A Clustering-based Differential Evolution for Global Optimization☆

Zhihua Cai[a,*], Wenyin Gong[a,b], Charles X. Ling[b], Harry Zhang[c]

[a] *School of Computer Science, China University of Geosciences, Wuhan 430074, P.R. China*
[b] *Department of Computer Science, The University of Western Ontario, London,Ontario,N6A 5B7,Canada*
[c] *School of Computer Science, The University of New Brunswick, Fredrecton 4000, Canada.*

Abstract

Hybridization with other different algorithms is an interesting direction for the improvement of differential evolution (DE). In this paper, a hybrid DE based on the one-step k-means clustering, called Clustering-based DE (CDE), is presented for the unconstrained global optimization problems. The one-step k-means clustering acts as several multi-parent crossover operators to utilize the information of the population efficiently, and hence it can enhance the performance of DE. To validate the performance of our approach, 30 benchmark functions of a wide range of dimensions and diversity complexities are employed. Experimental results indicate that our approach is effective and efficient. Compared with other state-of-the-art DE approaches, our approach performs better, or at least comparably, in terms of the quality of the final solutions and the reduction of the number of fitness function evaluations (NFFEs).

Key words: Differential evolution, k-means clustering, hybridization, global optimization

1. Introduction

Without loss of generality, a global minimization problem can be formalized as a pair (S, f) , where $S \subseteq R^D$ is a bounded set on R^D and $f : S \rightarrow R$ is a D-dimensional real-valued function. The problem is to find a point $X^* \in S$ such that $f(X^*)$ is a global minimum on S [1]. More specifically, it is required to find an $X^* \in S$ such that

$$\forall X \in S : f(X^*) \leq f(X) \tag{1}$$

where f does not need to be continuous but it must be bounded. Generally, for each variable x_i it satisfies a constrained boundary:

$$l_i \leq x_i \leq u_i, i = 1, 2, \cdots, D \tag{2}$$

Global optimization problems are frequently arisen in almost every field of engineering design, applied sciences, molecular biology and other scientific applications. Many of these problems cannot be solved analytically, and consequently, they have to be addressed by numerical algorithms. Moreover, in many cases, global optimization problems are non-differentiable, noisy and simulation-based. Hence the gradient-based methods cannot be used for finding the global minimum of such problems. As a result, many researchers have devoted themselves in finding some reliable stochastic global optimization methods that do not require any computation of the gradients of the objective function. In global optimization problems, the major challenge is that an algorithm may be trapped in the local optima of the objective function. This issue is particularly challenging when the dimension is high and there are numerous local optima. Recently, using the Evolutionary Computation (EC) [2] to solve the global optimization has been very active, producing different kinds of EC for optimization in the continuous domain, such as genetic algorithms (GAs) [3] - [4], evolution strategy (ES) [5], evolutionary programming (EP) [1, 6], particle swarm optimization (PSO) [7], differential evolution (DE) [8], etc.

Differential evolution (DE) [9] algorithm is a novel evolutionary algorithm (EA) for global optimization, which mutation operator is based on the distribution of solutions in the population. It won the third place at the first International Contest on Evolutionary Computation on a real-valued function testsuite [10]. DE is a simple yet powerful population-based, direct search algorithm with the generation-and-test feature for globally optimizing functions using real-valued parameters. Among DE's advantages are its simple structure, ease of use, speed and robustness. Price & Storn [9] gave the working principle of DE with single scheme. Later on, they suggested ten different schemes of DE [10, 11]. DE has been successfully applied to a whole host of engineering problems including the design

☆This work was supported in part by the Fund for Outstanding Doctoral Dissertation of China University of Geosciences, China Scholarship Council under Grant No. 2008641008, the National High Technology Research and Development Program of China under Grant No. 2009AA12Z117, and the Research Fund for the Doctoral Program of Higher Education under Grant No. 20090145110007.

*Corresponding author.
Email addresses: zhcai@cug.edu.cn (Zhihua Cai), cug11100304@yahoo.com.cn (Wenyin Gong), cling@csd.uwo.ca (Charles X. Ling), hzhang@unb.ca (Harry Zhang)

of digital filters, mechanical design optimization, aerodynamic design and multiprocessor synthesis [10, 11, 12]. However, DE has been shown to have certain weaknesses, especially if the global optimum should be located using a limited number of fitness function evaluations (NFFEs). In addition, DE is good at exploring the search space and locating the region of global minimum, but it is slow at exploitation of the solution [13].

The main contribution of this paper is the hybridization of the one-step k-means clustering with DE, which makes the original DE more effective and efficient. The one-step k-means clustering acts as several multi-parent crossover operators to utilize the information of the population efficiently. After incorporating the one-step k-means clustering, the hybrid DE approach, called Clustering-based DE (CDE), can balance the exploration and exploitation in the evolutionary process. The advantages of CDE are its simplicity, efficiency and flexibility. To verify the performance of our approach, 30 benchmark functions (including 8 new test functions provided by CEC2005 special session [14]) are selected from the literature. Experimental results indicate that our approach is effective and efficient. Compared with other state-of-the-art DE approaches, our approach performs better, or at least comparably, in terms ofthe quality of the final solutions and the reduction of the NFFEs.

The remainder of this paper is organized as follows. In Section 2, we briefly introduce the DE algorithm. In addition, some improved variants of DE are reviewed. Section 3 briefly describes the k-means clustering used in this work. Our proposed approach is presented in detail in Section 4. In Section 5, we verify our approach through 30 benchmark functions. Moreover, the experimental results are compared with those of some state-of-the-art DE approaches. The last section, Section 6, is devoted to conclusions and future work.

2. Differential Evolution

DE [9] is a simple EA that creates new candidate solutions by combining the parent individual and several other individuals of the same population. A candidate replaces the parent only if it has better fitness. Among DE's advantages are its simple structure, ease of use, speed and robustness. Due to these advantages, it has many real-world applications, such as data mining [15, 16], pattern recognition, digital filter design, neural network training, etc. [11, 12].

The pseudo-code of DE is shown in Algorithm 1. Where D is the number of decision variables, NP is the population size, F is the mutation scaling factor, CR is the probability of crossover operator, U^i is the offspring, $\text{rndint}(1, D)$ is a uniformly distributed random integer number between 1 and D, and $\text{rnd}_j[0, 1)$ is a uniformly distributed random real number in $[0, 1)$. Many schemes of creation of a candidate are possible. We use the DE/rand/1/exp scheme (see lines 6 - 13 of Algorithm 1) described in Algorithm 1 (more details on DE/rand/1/exp and other DE schemes can be found in [10] and [11]).

From Algorithm 1, we can see that there are only three control parameters in this algorithm. These are NP, F and CR. For the terminal conditions, one can either fix the maximum NFFEs

Algorithm 1 DE algorithm with DE/rand/1/exp

1: Generate the initial population P
2: Evaluate the fitness for each individual in P
3: **while** The halting criterion is not satisfied **do**
4:   **for** $i = 1$ to NP **do**
5:     Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq i$
6:     $j_{rand} = \text{rndint}(1, D)$
7:     $L = 0$
8:     $U^i = P^i$
9:     **repeat**
10:       $U^i_j = X^{r_1}_j + F \times (X^{r_2}_j - X^{r_3}_j)$
11:       $j_{rand} = (j_{rand} + 1) \bmod D$
12:       $L = L + 1$
13:     **until** $\text{rnd}_j[0, 1) > CR$ or $L > D$
14:     Evaluate the offspring U^i
15:     **if** U^i is better than P^i **then**
16:       $P^i = U^i$ {Replace the parent *immediately*}
17:     **end if**
18:   **end for**
19: **end while**

(*Max_NFFEs*) or the precision of a desired solution value to reach (*VTR*).

In the original DE algorithm, many schemes have been proposed [10, 11] that use different learning strategies and/or recombination operations in the reproduction stage. In order to distinguish among its schemes, the notation "$DE/a/b/c$" is used, where "DE" denotes the Differential Evolution; "a" specifies the vector to be mutated (which can be random or the best vector); "b" is the number of difference vectors used; and "c" denotes the crossover scheme, *binomial* or *exponential*. The exponential crossover scheme is presented in Algorithm 1 and in case of exponential crossover, the crossover probability CR regulates how many consecutive mutated genes are copied to the trial individual U^i. Using this notation, the DE strategy described in Algorithm 1 above can be denoted as DE/rand/1/exp. Other well-known schemes are DE/best/1/exp, DE/rand/2/exp, and DE/best/2/exp which can be implemented by (3) - (5), respectively.

$$U^i = X^{best} + F \times (X^j - X^h) \tag{3}$$

$$U^i = X^j + F \times (X^h - X^l) + F \times (X^s - X^t) \tag{4}$$

$$U^i = X^{best} + F \times (X^j - X^h) + F \times (X^l - X^s) \tag{5}$$

where X^{best} represents the best individual in the current generation, i, j, h, l, s and $t \in \{1, \cdots, D\}$, and $i \neq j \neq h \neq l \neq s \neq t$. Again, each of the above algorithms can be configured to use the binomial crossover.

Recently, many researchers are working on the improvement of DE, and many variants of DE are presented. Hybridization with other different algorithms is one direction for improvement. Fan and Lampinen [17] proposed a new version of DE which uses an additional mutation operation called trigonometric mutation operation. They showed that the modified DE algorithm can outperform the classic DE algorithm for some benchmarks and real-world problems. Sun *et al*. [18] proposed

Table 1: The 22 benchmark functions used in our experimental study, where D is the number of variables, "optimal" is the minimum value of the function, and $S \subseteq R^D$. A detailed description of all functions can be found in [1].

Test Functions	D	S	optimal
$f01 = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0
$f02 = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	30	$[-10, 10]^D$	0
$f03 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]^D$	0
$f04 = \max_i\{\lvert x_i\rvert, 1 \leq i \leq D\}$	30	$[-100, 100]^D$	0
$f05 = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^D$	0
$f06 = \sum_{i=1}^{D} (\lfloor x_i + 0.5\rfloor)^2$	30	$[-100, 100]^D$	0
$f07 = \sum_{i=1}^{D} x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]^D$	0
$f08 = \sum_{i=1}^{D} (-x_i \sin(\sqrt{\lvert x_i\rvert}))$	30	$[-500, 500]^D$	-418.982887×D
$f09 = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	30	$[-5.12, 5.12]^D$	0
$f10 = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + \exp(1)$	30	$[-32, 32]^D$	0
$f11 = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600, 600]^D$	0
$f12 = \frac{\pi}{D}\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	30	$[-50, 50]^D$	0
$f13 = \frac{1}{10}\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)^2[1 + \sin^2(2\pi x_D)]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	30	$[-50, 50]^D$	0
$f14 = \left[\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right]^{-1}$	2	$[-65.536, 65.536]^D$	1
$f15 = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]^D$	0.003075
$f16 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^D$	-1.0316285
$f17 = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
$f18 = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]^D$	3
$f19 = -\sum_{i=1}^{4} c_i\exp[-\sum_{j=1}^{D} a_{ij}(x_j - p_{ij})^2]$	3	$[0, 1]^D$	-3.86
$f20 = -\sum_{i=1}^{4} c_i\exp[-\sum_{j=1}^{D} a_{ij}(x_j - p_{ij})^2]$	6	$[0, 1]^D$	-3.32
$f21 = -\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]^D$	-10
$f22 = -\sum_{i=1}^{7}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]^D$	-10

a new hybrid algorithm based on a combination of DE and Estimation of Distribution Algorithm (EDA). This technique uses a probability model to determine promising regions in order to focus the search process on those areas. Gong *et al*. [19] employed the two level orthogonal crossover to improve the performance of DE. They showed that the proposed approach performs better than the classical DE in terms of the quality, speed, and stability of the final solutions. Noman and Iba [20] proposed fittest individual refinement, a crossover-based local search (LS) method DE to solve the high dimensional problems. They showed that the improved DE method accelerates the convergence rate for high dimensional benchmark functions. Based on their previous work, Noman and Iba incorporated LS into the classical DE in [13]. They presented an LS technique to solve this problem by adaptively adjusting the length of the search, using a hill-climbing heuristic. Through the experiments, they showed that the proposed new version of DE performs better, or at least comparably, to classic DE algorithm. Kaelo and Ali [21] adopted the attraction-repulsion concept of electromagnetism-like algorithm to boost the mutation operation of the original DE. Yang *et al*. [22] proposed a neighborhood search based DE. Experimental results showed that DE with neighborhood search has significant advantages over other existing algorithms on a broad range of different benchmark functions [22]. Wang *et al*. [23] proposed a dynamic clustering-based DE for global optimization, where a hierarchical clustering method is dynamically incorporated in DE. Experiments on 28 benchmark problems, including 13 high dimensional functions, showed that the new method is able to find near optimal solutions efficiently [23].

Some other studies focus on adapting DE's control parameters. Liu and Lampinen [24] proposed a fuzzy adaptive DE (FADE) which uses fuzzy logic controllers to adapt the mutation and crossover control parameters. Brest *et al*. [8] proposed

Table 2: Best Error Values of DE and CDE on All Test Functions, Where "Mean" Indicates the Mean Best Error Values Found in the Last Generation, "Std Dev" Stands For the Standard Deviation. "Time" Indicates the Average Running Time in Seconds. Hereafter, A Result With **Boldface** Means Better Value Found.

F	*D*	Max_NFFEs	DE				CDE				DE-CDE
			Mean	Std Dev	SR	Time (s)	Mean	Std Dev	SR	Time (s)	*t-test*
f01	30	150 000	2.01E-17	1.14E-17	50	**0.5669**	**1.07E-28**	7.65E-29	50	0.6218	12.51†
f02	30	200 000	3.86E-14	9.28E-15	50	**1.1814**	**4.21E-21**	1.85E-21	50	1.2611	29.41†
f03	30	500 000	5.04E-11	2.46E-11	50	**3.1936**	**1.64E-34**	9.18E-34	50	3.4359	14.46†
f04	30	500 000	8.81E-08	2.39E-08	0	**2.8315**	**6.48E-22**	1.18E-21	**50**	3.1190	26.02†
f05	30	500 000	5.15E-22	1.21E-21	50	**2.5813**	**0**	0	50	2.8377	3.00†
f06	30	150 000	0	0	50	**0.7424**	0	0	50	0.8096	0
f07	30	300 000	0.0078	0.0017	50	**2.2469**	**0.0013**	7.37E-04	50	2.3298	24.62†
f08	30	300 000	0	0	50	**2.1249**	0	0	50	2.2004	0
f09	30	300 000	0	0	50	**1.7517**	0	0	50	1.8766	0
f10	30	150 000	1.21E-09	3.14E-10	50	**0.9142**	**5.28E-15**	1.67E-15	50	0.9612	27.24†
f11	30	200 000	0	0	50	**1.3484**	0	0	50	1.4278	0
f12	30	150 000	1.46E-18	7.33E-19	50	**2.4140**	**1.79E-30**	1.50E-30	50	2.4234	14.11†
f13	30	150 000	1.59E-16	6.79E-17	50	**2.2264**	**9.42E-29**	8.40E-29	50	2.2591	16.53†
f14	2	10 000	0	0	50	0.0986	0	0	50	**0.0983**	0
f15	4	40 000	1.85E-19	4.00E-19	50	0.1657	**1.03E-19**	3.12E-19	50	**0.1643**	1.14
f16	2	10 000	1.28E-14	4.71E-14	50	0.0389	**7.99E-16**	4.44E-15	50	**0.0333**	1.79
f17	2	10 000	1.74E-11	6.58E-11	50	0.0311	**4.33E-13**	1.46E-12	50	**0.0299**	1.82
f18	2	10 000	7.08E-15	1.43E-14	50	**0.0310**	**4.69E-15**	4.93E-15	50	0.0315	1.11
f19	3	10 000	0	0	50	**0.0440**	0	0	50	0.0454	0
f20	6	20 000	2.92E-12	2.04E-11	50	0.0793	**1.40E-14**	7.04E-14	50	**0.0780**	1.01
f21	4	10 000	1.91E-08	3.75E-08	30	0.0377	**1.67E-08**	3.90E-08	**37**	**0.0362**	0.32
f22	4	10 000	**4.98E-09**	2.54E-08	**48**	**0.0346**	5.60E-09	1.59E-08	45	0.0360	-0.14
F01	30	300 000	0	0	50	**8.6502**	0	0	50	8.6971	0
F02	30	300 000	3.12E-04	1.28E-04	0	**8.9969**	**3.60E-16**	3.97E-16	**50**	9.1859	17.21†
F03	30	300 000	1.03E+06	557411.46	0	**9.4954**	**892897.03**	306444.77	0	9.5035	1.54
F04	30	300 000	4.11E-04	1.94E-04	0	**8.9765**	**4.52E-16**	4.93E-16	**50**	9.1330	14.99†
F06	30	300 000	0.0091	0.019	0	**10.855**	**0.0041**	0.014	**7**	10.924	1.39
F07	30	300 000	**2.03E-05**	1.57E-05	0	**8.8908**	0.0041	0.0062	**32**	9.2186	-4.64†
F08	30	300 000	20.95	0.061	0	**9.0375**	**20.93**	0.082	0	9.4346	1.33
F09	30	300 000	0	0	50	**8.9830**	0	0	50	9.0860	0

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

self-adapting control parameter settings. Salman *et al.* [25] proposed a self-adaptive DE (SDE) algorithm which eliminates the need for manual tuning of control parameters. In SDE, the mutation weighting factor F is self-adapted by a mutation strategy similar to the mutation operator of DE. Nobakhti and Wang [26] proposed a Randomized Adaptive Differential Evolution (RADE) method, where a simple randomized self-adaptive scheme is proposed for the DE scaling factor F. Qin and Suganthan [27] proposed a self-adaptive DE algorithm. The aim of their work was to allow DE to switch between two schemes: "DE/rand/1/bin" and "DE/best/2/bin" and also to adapt the F and CR values. The approach performed well on several benchmark problems. Das *et al.* [28] proposed two variants of DE, DERSF and DETVSF, that use varying scale factors. They concluded that those variants outperform the original DE. Teo [29] presented a dynamic self-adaptive populations DE, where the population size is self-adapting. Brest and Mauěc [30] proposed an improved DE method, where the population size is gradually reduced. They concluded that their approach improved efficiency and robustness of DE.

