ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/317768719

A new differential evolution algorithm for solving multimodal optimization problems with high dimensionality

Article *in* Soft Computing · June 2017

DOI: 10.1007/s00500-017-2632-5

CITATIONS
2

READS
370

4 authors:

Shouheng Tuo
Xi'an University of Posts and Telecommunications,China,Xi'an
124 PUBLICATIONS 1,735 CITATIONS

SEE PROFILE

Junying Zhang
Xidian University
169 PUBLICATIONS 1,619 CITATIONS

SEE PROFILE

Xiguo Yuan
Virginia Polytechnic Institute and State University
76 PUBLICATIONS 573 CITATIONS

SEE PROFILE

Longquan Yong
Shaanxi University of Technology
98 PUBLICATIONS 485 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project gene annotation View project

Project machine learning View project

All content following this page was uploaded by Shouheng Tuo on 06 October 2018.

The user has requested enhancement of the downloaded file.

Soft Comput
DOI 10.1007/s00500-017-2632-5

CrossMark

METHODOLOGIES AND APPLICATION

A new differential evolution algorithm for solving multimodal optimization problems with high dimensionality

Shouheng Tuo[1,2] · Junying Zhang[1] · Xiguo Yuan[1] · Longquan Yong[2]

© Springer-Verlag Berlin Heidelberg 2017

Abstract Differential evolution (DE) is an efficient intelligent optimization algorithm which has been widely applied to real-world problems, however poor in solution quality and convergence performance for complex multimodal optimization problems. To tackle this problem, a new improving strategy for DE algorithm is presented, in which crossover operator, mutation operator and a new local variables adjustment strategy are integrated together to make the DE more efficient and effective. An improved dynamic crossover rate is adopted to manage the three operators, so to decrease the computational cost of DE. To investigate the performance of the proposed DE algorithm, some frequently referred mutation operators, i.e., DE/rand/1, DE/Best/1, DE/current-to-best/1, DE/Best/2, DE/rand/2, are employed, respectively, in proposed method for comparing with standard DE algorithm which also uses the same mutation operators as our method. Three state-of-the-art evolutionary algorithms (SaDE, CoDE and CMAES) and seven large-scale optimization algorithms on seven high-dimensional optimization problems of CEC2008 are compared with the proposed algorithm. We employ Wilcoxon Signed-Rank Test to further test the difference significance of performance between our algorithm and other compared algorithms. Experimental results demonstrate that the proposed algorithm is more effective in solution quality but with less CPU time (e.g., when dimensionality equals 1000, its mean optimal fitness is less than 1e−9 and the CPU time reduces by about 19.3% for function Schwefel 2.26), even with a very small population size, no matter which mutation operator is adopted.

Keywords Differential evolution · Dynamic crossover operator · Local adjustment strategy · High dimensionality · Multimodal optimization problems

Communicated by V. Loia.

✉ Junying Zhang
jyzhang@mail.xidian.edu.cn

Shouheng Tuo
tuo_sh@126.com

1 School of Computer Science and Technology, Xidian University, Xi'an 710071, People's Republic of China

2 School of Mathematics and Computer Science, Shaanxi University of Technology, Hanzhong 723000, People's Republic of China

1 Introduction

Since the differential evolution (DE) algorithm was proposed by Storn and Price (1995), Storn and Price (1997), Price et al. (2006), Storn and Price (1996), it has become one of hot areas of research quickly. With advantages of less parameter, strong search ability, and simplicity, it has been widely employed to tackle engineering optimization problems (Wang et al. 2010; Pan et al. 2011; Kitayama et al. 2011; Cai et al. 2011; Liu et al. 2014; Xu et al. 2007; Liao 2010; Joshi and Sanderson 1997). Although researches on DE have reached a very high level during the last several years, there still exist some shortcomings in DE, such as weak global search ability for multimodal optimization problems, and expensive running cost for high-dimensional problems.

