_{CR} is adaptive:

$$\sigma_{CR} = \max(\mu_{CR}, 1 - \mu_{CR}). \tag{6}$$

The value of σ_{CR} is within $[0.5, 1)$. The normal distributed numbers can cover the range $[0, 1]$.

At each generation, the crossover probability CR_i of each individual is generated as follows

$$CR_i = \text{randn}_i(\mu_{CR}, \sigma_{CR}) \tag{7}$$

and then truncated to $[0, 1]$.

C. The Improved JADE Algorithm

The JADE algorithm, called JADE2, applies the improved adaptation of CR (see Section II-B) at each iteration (see the CR_i in the step 8 in Algorithm 1 which is the only difference between JADE2 and JADE). Algorithm 1 presents the pseudo

Algorithm 1 JADE2 Algorithm

1: Set μ_{CR}=0.5, μ_F=0.5 and A=$\emptyset$;
2: Generate uniform random individuals for the initial population P_0;
3: Evaluate the fitness for each individual in P_0;
4: $G = 1$;
5: **while** the stop criterion is not satisfied **do**
6: S_F=$\emptyset$, S_{CR}=$\emptyset$;
7: **for** each individual $\mathbf{x}_{i,G} \in P_G$ **do**
8: Generate $CR_i = \text{randn}_i(\mu_{CR}, \sigma_{CR})$, $F_i = \text{randc}_i(\mu_F, 0.1)$;
9: Randomly choose $\mathbf{x}^p_{best,G}$ as one of the $100p\%$ best vectors;
10: Randomly choose $\mathbf{x}_{r_1,G} \neq \mathbf{x}_{i,G}$ from the current population P_G;
11: Randomly choose $\tilde{\mathbf{x}}_{r_2,G} \neq \mathbf{x}_{r_1,G} \neq \mathbf{x}_{i,G}$ from $P_G \bigcup A$;
12: $\mathbf{v}_{i,G} = \mathbf{x}_{i,G} + F_i \cdot (\mathbf{x}^p_{best,G} - \mathbf{x}_{i,G}) + F_i \cdot (\mathbf{x}_{r_1,G} - \tilde{\mathbf{x}}_{r_2,G})$;
13: j_{rand}=rndint(1,D);
14: **for** j=1 to D **do**
15: **if** rnd(0,1)< CR_i or $j == j_{rand}$ **then**
16: $u_{i,j,G} = v_{i,j,G}$;
17: **else**
18: $u_{i,j,G} = x_{i,j,G}$;
19: **end if**
20: **end for**
21: **if** $u_{i,j,G} \notin [x_{low,j}, x_{up,j}]$ **then**
22: Use Eq. (8) to map $u_{i,j,G}$ to be in the search range $[x_{low,j}, x_{up,j}]$;
23: **end if**
24: Evaluate the offspring $\mathbf{u}_{i,G}$;
25: **if** $\mathbf{u}_{i,G}$ is not worse than $\mathbf{x}_{i,G}$ **then**
26: $\mathbf{x}_{i,G+1} = \mathbf{u}_{i,G}$; $\mathbf{x}_{i,G} \rightarrow A$; $CR_i \rightarrow S_{CR}$; $F_i \rightarrow S_F$;
27: **else**
28: $\mathbf{x}_{i,G+1} = \mathbf{x}_{i,G}$;
29: **end if**
30: **end for**
31: Randomly remove solutions from A so that $|A| \leq NP$;
32: **if** u_F and u_{CR} are not empty **then**
33: $\mu_{CR} = (1-c) \cdot \mu_{CR} + c \cdot mean_A(S_{CR})$;
34: $\mu_F = (1-c) \cdot \mu_F + c \cdot mean_L(S_F)$;
35: **end if**
36: $G = G + 1$;
37: **end while**

code of JADE2. In JADE algorithm, at a generation, if no non-worse trial vector is generated, the set S_F is empty. In this case, the divisor in Eq. (5) is equl to 0. This is an error. In the experiments, for all the JADE algorithms including the improved JADE algorithm, the error is corrected by judging whether μ_F and μ_{CR} are empty (see the step 32 in Algorithm 1). μ_{CR} and μ_F are only updated in the case that μ_F and μ_{CR} are not empty.

