Improving the JADE Algorithm by Clustering Successful Parameters

Zhijian Li

School of Computer Science,
Central China Normal University, Wuhan 430079, China
E-mail: lzj1769@163.com

Jinglei Guo*

School of Computer Science,
Central China Normal University, Wuhan 430079, China
E-mail: guojinglei@mail.ccnu.edu.cn
*Corresponding author

Shengxiang Yang

Centre for Computational Intelligence (CCI)
School of Computer Science and Informatics, De Montfort Univesity, Leicester LE1 9BH, U.K.
E-mail: syang@dmu.ac.uk

Abstract: Differential evolution (DE) is one of the most powerful and popular evolutionary algorithms for real parameter global optimization problems. However, the performance of DE highly depends on the selection of control parameters, e.g., the population size, scaling factor and crossover rate. How to set these parameters is a challenging task because they are problem dependent. In order to tackle this problem, a JADE variant, denoted CJADE, is proposed in this paper. In the proposed algorithm, the successful parameters are clustered with the K-means clustering algorithm to reduce the impact of poor parameters. Simulation results show that CJADE is better than, or at least comparable to, several state-of-the-art DE algorithms.

Keywords: differential evolution algorithm; k-means; successful parameters.

Biographical notes: Zhijian Li received the M.S. degree in Computer Science from the Central China Normal University in 2016. His current research interests focus on evolutionary computation, optimization algorithm and machine learning.

Jinglei Guo received the Ph.D. in Computer Science from Wuhan University in 2011. Currently, she is an Associate Professor at the School of Computer, Central China Normal University. Her research interests include artificial intelligence and evolutionary computation.

Shengxiang Yang received the B.Sc. and M.Sc. degrees in automatic control and the Ph.D. degress in system engineering from Northeastern University, China. He is currently a Professor in Computational Intelligence and Director of the Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, U.K. His current research interests include evolutionary and genetic algorithms, swarm intelligence, computational intelligence in dynamic and uncertain environments, artificial neural networks for scheduling, and relevant real-world applications.

1 Introduction

Differential evolution (DE), proposed in R. Storn and K. Price (1995), is a simple and efficient evolutionary algorithm (EA) for global optimization in the continuous space. However, unlike other EAs, DE perturbs the current population members with the scaled difference of randomly selected distinct population members (Das and Suganthan, 2011). DE has been successfully applied to various fields of real-world optimization problems, such as pattern recognition (Koloseni et al., 2012; Kim et al, 2008), image processing (Su et al., 2012), and engineering design (Ghosh et al., 2012; Guo et al., 2001). A detailed survey on DE can be found in Das and Suganthan (Das and Suganthan, 2011).

There are three main control parameters within the DE algorithm: the mutation scale factor F, the crossover constant Cr, and the population size NP. These parameters have significant impact on the performance of DE. In order to minimize the effects of these parameters, a number of enhanced DE variants have been proposed in the past decade. Most of those algorithms are based on adaptive or self-adaptive mechanisms and consider F and Cr separately. In this paper, we propose a new self-adaptive scheme, where F and Cr are considered as a point in the two-dimensional space and we utilize the cluster analysis to reduce the impact of poor parameters and to update the control parameters automatically.

The rest of this paper is organized as follows. In Section 2, we introduce the classical DE algorithm and related work. JADE is described in Section 3. The proposed approach is introduced in detail in Section 4. Experimental studies are reported in Section 5. Finally, Section 6 concludes this paper.

2 Background

2.1 Classical differential evolution

DE begins with a randomly initiated population of NP individuals, each of which is a D-dimensional real parameter vector to represent a candidate solution, where NP is the population size and D is the number of dimensions of the problem. The jth dimension of the ith individual can be initialized according to:

$$x_{j,i} = x_{j,min} + rand_{i,j}(0,1) \cdot (x_{j,max} - x_{j,min}) \tag{1}$$

where $x_{j,min}$ and $x_{j,max}$ are the predefined lower and upper bounds of the jth dimension, $rand_{i,j}(0,1)$ is a uniformly distributed random number in $(0,1)$.

After initialization, DE generates a mutant individual $V_{i,g}$, called *donor* individual, for each population member or *target* individual $X_{i,g}$ in the current generation by mutation. The most frequently used strategy is defined as follows:

$$V_{i,g} = X_{r_1^i,g} + F \cdot (X_{r_2^i,g} - X_{r_3^i,g}) \tag{2}$$

where the indices r_1^i, r_2^i, r_3^i are mutually exclusive integers randomly chosen from the range $[1, NP]$, which are also different from the base individual index i. The scaling factor F is a positive real number for scaling the difference vectors.