To enhance the performance of solving complex multimodal problems, the improvements of many DE variants concentrate mainly on the mutation operator, crossover operation, and parameters [such as crossover rate (CR), scale factor (F), and population size (Islam et al. 2012)]. Some of them Cai et al. (2011), Liu et al. (2014), Xu et al. (2007), Liao (2010), Joshi and Sanderson (1997), Islam et al. (2012), Qin et al. (2009), Wang et al. (2011), Wang et al. (2012),

Published online: 03 June 2017

Kovačević et al. (2014), Zhou et al. (2013), Ghosh et al. (2011) have successfully improved the performance for solving low-dimensional optimization problems. However, for some complex multimodal optimization problems with high dimensionality (≥200), the DE and variants are still unsatisfactory. That is because the equilibrium between population diversity (global exploration power) and convergence of algorithm (local exploitation power) is extremely difficult owing to some internal intrinsic difficulties on complex multimodal optimization problems. It must be assured that the population diversity on each decision variable can be maintained before the region in which the global optimal solution is contained is found. If the population on one decision variable loses the diversity prematurely, the DE algorithm will disable to find the global optimal solution on this decision variable. In this study, a novel crossover operator combined with a new local adjustment strategy is presented. It can contribute to maintain a strong global exploration power in the early stage of searching process by using a large crossover probability and a large adjustment-step. Moreover, it can intensify the local exploitation power in the later period of evolution using a small crossover rate and fine-tuning strategy.

In this paper, the improved strategy includes three respects as follows,

(1) The mutation operator and the crossover operator are integrated together to decrease the computational cost for solving a high-dimensional problem.
(2) A local adjustment strategy is combined with crossover operator, which is conducive to enhance the population diversity due to a large disturbance step in the early period of evolution process, and intensify the local exploitation power using a small adjustment-step in the later stage of evolution period.
(3) The crossover rate (CR) decreases dynamically with the increasing of the iteration times. In this way, a large CR in the beginning is beneficial to enhance the learning with other individuals in the early stage, and a small CR can intensify the local exploitation power in the later stage.

The rest of this paper is organized in the following way: Sect. 2 introduces the standard differential evolution algorithm. Section 3 reviews the related works on DE in briefly. The proposed algorithm is presented in Sect. 4 in detail. Experimental results are investigated in Sect. 5. Finally, Sect. 6 summarizes this work.

2 Differential evolution algorithm

In this study, the optimization problems with bound constraint are expressed as:

$$\underset{X}{\text{minimize}}\ f(X)$$

$$X = (x_1, x_2, \ldots, x_D) \in S = \prod_{j=1}^{D} [x_{\mathrm{L}_j}, x_{\mathrm{U}_j}]$$

where $f(X)$ is the objective function, D is the dimensionality of problem. x_{L_j} and x_{U_j} are the lower bound and upper bound of variable x_j.

Differential evolution (DE), as a swarm intelligent optimization algorithm, needs a population with NP individuals. Each individual in DE represents a point of search space and improves the fitness value by finding the difference between itself and other individuals.

The algorithmic framework of DE is expressed as follows:

Step 1 Set the parameters: scaling factor (F), crossover rate (CR), number of population (NP) and the terminal condition.

Step 2 Initialize the individuals $X^1(t_0)$, $X^2(t_0), \ldots, X^{\mathrm{NP}}(t_0)$ randomly from search space S to form the initial population.

$$X^i(t_0) = xL + \mathrm{rand}(1, D). * (x_U - x_L)$$

where rand$(1, D)$ is a vector that includes D uniformly distributed random numbers lying between 0 and 1; ".∗" denotes the Hadamard product.

Step 3 For each individual X^i $(i = 1, 2, \ldots, \mathrm{NP})$ in the population, do the following operations:

(3.1) *Mutation operation* Create a donor vector $V^i(t) = \left(v_1^i(t), v_2^i(t), \ldots, v_D^i(t)\right)$ corresponding to individual $X^i(t) = \left(x_1^i(t), x_2^i(t), \ldots, x_D^i(t)\right)$ by mutation operation. The typical mutation operations are listed in the following:

(1) DE/rand/1: $v_j^i(t) = x_j^{r_1}(t) + F \times \left(x_j^{r_2}(t) - x_j^{r_3}(t)\right)$

(2) DE/best/1: $v_j^i(t) = x_j^{\text{best}}(t) + F \times \left(x_j^{r_1}(t) - x_j^{r_2}(t)\right)$

(3) DE/current-to-best/1: $v_j^i(t) = x_j^i(t) + F \times \left(x_j^{\text{best}}(t) - x_j^i(t)\right) + F \times \left(x_j^{r_1}(t) - x_j^{r_2}(t)\right)$

(4) DE/best/2: $v_j^i(t) = x_j^{\text{best}}(t) + F \times \left(x_j^{r_1}(t) - x_j^{r_2}(t)\right) + F \times \left(x_j^{r_3}(t) - x_j^{r_4}(t)\right)$

(5) DE/rand/2: $v_j^i(t) = x_j^{r_1}(t) + F \times \left(x_j^{r_2}(t) - x_j^{r_3}(t)\right) + F \times \left(x_j^{r_4}(t) - x_j^{r_5}(t)\right)$

where F is the scale factor; r_1, r_2, r_3, r_4 and r_5 are integers which are chosen from the set $\{1, 2, \ldots, \mathrm{NP}\}$ randomly, and they are mutually exclusive; t is the iteration times; "best" is the index of the optimal individual in population.