When $u_{i,j}$ exceeds the search range after the mutation, we map $u_{i,j}$ to be legal as follows:

$$u_{i,j} = \begin{cases} (F_{max} \cdot x_{up,j} - x_{low,j} + u_{i,j})/F_{max} & \text{if } u_{i,j} < x_{low,j} \\ (F_{max} \cdot x_{low,j} - x_{up,j} + u_{i,j})/F_{max} & \text{if } u_{i,j} > x_{up,j} \end{cases} \quad (8)$$

where F_{max} is the max value of $F_{i,j,G}$; $x_{low,j}$ and $x_{up,j}$ are predefined lower and upper bounds, respectively. In this paper, this method is used for all algorithms to handle such situation. For the JADE, JADE2 and DE/rand/1/bin algorithms, F_{max}=1.0.

III. Experimental Study

A. Comparison of Dim-jDE with Other Algorithms

1) Benchmark Functions: A set of 25 scalable benchmark functions for the competition on IEEE CEC05 [20], are used with dimensionality of D=30 and D=50. Table I presents the details of these functions, including shifted functions, rotated functions, rotated shifted functions, and the complex hybrid composition functions proposed in [21]. A more detailed description and parameter settings of these functions can be found in [20]. In these functions, f_1–f_5 are unimodal functions, f_6–f_{12} are basic multimodal functions, f_{13}–f_{14} are expanded multimodal functions, and f_{15}–f_{25} are hybrid composition functions. Note that, functions f_7 and f_{25} have no boundary constraint. Therefore, for these two functions, we do not re-map individuals that are beyond of the initial bounds by Eq. (8) for all algorithms in this paper.

2) Peer Algorithms: To investigate the effect of the improved adaptation of CR, JADE2 is compared with DE/rand/1/bin [1] and JADE [19]. We use the JADE algorithm with an archive in this paper since it showed promising results compared with JADE without an archive in [19]. The maximal number of fitness evaluations $MaxFEs = 10000 \times D$ is used as the stop criteria for all the algorithms on each function. The parameter settings of all the algorithms are as follows:

a) DE/rand/1/bin: F=0.5 and CR=0.9 as used or recommended in [1], [6], [22]. NP=100 for D=30 and NP=200 for D=50, as used in [12], [17], [19].
b) JADE: c=0.1, p=0.2 as recommended in [19]. NP=100 for D=30 and NP=200 for D=50, as used in [17], [19].
c) JADE2: The parameters use the same settings as JADE.

3) Performance Comparison: Table II summarizes the average error results of 30 independent runs for each algorithm on each function with D=30 and D=50. For each function, the best value of the results obtained by all the algorithms is shown in bold font. b/n/w summarizes the statistical results: b, n, and w denote the number of functions for which JADE2 performs significantly better, not significantly different and significantly worse than its peer, respectively. Note that, for the results in Table II, we only show three significant digits. Therefore, although some results shown in the two tables are same, the values of these results are different.

For the 30-dimensional problems in Table II, it can be seen that there are 8 functions for which JADE2 archives the best

TABLE I

CEC05 BENCHMARK FUNCTIONS. D DENOTES THE DIMENSIONALITY OF THE TEST PROBLEM, S DENOTES THE RANGES OF THE VARIABLES, AND f_{min} IS THE FUNCTION VALUE OF GLOBAL OPTIMUM.