Then, DE employs crossover operator to enhance the potential diversity of the population. In the classical version, a binomial crossover operation is employed by DE as follows:

$$u_{i,j,g} = \begin{cases} v_{i,j,g}, & if\ (rand_{i,j}(0,1) \le Cr\ or\ j = j_{rand}) \\ x_{i,j,g}, & otherwise \end{cases} \tag{3}$$

where $j_{rand} \in [1, 2, \dots, D]$ is a randomly chosen index, which ensures that $u_{i,j,g}$ dose not duplicate $x_{i,j,g}$.

Finally, a selection operator is used by DE to determine whether the target or trial vector survives to the next generation. The selection operation is described as follows:

$$X_{i,g+1} = \begin{cases} U_{i,g}, & if\ f(U_{i,g}) \le f(X_{i,g}) \\ X_{i,g}, & otherwise \end{cases} \tag{4}$$

where $f(X)$ is the objective function (without loss of generality, we assume minimization problems here).

2.2 Related Work

Zhang and Sanderson (Zhang et al., 2009) proposed an adaptive DE with an optional external archive (JADE), where the scaling factor F and the crossover rate Cr are generated according to a normal distribution and a Cauchy distribution, respectively. Fei et al. (2009) applied two strategies together on the original JADE, to dedicatedly improve the reliability of it (rJADE). They first modified the control parameter adaptation strategy of JADE by adding a weighting strategy. Then, a restart with knowledge transfer strategy was applied by utilizing the knowledge obtained from previous failures to guide the subsequent search. Experimental studies showed that the proposed rJADE achieved significant improvements on a set of widely used benchmark functions. Li et al. (2015) proposed a predictive approach to the reproduction mechanism of new individuals for differential evolution (DE) algorithms utilizing cumulative correlation information already existing in an evolutionary process. DE uses a distributed model (DM) to generate new individuals, which is relatively explorative, whilst evolution strategy (ES) uses a centralized model (CM) to generate offspring, which through adaptation retains a convergence momentum.

Pan, X. and Li, R. (2015) proposed an improved differential evolution (DE) to solve parameter optimisation problems. The new approach is called ICBBDE, which is an enhanced version of bare bones DE (BBDE). The ICBBDE employs an adaptive strategy to dynamically adjust the crossover rate. Moreover, a Cauchy mutation is used to improve the exploration ability. Experiments are conducted on a set of benchmark functions and two real-world parameter optimisation problems. Simulation results demonstrate the efficiency and effectiveness of our approach. Zhang et al. (2015) presented a novel DE variant, Top-k elites-based Oppositional Differential Evolution (TEODE), which is based on a new opposition-based learning strategy using the top-k elites (TEOBL) in the current generation and employs similar schemes of ODE for population initialisation and generation jumping with TEOBL. Experiments are conducted on 17 benchmark functions. The results confirm that TEODE outperforms classical DE, ODE and COODE (opposition-based differential evolution using the current optimum). You et al. (2016) designed a new mutation operator to improve the exploitation ability of DE. Experiments are carried out on 13 classical test functions. Simulation results show that the new mutation scheme can help DE to find better solutions than three other classical DE mutation strategies.

3 Introduction of JADE

JADE, proposed by Zhang and Sanderson Zhang et al. (2009), is a well-known adaptive DE, which uses a novel mutation strategy named DE/Current-to-pbest and an external archive for storing the inferior solutions. In addition, F and Cr are generated according to a normal distribution $N(\mu_{Cr}, 0.1)$ and a Cauchy distribution $C(\mu_F, 0.1)$, respectively.

3.1 DE/Current-to-pbest

In view of the fast convergence but less reliable performance of DE/current-to-best and DE/best/1 strategy, Zhang (Zhang et al., 2009) proposed a new mutation strategy, named DE/current-to-pbest, with an optional archive. In DE/current-to-pbest/1 (with archive), a mutation vector is generated in the following manner:

$$V_{i,g} = X_{i,g} + F_i(X^p_{best,g} - X_{i,g}) + F_i(X_{r_1,g} - X_{r_2,g}) \quad (5)$$

where $X^p_{best,g}$ is randomly chosen as one of the top $100p\%$ individuals in the current population with $p \in (0, 1]$, and F_i is the mutation factor that is associated with X_i and is re-generated at each generation by the adaptation process introduced later in Eq. (6).