(3.2) *Crossover operation* Create a trial vector $U^i(t) = \left(u_1^i(t), u_2^i(t), \ldots, u_D^i(t)\right)$ through mixing the donor vector $V^i(t)$ and the target vector $X^i(t)$ under crossover operation. The most frequently referred crossover operator is the binomial crossover, as follow

$$u_j^i(t) = \begin{cases} v_j^i(t), & \text{if } \mathrm{rand}(0,1) \le \mathrm{CR} \text{ or } j == J \\ x_j^i(t), & \text{otherwise} \end{cases}, \quad j = 1, 2, \ldots, D$$

where J is an integer that is chosen from the range $[1, D]$ randomly; CR is the crossover rate.

(3.3) *Selection operation* Determine whether the trial vector $U^i(t)$ can replace the target vector $X^i(t)$. The selection operation is as follow

$$X^i(t+1) = \begin{cases} U^i(t) & , \text{If } f\left(U^i(t)\right) \le f\left(X^i(t)\right) \\ X^i(t) & , \text{If } f\left(U^i(t)\right) > f\left(X^i(t)\right) \end{cases}$$

3 Previous work on DE

Performance of DE is limited on convergence property and solution quality. It is claimed that solution quality is sensitive to parameter setting and the characteristics (unimodal, multimodal) of problem (Gamperle et al. 2002). Many variants of DE have been proposed to improve the performance, which are generally classified into four categories (Wang et al. 2012):

(1) Parameters (i.e., F, CR and NP) are adjusted with dynamic adjustment strategy or self-adaptive strategy (Islam et al. 2012; Qin et al. 2009; Wang et al. 2011; Kovačević et al. 2014; Ghosh et al. 2011; Wang and Dang 2007; Weber et al. 2011; Brest et al. 2006; Zhang and Sanderson 2009). Many attempts have been made to improve global search ability by tuning the parameters with dynamic strategy or adaptive strategy. Liu and Lampinen (2005) introduced a fuzzy adaptive DE algorithm, which adjusts scaling factor F and crossover rate CR by using a fuzzy knowledge-based system. Zhang and Sanderson (2009), respectively, employed normal distribution and Cauchy distribution to generate dynamic F and CR. SaDE is proposed by Qin et al. (2009), in which the trial vector generation strategies and parameters (F, CR) are adjusted adaptively by learning from historical search. Wang et al. (2011) introduced a composite DE (CoDE) by combining several effective trial vector generation strategies.

(2) New mutation or crossover operators are introduced Fan and Lampinen (2003), Bhowmik et al. (2010), Das et al. (2009), Zhang and Sanderson (2009), Das et al. (2005). Fan and Lampinen (2003) presented a trigonometric mutation operator, so to accelerate the convergence speed. Das et al. (2009) proposed a neighborhood-based mutation operator which is combined with global mutation model.

(3) DE algorithm is hybridized with other methods (Wang et al. 2007, 2001; Jia et al. 2011; Neri et al. 2011; Noman and Iba 2008; Rahnamayan et al. 2008) to enhance the performance of DE for complex problems.

(4) The multi-population strategies (Tasgetiren and Suganthan 2006; Zaharie and Petcu 2003) and dynamic population size Brest et al. (2008) are proposed to enhance the population diversity and improve global exploration power. DE is combined with cooperative co-evolution as a dimension decomposition mechanism (Zamuda et al. 2008).

4 The proposed algorithm

For a multimodal optimization problem with high dimensionality, the computing cost is very expensive for many intelligent algorithms, which are easily trapped into a local search. The DE and variants also have the shortcomings. To tackle the problems, the most commonly method is to improve the diversity of population, such as setting large size population (NP), multi-population strategy and hybridizing with other complex technique. However, the strategy of large size of NP and multi-population would make the computed amount increase greatly and reduce the convergence speed of algorithm. Hybridizing DE combined with other methods usually makes algorithm very complex.

To tackle these problems, we proposed a new DE algorithm with small population size, in which the mutation operator, the crossover operator and a proposed local adjustment strategy are integrated together for decreasing the computational cost, and the CR is adjusted dynamically with evolution process for balancing the exploration power and the exploitation power.