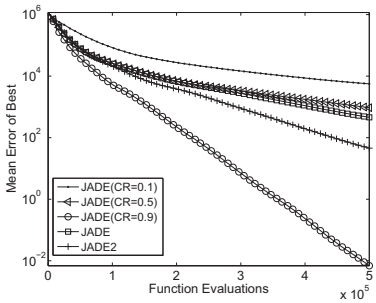
F	*Name*	S	f_{min}
f_1	Shifted Sphere Function(F_1)	$[-100, 100]^D$	-450
f_2	Shifted Schwefel's Problem 1.2(F_2)	$[-100, 100]^D$	-450
f_3	Shifted Rotated High Conditioned Elliptic Function(F_3)	$[-100, 100]^D$	-450
f_4	Shifted Schwefel's Problem 1.2 with Noise in Fitness(F_4)	$[-100, 100]^D$	-450
f_5	Schwefel's Problem 2.6 with Global Optimum on Bounds(F_5)	$[-100, 100]^D$	-310
f_6	Shifted Rosenbrock's Function(F_6)	$[-100, 100]^D$	390
f_7	Shifted Rotated Griewank's Function without Bounds(F_7)	$[0, 600]^D$	-180
f_8	Shifted Rotated Ackley's Function with Global Optimum on Bounds(F_8)	$[-32, 32]^D$	-140
f_9	Shifted Rastrigin's Function(F_9)	$[-5, 5]^D$	-330
f_{10}	Shifted Rotated Rastrigin's Function(F_{10})	$[-5, 5]^D$	-330
f_{11}	Shifted Rotated Weierstrass Function(F_{11})	$[-0.5, 0.5]^D$	90
f_{12}	Schwefel's Problem 2.13(F_{12})	$[-\pi, \pi]^D$	-460
f_{13}	Expanded Extended Griewank's plus Rosenbrock's Function(F_{13})	$[-5, 5]^D$	-130
f_{14}	Shifted Rotated Expanded Scaffer's F6(F_{14})	$[-100, 100]^D$	-300
f_{15}	Hybrid Composition Function(F_{15})	$[-5, 5]^D$	120
f_{16}	Rotated Hybrid Composition Function(F_{16})	$[-5, 5]^D$	120
f_{17}	Rotated Hybrid Composition Function with Noise in Fitness(F_{17})	$[-5, 5]^D$	120
f_{18}	Rotated Hybrid Composition Function(F_{18})	$[-5, 5]^D$	10
f_{19}	Rotated Hybrid Composition Function with a Narrow Basin for the Global Optimum(F_{19})	$[-5, 5]^D$	10
f_{20}	Rotated Hybrid Composition Function with the Global Optimum on the Bounds(F_{20})	$[-5, 5]^D$	10
f_{21}	Rotated Hybrid Composition Function(F_{21})	$[-5, 5]^D$	360
f_{22}	Rotated Hybrid Composition Function with High Condition Number Matrix(F_{22})	$[-5, 5]^D$	360
f_{23}	Non-Continuous Rotated Hybrid Composition Function(F_{23})	$[-5, 5]^D$	360
f_{24}	Rotated Hybrid Composition Function(F_{24})	$[-5, 5]^D$	260
f_{25}	Rotated Hybrid Composition Function without Bounds(F_{25})	$[2, 5]^D$	260

TABLE II

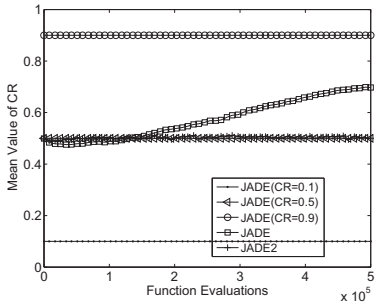
AVERAGE ERROR VALUES ARCHIVED FOR 25 30-DIMENSIONAL AND 50-DIMENSIONAL CEC05 BENCHMARK FUNCTIONS OVER 30 INDEPENDENT RUNS: THE MEAN BEST RESULT AND CORRESPONDING STANDARD DEVIATION VALUE.