3.2 Parameter Adaptation

At each generation g, the mutation factor F_i and crossover probability Cr_i of each individual X_i is independently generated:

$$F_i = randc_i(\mu F, 0.1) \quad (6)$$

$$Cr_i = randn_i(\mu Cr, 0.1) \quad (7)$$

where $randc_i(\mu F, 0.1)$ and $randn_i(\mu Cr, 0.1)$ are the random number generated according to Cauchy distribution of $(\mu F, 0.1)$ and normal distribution of $(\mu Cr, 0.1)$. F_i is truncated as 1 when $F_i > 1$ and regenerated when $F_i \leq 0$. If $Cr_i > 1$ or $Cr_i < 0$, Cr_i is set as 1 or 0. Denote S_F and S_{Cr} as the set of all successful mutation factors and crossover probabilities at generation g. μF and μCr are initialized to be 0.5 and then updated at the end of each generation as follows:

$$\mu F = (1 - c) \cdot \mu F + c \cdot mean_L(S_F) \quad (8)$$

$$\mu Cr = (1 - c) \cdot \mu Cr + c \cdot mean_A(S_{Cr}) \quad (9)$$

where c is a positive constant between 0 and 1, $mean_A(\cdot)$ is the usual arithmetic mean, and $mean_L(\cdot)$ is the Lehmer mean, defined as follows:

$$mean_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F}. \quad (10)$$

Algorithm 1 CJADE algorithm

```
1: Initialization
2: $g := 0; \{\mu_F^1, \mu_F^2, \cdots, \mu_F^K\} := \{0.5\};$
3: $\{\mu_{Cr}^1, \mu_{Cr}^2, \cdots, \mu_{Cr}^K\} := \{0.5\}; p := 0.05; A := \emptyset;$
4: Initialize population $P_g = (x_{1,g}, \cdots, x_{NP,g})$ randomly;
5: while The termination criteria are not met do
6:   $S_F := \emptyset, S_{Cr} := \emptyset;$
7:   for $i := 1$ to $NP$ do
8:     Randomly choose $\mu_F, \mu_{Cr}$ from $\{\mu_F^1, \mu_F^2, \cdots, \mu_F^K\}$ and $\{\mu_{Cr}^1, \mu_{Cr}^2, \cdots, \mu_{Cr}^K\}$
9:     $F_i := randc_i(\mu_F, 0.1);$
10:    $Cr_i := randn_i(\mu_{Cr}, 0.1);$
11:    Generate $V_{i,g}$ according to Eq. (5);
12:    Generate $U_{i,g}$ according to Eq. (3);
13:    if $f(U_{i,g}) \leq f(X_{i,g})$ then
14:      $X_{i,g+1} := U_{i,g}, X_{i,g} \to A;$
15:      $F_i \to S_F, Cr_i \to S_{Cr};$
16:    else
17:      $x_{i,g+1} := x_{i,g};$
18:    end if
19:  end for
20:  Remove solutions randomly from A so that $|A| \leq NP$;
21:  if $S_F \neq \emptyset$ and $S_{Cr} \neq \emptyset$ then
22:    Select randomly K initial cluster centers from points set $I := \{(F_i, Cr_i) | F_i \in S_F, Cr_i \in S_{Cr}\};$
23:    for $i := 1$ to $Iterations$ do
24:      for $j := 1$ to $|I|$ do
25:        For each point $I_j$, find its closest cluster center and assign point $I_j$ to cluster;
26:      end for
27:      Update the cluster centers to be the average of points contained within them;
28:    end for
29:    K clusters $\{(C_F^1, C_{Cr}^1), (C_F^2, C_{Cr}^2), \cdots, (C_F^K, C_{Cr}^K)\}$
30:    for $i := 1$ to $K$ do
31:      $\mu_F^i = (1 - c) \cdot \mu_F^i + c \cdot mean_L(C_F^i)$
32:      $\mu_{Cr}^i = (1 - c) \cdot \mu_{Cr}^i + c \cdot mean_A(C_{Cr}^i)$
33:    end for
34:  end if
35:  $g + +;$
36: end while
```

4 THE PROPOSED APPROACH

As mentioned above, the performance of DE highly depends on the control parameters F and Cr. Therefore, it appears increasing to the DE community to study adaptive or self-adaptive schemes to select suitable parameter values during the evolution process. A good volume of research work has been undertaken so far to improve the ultimate performance of DE by tuning its control parameters Das and Suganthan (2011). To the best of our knowledge, most of these work consider F and Cr separately. The trial vector, however, is a result of the combination of F and Cr. Hence, in this paper, we consider each F and Cr as a point in the two dimensional vector space and update the control parameters automatically by clustering the successful parameters with K-means method.