In Fig. 1, the proposed algorithm is described in detail.

As shown in Fig. 1, there are three primary difference between our algorithm and the standard DE algorithm.

(1) Each trial vector $U^i(t)(i = 1, 2, \ldots, \mathrm{NP})$ is assigned to $X^i(t)$ before mutation and crossover operation.
(2) A new local adjustment strategy is incorporated into the crossover operator to enhance the capability of global search.
(3) Both the mutation and the crossover are controlled by parameter CR, which will be analyzed in Sect. 4.1, 4.2 and 4.3.

Algorithm: proposed DE algorithm

Input : Objective function, *D, NP*, MaxIteration, x_L, x_U, *F*, CR_{max}, CR_{min}, AS_{max}, AS_{min}, AS_{mid}, LAR_{max}, LAR_{min}.
Output : global optimal solution and fitness;

Initialization :
MaxFEs=MaxIteration $\times$ NP;
  For i=1 to NP
    $X^i(t_0) = xL + rand(1, D) * (x_U - x_L)$;
    $F^i = fitness\left(X^i(t_0)\right)$;
  EndFor
Evolutionary process :
For t=1 to MaxIteration
  Calculate the $CR(t)$, $LAR(t)$ and $AS(t)$ with equation (1),(2) and (3);
  For each target vector $X^i\,(i = 1, 2, \cdots, NP)$, *do*
    $U^i(t) = X^i(t)$;
    $J = \lceil rand(0,1) * D \rceil$;
    for each j in $\{1, 2, \cdots, D\}$ do
      if $rand(0,1) \le CR(t)$ *or* $j == J$
        $v = Call\_Mutation\left(x_j^i(t)\right)$; // (1) *Mutation*
        $u_j^i(t) = v$; // **(2)Crossover**
        if $rand(0,1) \le LAR(t)$
          $u_j^i(t) = u_j^i(t) + (0.5 - rand(0,1)) * AS_j(t)$;
        elseif $rand(0,1) \le 0.1 * LAR(t)$
          $u_j^i(t) = xL_j + rand(0,1) * \left(x_{U_j} - x_{L_j}\right)$;
        endif (3)Local Adjustment
      endif
    endfor
    $\overline{F}^i = fitness\left(U^i(t)\right)$;
    if $\overline{F}^i \le F^i$ (4) Selection
      $X^i(t+1) = U^i(t)$; $F^i(t+1) = \overline{F}^i$;
    endif
  EndFor
EndFor

Fig. 1 The flow chart of proposed algorithm

4.1 The dynamic crossover rate

Unlike conventional DE fixed the CR from beginning to end, in the proposed algorithm, CR is dependent on a dynamic function. At the early stage of evolution, the CR obtains a large probability value for mutation, crossover and local adjustment. At the later stage of evolution, the CR gets a small value to tune the decision variables for finding high-precision solution. The expression of CR is given in Eq. (1), and changing curve of CR is shown in Fig. 2a.

$$\mathrm{CR}(t) = \mathrm{CR}_{\max} \times \left(\frac{\mathrm{CR}_{\min}}{\mathrm{CR}_{\max}}\right)^{\left(\frac{t}{T}\right)^2} \quad (1)$$

4.2 The local adjustment strategy

Local adjustment (LA) strategy is proposed for jumping out of local search in the early stage of search and for exploiting high-precision solutions in the later stage of evolution. The LA consists of step disturbance and random perturbation. The step disturbance is used for adjusting trial solution with local adjustment rate (LAR) in the neighborhood of current position. Random perturbation is for exploring new unknown region in the feasible space with probability 0.1* LAR, which can effectively avoid trapping into local search. The value of LAR increases from $\mathrm{LAR}_{\min}$ to $\mathrm{LAR}_{\max}$ with the increasing t (see Eq. 2) and the change curve is shown in Fig. 2b.