Most recently, Rahnamayan *et al.* [31, 32, 33] proposed a novel initialization approach which employs opposition-based learning to generate initial population. Through a comprehensive set of benchmark functions they showed that replacing the random initialization with the opposition-based population initialization in DE can accelerate convergence speed.

Although there are many hybrid DE variants for the improvement of DE, only a little work studied the hybridization of clustering techniques with the DE method [23]. To the best of our knowledge, the k-means clustering is not used to enhance the performance of DE until date.

3. K-Means Clustering

Clustering is a process that organizes a data (pattern) set into a number of groups (clusters) such that patterns within a cluster are more similar to each other than patterns belonging to different clusters; in other words, clustering is an important technique for discovering the inherent structure in any given pattern set.

Clustering algorithms proposed in the literature can be divided into two main categories: crisp (or hard) clustering procedures where each data point belongs to only one cluster, and fuzzy clustering techniques where every data point belongs to every cluster with a specific degree of membership [34].

There are many clustering algorithms in the literature. The k-means clustering is employed in this study and shortly described as follows. K-means clustering [34], which is an iterative hill climbing algorithm, is one of the widely used clustering techniques. It consists of the following steps:

1) Choose k initial cluster centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_k$ randomly from the n points $\{X_1, X_2, \cdots, X_n\}$.

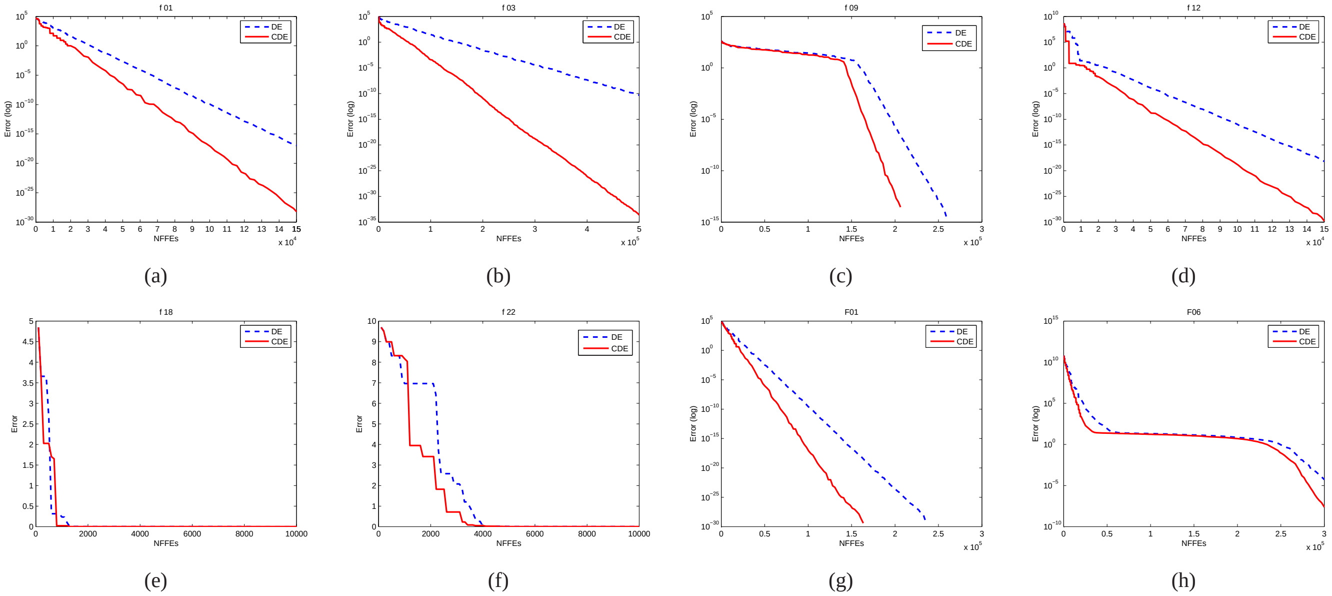


Figure 1: Mean error curves of DE and CDE for selected functions. (a) f01. (b) f03. (c) f09. (d) f12. (e) f18. (f) f22. (g) F01. (h) F06.

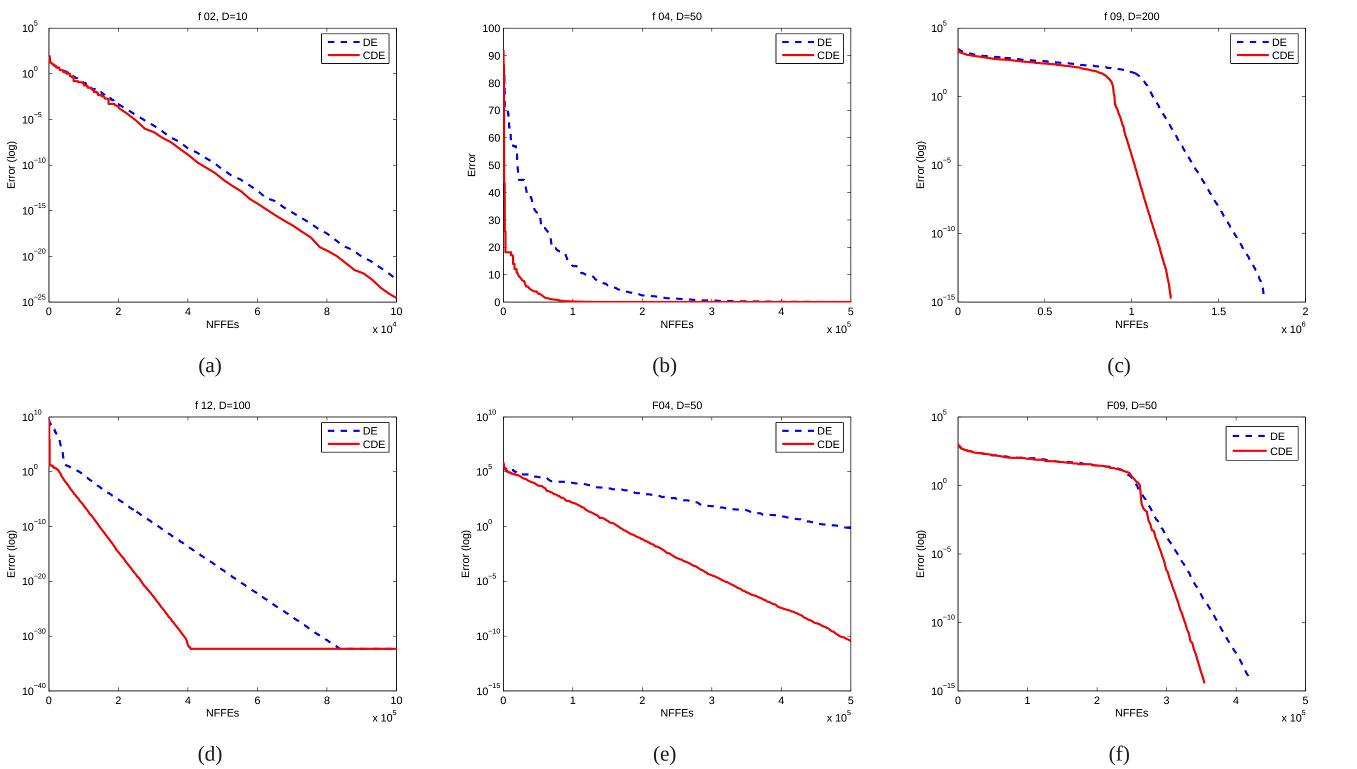


Figure 2: Mean error curves of DE and CDE for selected functions at different dimensions. (a) f02 (D = 10). (b) f04 (D = 50). (c) f09 (D = 200). (d) f12 (D = 100). (e) F04 (D = 50). (f) F09 (D = 50).

Table 3: NFFEs Required to Obtain Accuracy Levels Less Than ϵ. "NA" Indicates the Accuracy Level is Not Obtained After 500000 NFFEs.

F	*D*	DE			CDE			DE-CDE
		Mean	Std Dev	SR	Mean	Std Dev	SR	*t-test*
f01	30	88638	1050.34	50	**56525.82**	1107.36	50	148.77[†]
f02	30	129962	1071.12	50	**87810.34**	1149.11	50	189.73[†]
f03	30	422024	6461.01	50	**155326.04**	4969.85	50	231.35[†]
f04	30	NA	NA	0	**208667.78**	5828.96	**50**	–
f05	30	345258	12824.57	50	**313882.82**	12660.44	50	12.31[†]
f06	30	32196	893.27	50	**18736.76**	1244.05	50	62.14[†]
f07	30	236198	44694.77	50	**36884.36**	18150.75	50	29.21[†]
f08	30	143724	2356.54	50	**117509.58**	2842.75	50	50.21[†]
f09	30	215304	3557.22	50	**188759.70**	4420.37	50	33.08[†]
f10	30	137056	1298.53	50	**88046.56**	1195.64	50	196.32[†]
f11	30	94812	3647.51	50	**59249.90**	2548.52	50	56.51[†]
f12	30	80520	1345.28	50	**47980.16**	1090.73	50	132.85[†]
f13	30	95080	1355.41	50	**56515.58**	1529.94	50	133.41[†]
f14	2	**4766**	543.46	50	4847.98	575.67	50	-0.73
f15	4	10038	775.62	50	**9927.02**	799.44	50	0.71
f16	2	**4688**	751.49	50	4759.02	694.93	50	-0.49
f17	2	6104	1149.61	50	**6103.16**	1188.93	50	0.003
f18	2	**3550**	299.14	50	3581.08	257.32	50	-0.55
f19	3	4174	259.36	50	**4149.48**	263.67	50	0.47
f20	6	11094	916.82	50	**10963.32**	1196.13	50	0.61
f21	4	9704	601.34	50	**9655.20**	585.99	50	0.41
f22	4	**9146**	392.38	50	9212.60	565.28	50	-0.68
F01	30	89600	1097.49	50	**59176.46**	1114.09	50	137.56[†]
F02	30	464600	8139.42	50	**188492.82**	3761.82	50	217.73[†]
F03	30	NA	NA	0	NA	NA	0	–
F04	30	467920	9807.97	50	**190753.02**	4294.27	50	183.04[†]
F06	30	357646	10250.37	50	**322388.76**	13471.89	50	14.73[†]
F07	30	428790	15032.58	50	**227382.08**	15566.52	50	59.91[†]
F08	30	NA	NA	0	NA	NA	0	–
F09	30	207102	4102.98	50	**186611.62**	4381.94	50	24.14[†]

[†] The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

2) Assign point X_i, $i = 1, 2, \cdots, n$ to cluster C_j, $j = 1, 2, \cdots, k$, if and only if $\| X_i - \boldsymbol{c}_j \| \le \| X_i - \boldsymbol{c}_p \|$ $p = 1, 2, \cdots, k$, and $p \neq j$, where $\| X_i - \boldsymbol{c}_j \|$ is the distance between X_i and $\boldsymbol{c}_j$. Ties are resolved arbitrarily.

3) Compute new cluster centers $\boldsymbol{c}'_1, \boldsymbol{c}'_2, \cdots, \boldsymbol{c}'_k$ as follows:

$$\boldsymbol{c}'_i = \frac{1}{n_i} \sum_{X_j \in C_i} X_j, i = 1, 2, \cdots, k$$

where n_i is the number of elements belonging to cluster C_i.

4) If $\boldsymbol{c}'_i = \boldsymbol{c}_i$, $\forall i \in \{1, 2, \cdots, k\}$, then the process is terminated and $\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_k$ are chosen as the cluster centers. Otherwise, assign each $\boldsymbol{c}_i$ with $\boldsymbol{c}'_i, i = 1, 2, \cdots, k$, and continue from step 2).

The distance measure used in the clustering algorithm is a very important issue. The most widely used distance measure is the Euclidean distance, which between any two d-dimensional vectors X_i and X_j is given by

$$d(X_i, X_j) = \sqrt{\sum_{p=1}^{d} (X_{i,p} - X_{j,p})^2} = \| X_i - X_j \| . \tag{6}$$

The Euclidean distance measure is a special case (when $\alpha = 2$) of the Minowsky metric, which is defined as

$$d^{\alpha}(X_i, X_j) = \left(\sum_{p=1}^{d} (|X_{i,p} - X_{j,p}|)^{\alpha} \right)^{1/\alpha} = \| X_i - X_j \|^{\alpha} . \tag{7}$$

When $\alpha = 1$, the measure is known as the Manhattan distance.

Another distance measure used in clustering algorithms is the cosine distance (or vector dot product), which is given by

$$\langle X_i, X_j \rangle = \frac{\sum_{p=1}^{d} (X_{i,p} \cdot X_{j,p})}{\| X_i \| \cdot \| X_j \|}. \tag{8}$$

Although the clustering algorithms are originally used to obtain suboptimal clusters, recently some researchers adopt them for global optimization problems, especially for multimodal function optimization. Wang *et al*. [23] proposed a dynamic clustering-based DE for global optimization, where a hierarchical clustering method is dynamically incorporated into DE. Imrani *et al.* [35] combined the sharing technique and a fuzzy clustering algorithm to improve the performance of GAs in multimodal function optimization. Experiments on 4 multimodal functions indicated their approach showed good performance. Pelikan and Goldberg [36] used the k-means clustering in EAs to help the algorithm to separate the two or more complementary parts of the solution space and to eliminate the problem of symmetry in combinational optimization. Damavandi and Safavi-Naeini [37] proposed a hybrid EP based on a density clustering algorithm to preserve the diversity of the population. They showed that the hybrid method improved the robustness of the algorithm for complex multimodal circuit optimization problems. Lu and Yao [38] incorporated k-means

Table 4: Comparison of DE and CDE for Different Population Size, Hereafter, (#) Indicates the Number of Successful Runs, [$a \pm b$] Denotes the Averaged NFFEs Required When the Global Minimum Achieved Before Using the Maximum Allowed NFFEs