The parameters that generate individuals survived to the next generation are called *successful parameters*, which tend to generate individuals that are more likely to survive and thus should be propagated to the following generations. As with JADE, S_F and S_{Cr} record the successful F and Cr in the current generation and a set of points is constructed as follows:

$$I = \{(F_i, Cr_i) | F_i \in S_F, Cr_i \in S_{Cr}\}. \tag{11}$$

We utilize the K-means cluster algorithm to divide the set I into K clusters $\{(C_F^1, C_{Cr}^1), (C_F^2, C_{Cr}^2), \cdots, (C_F^K, C_{Cr}^K)\}$. For the ith cluster, μ_F^i and μ_{Cr}^i are updated as follows:

$$\mu_F^i = (1-c) \cdot \mu_F^i + c \cdot mean_L(C_F^i) \tag{12}$$

$$\mu_{Cr}^i = (1-c) \cdot \mu_{Cr}^i + c \cdot mean_A(C_{Cr}^i) \tag{13}$$

where μ_F^i and μ_{Cr}^i are initialized to 0.5 at the beginning according to Zhang et al. (2009), and $mean_L(C_F^i)$ and $mean_A(C_{Cr}^i)$ are calculated according to Eqs. (8) and (9) respectively. The overall CJADE algorithm is shown in Algorithm 1. Here, the number of iterations is set to a fixed value to simplify the K-means algorithm, and we use the Euclidean metric to measure the distance between two points. Obviously, JADE is a special case of CJADE with $K = 1$.

5 EXPERIMENTAL STUDY

5.1 Benchmark Functions

In order to verify the performance of our proposed approach, we employ the benchmark functions used for the 2014 IEEE Congress on Evolutionary Computation (IEEE CEC) Competition on Single Objective Real-parameter Numerical Optimization, which consists of 30 single objective benchmark functions Liang et al. (2013) and is denoted *IEEE CEC 2014 benchmark functions* in this paper. Based on their characteristics, the IEEE CEC 2014 benchmark functions can be divided into the following four classes. Functions $cf_1 - cf_3$ are unimodal, functions $cf_4 - cf_{16}$ are simple multimodal functions, functions $cf_{17} - cf_{22}$ are hybrid, and functions $cf_{23} - cf_{30}$ are composition functions with a huge number of local minima. A thorough description of this suit is provided in Liang et al. (2013).

5.2 Effect of the Number of Iterations and Cluster Centers

CJADE introduces a new parameter that need to be predefined: the number of clusters K in the cluster analysis. We compare the performance of $K = 3$ with the performance of $K = 2$ in Table 1. It can be seen from Table 1 that the number of cluster centers has significant impact on the performance of CJADE on CEC2014 benchmark functions. Specifically, the results of $K = 2$ are better than those of $K = 3$ on 17 functions, while worse than those of $K = 3$ on 4 functions.

5.3 Comparison of CJADE with Other DE Algorithms

In this section, we compare CJADE with four other state-of-the-art DE algorithms. For fair comparison, we set the parameters of CJADE to be fixed: $p = 0.05$, $Iterations = 10$, $K = 2$ and $NP = 100$. We follow the parameter settings in the original paper of JADE (Zhang et al., 2009), EPSDE (Mallipeddi et al., 2011), CoDE (Wang et al., 2011), and MDE_pBX (Islam et al., 2012). For all algorithms, the maximum number of function evaluations was set to 10000D. In addition, we judge the results by performing the Wilcoxon's rank-sum test at the 0.05 significance level.

Table 2 summarizes the average error results of 51 independent runs for each algorithm on each 30-dimensional function. Error value smaller than 10^{-8} will be taken as zero (Liang et al., 2013). 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 CJADE performs significantly better, not significantly different and significantly worse than its peer, respectively.

1. *Unimodal Functions* $cf_1 - cf_3$: Clearly, CJADE is the best among the five methods on these three unimodal functions. It outperforms JADE, EPSDE, CoDE and MDE_pBX on one, one, one and three test functions, respectively.

2. *Multimodal Functions* $cf_4 - cf_{16}$: On these thirteen test functions, CJADE is significantly better than JADE, EPSDE, CoDE, and MDE_pBX on four, twelve, twelve and ten test functions, respectively. MDE_pBX outperforms CJADE on one test function, and JADE, EPSDE, and CoDE cannot be significantly better than CJADE on any test function. Hence, CJADE is the winner on these thirteen test functions.

3. *Hybrid Functions* $cf_{17} - cf_{22}$: For these six functions, CJADE is slightly better than JADE and CoDE. CJADE outperforms EPSDE on one function (cf_{17}) and is better than MDE_pBX on five functions ($cf_{17} - cf_{19}, cf_{21}, cf_{22}$).