F	D=30			D=50		
	JADE2	JADE	DE/rand/1/bin	JADE2	JADE	DE/rand/1/bin
f_1	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	1.33e-014±2.45e-014†	8.59e-008±4.46e-008†
f_2	4.23e-013±5.44e-013	**0.00e+000±0.00e+000**‡	1.90e-004±1.96e-004†	2.88e-003±1.64e-003	**3.43e-009±4.27e-009**‡	1.03e+004±2.48e+003†
f_3	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	3.60e-014±2.79e-014†	3.78e-005±1.97e-005†
f_4	3.30e-005±7.22e-005	**3.90e-010±1.08e-009**‡	4.10e-002±3.14e-002†	5.16e+001±3.01e+001	**9.40e-001±2.07e+000**‡	3.40e+004±6.90e+003†
f_5	2.32e-004±2.10e-004	**2.85e-006±5.78e-006**‡	3.84e-001±4.28e-001†	**4.57e+001±2.54e+001**	4.59e+002±8.68e+002†	6.13e+003±1.14e+003†
f_6	**1.33e-001±7.28e-001**	3.13e+000±8.12e+000†	3.25e+000±1.36e+000†	1.91e+001±2.24e+000	**6.97e+000±2.06e+000**‡	4.10e+001±5.43e-001†
f_7	3.61e-003±5.99e-003	5.50e-003±7.48e-003	**0.00e+000±0.00e+000**‡	1.08e+000±2.71e-001	**1.40e-003±4.65e-003**‡	2.42e+000±8.65e-001†
f_8	2.09e+001±4.96e-002	**2.09e+001±2.20e-001**	2.09e+001±5.85e-002	**2.11e+001±3.87e-002**	2.11e+001±4.25e-002	2.11e+001±4.12e-002
f_9	**0.00e+000±0.00e+000**	**0.00e+000±0.00e+000**	1.27e+002±2.04e+001†	7.43e+001±4.92e+000	**2.71e-005±1.67e-005**‡	3.76e+002±1.62e+001†
f_{10}	7.30e+001±9.08e+000	**5.39e+001±7.26e+000**‡	1.83e+002±1.00e+001†	2.08e+002±1.31e+001	**1.55e+002±1.30e+001**‡	3.92e+002±1.71e+001†
f_{11}	2.85e+001±1.46e+000	**2.66e+001±1.47e+000**‡	3.91e+001±1.30e+000†	5.84e+001±2.13e+000	**5.38e+001±2.20e+000**‡	7.28e+001±2.03e+000†
f_{12}	3.66e+004±9.61e+003	1.75e+004±5.49e+003‡	**4.96e+003±4.75e+003**‡	3.78e+005±4.41e+004	**1.76e+005±2.36e+004**‡	1.95e+006±2.67e+005†
f_{13}	2.34e+000±3.38e-001	1.69e+000±1.93e-001‡	**1.26e+000±2.94e-001**‡	4.84e+000±1.04e+000	**3.50e+000±5.15e-001**‡	4.90e+000±1.29e+000
f_{14}	1.28e+001±2.38e-001	**1.27e+001±1.77e-001**	1.35e+001±1.36e-001†	2.26e+001±1.71e-001	**2.25e+001±1.99e-001**	2.32e+001±1.39e-001†
f_{15}	**1.00e+002±4.16e-013**	1.00e+002±3.47e-013†	1.20e+002±1.09e+002†	**4.00e+002±2.89e-013**	4.32e+002±1.23e+002†	4.10e+002±3.05e+001†
f_{16}	**1.00e+002±5.52e-013**	1.00e+002±2.89e-014†	1.00e+002±4.38e-013†	5.03e+002±3.10e+002	9.11e+002±2.42e+001†	**1.51e+002±1.14e+002**‡
f_{17}	1.22e+002±1.10e+002	3.74e+002±3.06e+002†	**1.12e+002±6.30e+000**‡	**8.69e+002±7.08e+000**	9.48e+002±1.96e+001†	1.01e+003±1.78e+001†
f_{18}	**3.00e+002±8.58e-013**	3.77e+002±2.35e+002†	3.00e+002±2.46e-012†	5.20e+002±3.44e+002	8.94e+002±3.37e+002†	**3.01e+002±5.90e-001**
f_{19}	**3.00e+002±8.25e-013**	3.78e+002±2.38e+002†	3.00e+002±1.51e-012†	6.70e+002±3.55e+002	1.04e+003±1.52e+002†	**3.01e+002±9.89e-001**‡
f_{20}	7.20e+002±3.74e+002	1.02e+003±1.98e+002†	**3.00e+002±2.79e-011**‡	6.36e+002±4.19e+002	1.13e+003±9.87e+000†	**3.01e+002±8.54e-001**
f_{21}	1.15e+003±1.05e+001	1.16e+003±2.19e+001†	**1.14e+003±6.35e+000**‡	**1.12e+003±1.12e+001**	1.21e+003±2.48e+001†	1.13e+003±1.52e+001
f_{22}	1.17e+003±1.20e+001	1.17e+003±1.10e+001	**9.18e+002±3.49e+002**‡	**1.12e+003±6.74e+000**	1.16e+003±6.13e+000†	1.14e+003±6.03e+000†
f_{23}	1.19e+003±8.05e+000	1.20e+003±1.03e+001†	**1.19e+003±6.21e+000**‡	1.16e+003±8.70e+000	1.26e+003±1.46e+001†	**1.16e+003±7.40e+000**
f_{24}	1.09e+003±4.05e+000	**1.09e+003±5.36e+000**‡	1.12e+003±2.77e+000†	**1.17e+003±4.65e+000**	1.18e+003±5.19e+000†	1.20e+003±3.24e+000†
f_{25}	1.20e+003±1.12e+001	1.20e+003±1.03e+001	**1.03e+003±4.25e+002**	**1.28e+003±7.19e+000**	1.30e+003±1.04e+001†	1.35e+003±6.04e+000†
b/n/w	—	9/8/8	13/4/8	—	14/2/9	17/6/2