F	D	$NP = 50$		$NP = 200$	
		DE	CDE	DE	CDE
f01	30	1.89E-40 ± 1.82E-40 (50)	**7.67E-63 ± 9.61E-63** (50)†	1.28E-06 ± 2.43E-07 (0)	**6.44E-12 ± 3.22E-12** (50)†
f02	30	8.23E-31 ± 4.89E-31 (50)	**1.61E-45 ± 1.46E-45** (50)†	2.81E-06 ± 3.75E-07 (0)	**1.78E-09 ± 5.74E-10** (50)†
f03	30	2.99E-30 ± 3.34E-30 (50)	**1.43E-57 ± 4.62E-57** (50)†	3.40E-03 ± 8.70E-04 (0)	**1.90E-16 ± 1.35E-16** (50)†
f04	30	**7.83E-18 ± 4.64E-18** (50)†	8.30E-02 ± 2.20E-01 (35)	3.20E-03 ± 3.97E-04 (0)	**1.09E-10 ± 3.64E-11** (50)†
f05	30	[3.90E+05 ± 3.62E+04] (50)	**[3.33E+05 ± 1.82E+06]** (50)†	4.10E-03 ± 7.56E-03 (0)	**3.87E-06 ± 8.17E-06** (6)†
f06	30	[1.57E+04 ± 4.96E+02] (50)	**[9.81E+03 ± 6.83E+02]** (50)†	[6.52E+04 ± 1.28E+03] (50)	**[3.67E+04 ± 2.24E+03]** (50)†
f07	30	4.05E-03 ± 7.90E-04 (50)	**1.30E-03 ± 5.13E-04** (50)†	1.50E-02 ± 3.50E-03 (40)	**2.20E-03 ± 1.20E-03** (50)†
f08	30	[7.91E+04 ± 1.79E+03] (50)	**[6.44E+04 ± 1.52E+03]** (50)†	1.25E-09 ± 2.05E-09 (49)	**[2.67E+05 ± 7.04E+03]** (50)†
f09	30	[1.25E+05 ± 2.74E+03] (50)	**[1.07E+05 ± 2.69E+03]** (50)†	5.44E+00 ± 1.35E+00 (0)	**3.03E+00 ± 1.54E+00** (0)†
f10	30	[1.24E+05 ± 6.49E+03] (50)	**[7.31E+04 ± 1.31E+03]** (50)†	3.18E-04 ± 3.88E-05 (0)	**6.95E-07 ± 2.14E-07** (0)†
f11	30	[7.03E+04 ± 2.25E+03] (50)	**[4.58E+04 ± 1.48E+03]** (50)†	4.06E-09 ± 4.93E-09 (50)	**[1.72E+05 ± 6.43E+04]** (50)†
f12	30	1.57E-32 ± 0.00E+00 (50)	1.57E-32 ± 0.00E+00 (50)	8.33E-08 ± 2.16E-08 (0)	**1.01E-13 ± 4.87E-14** (50)†
f13	30	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)	1.25E-05 ± 4.02E-06 (0)	**9.09E-12 ± 5.39E-12** (50)†
f14	2	[4.26E+03 ± 2.71E+02] (50)	**[4.21E+03 ± 2.81E+02]** (50)	2.61E-07 ± 1.23E-06 (40)	**2.96E-08 ± 8.79E-08** (41)
f15	4	9.56E-09 ± 5.45E-09 (49)	**1.32E-09 ± 2.17E-10** (50)	1.41E-18 ± 1.49E-18 (50)	1.41E-18 ± 1.49E-18 (50)
f16	2	1.85E-18 ± 0.00E+00 (50)	1.85E-18 ± 0.00E+00 (50)	**3.25E-08 ± 3.31E-08** (45)	3.75E-08 ± 3.45E-08 (42)
f17	2	6.82E-32 ± 0.00E+00 (50)	6.82E-32 ± 0.00E+00 (50)	**2.32E-06 ± 3.31E-06** (0)	3.51E-06 ± 4.96E-06 (0)
f18	2	4.63E-25 ± 0.00E+00 (50)	4.63E-25 ± 0.00E+00 (50)	2.35E-12 ± 1.60E-12 (50)	2.35E-12 ± 1.60E-12 (50)
f19	3	3.35E-35 ± 0.00E+00 (50)	3.35E-35 ± 0.00E+00 (50)	4.65E-10 ± 2.05E-10 (50)	**3.25E-10 ± 1.88E-10** (50)
f20	6	[9.32E+03 ± 7.22E+02] (50)	**[9.10E+03 ± 6.43E+02]** (50)	2.52E-05 ± 3.98E-05 (0)	**1.68E-05 ± 6.32E-06** (0)
f21	4	[6.82E+03 ± 5.69E+02] (50)	**[6.73E+03 ± 1.95E+02]** (50)	**2.30E-02 ± 5.32E-02** (0)	3.81E-02 ± 5.85E-02 (0)
f22	4	1.34E-01 ± 1.92E-02 (49)	**[6.91E+03 ± 1.05E+02]** (50)†	3.50E-02 ± 1.26E-02 (0)	**1.10E-02 ± 6.51E-03** (0)
F01	30	[1.13E+05 ± 1.23E+03] (50)	**[7.94E+04 ± 1.02E+03]** (50)†	5.99E-17 ± 1.81E-17 (50)	**3.42E-26 ± 1.74E-26** (50)†
F02	30	7.16E-15 ± 9.92E-15 (50)	**2.52E-28 ± 2.08E-28** (50)†	6.02E+00 ± 1.28E+00 (0)	**9.79E-06 ± 4.98E-06** (0)†
F03	30	6.35E+05 ± 3.05E+05 (0)	**6.09E+05 ± 2.46E+05** (0)	1.28E+07 ± 3.04E+06 (0)	**1.31E+06 ± 7.64E+05** (0)†
F04	30	9.34E-15 ± 1.23E-14 (50)	**3.36E-28 ± 3.08E-28** (50)†	7.89E+00 ± 2.34E+00 (0)	**1.27E-05 ± 6.38E-06** (0)†
F06	30	**5.60E-02 ± 3.90E-01** (42)†	2.72E+00 ± 2.90E+00 (22)	1.22E+01 ± 6.20E-01 (0)	**9.95E+00 ± 6.48E-01** (0)†
F07	30	**8.87E-04 ± 2.70E-03** (45)†	1.30E-02 ± 1.10E-02 (26)	1.10E-01 ± 4.20E-02 (0)	**8.38E-04 ± 2.40E-03** (0)†
F08	30	2.09E+01 ± 6.60E-02 (0)	**2.04E+01 ± 2.60E-01** (0)†	2.09E+01 ± 5.10E-02 (0)	2.09E+01 ± 5.50E-02 (0)
F09	30	[1.22E+05 ± 2.59E+03] (50)	**[1.05E+05 ± 2.69E+03]** (50)†	**1.95E+00 ± 1.10E+00** (0)	1.99E+00 ± 1.23E+00 (0)

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

clustering with EDA to break the single Gaussian distribution assumption. In addition, they used the rival penalized competitive learning [39] to select the number of clusters during learning automatically. Experimental results showed that the proposed approaches can perform very well when dealing with multimodal functions that do not contain too many local optima. However, their approach failed to solve the functions with many local optima. Song and Yu [40] incorporated hierarchical clustering and sharing technique with GA to solve multi-peak function optimization. Alami *et al.* [41] combined the cultural algorithms and a fuzzy clustering algorithm for multimodal function optimization. Zhang *et al.* [42] adopted the k-means clustering algorithm to cluster the distribution of the population in the search space at each generation. Thereafter, the fuzzy logic was used to adaptively adjust the probability of mutation and crossover. Experiments conducted on some benchmark functions and the design of a buck regulator, they showed that the new method not only improves the convergence rate of the GA, but also prevents the solution from becoming trapped in a local optimum point. Ling *et al.* [43] presented a crowding clustering-based GA for multimodal function optimization. They concluded that their approach is superior to both standard crowding and deterministic crowding in quantity, quality and precision of multi-optimum search [43].

4. Clustering-based Differential Evolution: CDE

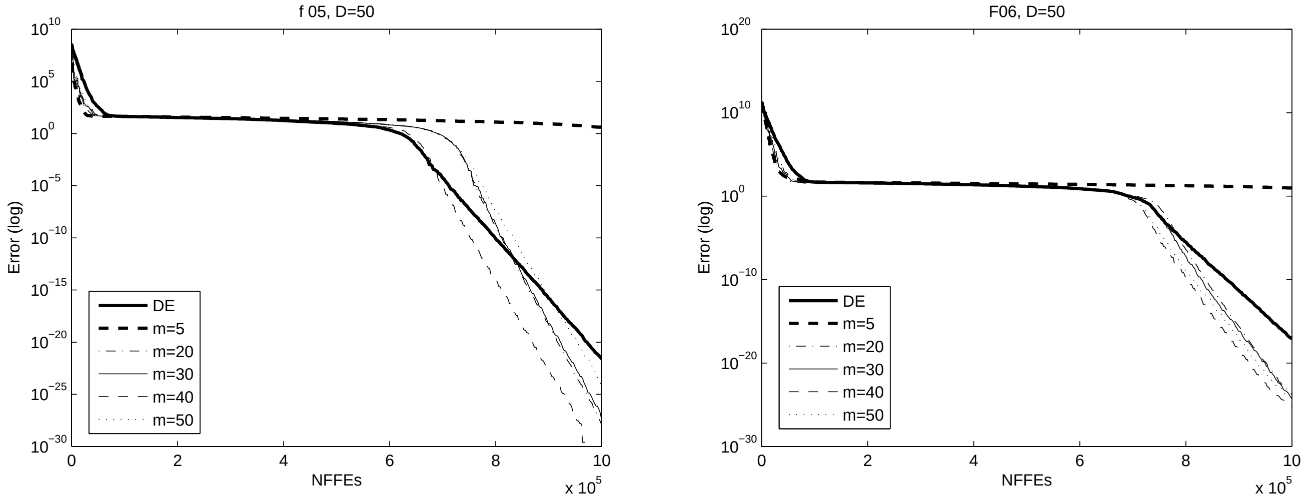
In order to accelerate the convergence rate and balance the exploration and exploitation of DE, in this study, we attempt to improve DE by integrating the one-step k-means clustering algorithm. Our proposed DE algorithm is named CDE. The pseudocode of CDE is described in Algorithm 2, where t is the generation counter, m is the *clustering period*, NP is the population size, and rndint[2, $\sqrt{NP}$] is a random integer number from [2, $\sqrt{NP}$]. Compared with the original DE algorithm, three crucial issues of CDE will be discussed as follows.

4.1. One-step K-Means Clustering

In this work, one-step k-means clustering is used to enhance the performance of DE. It acts as several multi-parent crossover operators to utilize the information of the population efficiently, and hence it can balance the exploration and exploitation in the evolutionary process. The one-step k-means clustering is described as follows.

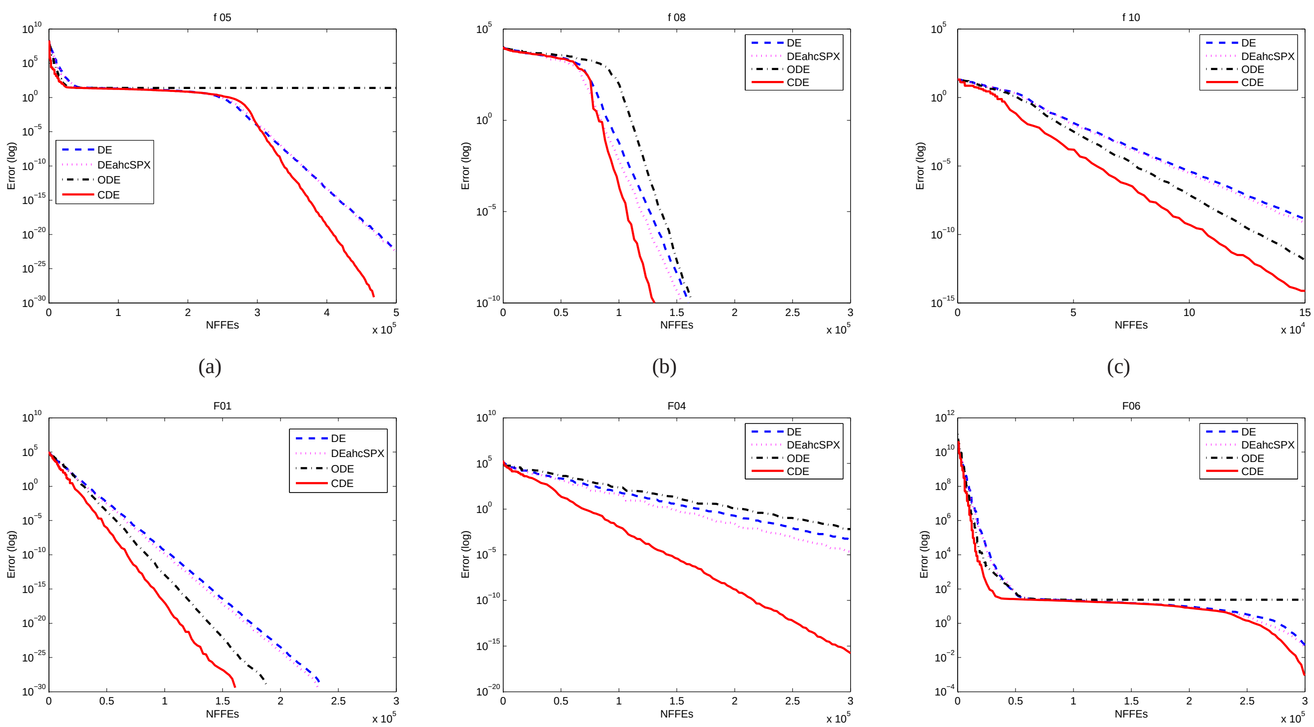
1) Choose k individuals as the initial cluster centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_k$ randomly from the current population $\{X_1, X_2, \cdots, X_{NP}\}$.

2) Assign point X_i, $i = 1, 2, \cdots, NP$ to cluster C_j, $j = 1, 2, \cdots, k$, if and only if $\| X_i - \boldsymbol{c}_j \| \leq \| X_i - \boldsymbol{c}_p \|$ $p = 1, 2, \cdots, k$, and $p \neq j$, where $\| X_i - \boldsymbol{c}_j \|$ is the distance between X_i and $\boldsymbol{c}_j$. Ties are resolved arbitrarily.



(a) (b)

Figure 3: Mean error curves of DE and CDE for functions f05 and F06. (a) f05 (D = 50). (b) F06 (D = 50).



(a) (b) (c)

(d) (e) (f)

Figure 4: Mean error curves of DE, DEahcSPX, and CDE for the selected functions. (a) f05. (b) f08. (c) f10. (d) F01. (e) F04. (f) F06.

Table 5: Comparison of DE and CDE for Different Problem Dimensionality. The Results Were Obtained After $D \times 10000$ NFFEs.

F	$D = 10$		$D = 50$	
	DE	CDE	DE	CDE
f01	3.07E-42 ± 2.94E-42 (50)	**7.21E-47 ± 1.84E-46** (50)†	2.36E-38 ± 1.25E-38 (50)	**2.71E-67 ± 3.04E-67** (50)†
f02	4.36E-23 ± 2.49E-23 (50)	**2.64E-25 ± 1.60E-25** (50)†	1.46E-21 ± 3.07E-22 (50)	**5.61E-36 ± 2.84E-36** (50)†
f03	4.12E-27 ± 4.04E-27 (50)	**5.12E-39 ± 1.19E-38** (50)†	3.12E-01 ± 9.27E-02 (0)	**1.20E-12 ± 1.48E-12** (50)†
f04	9.39E-13 ± 5.33E-13 (50)	**1.61E-15 ± 9.02E-16** (50)†	2.08E-02 ± 2.36E-03 (0)	**2.27E-08 ± 9.36E-08** (42)†
f05	2.99E-15 ± 1.39E-14 (50)	**2.05E-18 ± 1.28E-17** (50)†	**1.25E+01 ± 1.18E+00** (0)†	1.68E+01 ± 1.19E+00 (0)
f06	[9.08E+03 ± 4.85E+02] (50)	**[8.09E+03 ± 4.78E+02]** (50)†	[5.63E+04 ± 1.18E+03] (50)	**[2.89E+04 ± 1.35E+03]** (50)†
f07	**1.49E-03 ± 6.00E-04** (50)	1.50E-03 ± 6.15E-04 (50)	1.54E-02 ± 2.87E-03 (0)	**1.49E-03 ± 7.12E-04** (50)†
f08	[5.23E+04 ± 1.84E+03] (50)	**[5.06E+04 ± 1.86E+03]** (50)	[2.75E+05 ± 3.12E+03] (50)	**[2.12E+05 ± 5.54E+03]** (50)†
f09	[8.18E+04 ± 2.09E+03] (50)	**[7.88E+04 ± 2.37E+03]** (50)†	[4.34E+05 ± 5.14E+03] (50)	**[3.50E+05 ± 6.38E+03]** (50)†
f10	6.60E-16 ± 5.02E-16 (50)	**5.89E-16 ± 0.00E+00** (50)	7.69E-15 ± 0.00E+00 (50)	**5.89E-16 ± 0.00E+00** (50)†
f11	2.71E-04 ± 1.91E-03 (44)	**5.53E-06 ± 3.88E-05** (48)	[2.40E+05 ± 2.28E+03] (50)	**[1.37E+05 ± 2.19E+03]** (50)†
f12	4.71E-32 ± 0.00E+00 (50)	4.71E-32 ± 0.00E+00 (50)	9.42E-33 ± 0.00E+00 (50)	9.42E-33 (0).00E+00 (50)
f13	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)	1.35E-32 (0).00E+00 (50)
F01	[7.26E+04 ± 8.69E+02] (50)	**[6.63+04 ± 8.46E+02]** (50)†	[3.40E+05 ± 2.32E+03] (50)	**[2.56E+05 ± 2.25E+03]** (50)†
F02	2.60E-22 ± 3.64E-22 (50)	**4.26E-27 ± 1.18E-26** (50)†	5.98E-01 ± 1.46E-01 (0)	**7.24E-11 ± 1.36E-10** (50)†
F03	2.92E-11 ± 2.95E-11 (50)	**2.65E-11 ± 1.88E-10 (50)**	7.99E+07 ± 1.73E+07 (0)	**5.86E+06 ± 1.68E+06** (0)†
F04	3.43E-22 ± 5.08E-22 (50)	**5.86E-27 ± 1.75E-26** (50)†	7.89E-01 ± 2.67E-01 (0)	**9.59E-11 ± 1.75E-10** (50)†
F06	1.48E-14 ± 4.27E-14 (50)	**7.23E-18 ± 4.41E-17** (50)†	**1.41E+01 ± 1.16E+00** (0)†	1.80E+01 ± 1.10E+00 (0)
F07	1.76E-01 ± 5.82E-02 (0)	**9.52E-02 ± 7.93E-02** (9)†	1.61E-02 ± 7.46E-03 (0)	**3.36E-03 ± 6.04E-03** (7)†
F08	**2.03E+01 ± 7.93E-02** (0)	2.04E+01 ± 6.90E-02 (0)	2.11E+01 ± 3.43E-02 (0)	2.11E+01 ± 3.33E-02 (0)
F09	[8.07E+04 ± 2.15E+03] (50)	**[7.73E+04 ± 2.82E+03]** (50)†	[4.21E+05 ± 4.94E+03] (50)	**[3.50E+05 ± 6.75E+03]** (50)†