4. *Composition Functions* $cf_{23} - cf_{30}$: These composition functions are much harder than others because they can have different properties for different variables subcomponents Liang et al. (2013). The performance of JADE and EPSDE is slightly better than that of CJADE. In contrast,

Table 1 CJADE Results on CEC2014 benchmarks using various K

	CJADE(K=2)	CJADE(K=3)			CJADE(K=2)	CJADE(K=3)	
Prob.	Mean St.D.	Mean St.D.		Prob.	Mean St.D.	Mean St.D.	
cf_1	**2.15e+03 2.40e+03**	4.39e+06 1.96e+06	−	cf_{16}	**9.31e+00 3.18e-01**	9.78e+00 2.72e-01	=
cf_2	**0.00e+00 0.00e+00**	**0.00e+00 0.00e+00**	=	cf_{17}	**1.40e+03 4.59e+02**	4.60e+05 3.12e+05	−
cf_3	**0.00e+00 0.00e+00**	**0.00e+00 0.00e+00**	=	cf_{18}	**9.51e+01 4.05e+01**	2.75e+02 2.46e+02	−
cf_4	**0.00e+00 0.00e+00**	2.28e+01 3.02e+01	−	cf_{19}	**4.83e+00 8.65e-01**	6.58e+00 1.18e+00	−
cf_5	2.03e+01 5.04e-02	**2.02e+01 9.17e-02**	+	cf_{20}	**1.00e+03 1.94e+03**	3.36e+01 2.29e+01	+
cf_6	**7.39e+00 3.38e+00**	1.06e+01 1.76e+00	−	cf_{21}	**1.77e+03 9.41e+03**	6.70e+04 3.15e+04	−
cf_7	**5.32e-04 2.15e-03**	4.92e-03 9.46e-03	−	cf_{22}	**1.57e+02 7.01e+01**	1.65e+02 6.11e+01	=
cf_8	**0.00e+00 0.00e+00**	**0.00e+00 0.00e+00**	=	cf_{23}	**3.15e+02 0.00e+00**	**3.15e+02 0.00e+00**	=
cf_9	**2.26e+01 3.96e+00**	3.23e+01 3.84e+00	−	cf_{24}	2.27e+02 4.45e+00	2.27e+02 3.18e+00	=
cf_{10}	**5.30e-03 9.06e-03**	1.99e-01 7.71e-01	−	cf_{25}	**2.08e+02 2.55e+00**	2.10e+02 1.42e+00	−
cf_{11}	**1.58e+03 2.34e+02**	1.88e+03 1.53e+02	−	cf_{26}	1.02e+02 1.39e+01	**1.00e+02 0.00e+00**	+
cf_{12}	**2.46e-01 4.02e-02**	2.49e-01 4.31e-02	=	cf_{27}	**3.50e+02 4.65e+01**	4.05e+02 6.09e+01	−
cf_{13}	**2.13e-01 3.62e-02**	2.43e-01 2.71e-02	−	cf_{28}	**7.99e+02 3.67e+01**	8.52e+02 2.96e+01	−
cf_{14}	2.29e-01 3.33e-02	**2.25e-01 3.62e-02**	+	cf_{29}	**7.46e+02 2.03e+01**	1.37e+03 1.79e+02	−
cf_{15}	**2.99e+00 4.29e-01**	3.84e+00 4.31e-01	=	cf_{30}	**1.64e+03 3.66e+02**	2.30e+03 5.98e+02	−
Total number of (+/=/-)		4/9/17					

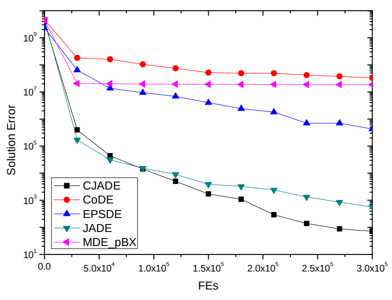
Table 2 The experimental results of 30-dimensional problems $cf_1 - cf_{30}$