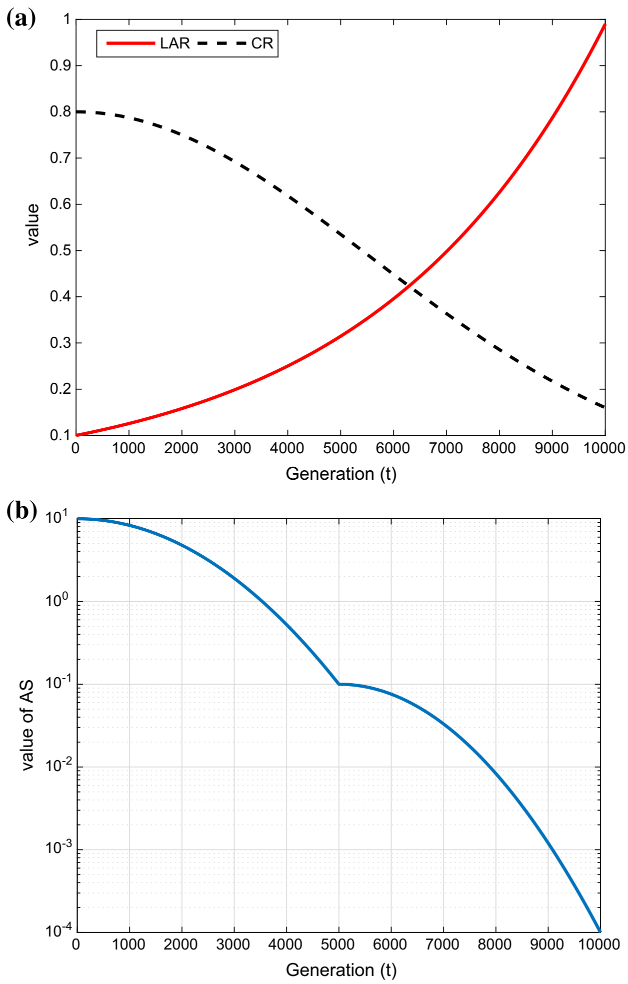

Fig. 2 **a** The change curves of CR and LAR. **b** The change curve of local adjustment step (AS)

$$\mathrm{LAR}(t) = \mathrm{LAR}_{\min} \times \left(\frac{\mathrm{LAR}_{\max}}{\mathrm{LAR}_{\min}}\right)^{\frac{t}{T}} \tag{2}$$

The adjustment step (AS) varies with iteration t dynamically, as shown in Eq. (3). In the beginning phase of evolution, the AS having a large value aims to explore new regions. With the increasing of t, the value of AS gradually decreases from $\mathrm{AS}_{\max}$ to $\mathrm{AS}_{\mathrm{mid}}$. When the region including global optimal solution has been located, the value of AS has also become very small relatively, the DE will begin to concentrate all efforts on exploiting the high-precision solution in the found region.

$$\mathrm{AS}(t) = \begin{cases} \mathrm{AS}_{\max} \times \left(\frac{\mathrm{AS}_{\mathrm{mid}}}{\mathrm{AS}_{\max}}\right)^{\left(\frac{t}{\mathrm{T}/2}\right)^2}, & t \le \frac{\mathrm{T}}{2} \\ \mathrm{AS}_{\mathrm{mid}} \times \left(\frac{\mathrm{AS}_{\min}}{\mathrm{AS}_{\mathrm{mid}}}\right)^{\left(\frac{t-\mathrm{T}/2}{\mathrm{T}/2}\right)^2}, & t > \frac{\mathrm{T}}{2} \end{cases} \tag{3}$$

where $\mathrm{AS}_{\min} < \mathrm{AS}_{\mathrm{mid}} < \mathrm{AS}_{\max}$.

In the Eq. (3), a large value of AS and a small value of LAR can keep strong perturbation capability in search space so as to find new region or jump out of local extreme area. When the globally optimal region has been located, small AS and large LAR are beneficial to find high-precision solution. The change curve of AS is shown in Fig. 2b.

4.3 Integrating the mutation operation and the crossover operation

In the standard DE algorithm, whether the crossover operation is carried out on $x_j^i(t)(i = 1, 2, \ldots, \mathrm{NP}; j = 1, 2, \ldots, D)$ or not, the mutation operation is always performed on each variable $x_j^i(t)$ of target vector $X^i(t) = \left(x_1^i(t), x_2^i(t), \ldots, x_D^i(t)\right)$ so as to generate the donor vector $V^i(t) = \left(v_1^i(t), v_2^i(t), \ldots, v_D^i(t)\right)$. However, usually, there are only part of values in $\left(v_1^i(t), v_2^i(t), \ldots, v_D^i(t)\right)$ could be adopted in trial vector $U^i(t) = \left(u_1^i(t), u_2^i(t), \ldots, u_D^i(t)\right)$, which is as follow

$$u_j^i(t) = \begin{cases} v_j^i(t), & \text{if rand} < \mathrm{CR} \text{ or } j == J \\ x_j^i(t), & \text{otherwise} \end{cases}$$

For a high-dimensional problem, the mutation operation is performed on only a few of variables if the value of $\mathrm{CR}(t)$ is very small, which causes much redundant computation. To reduce the redundant calculation, the mutation operation is integrated with the crossover operation in the proposed DE algorithm, in which the mutation operation will be performed on only the variables that have been selected to perform the crossover operation.