F	$D = 100$		$D = 200$	
	DE	CDE	DE	CDE
f01	6.84E-38 ± 1.83E-38 (50)	**2.27E-74 ± 2.38E-74** (50)†	1.64E-37 ± 3.85E-38 (50)	**2.41E-79 ± 1.62E-79** (50)†
f02	3.77E-21 ± 6.41E-22 (50)	**6.14E-40 ± 2.88E-40** (50)†	8.50E-21 ± 1.04E-21 (50)	**2.95E-43 ± 1.81E-43** (50)†
f03	1.42E+02 ± 1.94E+01 (0)	**7.70E-05 ± 4.16E-05** (0)†	3.90E+03 ± 4.29E+02 (0)	**7.68E+00 ± 4.04E+00** (0)†
f04	7.12E-01 ± 3.92E-02 (0)	**1.54E-01 ± 1.55E-01** (1)†	6.10E+00 ± 1.63E-01 (0)	**3.13E+00 ± 8.67E-01** (0)†
f05	**6.30E+01 ± 9.96E-01** (0)†	7.17E+01 ± 1.21E+00 (0)	**1.62E+02 ± 1.02E+00** (0)†	1.76E+02 ± 2.05E+00 (0)
f06	[1.17E+05 ± 1.79E+03] (50)	**[5.34E+04 ± 2.21E+03]** (50)†	[2.40E+05 ± 2.21E+03] (50)	**[1.01E+05 ± 3.70E+03]** (50)†
f07	3.68E-02 ± 4.14E-03 (0)	**5.04E-03 ± 1.22E-03** (50)†	8.61E-02 ± 5.06E-03 (0)	**3.42E-02 ± 5.86E-03** (0)†
f08	[5.55E+05 ± 7.12E+03] (50)	**[4.05E+05 ± 5.12E+03]** (50)†	[1.14E+06 ± 7.21E+03] (50)	**[7.89E+05 ± 8.70E+03]** (50)†
f09	[8.74E+05 ± 7.53E+03] (50)	**[6.60E+05 ± 1.63E+04]** (50)†	[1.756E+06 ± 8.38E+03] (50)	**[1.22E+06 ± 3.05E+04]** (50)†
f10	1.56E-14 ± 1.53E-15 (50)	**5.89E-16 ± 0.00E+00** (50)†	3.26E-14 ± 0.00E+00 (50)	**5.89E-16 ± 0.00E+00** (50)†
f11	[4.75E+05 ± 2.52E+03] (50)	**[2.46E+05 ± 2.62E+03]** (50)†	1.11E-16 ± 0.00E+00 (50)	**[4.59E+05 ± 2.54E+03]** (50)†
f12	4.71E-33 ± 0.00E+00 (50)	4.71E-33 ± 0.00E+00 (50)	2.36E-33 ± 0.00E+00 (50)	2.36E-33 ± 0.00E+00 (50)
f13	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)	1.35E-32 ± 0.00E+00 (50)

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

Algorithm 2 Clustering-based DE: CDE

```
Generate the initial population P randomly
Evaluate the fitness for each individual in P
Initialize the generation counter t = 1
while The halting criterion is not satisfied do
    Use DE to update the population (see lines 4 - 18 in Algorithm 1)
    if t%m == 0 then
        Randomly generate k = rndint[2, √NP]
        Adopt the one-step k-means clustering to create k offspring (the set A)
        Choose k parents (the set B) randomly from the population P
        From the combined set A ∪ B, choose k best solutions and put them in B′. Update P as P = (P\B) ∪ B′
    end if
    t = t + 1
end while
```

3) Compute new cluster centers $\boldsymbol{c}'_1, \boldsymbol{c}'_2, \cdots, \boldsymbol{c}'_k$ as follows:

$$\boldsymbol{c}'_i = \frac{1}{n_i} \sum_{X_j \in C_i} X_j, i = 1, 2, \cdots, k$$

where n_i is the number of elements belonging to cluster C_i.

4) Replace each $\boldsymbol{c}_i$ with $\boldsymbol{c}'_i$, $i = 1, 2, \cdots, k$, and evaluate these k individuals. The process is terminated.

We choose the one-step k-means clustering[1] for its simplicity and linear time complexity. Other clustering approaches can also be employed as well[2]. Note that k is generated from $[2, \sqrt{NP}]$ randomly. Here, the upper bound of the number of clusters is taken to be $\sqrt{NP}$, which is a rule of thumb used by many investigators in the literature [45].

[1]In our experiment, we also implement the multi-step k-means clustering, which needs more computational time. However, it does not bring any important advantage in our CDE approach. Due to the tight space restrictions however, we omit these results in this paper.

[2]After this paper was submitted, we extended this work by adopting the Fuzzy C-means clustering to improve DE [44]. The proposed FCDE approach can also obtain the better results than the classical DE algorithm.

Table 6: Effect of the Clustering Period on the Performance of CDE.

F	D	m = 2	m = 5	m = 15	m = 20
f01	30	**1.33E-38 ± 2.08E-38** (50)	3.44E-33 ± 2.58E-33 (50)	3.77E-26 ± 3.73E-26 (50)	1.35E-24 ± 9.05E-25 (50)
f02	30	**5.51E-29 ± 2.14E-29** (50)	2.20E-24 ± 9.15E-25 (50)	1.86E-19 ± 9.85E-20 (50)	1.67E-18 ± 6.36E-19 (50)
f03	30	4.65E-29 ± 1.26E-28 (50)	**5.20E-36 ± 1.59E-35** (50)	3.60E-33 ± 9.54E-33 (50)	1.46E-31 ± 1.64E-31 (50)
f04	30	3.49E-01 ± 3.64E-01 (2)	6.30E-17 ± 3.32E-16 (50)	**2.40E-20 ± 1.57E-20** (50)	6.63E-19 ± 5.28E-19 (50)
f05	30	3.19E-01 ± 1.09E+00 (46)	[4.71E+05 ± 1.19E+04] (50)	**[4.62E+05 ± 1.26E+04]** (50)	[4.68E+05 ± 1.18E+04] (50)
f06	30	**[1.43E+04 ± 7.60E+02]** (50)	[1.6E+04 ± 8.64E+02] (50)	[2.05E+04 ± 1.22E+03] (50)	[2.17E+04 ± 1.38E+03] (50)
f07	30	**1.11E-03 ± 4.20E-04** (50)	1.18E-03 ± 5.38E-04 (50)	1.79E-03 ± 9.16E-04 (50)	1.82E-03 ± 8.97E-04 (50)
f08	30	**[1.18E+05 ± 2.64E+03]** (50)	[1.24E+05 ± 2.32E+03] (50)	[1.35E+05 ± 1.40E+03] (50)	[1.38E+05 ± 4.55E+03] (50)
f09	30	**[1.89E+05 ± 8.88E+03]** (50)	[2.07E+05 ± 4.33E+03] (50)	[2.25E+05 ± 4.16E+03] (50)	[2.30E+05 ± 3.99E+03] (50)
f10	30	**5.89E-16 ± 0.00E+00** (50)	**5.89E-16 ± 0.00E+00** (50)	4.76E-14 ± 1.91E-14 (50)	2.89E-13 ± 1.33E-13 (50)
f11	30	**[7.16E+04 ± 1.73E+03]** (50)	[8.20E+04 ± 2.57E+03] (50)	[1.02E+05 ± 2.80E+03] (50)	[1.07E+05 ± 2.66E+03] (50)
f12	30	**1.57E-32 ± 0.00E+00** (50)	**1.57E-32 ± 0.00E+00** (50)	5.32E-28 ± 3.74E-28 (50)	2.66E-26 ± 2.42E-26 (50)
f13	30	**1.35E-32 ± 0.00E+00** (50)	**1.35E-32 ± 0.00E+00** (50)	4.36E-26 ± 4.79E-26 (50)	2.19E-24 ± 1.62E-24 (50)
f14	2	[8.46E+03 ± 3.77E+02] (50)	**[8.30E+03 ± 5.34E+02]** (50)	[8.57E+03 ± 4.20E+02] (50)	[8.44E+03 ± 4.27E+02] (50)
f15	4	3.23E-18 ± 2.67E-19 (50)	4.01E-18 ± 3.79E-19 (50)	**8.12E-19 ± 4.08E-19** (50)	**8.12E-19 ± 4.08E-19** (50)
f16	2	4.26E-14 ± 4.35E-15 (50)	4.15E-14 ± 5.27E-15 (50)	**3.52E-14 ± 1.61E-14** (50)	**3.52E-14 ± 1.61E-14** (50)
f17	2	1.11E-10 ± 3.16E-11 (50)	**8.92E-11 ± 2.21E-11** (50)	**8.92E-11 ± 2.21E-11** (50)	**8.92E-11 ± 2.21E-11** (50)
f18	2	1.32E-35 ± 0.00E+00 (50)	1.32E-35 ± 0.00E+00 (50)	1.32E-35 ± 0.00E+00 (50)	1.32E-35 ± 0.00E+00 (50)
f19	3	2.38E-40 ± 0.00E+00 (50)	2.38E-40 ± 0.00E+00 (50)	2.38E-40 ± 0.00E+00 (50)	2.38E-40 ± 0.00E+00 (50)
f20	6	**3.56E-18 ± 0.00E+00** (50)	5.68E-14 ± 5.65E-15 (50)	2.35E-11 ± 1.47E-12 (50)	8.97E-14 ± 3.31E-14 (50)
f21	4	**3.20E-08 ± 4.39E-09** (47)	2.35E-07 ± 4.42E-08 (42)	3.23E-07 ± 3.31E-08 (44)	1.25E-07 ± 1.66E-08 (45)
f22	4	5.60E-08 ± 1.64E-09 (48)	1.36E-07 ± 3.77E-08 (46)	**4.56E-08 ± 4.60E-09** (48)	1.29E-07 ± 1.33E-08 (47)
F01	30	**[1.29E+05 ± 1.74E+03]** (50)	[1.47E+05 ± 1.42E+03] (50)	[1.76E+05 ± 1.64E+03] (50)	[1.84E+05 ± 2.22E+03] (50)
F02	30	5.84E-15 ± 1.35E-14 (50)	**1.10E-17 ± 1.74E-17** (50)	1.04E-14 ± 8.01E-15 (50)	1.18E-13 ± 8.59E-14 (50)
F03	30	1.37E+06 ± 5.39E+05 (0)	1.09E+06 ± 5.19E+05 (0)	1.07E+06 ± 5.19E+05 (0)	**9.46E+05 ± 3.38E+05** (0)
F04	30	8.66E-15 ± 2.45E-14 (50)	**1.38E-17 ± 1.99E-17** (50)	1.32E-14 ± 9.83E-15 (50)	1.52E-13 ± 1.10E-13 (50)
F06	30	4.43E+00 ± 2.09E+00 (4)	1.50E-01 ± 5.81E-01 (10)	**1.43E-03 ± 4.90E-03** (31)	2.68E-03 ± 1.59E-02 (29)
F07	30	1.40E-02 ± 1.12E-02 (8)	7.73E-03 ± 9.84E-03 (24)	3.65E-03 ± 4.98E-03 (31)	**2.66E-03 ± 4.69E-03** (36)
F08	30	**2.03E+01 ± 1.28E-01** (0)	2.09E+01 ± 2.42E-01 (0)	2.09E+01 ± 5.78E-02 (0)	2.09E+01 ± 6.17E-02 (0)
F09	30	**[2.01E+05 ± 4.06E+03]** (50)	[2.08E+05 ± 3.93E+03] (50)	[2.21E+05 ± 3.53E+03] (50)	[2.25E+05 ± 3.97E+03] (50)

As mentioned above, the distance metric used in the clustering algorithm is a very important issue and for very complex problems this may lead to anomalous results. In this study the Euclidean distance is used as the distance measure. Moreover, the distance measure can be used in decision space or in objective space. Here, unless otherwise mentioned, we calculate the distance in decision space in the following experiments. In addition, the influence of the different distance metrics used in the one-step k-means method is provided in Section 5.10. Moreover, the effect of using the objective space distance measure is described in Section 5.11.

4.2. Population Update

After using one-step k-means clustering to create k offspring, the population needs to be updated by them. Deb [46] proposed a generic population-based algorithm-generator for real-parameter optimization, where the optimization task is divided into four independent plans: i) selection plan, ii) generation plan, iii) replacement plan, and iv) update plan. In lines 8 - 10 of Algorithm 2, our improvement can also be described with the population-update-algorithm proposed in [46].

- **Selection plan**: Choose k individuals from current population randomly (step 1 in the one-step k-means clustering).
- **Generation plan**: Create k offspring (the set A) using the one-step k-means clustering (steps 2 - 4 in the one-step k-means clustering).
- **Replacement plan**: Choose k solutions (the set B) from current population randomly for replacement.
- **Update plan**: From the combined set $A \cup B$, choose k best solutions and put them in B'. Update P as $P = (P \backslash B) \cup B'$.

The population-update-algorithm used in this work is similar to the G3 model in [47, 46]. In the update plan, the k best solutions are chosen from the combined set $A \cup B$, thereby the elite-preservation is ensured.

4.3. Clustering Period

In order to exploit the search space efficiently, the clustering is performed periodically in our proposed hybrid DE. It is similar to the method used in [37]. The reason for performing the clustering periodically is that DE needs time to explore the search place and form clusters. An attempt to perform the clustering very early will lead to a false identification of clusters [37]. Consequently, it is important to choose a clustering period that is large enough so that DE has time to completely form stable clusters. In our approach an additional parameter m is adopted to control the clustering period. The influence of m is given in Section 5.6.

It is worth pointing out that the clustering period used in CDE approach is similar to Damavandi's technique proposed in [37]. Compared with Damavandi's technique, our approach has two main differences: i) We don't use the deterministic method to refine the cluster centers; and ii) We propose a population update method to update the population after the clustering technique is conducted.

Table 7: Influence of the Number of Cluster Centers on the Performance of CDE.