	CJADE	JADE	EPSDE	CoDE	MDE_pBX
cf_1	2.15e+03±2.40e+03	**1.78e+03±2.09e+03**	2.72e+05±2.14e+05†	1.35e+07±5.96e+06†	1.03e+07±6.04e+06†
cf_2	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	3.92e+07±5.19e+07†
cf_3	**0.00e+00±0.00e+00**	1.34e+00±2.13e+00†	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	8.03e-01±2.02e+00†
cf_4	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	3.01e-03±1.05e-02†	1.27e+02±1.29e+01†	1.45e+02±3.64e+01†
cf_5	**2.03e+01±5.04e-02**	**2.03e+01±3.97e-02†**	2.06e+01±6.04e-02†	2.05e+01±4.79e-02†	2.05e+01±1.02e-01†
cf_6	7.39e+00±3.38e+00	9.29e+00±2.56e+00†	1.99e+01±1.37e+00†	1.81e+01±1.38e+00†	**4.53e+00±1.53e+00‡**
cf_7	5.32e-04±2.15e-03	**0.00e+00±0.00e+00**	1.45e-04±1.03e-03	1.88e-07±9.26e-07†	1.30e+00±9.85e-01†
cf_8	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	2.37e+01±2.78e+00†	**0.00e+00±0.00e+00**	1.54e+01±4.46e+00†
cf_9	**2.26e+01±3.96e+00**	2.52e+01±3.73e+00†	1.17e+02±8.74e+00†	1.14e+02±8.82e+00†	2.73e+01±6.36e+00†
cf_{10}	**5.30e-03±9.06e-03**	**5.30e-03±1.08e-02**	9.25e+02±1.59e+02†	1.73e-01±1.37e-01†	2.10e+02±1.54e+02†
cf_{11}	**1.58e+03±2.34e+02**	1.66e+03±2.14e+02	4.82e+03±2.73e+02†	3.94e+03±2.71e+02†	2.62e+03±6.52e+02†
cf_{12}	**2.46e-01±4.02e-02**	2.61e-01±3.99e-02	1.04e+00±1.26e-01†	7.57e-01±1.00e-01†	4.15e-01±1.53e-01†
cf_{13}	2.13e-01±3.62e-02	2.05e-01±2.91e-02	3.01e-01±3.24e-02†	4.37e-01±6.04e-02†	**2.04e-01±4.54e-02**
cf_{14}	**2.29e-01±3.33e-02**	2.31e-01±3.37e-02	2.55e-01±3.35e-02†	2.67e-01±3.28e-02†	2.52e-01±4.42e-02†
cf_{15}	**2.99e+00±4.29e-01**	3.17e+00±3.15e-01†	1.17e+01±8.60e-01†	1.30e+01±1.19e+00†	4.03e+00±1.62e+00†
cf_{16}	**9.31e+00±3.18e-01**	9.37e+00±3.30e-01	1.15e+01±3.05e-01†	1.10e+01±2.78e-01†	9.43e+00±8.27e-01
cf_{17}	1.40e+03±4.59e+02	**1.15e+03±3.76e+02‡**	2.35e+03±8.84e+02†	1.60e+04±2.30e+04†	5.59e+04±7.53e+04†
cf_{18}	9.51e+01±4.05e+01	1.08e+02±2.25e+02†	**4.75e+01±6.10e+00‡**	1.41e+03±2.24e+03†	1.11e+03±1.21e+03†
cf_{19}	4.83e+00±8.65e-01	**4.52e+00±7.63e-01‡**	4.63e+00±5.35e-01	7.65e+00±8.64e-01†	1.42e+01±1.78e+01†
cf_{20}	1.00e+03±1.94e+03	2.34e+03±2.27e+03†	**2.71e+01±3.42e+00‡**	1.96e+02±4.34e+02‡	7.60e+01±3.68e+01‡
cf_{21}	1.77e+03±9.41e+03	8.41e+03±3.89e+04†	**7.24e+02±1.38e+02‡**	4.80e+03±7.37e+03†	3.05e+03±3.22e+03†
cf_{22}	1.57e+02±7.01e+01	1.43e+02±6.11e+01	**1.41e+02±6.13e+01‡**	1.57e+02±9.25e+01	2.40e+02±1.13e+02†
cf_{23}	**3.15e+02±0.00e+00**	**3.15e+02±0.00e+00**	**3.15e+02±0.00e+00**	**3.15e+02±0.00e+00**	3.17e+02±1.41e+00†
cf_{24}	2.27e+02±4.45e+00	2.25e+02±1.91e+00‡	**2.23e+02±9.09e-01‡**	2.25e+02±5.93e-01‡	2.33e+02±5.52e+00†
cf_{25}	2.08e+02±2.55e+00	2.04e+02±1.81e+00‡	**2.03e+02±2.35e-01‡**	2.09e+02±1.17e+00†	2.11e+02±1.19e+00†
cf_{26}	1.02e+02±1.39e+01	1.02e+02±1.39e+01	**1.00e+02±0.00e+00**	**1.00e+02±2.69e-01**	1.29e+02±4.56e+01†
cf_{27}	3.50e+02±4.65e+01	**3.38e+02±4.66e+01‡**	3.73e+02±4.50e+01	4.21e+02±3.71e+01†	4.34e+02±4.51e+01†
cf_{28}	7.99e+02±3.67e+01	**7.90e+02±3.94e+01**	9.58e+02±2.69e+01†	9.84e+02±3.32e+01†	9.03e+02±4.18e+01†
cf_{29}	**7.29e+02±2.03e+01**	7.46e+02±1.27e+01†	8.55e+02±2.05e+02†	1.23e+03±1.70e+02†	2.34e+03±9.65e+02†
cf_{30}	1.64e+03±3.66e+02	1.59e+03±5.25e+02	**1.01e+03±1.39e+02‡**	2.30e+03±3.55e+02†	7.30e+03±3.98e+03†
	$b/n/w$	9/16/5	16/7/7	21/7/2	26/2/2