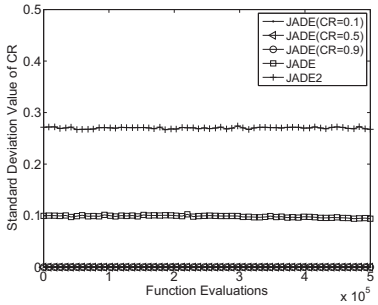
† JADE2 performs significantly better than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.
‡ JADE2 performs significantly worse than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.



(a) The convergence graph

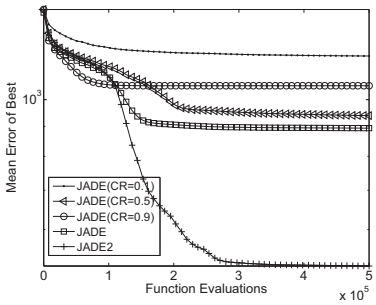


(b) The mean value of CR_i of all individuals

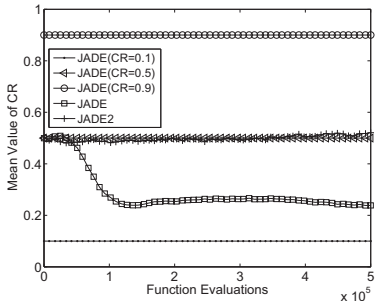


(c) The standard deviation value of CR_i of all individuals

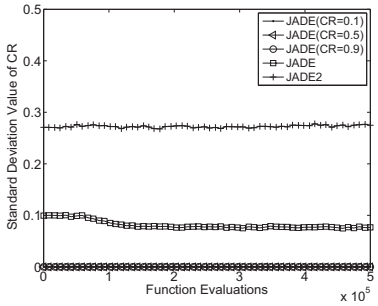
Fig. 3. The results for the test function f_5 when D=50 over 30 independent runs.



(a) The convergence graph



(b) The mean value of CR_i of all individuals



(c) The standard deviation value of CR_i of all individuals

Fig. 4. The results for the test function f_{18} when D=50 over 30 independent runs.

error; while there are 11 functions for which DE/rand/1/bin archives the smallest error. From the statistical significant results, we can see that JADE2 performs much better than DE/rand/1/bin. JADE2 performs slightly better than JADE. For the 50-dimensions problems, there are 10 functions for which JADE2 got the best results. And JADE2 performs significantly better than DE/rand/1/bin and much better than JADE. Through comparing the results in Table II, JADE2 is much more superior than the other algorithms when the dimension of problems is high. From the above results, we can conclude that the improved adaptation of CR can improve the performance of the JADE algorithm, especially for high-dimensional problems.

Fig. 3 and Fig. 4 show the results of f_5 and f_{18} over 30 independent runs, respectively. In these figures, JADE(CR=0.1), JADE(CR=0.5) and JADE(CR=0.9) are the JADE algorithms in which the probability of each individual are not adaptive, and in these three algorithms all individuals have the fixed crossover probability CR=0.1, CR=0.5 and CR=0.9, respectively.

For f_5, JADE(CR=0.9) performs better than JADE(CR=0.5), and JADE(CR=0.5) performs better than JADE(CR=0.1) (see Fig. 3a). This indicates that the larger value of crossover probability, the JADE algorithm performs better on this problem. After 200000 function evaluations, although JADE2 has a slightly smaller mean value of CR_i of all individuals than JADE (see Fig. 3b), JADE2 has a larger standard deviation value of CR_i (see Fig. 3c). JADE2 can generate large value of CR_i with a large probability. Therefore, JADE2 performs better than JADE (see Fig. 3a).

For f_{18}, JADE(CR=0.9) and JADE(CR=0.1) perform worse than JADE(CR=0.5) (see Fig. 4a). This indicates that large or small values of CR_i are not the best for solving f_8. At a generation, for JADE2, the mean value of CR_i of all individuals is equal to about 0.5, the middle value of the range [0,1] (see Fig. 4b). JADE2 has a larger standard deviation value of CR_i than JADE (see Fig. 4c). JADE2 can generate CR_i whose value covers the range [0,1], which is why JADE2 performs better than JADE (see Fig. 4a).