F	*D*	$k = 2$	$k = 5$	$k = 8$	$k = 10$
f01	30	6.07E-28 ± 5.88E-28 (50)	3.73E-30 ± 2.51E-30 (50)	5.59E-31 ± 2.40E-31 (50)	**1.68E-31** ± **6.65E-32** (50)
f02	30	9.18E-21 ± 4.20E-21 (50)	3.12E-22 ± 1.17E-22 (50)	7.22E-23 ± 2.95E-23 (50)	**3.27E-23** ± **1.07E-23** (50)
f03	30	9.54E-30 ± 9.82E-30 (50)	5.49E-32 ± 6.34E-32 (50)	4.40E-33 ± 5.65E-33 (50)	**9.41E-34** ± **9.10E-34** (50)
f04	30	**1.49E-21** ± **9.73E-22** (50)	4.08E-20 ± 1.28E-19 (50)	4.74E-20 ± 1.40E-19 (50)	5.42E-21 ± 9.13E-21 (50)
f05	30	[4.63E+05 ± 1.68E+04] (50)	**[4.60E+05** ± **7.68E+03]** (50)	[4.65E+05 ± 1.4E+04] (50)	[4.60E+05 ± 1.18E+04] (50)
f06	30	[1.85E+04 ± 7.00E+02] (50)	[1.76E+04 ± 8.47E+02] (50)	**[1.74E+04** ± **1.33E+03]** (50)	[1.80E+04 ± 1.07E+03] (50)
f07	30	9.11E-04 ± 6.60E-04 (50)	1.05E-03 ± 3.18E-04 (50)	**8.56E-04** ± **4.04E-04** (50)	1.04E-03 ± 3.09E-04 (50)
f08	30	[1.23E+05 ± 3.37E+03] (50)	[1.20E+05 ± 7.34E+03] (50)	[1.18E+05 ± 3.66E+03] (50)	**[1.18E+05** ± **1.75E+03]** (50)
f09	30	**[1.91E+05** ± **4.12E+03]** (50)	[2.13E+05 ± 4.36E+03] (50)	[2.07E+05 ± 5.62E+03] (50)	[2.09E+05 ± 4.19E+03] (50)
f10	30	8.05E-15 ± 2.02E-15 (50)	**4.14E-15** ± **0.00E+00** (50)	**4.14E-15** ± **0.00E+00** (50)	**4.14E-15** ± **0.00E+00** (50)
f11	30	[9.62E+04 ± 2.03E+03] (50)	[9.04E+04 ± 1.32E+03] (50)	[8.99E+04 ± 1.99E+03] (50)	**[8.97E+04** ± **4.21E+03]** (50)
f12	30	6.97E-30 ± 3.69E-30 (50)	8.26E-32 ± 5.98E-32 (50)	3.07E-32 ± 2.17E-32 (50)	**1.98E-32** ± **5.99E-33** (50)
f13	30	6.10E-28 ± 3.35E-28 (50)	9.89E-30 ± 1.33E-29 (50)	1.29E-30 ± 8.18E-31 (50)	**6.25E-31** ± **4.84E-31** (50)
f14	2	**[8.52E+03** ± **5.57E+02]** (50)	[8.61E+03 ± 6.08E+02] (50)	[8.77E+03 ± 4.51E+02] (50)	[8.57E+03 ± 3.98E+02] (50)
f15	4	4.12E-19 ± 5.32E-19 (50)	**2.00E-19** ± **4.24E-19** (50)	2.06E-19 ± 4.34E-19 (50)	3.09E-19 ± 4.98E-19 (50)
f16	2	**9.02E-16** ± **2.52E-15** (50)	6.02E-15 ± 1.58E-14 (50)	5.00E-15 v 9.71E-15 (50)	1.30E-14 ± 2.36E-14 (50)
f17	2	**7.83E-13** ± **1.65E-12** (50)	8.87E-12 ± 2.71E-11 (50)	3.12E-10 ± 9.82E-10 (50)	1.15E-10 ± 3.39E-10 (50)
f18	2	8.78E-14 ± 4.12E-15 (50)	**3.91E-15** ± **5.05E-15** (50)	7.82E-15 ± 4.12E-15 (50)	6.84E-15 ± 4.72E-15 (50)
f19	3	**[7.64E+03** ± **2.98E+02]** (50)	[7.74E+03 ± 1.05E+02] (50)	[7.70E+03 ± 3.15E+02] (50)	[7.90E+03 ± 3.11E+02] (50)
f20	6	[1.75E+03 ± 5.29E+02] (50)	**[1.73E+04** ± **1.07E+03]** (50)	[1.77E+04 ± 9.02E+02] (50)	[1.78E+04 ± 5.36E+02] (50)
f21	4	7.28E-09 ± 1.61E-08 (45)	**5.61E-09** ± **5.91E-09** (50)	6.72E-09 ± 7.46E-09 (50)	1.06E-08 ± 2.04E-08 (43)
f22	4	4.02E-09 ± 7.51E-09 (46)	4.61E-09 ± 5.81E-09 (50)	1.69E-09 ± 1.51E-09 (50)	**1.30E-09** ± **1.87E-09** (50)
F01	30	[1.71E+05 ± 1.00E+03] (50)	[1.62E+05 ± 1.65E+03] (50)	[1.59E+05 ± 1.59E+03] (50)	**[1.57E+05** ± **1.11E+03]** (50)
F02	30	1.25E-15 ± 8.77E-16 (50)	1.71E-16 ± 1.38E-16 (50)	1.08E-16 ± 1.01E-16 (50)	**9.81E-17** ± **6.87E-17** (50)
F03	30	**7.74E+05** ± **3.57E+05** (0)	1.04E+06 ± 4.01E+05 (0)	8.44E+05 ± 4.01E+05 (0)	9.79E+05 ± 3.97E+05 (0)
F04	30	1.35E-15 ± 1.06E-15 (50)	1.84E-16 ± 1.52E-16 (50)	1.16E-16 ± 1.10E-16 (50)	**1.01E-16** ± **6.94E-17** (50)
F06	30	**2.35E-03** ± **5.97E-03** (10)	4.34E-03 ± 2.99E-02 (10)	3.59E-03 ± 8.16E-03 (9)	5.74E-03 ± 1.08E-01 (8)
F07	30	**3.45E-03** ± **6.48E-03** (34)	4.68E-03 ± 7.29E-03 (33)	5.16E-03 ± 9.64E-03 (22)	6.65E-03 ± 6.87E-03 (30)
F08	30	2.10E+01 ± 6.07E-02 (0)	2.09E+01 ± 3.96E-02 (0)	2.08E+01 ± 4.61E-02 (0)	2.11E+01 ± 7.08E-02 (0)
F09	30	[2.18E+05 ± 2.48E+03] (50)	[2.10E+05 ± 4.08E+03] (50)	[2.07E+05 ± 2.82E+03] (50)	**[2.05E+05** ± **3.31E+03]** (40)

5. Experimental Results and Analysis

In order to validate the performance of CDE, we have carried out different experiments using a test suite, which consists of 30 unconstrained single-objective benchmark functions with different characteristics chosen from the literature. All of the functions are minimization problems. f01 - f22 are chosen from [1]. Since we do not make any changes to these problems, we only briefly describe them in Table 1. More details can be found in [1]. The rest 8 functions (F01 - F04 and F06 - F09) are the new test functions provided by the CEC2005 special session [14]. Functions f01 - f13 are high-dimensional problems. Functions f01 - f05[3] are unimodal. Function f06 is the step function, which has one minimum and is discontinuous. Function f07 is a noisy quartic function, where *random* [0,1) is a uniformly distributed random variable in [0,1). Functions f08 - f13 are multimodal functions where the number of local minima increases exponentially with the problem dimension. They appear to be the most difficult class of problems for many optimization algorithms. Functions f14 - f23 are low-dimensional functions that have only a few local minima. Functions F01 - F04 are unimodal. Functions F06 - F09 are multimodal. Functions F01 and F09 are separable, and the remaining 6 functions are non-separable. The shifted and/or rotated features make these 8 functions are very difficult to solve.

5.1. Experimental Setup

For CDE, there are four control parameters. Three of them belong to the original DE algorithm, namely, population size NP, scaling factor F, and crossover probability CR. These parameters are problem dependent [49], and they are studied elsewhere [49], [24]. Another parameter is the clustering period m, which will be discussed later. For all experiments, we use the following parameters unless a change is mentioned.

- Population size: $NP = 100$;
- Scaling factor: $F = 0.5$;
- Crossover probability: $CR = 0.9$;
- Clustering period: $m = 10$;
- DE scheme: DE/rand/1/exp (different schemes will be discussed in Section 5.8);
- Value to reach: VTR = 10^{-8}, except for f07 of VTR = 10^{-2};
- Maximum NFFEs[4]: For f01, f06, f10, f12, and f13, Max_NFFEs = 150000; for f03 - f05, Max_NFFEs = 500000; for f02 and f11, Max_NFFEs = 200000; For f07 - f09, F01 - F04, and F06 - F09, Max_NFFEs = 300000; for f14, f16 - f19, f21, and f22, Max_NFFEs = 10000; for f15, Max_NFFEs = 40000; and for f20, Max_NFFEs = 20000.

[3] In fact, the generalized Rosenbrock's function f05 is a multimodal function when $D > 3$ [48].

[4] The function evaluations required to process the cluster points are added in the Maximum NFFEs.

Table 8: Comparison of DE and CDE for Different Schemes.

F	DE/rand/1/bin		DE/rand/2/exp		DE/rand/2/bin	
	DE	CDE	DE	CDE	DE	CDE
f01	2.35E-16 ± 2.04E-16	**2.35E-30 ± 2.46E-30**†	8.37E-08 ± 2.24E-08	**1.80E-16 ± 1.73E-16**†	1.18E+02 ± 3.30E+01	**3.71E-12 ± 2.73E-12**†
f02	1.33E-11 ± 6.51E-12	**4.37E-22 ± 2.38E-22**†	4.62E-07 ± 7.40E-08	**2.04E-12 ± 9.24E-13**†	1.44E+01 ± 3.99E+00	**6.98E-09 ± 2.85E-09**†
f03	4.36E-13 ± 7.77E-13	**1.18E-24 ± 2.72E-24**†	1.97E-02 ± 5.69E-03	**4.59E-26 ± 3.72E-26**†	8.18E+02 ± 2.38E+02	**2.43E-18 ± 4.20E-18**†
f04	**9.74E-02 ± 1.69E-01**†	1.62E+00 ± 1.19E+00	6.81E-03 ± 1.04E-03	**7.16E-14 ± 3.18E-14**†	6.59E+00 ± 1.02E+00	**9.86E-03 ± 1.33E-02**†
f05	2.11E-17 ± 7.49E-17	**3.25E-27 ± 1.52E-26**†	1.15E-10 ± 7.14E-11	**3.73E-28 ± 9.29E-28**†	2.88E+01 ± 1.92E+00	**1.05E-16 ± 1.65E-16**†
f06	[3.47E+04 ± 1.44E+03]	**[1.9E+04 ± 1.02E+03]**†	[5.91E+04 ± 1.53E+03]	**[2.77E+04 ± 1.53E+03]**†	1.25E+02 ± 2.79E+01	**[3.91E+04 ± 2.70E+03]**†
f07	4.57E-03 ± 1.29E-03	**1.97E-03 ± 6.49E-04**†	1.73E-02 ± 2.86E-03	**1.28E-03 ± 6.83E-04**†	7.15E-02 ± 1.76E-02	**3.33E-03 ± 1.71E-03**†
f08	6.61E+03 ± 6.56E+02	**5.66E+03 ± 9.21E+02**†	2.82E-10 ± 2.25E-10	**[1.91E+05 ± 5.29E+03]**†	**7.35E+03 ± 2.73E+02**	7.37E+03 ± 2.35E+02
f09	1.27E+02 ± 2.18E+01	**5.55E+01 ± 2.45E+01**†	1.68E-03 ± 2.13E-03	**1.13E-06 ± 1.83E-06**†	2.21E+02 ± 1.14E+01	**7.75E+01 ± 6.62E+01**†
f10	4.85E-09 ± 2.05E-09	**3.72E-15 ± 1.17E-15**†	8.99E-05 ± 1.24E-05	**3.13E-09 ± 1.09E-09**†	4.40E+00 ± 3.07E-01	**5.93E-07 ± 2.04E-07**†
f11	[1.51E+05 ± 2.75E+03]	**[8.85E+04 ± 2.39E+03]**†	4.74E-07 ± 1.63E-06	**[1.48E+05 ± 4.33E+03]**†	1.20E+00 ± 6.20E-02	**[1.92E+5 ± 3.55E+03]**†
f12	3.52E-17 ± 3.63E-17	**2.47E-32 ± 2.31E-32**†	8.11E-09 ± 2.50E-09	**3.07E-18 ± 2.78E-18**†	3.35E+01 ± 5.00E+01	**6.86E-14 ± 5.39E-14**†
f13	7.70E-15 ± 8.02E-15	**7.83E-27 ± 3.89E-26**†	1.96E-06 ± 6.18E-07	**3.77E-16 ± 2.91E-16**†	3.69E+03 ± 3.75E+03	**1.00E-11 ± 9.75E-12**†
f14	[7.97E+03 ± 2.11E+03]	**[7.86E+03 ± 2.03E+03]**	9.13E-13 ± 2.55E-12	**5.63E-13 ± 1.72E-12**	2.87E-12 ± 8.35E-12	**1.17E-12 ± 5.51E-12**
f15	**3.50E-19 ± 4.93E-19**	3.71E-19 ± 4.99E-19	9.27E-19 ± 3.12E-19	9.27E-19 ± 3.12E-19	9.89E-19 ± 2.04E-19	**8.45E-19 ± 4.00E-19**†
f16	**1.20E-15 ± 4.35E-15**	1.80E-15 ± 5.22E-15	3.92E-10 ± 4.00E-10	**2.24E-10 ± 2.58E-10**†	3.25E-10 ± 5.42E-10	**2.52E-10 ± 3.08E-10**
f17	3.75E-11 ± 1.52E-10	**1.48E-11 ± 5.12E-11**	4.46E-06 ± 9.30E-06	**3.83E-06 ± 6.87E-06**	3.07E-06 ± 7.58E-06	**2.78E-06 ± 5.38E-06**
f18	6.06E-15 ± 4.79E-15	**5.08E-15 ± 4.93E-15**	1.49E-14 ± 7.38E-15	**1.39E-14 ± 6.44E-15**	1.57E-14 ± 6.15E-15	**1.35E-14 ± 6.05E-15**
f19	1.05E-19 ± 0.00E+00	1.05E-19 ± 0.00E+00	5.56E-14 ± 4.33E-14	**2.94E-14 ± 2.75E-14**†	3.99E-14 ± 4.00E-14	**2.32E-14 ± 2.30E-14**†
f20	7.13E-03 ± 2.85E-02	**2.38E-03 ± 1.68E-02**	2.38E-03 ± 1.68E-02	**1.38E-07 ± 4.69E-07**†	1.19E-02 ± 3.60E-02	**4.98E-07 ± 1.18E-06**†
f21	**9.07E-09 ± 2.65E-08**	1.05E-08 ± 4.39E-08	2.15E-02 ± 6.39E-02	**1.07E-02 ± 2.41E-02**	3.50E-02 ± 8.73E-02	**8.27E-03 ± 1.36E-02**
f22	1.30E-09 ± 4.16E-09	**1.21E-09 ± 5.69E-09**	**1.72E-03 ± 5.75E-03**	3.44E-03 ± 1.81E-02	3.81E-03 ± 2.38E-02	**3.79E-04 ± 9.83E-04**
F01	3.33E-29 ± 7.47E-29	**2.83E-29 ± 7.08E-29**	2.62E-19 ± 1.03E-19	**[2.60E+05 ± 3.20E+03]**†	7.28E-01 ± 2.34E-01	**4.13E-26 ± 3.15E-26**†
F02	2.72E-05 ± 3.79E-05	**5.50E-10 ± 7.75E-10**†	1.32E+01 ± 2.78E+00	**1.27E-09 ± 8.50E-10**†	7.65E+03 ± 1.11E+03	**3.07E-05 ± 3.69E-05**†
F03	3.80E+05 ± 2.29E+05	**2.25E+05 ± 1.15E+05**†	2.69E+07 ± 6.04E+06	**1.16E+06 ± 6.67E+05**†	5.28E+07 ± 1.25E+07	**4.54E+05 ± 2.77E+05**†
F04	3.84E-05 ± 6.54E-05	**7.24E-10 ± 1.05E-09**†	1.74E+01 ± 5.22E+00	**1.64E-09 ± 1.11E-09**†	1.00E+04 ± 2.23E+03	**3.96E-05 ± 4.46E-05**†
F06	**8.62E-02 ± 1.70E-01**†	8.77E-01 ± 1.67E+00	9.34E-01 ± 4.49E-01	**2.08E-07 ± 3.55E-07**†	4.63E+03 ± 2.41E+03	**6.26E-01 ± 7.72E-01**†
F07	**1.48E-04 ± 1.05E-03**†	4.09E-03 ± 5.21E-03	3.60E-01 ± 5.35E-02	**1.23E-03 ± 3.73E-03**†	2.40E+00 ± 3.92E-01	**1.23E-03 ± 3.16E-03**†
F08	2.09E+01 ± 5.30E-02	2.09E+01 ± 7.38E-02	2.09E+01 ± 5.20E-02	2.09E+01 ± 5.40E-02	2.09E+01 ± 5.88E-02	2.09E+01 ± 6.02E-02
F09	1.28E+02 ± 2.58E+01	**6.99E+01 ± 1.71E+01**†	2.62E-04 ± 3.73E-04	**1.77E-07 ± 3.43E-07**†	2.13E+02 ± 1.21E+01	**1.80E+02 ± 1.18E+01**†

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

Moreover, in our experiments, each function is optimized over 50 independent runs. We also use the same set of initial random populations to evaluate different algorithms in a similar way done in [13]. All the algorithms are implemented in standard C++ and the experiments are done on a P-IV 3.0 GHz machine with 1.5 GB RAM under WIN-XP platform.