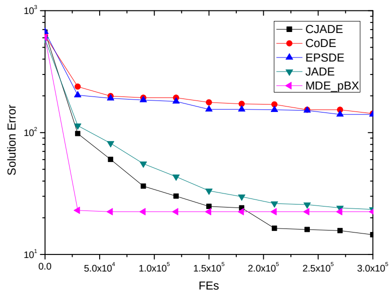
†CJADE performs better than the algorithm at a 0.05 level of significance by the Wilcoxon rank-sum test.
‡CJADE performs worse than the algorithm at a 0.05 level of significance by the Wilcoxon rank-sum test.

CJADE outperforms CoDE and MDE_pBX on five and eight functions, respectively.

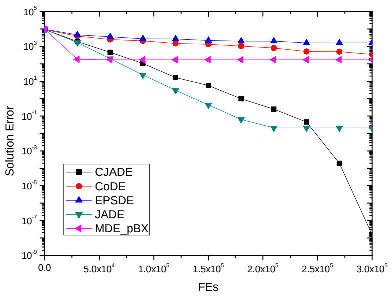
Fig. 1 shows the convergence graphs for cf_1, cf_9, cf_{10} and cf_{12} in detail. Obviously, CJADE maintains faster convergence speed and higher solution precision simultaneously, which shows that CJADE is better than the four competitors.



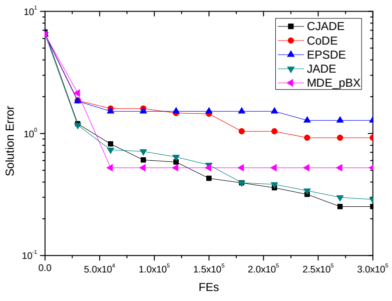
a) cf_1



b) cf_9

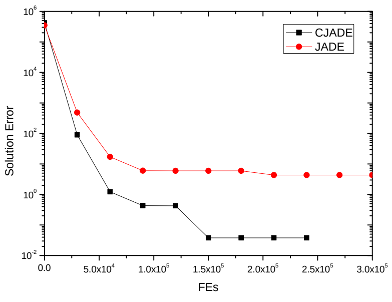


c) cf_{10}

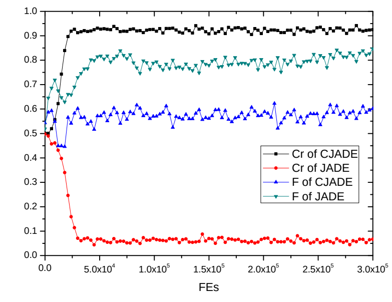


d) cf_{12}

Figure 1 The convergence curves of CJADE, CoDE, EPSDE, JADE and MDE_pBX on cf_1, cf_9, cf_{10} and cf_{12}.



a) convergence curves of cf_3



b) mean values of Cr and F of cf_3

Figure 2 The convergence curves and of CJADE and JADE and their mean values of Cr and F on cf_3.

5.4 The Advantage of Improved Adaptation of Parameters

As mentioned in Section 4, JADE is a special case of CJADE with $K = 1$. The procedure of CJADE is the same as JADE except for the adaptation of parameters. In this section, we will study the improved adaptation of parameters and show the advantage of CJADE compared with JADE. According to the results of Table 2, we take cf_3 as example. The evolutionary curves of solution error and parameters of CJADE and JADE are plotted in Fig. 2. When the objective function is separable, a value for Cr from the range (0.0, 0.2) is best because then each trial vector frequently competes with a target vector from which it differs by a single parameter (Ronkkonen et al., 2005). In contrast, a large value of Cr is very suitable for non-separable problems, since in this case all parameters may have to be adjusted simultaneously for the search to remain efficient. cf_3 is an unimodal and non-separable functions (Liang et al., 2013). As shown in Fig. 2, CJADE has a larger mean value of Cr of all individuals than JADE, which is why CJADE performs better than JADE on cf_3.