The above results show that JADE2 can diversify the values of CR_i of the population. Therefore, for JADE2 there are more suitable CR_i to generate better trial vectors.

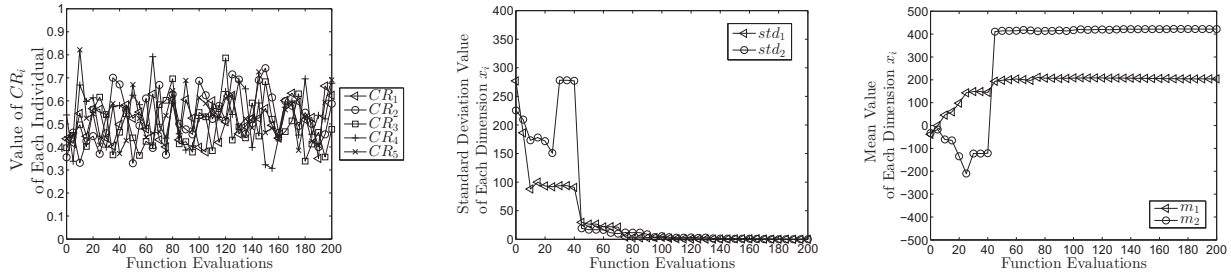
B. The Advantage of Improved Adaptation of CR

In this section, we will study the improved adaptation of CR and show the advantage of JADE2 compared with JADE. Take a 2-dimensional *Schwefel* function [23] for example, which is a classical multimodal minimum optimization function. The 2-dimensional *Schwefel* function is as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{D} \left(- x_i \sin\left(\sqrt{|x_i|}\right)\right), x_i \in [-500, 500], \quad (9)$$

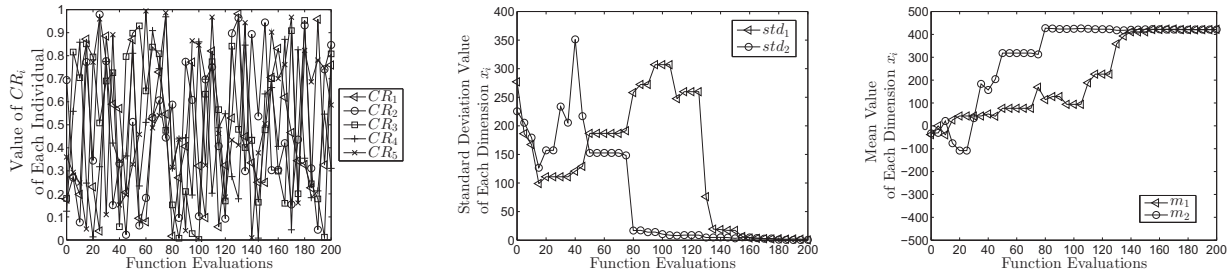
D=2 here. The global optimum of the *Schwefel* function is $\mathbf{x} \approx$\{420.9687,. . .,420.9687\}.

The mean and standard deviation of individuals' variables



(a) (b) (c)

Fig. 5. The results of JADE algorithm on the 2-dimensional *Schwefel* function with NP=5 in a typical run.



(a) (b) (c)

Fig. 6. The results of JADE2 algorithm on the 2-dimensional *Schwefel* function with NP=5 in a typical run.

in the j-th dimension at the G-th generation is:

$$m_{j,G} = \frac{1}{NP}\sum_{i=1}^{NP} x_{i,j,G}, \tag{10}$$

$$std_{j,G} = \sqrt{\frac{1}{NP}\sum_{i=1}^{NP}(x_{i,j,G} - m_{j,G})^2}, \tag{11}$$

where $m_{j,G}$ and $std_{j,G}$ are the mean and standard deviation of the population in the j-th dimension at generation G, respectively.

Fig. 5 and Fig. 6 show a typical run of JADE [19] and JADE2 (see Algorithm 1) algorithms for the 2-dimensional *Schwefel* function with NP=5. Note that, the initial population of JADE and JADE2 algorithms are the same.