5.2. *Performance Criteria*

Four performance criteria are selected from [14] to evaluate the performance of the algorithms. These criteria are also used in [13] and described as follows.

- **Error**: The error of a solution X is defined as $f(X) - f(X^*)$, where X^* is the global optimum of the function. The minimum error is recorded when the Max_NFFEs is reached in 50 runs and the average and standard deviation of the error values are calculated.
- **NFFEs**: The number of fitness function evaluations (NFFEs) is also recorded when the VTR is reached. The average and standard deviation of the NFFEs values are calculated.
- **Number of successful runs (SR)**: The number of successful runs is recorded when the VTR is reached before the Max_NFFEs condition terminates the trial.
- **Convergence graphs**: The convergence graphs show the mean error performance of the total runs, in the respective experiments.

5.3. *Comparison between DE and CDE*

In this section, we compare our approach with the original DE algorithm to show the superiority of CDE. The parameters used for DE and CDE are the same as described in Section 5.1. All functions are conducted for 50 independent runs. Table 2 shows the best error values of DE and CDE on all test functions. The average and standard deviation of NFFEs are shown in Table 3. Note that Table 3 the Max_NFFEs for all functions is 500 000. Additionally, some representative convergence graphs of DE and CDE are shown in Fig. 1.

From Table 2 we can see that CDE is significantly better than DE on 11 functions. For eight functions (f06, f08, f09, f11, f14, f19, F01, and F09), both CDE and DE can obtain the global optimum on all 50 runs. For function F07, DE is significantly better than CDE. However, CDE can reach the VTR (i.e. $f(X) - f(X^*) < 10^{-8}$) in 32 out of 50 runs, but DE can not reach the VTR on all 50 runs. Moreover, from Table 3 it can be seen that when the Max_NFFEs is extended to 500 000, both CDE and DE can reach the VTR on all 50 runs, however the NFFEs required by CDE is nearly half of that required by DE. For the other 10 functions, there are no significant difference between CDE and DE. CDE is slightly better than DE on these functions except for f22. For the low-dimensional functions (f14 - f22), the results of CDE do not differ significantly from DE. The reason might be that these functions are easy to solve for both DE and CDE. Furthermore, the standard deviations of the best error values obtained by CDE are relative small, which means that the solution quality of CDE is stable. In addition, we can

Table 9: Comparison of DE and CDE With the Self-Adaptive Control Parameter.

F	D	SaDE	SaCDE	F	D	SaDE	SaCDE
f01	30	1.48E-18 ± 9.28E-19 (50)	**3.00E-28 ± 3.34E-28** (50)†	f16	2	[6.51E+03 ± 1.07E+02] (50)	**[6.23E+03 ± 8.95E+01]** (50)
f02	30	3.16E-15 ± 1.34E-15 (50)	**1.98E-21 ± 1.16E-21** (50)†	f17	2	[9.42E+03 ± 1.25E+02] (50)	**[7.84E+03 ± 1.04E+02]** (50)†
f03	30	4.02E-20 ± 4.89E-20 (50)	**1.94E-36 ± 4.22E-36** (50)†	f18	2	**[6.40E+03 ± 8.03E+01]** (50)	[6.44E+03 ± 5.64E+01] (50)
f04	30	8.01E-10 ± 3.49E-10 (50)	**5.54E-17 ± 1.54E-16** (50)†	f19	3	2.03E-14 ± 3.62E-15 (50)	2.03E-14 ± 2.68E-15 (50)
f05	30	7.97E-02 ± 5.64E-01 (49)	**[4.10E+05 ± 9.98E+03]** (50)†	f20	6	3.50E-02 ± 1.68E-02 (49)	**3.61E-12 ± 1.19E-13** (50)†
f06	30	[3.01E+04 ± 1.07E+03] (50)	**[1.96E+04 ± 1.13E+03]** (50)†	f21	4	2.35E-05 ± 1.06E-06 (32)	**8.53E-06 ± 9.02E-07** (38)
f07	30	6.21E-03 ± 1.42E-03 (50)	**1.66E-03 ± 6.89E-04** (50)†	f22	4	5.63E-06 ± 1.44E-07 (36)	**1.23E-06 ± 1.31E-07** (40)
f08	30	[1.53E+05 ± 3.17E+03] (50)	**[1.25E+05 ± 2.92E+03]** (50)†	F01	30	[2.21E+05 ± 2.56E+03] (50)	**[1.62E+05 ± 1.96E+03]** (50)†
f09	30	[2.49E+05 ± 5.48E+03] (50)	**[2.09E+05 ± 7.46E+03]** (50)†	F02	30	1.30E-09 ± 9.14E-10 (50)	**8.12E-17 ± 1.26E-16** (50)†
f10	30	3.08E-10 ± 8.70E-11 (50)	**7.34E-15 ± 2.06E-15** (50)†	F03	30	4.76E+05 ± 2.50E+05 (0)	**4.56E+05 ± 2.33E+05** (0)
f11	30	[1.39E+05 ± 3.66E+03] (50)	**[9.49E+04 ± 2.51E+03]** (50)†	F04	30	1.79E-09 ± 1.68E-09 (50)	**1.01E-16 ± 1.56E-16** (50)†
f12	30	4.48E-20 ± 3.10E-20 (50)	**2.12E-30 ± 2.20E-30** (50)†	F06	30	6.98E-08 ± 1.40E-07 (42)	**1.40E-12 ± 2.90E-12** (50)†
f13	30	4.83E-18 ± 3.96E-18 (50)	**1.37E-28 ± 1.27E-28** (50)†	F07	30	**3.84E-03 ± 6.98E-03** (36)†	1.14E-02 ± 1.02E-02 (13)
f14	2	**[9.33E+03 ± 4.96E+02]** (50)	[9.38E+03 ± 3.76E+02] (50)	F08	30	2.09E+01 ± 5.87E-02 (0)	**2.08E+01 ± 1.95E-02** (0)†
f15	4	[3.51E+04 ± 2.89E+02] (50)	**[2.26E+04 ± 1.98E+02]** (50)†	F09	30	[2.37E+05 ± 5.75E+03] (50)	**[2.06E+05 ± 5.06E+03]** (50)†

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.

see that the computational time of CDE is only slightly higher than that of DE for the majority of functions.

Considering the NFFEs required by CDE and DE to reach the VTR, Table 3 clearly shows that for all high dimensional functions CDE is significantly better than DE, except for F03 and F08, in which both CDE and DE fail to reach the VTR after 500 000 NFFEs. For the low dimensional functions (f14 - f22), there is no significant difference for CDE and DE.

From Fig. 1 it can be seen that for the high dimensional functions CDE converges faster than DE. However, for the low dimensional functions the difference is not significant.

In general, the overall performance of CDE is better than that of DE for the high dimensional functions. For the low dimensional functions, since these functions are simple, both DE and CDE are able to solve these functions, and CDE is slightly better than DE. Moreover, our proposed CDE can accelerate the original DE algorithm and reduce the NFFEs to reach the VTR for high-dimensional functions significantly.

5.4. *Influence of Population Size*

In [49], the authors concluded that the performance of DE is sensitive to the choice of the population size. Increasing the population size will increase the diversity of possible movements, promoting the exploration of the search space. However, the probability to find the correct search direction decreases considerably [50]. The influence of population size is investigated in this section. For both CDE and DE, all the parameter settings are the same as mentioned in Section 5.1 only except for $NP = 50$ and $NP = 200$. The results for $NP = 50$ and $NP = 200$ are shown in Table 4.

For $NP = 50$, CDE is significantly better than DE in 16 functions. However, for three functions (f04, F06, and F07), DE is significantly better than CDE. For the rest 11 functions, there are no significant difference for CDE and DE. For 12 functions, CDE can obtain the global optimum on all 50 runs.

For $NP = 200$, Table 4 shows that for all the high dimensional functions CDE is significantly better than DE, except for F08 and F09, where both CDE and DE obtain similar results. For the low dimensional functions there are no significant difference between CDE and DE.

In summary, according to the results of Table 2 - Table 4, we can conclude that i) CDE and DE can provide better results using a smaller population size for some functions. ii) For the majority of functions, CDE is always better than DE. iii) CDE provides a faster convergence rate and greater robustness for different population size compared with DE.

5.5. *Scalability Study*

In order to study the effect of the problem dimensionality on the performance of CDE, a scalability study is conducted for the scalable functions in the test suit. For functions f01 - f13, $D = 10, 50, 100, 200$. For F01 - F04 and F06 - F09, $D = 10, 50$, since these functions are defined up to $D = 50$ dimensions [14]. The results are recorded after $D \times 10000$ NFFEs. All other control parameters are unchanged from their values mentioned in Section 5.1. The results of CDE and DE are given in Table 5, and some representative convergence graphs are shown in Fig. 2.

From Table 5, the results indicate that CDE outperforms DE for the majority of the test scalable functions at every dimension. In addition, the higher the problem dimensionality, the better the performance of CDE obtained. For example, for $D = 100$ and $D = 200$, CDE is significantly better than DE for 10 out of 13 functions. Both CDE and DE provide the same results for functions f12 and f13. Only for one function f05, DE is better than CDE. Moreover, Fig. 2 shows that CDE is able to accelerate the convergence rate in general. However, by carefully looking at the results in Table 5, we can see that for $D = 50$, DE is significantly better than CDE for functions f05 and F06. For $D = 100$ and $D = 200$, DE is significantly better than CDE for function f05. With respect to f05 and F06, they are the generalized Rosenbrock's function and shifted Rosenbrock's function, respectively. CDE converges to the local optima for the two functions. This might be caused by the fact that the small clustering period of $m = 10$ is used, which leads to CDE exploring the search place insufficiently. The influence

Table 10: Influence of the Different Distance Measure Used in the One-step K-Means in the Decision Space.

F	*D*	DE	CDE	CDE_cos	CDE_man
f01	30	2.01E-17 ± 1.14E-17 (50)	1.07E-28 ± 7.65E-29 (50)‡	**1.68E-31 ± 1.40E-31** (50)‡	2.22E-30 ± 1.35E-30 (50)‡
f02	30	3.86E-14 ± 9.28E-15 (50)	4.21E-21 ± 1.85E-21 (50)‡	**4.83E-23 ± 1.81E-23** (50)‡	4.15E-22 ± 2.22E-22 (50)‡
f03	30	5.04E-11 ± 2.46E-11 (50)	**1.64E-34 ± 9.18E-34** (50)‡	8.85E-32 ± 1.38E-31 (50)‡	1.11E-31 ± 2.15E-31 (50)‡
f04	30	8.81E-08 ± 2.39E-08 (39)	6.48E-22 ± 1.18E-21 (50)‡	6.20E-18 ± 4.38E-17 (50)‡	**1.88E-22 ± 5.75E-22** (50)‡
f05	30	5.15E-22 ± 1.21E-21 (50)	[4.61+E05 ± 1.28E+04] (50)‡	[4.06E+05 ± 9.85E+03] (50)‡	**[3.96E+05 ± 9.84E+03]** (50)‡
f06	30	[3.30E+04 ± 1.12E+03] (50)	[1.87E+04 ± 1.05E+03] (50)‡	**[1.75E+04 ± 6.57E+02]** (50)‡	[1.75E+04 ± 8.15E+02] (50)‡
f07	30	7.84E-03 ± 1.74E-03 (50)	**1.27E-03 ± 7.37E-04** (50)‡	2.16E-03 ± 8.13E-04 (50)‡	2.63E-03 ± 1.09E-03 (50)‡
f08	30	[1.59E+05 ± 1.37E+03] (50)	**[1.30E+05 ± 2.40E+03]** (50)‡	[1.39E+05 ± 6.59E+03] (50)‡	[1.38E+05 ± 6.09E+03] (50)‡
f09	30	[2.56E+05 ± 4.18E+03] (50)	[2.17E+05 ± 4.92E+03] (50)‡	[2.09E+05 ± 1.42E+03] (50)‡	**[2.06E+05 ± 1.62E+04]** (50)‡
f10	30	1.21E-09 ± 3.14E-10 (50)	5.28E-15 ± 1.67E-15 (50)‡	1.51E-15 ± 1.57E-15 (50)‡	**1.37E-15 ± 1.49E-15** (50)‡
f11	30	[1.49E+05 ± 3.35E+03] (50)	[9.42E+04 ± 2.46E+03] (50)‡	**[8.38E+04 ± 2.25E+03]** (50)‡	[8.49E+04 ± 2.51E+03] (50)‡
f12	30	1.46E-18 ± 7.33E-19 (50)	1.79E-30 ± 1.50E-30 (50)‡	**1.57E-32 ± 0.00E+00** (50)‡	**1.57E-32 ± 0.00E+00** (50)‡
f13	30	1.59E-16 ± 6.79E-17 (50)	9.42E-29 ± 8.40E-29 (50)‡	**1.35E-32 ± 0.00E+00** (50)‡	1.37E-32 ± 1.39E-33 (50)‡
f14	2	**[8.30E+03 ± 4.47E+02]** (50)	[8.40E+03 ± 4.06E+02] (50)	2.60E-15 ± 1.70E-14 (50)	5.78E-14 ± 3.91E-13 (50)
f15	4	1.85E-19 ± 4.00E-19 (50)	**1.03E-19 ± 3.12E-19** (50)‡	2.47E-19 ± 4.44E-19 (50)	2.27E-19 ± 4.31E-19 (50)
f16	2	1.28E-14 ± 4.71E-14 (50)	7.99E-16 ± 4.44E-15 (50)‡	**4.00E-16 ± 1.98E-15** (50)‡	1.60E-15 ± 4.22E-15 (50)‡
f17	2	1.74E-11 ± 6.58E-11 (50)	**4.33E-13 ± 1.46E-12** (50)‡	7.41E-12 ± 3.84E-11 (50)‡	4.10E-11 ± 1.83E-10 (50)
f18	2	7.08E-15 ± 1.43E-14 (50)	**4.69E-15 ± 4.93E-15** (50)‡	8.02E-15 ± 4.30E-15 (50)	7.43E-15 ± 4.68E-15 (50)
f19	3	[8.90E+03 ± 3.16E+02] (50)	[8.69E+03 ± 2.68E+02] (50)‡	**[7.73E+03 ± 3.18E+02]** (50)‡	[7.84E+03 ± 2.75E+02] (50)‡
f20	6	2.92E-12 ± 2.04E-11 (50)	1.40E-14 ± 7.04E-14 (50)‡	**[1.75E+04 ± 8.65E+02]** (50)‡	2.92E-14 ± 1.99E-13 (50)‡
f21	4	1.91E-08 ± 3.75E-08 (30)	**1.67E-08 ± 3.90E-08** (37)‡	2.78E-08 ± 6.89E-08 (28)	3.49E-08 ± 1.44E-07 (35)
f22	4	4.98E-09 ± 2.54E-08 (48)	5.60E-09 ± 1.59E-08 (45)	8.66E-09 ± 5.03E-08 (48)	**1.37E-09 ± 1.96E-09** (50)‡
F01	30	[2.35E+05 ± 1.78E+03] (50)	[1.65E+05 ± 1.98E+03] (50)‡	**[1.41E+05 ± 1.47E+03]** (50)‡	[1.43E+05 ± 1.59E+03] (50)‡
F02	30	3.12E-04 ± 1.28E-04 (0)	3.60E-16 ± 3.97E-16 (50)‡	3.61E-19 ± 1.06E-18 (50)‡	**3.17E-19 ± 4.01E-19** (50)‡
F03	30	1.03E+06 ± 5.57E+05 (0)	8.93E+05 ± 3.06E+05 (0)‡	**4.68E+05 ± 2.41E+05** (0)‡	4.79E+05 ± 2.33E+05 (0)‡
F04	30	4.11E-04 ± 1.94E-04 (0)	4.52E-16 ± 4.93E-16 (50)‡	5.43E-19 ± 1.83E-18 (50)‡	**4.23E-19 ± 5.45E-19** (50)‡
F06	30	8.90E-03 ± 1.93E-02 (0)	**4.13E-03 ± 1.45E-02** (7)‡	3.99E-01 ± 1.21E+00 (45)	2.39E-01 ± 9.56E-01 (47)
F07	30	**2.03E-05 ± 1.57E-05** (0)	4.14E-03 ± 6.28E-03 (32)	1.07E-02 ± 1.03E-02 (14)	1.39E-02 ± 1.08E-02 (8)
F08	30	**2.09E+01 ± 6.10E-02** (0)	2.09E+01 ± 8.21E-02 (0)	2.11E+01 ± 5.28E-02 (0)	2.11E+01 ± 5.75E-02 (0)
F09	30	[2.51E+05 ± 4.67E+03] (50)	**[2.16E+05 ± 4.16E+03]** (50)‡	[2.40E+05 ± 1.87E+04] (50)‡	[2.32E+05 ± 1.54E+04] (50)‡

‡ It indicates DE is worse than its competitor.

of the clustering period for the two functions is discussed in the following Section 5.6.