6 CONCLUSION

DE is an efficient population-based optimization algorithm. However, its performance highly depends on its control parameters. In this paper, a variant of JADE algorithm based on the clustering of successful parameters, denoted CJADE, is proposed. It considers each F and Cr as a point of the two-dimensional vector space and utilizes the K-means to cluster the successful parameters and to update the control parameters automatically.

The experimental studies are carried on 30 benchmark functions used for the 2014 IEEE CEC Competition on Single Objective Real-parameter Numerical Optimization, which includes simple unimodal and multimodal function, hybrid and composition functions. Compared with four DE variants, namely, JADE, jDE, MGBDE and RADE, CJADE shows good performance on the majority of the test functions.

Further research work includes the analysis of control parameters during the evolution process, as well as experiments with some other methods of cluster analysis, such as distribution-based and density-based clustering.

ACKNOWLEDGMENT

This work is supported by the self-determined research funds of CCNU from the colleges' basic research and operation of MOE (No. CCNU15A05063).

References

S. Das and P. N. Suganthan (2011) 'Differential evolution: a survey of the state-of-the-art', *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31.

Ghosh P, Das S and Zafar H. (2012) 'Adaptive-differential-evolution-based design of two-channel quadrature mirror filter banks for sub-band coding and data transmission', *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 6, pp. 1613–1623.

S. M. Guo, L. S. Shieh, G. Chen and N. P. Coleman. (2000) 'Observer-type Kalman innovation filter for uncertain linear systems', *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 4, pp. 1406–1418.

S. M. Islam, S. Das, S. Ghosh, S. Roy and P. N. Suganthan. (2012) 'An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 482–500.

K. J. Kim and S. B. Cho. (2008) 'An evolutionary algorithm approach to optimal ensemble classiers for DNA microarray data analysis', *IEEE Transactions on Evolutionary Computation*. vol. 12, no. 3, pp. 377–388.

D. Koloseni, J. Lampinen and P. Luukka. (2012) 'Optimized distance metrics for differential evolution based nearest prototype classifier', *Expert Systems With Applications*, vol. 39, no. 12, pp. 10564–10570.

J. J. Liang, B. Qu and P. N. Suganthan. (2013) 'Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization', *Technical Report*, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, 2013.

R. Mallipeddi, P. N. Suganthan, Q. K. Pan and M. F. Tasgetiren. (2011) 'Differential evolution algorithm with ensemble of parameters and mutation strategies', *Applied Soft Computing*, vol. 11, no. 2, pp. 1679–1696.

J. Ronkkonen, S. Kukkonen and K. V. Price. (2005) 'Real-parameter optimization with differential evolution', *Proc. 2005 IEEE Congr. Evol. Comput.*, vol. 1, pp. 506–513.

R. Storn and K. Price. (1995) 'Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces', *Technical Report*, International Computer Science Institute, Berkeley, USA, 1995.

Q. Su, Z. Huang, Z. Hu and X. Wang. (2012) 'Binarization algorithm based on differential evolution algorithm for gray images', *Proc. 9th Int. Conf. Fuzzy Systems and Knowledge Discovery*, vol. 1, pp. 2611–2615.

Y. Wang, Z. Cai and Q. Zhang. (2011) 'Differential evolution with composite trial vector generation strategies and control parameters', *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66.

J. Zhang and A. C. Sanderson. (2009) 'JADE: adaptive differential evolution with optional external archive', *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958.

F.Peng, K.Tang, G.Chen and X.Yao. (2009) 'Multi-start JADE with knowledge transfer for numerical optimization', *2009 IEEE Congress on Evolutionary Computation*, vol. 1, pp. 1889–1895.

Y.Li, Z. Zhan, Y. Gong , W. Chen and J. Zhang and Y. Li. (2015) 'Differential evolution with an evolution path: A DEEP evolutionary algorithm', *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 1798–1810.

X.Pan and R.Li. (2015) 'An improved differential evolution for parameter optimisation', *International Journal of Wireless and Mobile Computing*, vol. 8, no. 4, pp. 394–400.

X.You, Y. Ma and Z. Liu. (2016) 'A new modified differential evolution for global optimisation', *International Journal of Wireless and Mobile Computing*, vol. 10, no. 1, pp. 56–61.

J. Zhang, W. Pan, J. Wu and J. Wang. (2015) 'Top–k elites based oppositional differential evolution', *International Journal of Wireless and Mobile Computing*, vol. 8, no. 2, pp. 166–174.