For the JADE algorithm, the values of CR of all individuals in population are larger than 0.3 (see Fig. 5a). Because of this slightly large value of CR, the value of std_j on each dimension drops sharply to 0 (see Fig. 5b). The population converges at a local optimum (see Fig. 5c). In Fig. 5c, when the number of function evaluations is equal to about 40, the population converges at the global optimum on the second dimension, but not on the first dimension. Because the population diversity on the first dimension is very poor (see Fig. 5b), the population can not get better any more.

For the JADE2 algorithm, the values of CR of all individuals in population can cover the range [0,1] (see Fig. 6a). The value of CR can be small at some generations. Although the population converges slowly (see Fig. 6b), the population can converge at the global optimum (see Fig. 6c). When the number of function evaluations is equal to about 80, the population converges at the global optimum on the second dimension, but not on the first dimension (see Fig. 6c). Because the population diversity on the first dimension is high at this moment (see Fig. 6b), the population can get better and converges at the global optimum finally.

From this example, it can be seen that the improved adaptation of CR can enhance the diversity of values of CR and it can generate more suitable CR to produce better trial vectors.

IV. Conclusions

The control parameters involved in DE are highly dependent on the problems to be solved. There are many adaptive algorithms proposed for parameters adaptation. As a very known adaptive DE algorithm, JADE adjusts the mutation factors and crossover probabilities based on their historical records of success. The crossover probability CR is generated according to a normal distribution with the fixed standard deviation 0.1. The 3-$sigma$ rule shows that the generated normal distributed numbers can not cover the range [0,1]. Therefore, the generated CR may not suitable for solving a problem.

To improve the adaptation of CR, we propose a method in which the standard deviation is adaptive. The values of CR generated by this normal distribution can cover the range [0,1]. The JADE algorithm, called JADE2, applies the improved adaptation of CR is tested on a set of 25 scalable benchmark functions. The results show that the improved adaptation of CR can significantly improve the performance of the JADE algorithm, especially for high-dimensional function optimization.

REFERENCES

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

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

[3] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61–106, Feb. 2010.

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

[5] V. Feoktistov and S. Janaqi, "Generalization of the strategies in differential evolution," in *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, April 2004, pp. 165–170.

[6] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, ser. GECCO2006. New York, NY, USA: ACM, 2006, pp. 485–492.

[7] R. Gämperle, S. D. Müller, and P. Koumoutsakos, "A parameter study for differential evolution," in *WSEAS Int. Conf. on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*. Press, 2002, pp. 293–298.

[8] J. Liu and J. Lampinen, "On setting the control parameter of the differential evolution method," in *Proc. 8th Int. Conf. Soft Computing*, 2002, pp. 11–18.

[9] R. Mallipeddi and P. Suganthan, "Empirical study on the effect of population size on differential evolution algorithm," in *IEEE Congress on Evolutionary Computation'08*, june 2008, pp. 3663–3670.

[10] J. Tvrdík, "Adaptation in differential evolution: a numerical comparison," *Applied Soft Computing*, vol. 9, pp. 1149–1155, 2009.

[11] 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. 448–462, 2005.

[12] A. Qin and P. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2005, pp. 1785–1791.

[13] J. Teo, "Differential evolution with self-adaptive populations," in *Knowledge-Based Intelligent Information and Engineering Systems - 9th International Conference, KES 2005*, Melbourne, Australia, 2005, pp. 1284–1290.

[14] ——, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Comput.: Fusion Found., Methodologies Applicat.*, vol. 10, no. 8, pp. 673–686, 2006.

[15] N. Teng, J. Teo, and M. Hijazi, "Self-adaptive population sizing for a tune-free differential evolution," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 13, pp. 709–724, 2009.

[16] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm," in *IEEE Congress on Evolutionary Computation*, May 2002, pp. 831–836.

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

[18] J. Brest, B. Boskovic, V. Z. S. Greiner, and M. Maucec, "Performance comparison of self-adaptive and adaptive differential evolution algorithms," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 11, no. 7, pp. 617–629, 2007.

[19] J. Zhang and C. Arthur, "JADE: Adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[20] 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 Technological University, Singapore, Tech. Rep., 2005.

[21] J. Liang, P. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, june 2005, pp. 68–75.

[22] J. Vesterstrom, R. Thomsen, B. R. Center, A. Univ., and Denmark, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. IEEE Congr. Evol. Comput.*, ser. CEC2004, 2004, pp. 1980–1987.

[23] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.