5.6. *Effect of Clustering Period*

In our proposed CDE, only one additional parameter m is included. This parameter makes the one-step k-means clustering perform periodically. In order to investigate the effect of m on the performance of CDE, a set of experiments has been performed. All other parameters are kept unchanged as mentioned in Section 5.1, and we only modify the clustering period parameter m as follows: $m = 2, 5, 15, 20$. For each m, we perform 50 independent runs per test functions. The results are presented in Table 6. From Table 6, it can be seen that a lower clustering period can achieve faster convergence rate, however this may lead to becoming trapped in a local optimum, e.g. f04, f05, etc. The higher clustering period makes the algorithm more robust but lowers the convergence rate.

As mentioned in Section 5.5, CDE is significantly worse than DE for the higher dimensional Rosenbrock's functions (f05 and F06). This means that CDE with $m = 10$ misleads the search on these functions. Here, we perform a preliminary experiments to study the effect of m on the performance of CDE for the two functions at $D = 50$. In these experiments the maximum NFFEs of Max_NFFEs = $20000 \times D$ is used to clearly show the effect of different m values. All the remaining parameters are kept unchanged. The convergence graphs of these functions are shown in Fig. 3. From Fig. 3, we can see that CDE is able to provide better results for higher clustering period ($m \geq 15$). The reason might be that higher clustering period makes CDE explore the search space sufficiently for the higher dimensional problems, and hence the one-step k-means clustering can exploit information efficiently.

According to the previous experiments given in Section 5.3 - 5.6, the parameter m working in the interval $[5, 40]$ could be more reliable for unknown optimization problems. Higher clustering period is reasonable for higher dimensional problems. However, the effect should be studied in more detail by varying the population size and the problem dimensionality which is beyond the scope of this work. We leave this task for a future study.

5.7. *Influence of the Number of Cluster Centers*

In our proposed CDE approach, the number of cluster centers k is generated randomly from $[2, \sqrt{NP}]$. In this section we perform additional experiment to show the influence of the number of cluster centers. k is set to 2, 5, 8, and 10 to replace the random number. The results are shown in Table 7. From Table 7, it can be seen that for the majority of functions no significant difference can be found. It indicates that k has a small effect on the performance of CDE.

5.8. *Effect of Different Schemes*

In DE there are more than ten different schemes [9, 10], and [11]. In [51], Mezura-Montes *et al.* presented an empirical comparison of some DE schemes to identify which one of them is more suitable to solve an optimization problem. Different schemes are suitable for different problems. In this section, we conduct a set of experiments to show the performance of CDE

Table 11: Influence of the Distance Measure of Clustering in Decision Space and Objective Space.

F	*D*	DE	CDE	CDE_obj
f01	30	2.01E-17 ± 1.14E-17 (50)	1.07E-28 ± 7.65E-29 (50)	**4.33E-34 ± 3.07E-34** (50)
f02	30	3.86E-14 ± 9.28E-15 (50)	4.21E-21 ± 1.85E-21 (50)	**3.93E-25 ± 2.17E-25** (50)
f03	30	5.04E-11 ± 2.46E-11 (50)	1.64E-34 ± 9.18E-34 (50)	**4.98E-35 ± 6.38E-35** (50)
f04	30	8.81E-08 ± 2.39E-08 (0)	**6.48E-22 ± 1.18E-21** (50)	8.68E-21 ± 5.58E-20 (50)
f05	30	5.15E-22 ± 1.21E-21 (50)	**[4.61+E05 ± 1.28E+04]** (50)	[4.71E+05 ± 1.41E+04] (50)
f06	30	[3.30E+04 ± 1.12E+03] (50)	[1.87E+04 ± 1.05E+03] (50)	**[1.60E+04 ± 8.52E+02]** (50)
f07	30	7.84E-03 ± 1.74E-03 (50)	1.27E-03 ± 7.37E-04 (50)	**8.62E-04 ± 3.30E-04** (50)
f08	30	[1.59E+05 ± 1.37E+03] (50)	[1.30E+05 ± 2.40E+03] (50)	**[1.20E+05 ± 2.70E+03]** (50)
f09	30	[2.56E+05 ± 4.18E+03] (50)	[2.17E+05 ± 4.92E+03] (50)	**[1.94E+05 ± 5.67E+03]** (50)
f10	30	1.21E-09 ± 3.14E-10 (50)	5.28E-15 ± 1.67E-15 (50)	**5.89E-16 ± 0.00E+00** (50)
f11	30	[1.49E+05 ± 3.35E+03] (50)	[9.42E+04 ± 2.46E+03] (50)	**[7.98E+04 ± 2.07E+03]** (50)
f12	30	1.46E-18 ± 7.33E-19 (50)	1.79E-30 ± 1.50E-30 (50)	**1.57E-32 ± 0.00E+00** (50)
f13	30	1.59E-16 ± 6.79E-17 (50)	9.42E-29 ± 8.40E-29 (50)	**1.35E-32 ± 0.00E+00** (50)
f14	2	[8.30E+03 ± 4.47E+02] (50)	[8.40E+03 ± 4.06E+02] (50)	**[8.18E+03 ± 4.49E+02]** (50)
f15	4	1.85E-19 ± 4.00E-19 (50)	**1.03E-19 ± 3.12E-19** (50)	**1.03E-19 ± 3.12E-19** (50)
f16	2	1.28E-14 ± 4.71E-14 (50)	**7.99E-16 ± 4.44E-15** (50)	2.20E-15 ± 5.81E-15 (50)
f17	2	1.74E-11 ± 6.58E-11 (50)	**4.33E-13 ± 1.46E-12** (50)	1.46E-11 ± 4.57E-11 (50)
f18	2	7.08E-15 ± 1.43E-14 (50)	4.69E-15 ± 4.93E-15 (50)	**3.71E-15 ± 4.79E-15** (50)
f19	3	[8.90E+03 ± 3.16E+02] (50)	[8.69E+03 ± 2.68E+02] (50)	**[8.21E+03 ± 2.50E+02]** (50)
f20	6	2.92E-12 ± 2.04E-11 (50)	1.40E-14 ± 7.04E-14 (50)	**[1.63E+04 ± 1.06E+03]** (50)
f21	4	1.91E-08 ± 3.75E-08 (30)	1.67E-08 ± 3.90E-08 (37)	**1.59E-08 ± 9.00E-08** (47)
f22	4	**4.98E-09 ± 2.54E-08** (48)	5.60E-09 ± 1.59E-08 (45)	6.75E-08 ± 4.71E-07 (44)
F01	30	[2.35E+05 ± 1.78E+03] (50)	[1.65E+05 ± 1.98E+03] (50)	**[1.42E+05 ± 1.79+E03]** (50)
F02	30	3.12E-04 ± 1.28E-04 (0)	3.60E-16 ± 3.97E-16 (50)	**2.40E-18 ± 3.88E-18** (50)
F03	30	1.03E+06 ± 5.57E+05 (0)	8.93E+05 ± 3.06E+05 (0)	**6.16E+05 ± 2.19E+05** (50)
F04	30	4.11E-04 ± 1.94E-04 (0)	4.52E-16 ± 4.93E-16 (50)	**3.08E-18 ± 4.65E-18** (50)
F06	30	8.90E-03 ± 1.93E-02 (0)	**4.13E-03 ± 1.45E-02** (7)	6.84E-02 ± 1.53E-02 (3)
F07	30	**2.03E-05 ± 1.57E-05** (0)	4.14E-03 ± 6.28E-03 (32)	1.29E-02 ± 9.03E-03 (4)
F08	30	2.09E+01 ± 6.10E-02 (0)	2.09E+01 ± 8.21E-02 (0)	**2.06E+01 ± 2.90E-01** (0)
F09	30	[2.51E+05 ± 4.67E+03] (50)	[2.16E+05 ± 4.16E+03] (50)	**[2.02E+05 ± 4.16+E03]** (50)

for different schemes. Three schemes, namely, DE/rand/1/bin, DE/rand/2/exp, and DE/ran/2/bin are chosen in these experiments. All remaining parameters are the same as mentioned in Section 5.1. The dimension per function is the same as shown in Table 2. Table 8 gives the results of DE and CDE for the three schemes.

According to Table 8, we can see that for the majority of the test problems CDE is significantly better than DE, especially compared to the DE schemes with two difference vectors. Generally speaking, the overall results of Table 2, 3, and 8 substantiate our claim that for the majority of the test problems the proposed CDE is able to improve the performance of DE for different schemes.

5.9. Influence of Self-adaptive Parameter Control

As mentioned above, the choice of the control parameters F and CR is sensitive for different problems [49]. In order to show that CDE can also improve the self-adaptive DE, in this section, we adopt the self-adaptive control parameter proposed in [8] to replace the fixed $F = 0.5$ and $CR = 0.9$ in the previous experiments. All other parameter settings are kept unchanged. The results for the self-adaptive DE (SaDE) and self-adaptive CDE (SaCDE) are given in Table 9. The results indicate that SaCDE is significantly better than SaDE on 22 out of 30 functions. Only for one function F07, SaDE is significantly better than SaCDE. For the other 7 functions, there is no significant difference for both SaCDE and SaDE. In general, integration of the one-step k-means clustering can improve the performance of SaDE.

5.10. Influence of Different Distance Measure

In order to test the influence of different distance measure used in the one-step k-means on the performance of CDE, a set of experiments is performed here. The results are shown in Table 10, where CDE_cos means the cosine distance measure is used in CDE, and CDE_man denotes the Manhattan distance measure is used. All parameters are kept unchanged. In addition, all experiments are conducted for 50 independent runs for each function. It can be seen from Table 10 that the three CDE approaches outperformed DE on the majority of the test functions. On 26 functions, CDE is better than DE. CDE_cos outperforms DE on 22 functions. CDE_man is also better than DE on 22 out of 30 functions. Generally speaking, the improvement of CDE is not significantly influenced by the different distance measure used in the one-step k-means clustering.

5.11. Effect of Distance in Objective Space

In our previous experiments, the Euclidean distance of the clustering is calculated in the decision space. The distance can also be calculated in the objective space, we only need to modify step 2 of the one-step k-means clustering described in Section 4.1 as "Assign point X_i, $i = 1, 2, \cdots, NP$ to cluster C_j, $j = 1, 2, \cdots, k$, if and only if $\| f(X_i) - f(\boldsymbol{c}_j) \| \leq \| f(X_i) - f(\boldsymbol{c}_p) \|$, $p = 1, 2, \cdots, k$, and $p \neq j$, where $f(X)$ is the fitness of solution X, and $\| f(X_i) - f(\boldsymbol{c}_j) \|$ is the distance between $f(X_i)$ and $f(\boldsymbol{c}_j)$. Ties are resolved arbitrarily.". The CDE using the distance in the objective space is named CDE_obj. All other parameters are the same as mentioned in Section 5.1. Table 11 shows the results of DE, CDE, and CDE_obj for all test functions. From Table 11, we can see that both CDE and CDE_obj outperform

Table 12: Comparison on the Error values between our approach and Wang's approach [23].

F	D	Wang's approach [23]			CDE			Wang-CDE
		NFFEs	Mean	Std Dev	NFFEs	Mean	Std Dev	*t-test*
f01	30	150 000	1.00E-06	2.00E-06	150 000	**1.07E-28**	7.65E-29	3.54†
f02	30	200 000	**0.00E+00**	0.00E+00	200 000	4.21E-21	1.85E-21	-16.09†
f03	30	500 000	1.10E-05	1.40E-05	500 000	**1.64E-34**	9.18E-34	5.56†
f04	30	500 000	7.50E-05	7.50E-05	500 000	**6.48E-22**	1.18E-21	7.07†
f05	30	2 000 000	1.83E+00	6.83E+00	**500 000**	**0.00E+00**	0.00E+00	1.89†
f06	30	150 000	0.00E+00	0.00E+00	150 000	0.00E+00	0.00E+00	0
f07	30	300 000	4.60E-04	3.60E-04	300 000	**1.30E-03**	7.37E-04	-7.24†
f08	30	900 000	1.25E+03[a]	5.03E+02	**300 000**	**0.00E+00**	0.00E+00	17.57†
f09	30	500 000	1.80E-05	2.30E-05	**300 000**	**0.00E+00**	0.00E+00	5.53†
f10	30	150 000	4.70E-05	4.50E-05	150 000	**5.28E-15**	1.67E-15	27.24†
f11	30	200 000	0.00E+00	0.00E+00	200 000	0.00E+00	0.00E+00	0
f12	30	150 000	**0.00E+00**	0.00E+00	150 000	1.79E-30	1.50E-30	-8.44†
f13	30	150 000	**0.00E+00**	0.00E+00	150 000	9.42E-29	8.40E-29	-7.93†

† The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by two-tailed test.
[a] indicates the error value is used based on the reported results.

DE on the majority of the test functions. In addition, CDE_obj is slightly better than CDE on 22 functions. However, the effect should also be studied in more detail by varying the population size and the problem dimensionality. We leave this task for our future work. In summary, the one-step k-means clustering with the distance measure in both decision space and objective space can enhance the performance of DE.

5.12. *Comparison with Wang's Approach [23]*

Since both our approach and Wang's approach [23] improved the original DE algorithm with the clustering algorithm, in this section, we compare the results between our approach and Wang's method. The results are shown in Table 12 on functions f01 - f13. From Table 12 we can see that on 6 out of 13 functions our approach is significantly better than Wang's approach. On 4 functions (f02, f07, f12, and f13), our approach is significantly outperformed by Wang's method. However, for functions f02, f12, and f13, our approach obtains good mean best results and approximates the global optimum in all 50 runs for the three functions. For the rest two functions (f06 and f11) both approaches can obtain the global optimum in all 50 runs. In addition, for three functions (f05, f08, and f09) our approach provides better mean best values in less NFFEs. In general, we can conclude that our approach obtains better results than Wang's approach [23] in terms of the quality of the final results.

5.13. *Comparison with Other DE Hybrids*

Finally, we make a comparison with other DE. Since there are many variants of DE, we only compare our approach with DEahcSPX proposed in [13] and ODE proposed in [32]. In DEahcSPX, a crossover-based adaptive local search operation to accelerate DE. The authors concluded that DEahcSPX outperforms the original DE algorithm in items of convergence rate in all experimental studies. In ODE, the opposition-based learning is used for the population initialization and generation jumping. In this section, we compare our proposed CDE with DE, DEahcSPX and ODE. All the parameter settings are the same as mentioned in Section 5.1. For DEahcSPX, the number of parents in SPX sets to be $n_p = 3$ [13]. For ODE, the jump rate $J_r = 0.3$ [32]. The results are given in Table 13. Some selected representative convergence graphs are shown in Fig. 4. From Table 13 and Fig. 4, it can be seen that i) CDE is better than DE, DEahcSPX and ODE on 19 out of 30 functions. ii) For three functions (f22, F03, and F07), DEahcSPX can obtain better results compared with DE, ODE and CDE. iii) ODE is able to get better results for 7 functions (f04, f14, and f16 - f20), however it may lead to be premature, e.g. f05 and F06. iv) CDE is able to provide the highest overall number of successful runs. And v) CDE can converge faster for the majority of the test functions compared with DE, DEahcSPX, and ODE.

6. Conclusion and Future Work

In order to make the DE algorithm more effective and more efficient, the one-step k-means clustering is integrated into DE in this paper. The hybrid clustering-based DE (CDE) can balance the exploration and the exploitation in the evolutionary process. It is worth noting that our proposed CDE is also simple and ease to use. CDE adds only one parameter, the clustering period m, to the original DE algorithm.

To evaluate the performance of our presented approach, 30 unconstrained single-objective benchmark functions with different characteristics are chosen from the literature. A comprehensive set of experiments is conducted in this paper, to study i) the effect of the one-step k-means clustering on DE; ii) the influence of the population size; iii) the effect of the problem dimensionality; iv) the effect of the clustering period m; v) the influence of the number of cluster centers; vi) the mutation schemes; vii) the influence of the self-adaptive parameter control on DE; viii) the effect of different distance measure used in the one-step k-means clustering; ix) the distance measure of clustering in decision space and objective space; and x) comparison with the AHCXLS-based DE (DEahcSPX) and the opposition-based DE (ODE). In addition, four criteria are selected for evaluating the performance of the algorithms.

Table 13: Comparison of DE, DEahcSPX, ODE, and CDE on All Test Functions.

F	*D*	DE	DEahcSPX	ODE	CDE
f01	30	2.01E-17 ± 1.14E-17 (50)	6.82E-18 ± 2.68E-18 (50)	5.61E-24 ± 5.24E-24 (50)	**1.07E-28** ± **7.65E-29** (50)
f02	30	3.86E-14 ± 9.28E-15 (50)	2.59E-14 ± 6.34E-15 (50)	6.73E-13 ± 2.17E-13 (50)	**4.21E-21** ± **1.85E-21** (50)
f03	30	5.04E-11 ± 2.46E-11 (50)	2.65E-12 ± 1.57E-12 (50)	2.95E-08 ± 2.19E-08 (0)	**1.64E-34** ± **9.18E-34** (50)
f04	30	8.81E-08 ± 2.39E-08 (0)	2.19E-08 ± 5.17E-09 (0)	**2.08E-37** ± **2.77E-37** (50)	6.48E-22 ± 1.18E-21 (50)
f05	30	5.15E-22 ± 1.21E-21 (50)	3.69E-22 ± 8.56E-22 (50)	2.37E+01 ± 1.50E+00 (0)	**[4.61+E05** ± **1.28E+04]** (50)
f06	30	[3.30E+04 ± 1.12E+03] (50)	[3.24E+04 ± 1.13E+03] (50)	[2.48E+04 ± 8.83E+02] (50)	**[1.87E+04** ± **1.05E+03]** (50)
f07	30	7.84E-03 ± 1.74E-03 (50)	5.84E-03 ± 1.54E-03 (50)	2.04E-03 ± 6.04E-04 (50)	**1.27E-03** ± **7.37E-04** (50)
f08	30	[1.59E+05 ± 1.37E+03] (50)	[1.59E+05 ± 1.13E+03] (50)	[1.53E+05 ± 5.98E+03] (50)	**[1.30E+05** ± **2.40E+03]** (50)
f09	30	[2.56E+05 ± 4.18E+03] (50)	[2.56E+05 ± 6.29E+03] (50)	[2.32E+05 ± 1.17E+04] (50)	**[2.17E+05** ± **4.92E+03]** (50)
f10	30	1.21E-09 ± 3.14E-10 (50)	7.16E-10 ± 1.74E-10 (50)	9.50E-13 ± 3.34E-13 (50)	**5.28E-15** ± **1.67E-15** (50)
f11	30	[1.49E+05 ± 3.35E+03] (50)	[1.45E+05 ± 3.28E+03] (50)	[1.20+E05 ± 6.81E+03] (50)	**[9.42E+04** ± **2.46E+03]** (50)
f12	30	1.46E-18 ± 7.33E-19 (50)	4.15E-19 ± 2.60E-19 (50)	8.14E-25 ± 8.63E-25 (50)	**1.79E-30** ± **1.50E-30** (50)
f13	30	1.59E-16 ± 6.79E-17 (50)	6.04E-17 ± 2.78E-17 (50)	5.99E-21 ± 9.39E-21 (50)	**9.42E-29** ± **8.40E-29** (50)
f14	2	[8.30E+03 ± 4.47E+02] (50)	[8.14E+03 ± 6.37E+02] (50)	**[7.73E+03** ± **5.92E+02]** (50)	[8.40E+03 ± 4.06E+02] (50)
f15	4	1.85E-19 ± 4.00E-19 (50)	3.50E-19 ± 4.93E-19 (50)	7.21E-19 ± 4.98E-19 (50)	**1.03E-19** ± **3.12E-19** (50)
f16	2	1.28E-14 ± 4.71E-14 (50)	2.60E-15 ± 9.24E-15 (50)	**[8.20E+03** ± **4.89E+02]** (50)	7.99E-16 ± 4.44E-15 (50)
f17	2	1.74E-11 ± 6.58E-11 (50)	3.49E-11 ± 1.43E-10 (50)	**1.49E-13** ± **2.72E-13** (50)	4.33E-13 ± 1.46E-12 (50)
f18	2	7.08E-15 ± 1.43E-14 (50)	8.30E-15 ± 1.95E-14 (50)	**2.93E-15** ± **4.72E-15** (50)	4.69E-15 ± 4.93E-15 (50)
f19	3	[8.90E+03 ± 3.16E+02] (50)	[8.76E+03 ± 3.25E+02] (50)	**[7.88E+03** ± **2.53E+02]** (50)	[8.69E+03 ± 2.68E+02] (50)
f20	6	2.92E-12 ± 2.04E-11 (50)	9.40E-14 ± 3.23E-13 (50)	**5.95E-15** ± **1.26E-14** (50)	1.40E-14 ± 7.04E-14 (50)
f21	4	1.91E-08 ± 3.75E-08 (30)	1.90E-08 ± 5.48E-08 (37)	2.50E-06 ± 6.43E-06 (26)	**1.67E-08** ± **3.90E-08** (37)
f22	4	4.98E-09 ± 2.54E-08 (48)	**2.16E-09** ± **3.29E-09** (48)	7.21E-08 ± 9.99E-08 (28)	5.60E-09 ± 1.59E-08 (45)
F01	30	[2.35E+05 ± 1.78E+03] (50)	[2.32E+05 ± 1.89E+03] (50)	[1.86E+05 ± 2.72E+04] (50)	**[1.65E+05** ± **1.98E+03]** (50)
F02	30	3.12E-04 ± 1.28E-04 (0)	3.96E-05 ± 1.64E-05 (0)	5.52E-03 ± 3.64E-03 (0)	**3.60E-16** ± **3.97E-16** (50)
F03	30	1.03E+06 ± 5.57E+05 (0)	**7.35E+05** ± **3.70E+05** (0)	2.57E+06 ± 1.04E+06 (0)	8.93E+05 ± 3.06E+05 (0)
F04	30	4.11E-04 ± 1.94E-04 (0)	5.25E-05 ± 2.62E-05 (0)	6.89E-03 ± 4.52E-03 (0)	**4.52E-16** ± **4.93E-16** (50)
F06	30	8.90E-03 ± 1.93E-02 (0)	2.76E-02 ± 5.64E-02 (0)	1.89E+01 ± 9.97E+00 (0)	**4.13E-03** ± **1.45E-02** (7)
F07	30	2.03E-05 ± 1.57E-05 (0)	**1.85E-06** ± **1.63E-06** (0)	3.94E-03 ± 9.50E-03 (42)	4.14E-03 ± 6.28E-03 (32)
F08	30	2.09E+01 ± 6.10E-02 (0)	2.10E+01 ± 4.46E-02 (0)	2.10E+01 ± 4.00E-02 (0)	2.09E+01 ± 8.21E-02 (0)
F09	30	[2.51E+05 ± 4.67E+03] (50)	[2.51E+05 ± 3.73E+03] (50)	[2.70E+05 ± 9.40E+03] (50)	**[2.16E+05** ± **4.16E+03]** (50)

The experimental results indicate that by integrating of the one-step k-means clustering in DE, our proposed CDE can enhance the performance of DE in terms of the quality of the final results and the reduction of NFFEs to approach the global optimum. In addition, experiments conducted on different population size, dimensionality, various mutation schemes, and self-adaptive parameter control also show that CDE is more effective and efficient than DE. Moreover, compared with DEahcSPX and ODE, two highly competitive variants of DE, CDE is able to obtain better performance for the majority of the test functions in terms of all four performance criteria used in this paper.

One additional parameter, clustering period m, is included in CDE. In this work, some preliminary experiments have been performed to verify its effect on the performance of CDE. In our future work, the effect will be studied in more detail by varying the population size and the problem dimensionality. In addition, we believe that some other clustering algorithms and other distance measures can also be used in CDE. Furthermore, another possible direction is applying the one-step k-means method to other EC algorithms, such as GAs, PSO, etc.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive suggestions on this work.

References

[1] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transations on Evolutionary Computation 3 (2) (1999) 82–102.

[2] T. Bäck, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms, Oxford University Press, Oxford, UK, 1996.

[3] Y.-W. Leung, Y. Wang, An orthogonal genetic algorithm with quantization for global numerical optimization, IEEE Transactions on Evolutionary Computation 5 (1) (2001) 41–53.

[4] Y. Wang, C. Dang, An evolutionary algorithm for global optimization based on level-set evolution and latin squares, IEEE Transactions on Evolutionary Computation 11 (5) (2007) 579–595.

[5] X. Yao, Y. Liu, G. Lin, Fast evolution strategies, Control and Cybernetics 26 (3) (1997) 467–496.

[6] C. Lee, X. Yao, Evolutionary programming using mutations based on the Lévy probability distribution, IEEE Transactions on Evolutionary Computation 8 (1) (2004) 1–13.

[7] J. J. Liang, A. K. Qin, P. N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evolutionary Computation 10 (3) (2006) 281–295.

[8] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Žumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE Transactions on Evolutionary Computation 10 (6) (2006) 646–657.

[9] R. Storn, K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.

[10] R. Storn, K. Price, Home page of differential evolution (2008).
URL `http://www.ICSI.Berkeley.edu/~storn/code.html`

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

[12] U. Chakraborty, Advances in Differential Evolution, Springer-Verlag, Berlin, 2008.

[13] N. Noman, H. Iba, Accelerating differential evolution using an adaptive local search, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 107–125.

[14] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization (2005).
URL `http://www.ntu.edu.sg/home/EPNSugan`

[15] B. Alatas, E. Akin, A. Karci, MODENAR: Multi-objective differential

evolution algorithm for mining numeric association rules, Applied Soft Computing 8 (1) (2008) 646–656.
[16] S. Das, A. Abraham, A. Konar, Automatic clustering using an improved differential evolution algorithm, IEEE Transaction on Systems Man and Cybernetics: Part A 38 (1) (2008) 218–237.
[17] H.-Y. Fan, J. Lampinen, A trigonometric mutation operation to differential evolution, J. of Global Optimization 27 (1) (2003) 105–129.
[18] J. Sun, Q. Zhang, E. Tsang, DE/EDA: A new evolutionary algorithm for global optimization, Information Science 169 (2004) 249–262.
[19] W. Gong, Z. Cai, C. X. Ling, ODE: A fast and robust differential evolution based on orthogonal design, in: AI 2006: Advances in Artificial Intelligence, 19th Australian Joint Conference on Artificial Intelligence, Hobart, Australia, December 4-8, 2006, Proceedings, Vol. 4304, Springer, Berlin, German, 2006, pp. 709–718.
[20] N. Noman, H. Iba, Enhancing differential evolution performance with local search for high dimensional function optimization, in: H.-G. Beyer, U.-M. O'Reilly (Eds.), Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, ACM, Washington DC, USA, June 25-29, 2005, pp. 967–974.
[21] P. Kaelo, M. M. Ali, Differential evolution algorithms using hybrid mutation, Comput. Optim. Appl. 37 (2) (2007) 231–246.
[22] Z. Yang, J. He, X. Yao, Making a difference to differential evolution, Advances in Metaheuristics for Hard Optimization (2007) 397–414.
[23] Y.-J. Wang, J.-S. Zhang, G.-Y. Zhang, A dynamic clustering based differential evolution algorithm for global optimization, European Journal of Operational Research 183 (1) (2007) 56–73.
[24] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, Soft Comput. 9 (6) (2005) 448–462.
[25] A. Salman, A. Engelbrecht, M. Omran, Empirical analysis of self-adaptive differential evolution, European Journal of Operational Research 183 (2) (2007) 785–804.
[26] A. Salman, A. P. Engelbrecht, M. G. H. Omran, Empirical analysis of self-adaptive differential evolution, European Journal of Operational Research 183 (2) (2007) 785–804.
[27] A. K. Qin, P. N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, in: IEEE Congress on Evolutionary Computation (CEC2005), IEEE, 2005, pp. 1785–1791.
[28] S. Das, A. Konar, U. K. Chakraborty, Two improved differential evolution schemes for faster global search, in: H.-G. Beyer, U.-M. O'Reilly (Eds.), Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005, ACM, 2005, pp. 991–998.
[29] J. Teo, Exploring dynamic self-adaptive populations in differential evolution, Soft Comput 10 (8) (2006) 673–686.
[30] J. Brest, M. S. Maučec, Population size reduction for the differential evolution algorithm, Applied Intelligence 29 (3) (2008) 228–247.
[31] S. Rahnamayan, H. Tizhoosh, M. Salama, A novel population initialization method for accelerating evolutionary algorithms, Computers and Mathematics with Applications 53 (10) (2007) 1605–1614.
[32] S. Rahnamayan, H. Tizhoosh, M. Salama, Opposition-based differential evolution, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 64–79.
[33] S. Rahnamayan, H. Tizhoosh, M. Salama, Opposition versus randomness in soft computing techniques, Applied Soft Computing 8 (2) (2008) 906–918.
[34] A. Jain, M. Murty, P. Flynn, Data clustering: A review, ACM Computing Surveys 31 (3) (1999) 264–323.
[35] A. Imrani, A. Bouroumi, H. Abidine, et al, A fuzzy clustering-based niching approach to multimodal function optimization, Journal of Cognitive Systems Research 1 (2) (2000) 119–133.
[36] M. Pelikan, D. Goldberg, Genetic algorithms, clustering, and the breaking of symmetry, in: the 6th International Conference on Parallel Problem Solving from Nature, 2000, pp. 385–394.
[37] N. Damavandi, S. Safavi-Naeini, A hybrid evolutionary programming method for circuit optimization, IEEE Transaction on Circuits ans Systems-I 52 (5) (2005) 902–910.
[38] Q. Lu, X. Yao, Clustering and learning gaussian distribution for continuous optimization, IEEE Transaction on Systems Man and Cybernetics C 35 (2) (2005) 195–204.
[39] L. Xu, A. Kŕzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, IEEE Transactions on Neural Network 4 (4) (1993) 636–649.
[40] S. Song, X. Yu, Multi-peak function optimization using a hierarchical clustering based genetic algorithm, in: the Sixth International Conference on Intelligent Systems Design and Applications, 2006, pp. 425–428.
[41] J. Alami, A. Imrani, A. Bouroumi, A multipopulation cultural algorithm using fuzzy clustering, Applied Soft Computing 7 (2) (2007) 506–519.
[42] J. Zhang, H. Chung, W. Lo, Clustering-based adaptive crossover and mutation probabilities for genetic algorithms, IEEE Transactions on Evolutionary Computation 11 (3) (2007) 326–335.
[43] Q. Ling, G. Wu, Z. Yang, et al, Crowding clustering genetic algorithm for multimodal function optimization, Applied Soft Computing 8 (1) (2008) 88–95.
[44] W. Gong, Z. Cai, C. Ling, J. Du, Hybrid differential evolution based on fuzzy c-means clustering, in: Genetic and Evolutionary Computation Conference (GECCO 2009), ACM Press, 2009, pp. 523 – 530.
[45] W. Sheng, S. Swift, L. Zhang, et al, A weighted sum validity function for clustering with a hybrid niching genetic algorithm, IEEE Transaction on Systems Man and Cybernetics B 35 (6) (2005) 1156–1167.
[46] K. Deb, A population-based algorithm-generator for real-parameter optimization, Soft Computing - A Fusion of Foundations, Methodologies and Applications 9 (4) (2005) 236–253.
[47] K. Deb, A. Anand, D. Joshi, A computationally efficient evolutionary algorithm for real-parameter optimization, Evolutionary Computation 10 (4) (2002) 371–395.
[48] Y.-W. Shang, Y.-H. Qiu, A note on the extended rosenbrock function, Evol. Comput. 14 (1) (2006) 119–126.
[49] R. Gäperle, S. Müler, P. Koumoutsakos, A parameter study for differential evolution, in: Proc. WSEAS Int. Conf. Advances Intell. Syst., Fuzzy Syst., Evol. Comput., 2002, pp. 293–298.
[50] V. Feoktistov, S. Janaqi, Generalization of the strategies in differential evolution, Vol. 7, IEEE Computer Society, Los Alamitos, CA, USA, 2004, p. 165a.
[51] E. Mezura-Montes, J. Velázquez-Reyes, C. A. C. Coello, A comparative study of differential evolution variants for global optimization, in: M. Cattolico (Ed.), Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, ACM, Seattle, Washington, USA, July 8-12, 2006, pp. 485–492.