5 Numerical experiments

5.1 Experimental environment and parameter setting

Sixteen complex benchmark functions (Hedar 2013; Tang et al. 2007a, b; Herrera et al. 2010) (shown in Table 3) are employed to investigate the performance of the proposed algorithm. The effect of parameters to our method is analyzed in Sect. 5.2. We compare the quality of solutions in Sect. 5.3 and the difference significance between our approach and compared algorithms using the Wilcoxon Signed-Rank Test in Sect. 5.4. The experimental analysis and the comparison of population diversity are list in Sects. 5.5 and 5.6, respectively.

All the experiments were performed on Windows XP 32 system with Intel(R) Core(TM) i3-2120 CPU@3.30 GHz and 2 GB RAM, and all the program codes were written in MATLAB R2009a.

In the experiments, five state-of-the-art mutation operators (DE/rand/1, DE/Best/1, DE/current-to-best/1, DE/Best/2, and DE/rand/2) are employed in our algorithm, respectively. We name the five improved mutation operators as

Table 1 The parameter setting for all comparison algorithms

Algorithm	NP	F	CR	Others
DE/rand/1	30	0.6	0.3	
DE/Best/1				
DE/current-to-best/1				
DE/Best/2				
DE/rand/2				
DE/rand/1_2	5	0.6	$\mathrm{CR}_{\max} = 100/D$, $\mathrm{CR}_{\min} = 30/D$	$\mathrm{LAR}_{\max} = 0.99$, $\mathrm{LAR}_{\min} = 0.1$; $\mathrm{AS}_{\max} = (x_U - x_L)/10$, $\mathrm{AS}_{\mathrm{mid}} = (x_U - x_L)/(1\mathrm{E}{+}4)$, $\mathrm{AS}_{\min} = (x_U - x_L)/(1\mathrm{E}{+}15)$
DE/Best/1_2 (*myDE*)				
DE/current-to-best/1_2				
DE/Best/2_2				
DE/rand/2_2				
SaDE	50	$F \sim N(0.5, 0.3)$	CR$\sim N(\mathrm{CR}_m, 0.1)$	$c = 0.1$; $p = 0.05$
CoDE	30	$F = [1.0, 1.0, 0.8]$	CR$=[0.1, 0.9, 0.2]$	
CMAES	50	$\delta = 0.25,\ \mu = \left\lfloor \frac{1}{2}\left(4 + \lfloor 2\log(N)\rfloor\right)\right\rfloor$, $\tau_1 = 0.1$, $\tau_2 = 0.1$		

DE/rand/1_2, DE/Best/1_2 (*also called myDE*), DE/current-to-best/1_2, DE/Best/2_2 and DE/rand/2_2. In the same way, the five mutation operators are adopted in standard DE algorithm respectively, which are called DE/rand/1, DE/Best/1, DE/current-to-best/1, DE/Best/2 and DE/rand/2, correspondingly.

Parameters setting for the all algorithms are shown in Table 1 (the parameters values of SaDE, CoDE and CMAES are identical to original papers Qin et al. 2009; Wang et al. 2011; Hansen and Ostermeier 2001).

The MATLAB source codes of SaDE, CMAES, CoDE were downloaded from: http://ist.csu.edu.cn/ paper%20and%20matlab%20code/Differential%20evolution%20with%20composite%20trial%20vector%20generation%20strategies%20and%20control%20parameters/CoDE.RAR) (Wang et al. 2011).

Our algorithm (myDE) also compare with seven state-of-the-art algorithms (Brest et al. 2008; Hsieh et al. 2008; Macnish and Yao 2008; Tseng and Chen 2008; Wang and Li 2008; Zamuda et al. 2008; Zhao et al. 2008) which are used for testing large-scale single objective global optimization with bound constraints in CEC'2008 special session.

5.2 Effect of NP and CR on performance of the proposed algorithm

5.2.1 Effect of NP on the accuracy of solution

To investigate the effect of NP (from 5 to 35) on the performance of proposed algorithm, sixteen complex benchmark functions are employed to verify myDE algorithm. The dimensionality of all functions is set to 1000, maximum function evaluation times (MaxFEs) is set to 5×10^6. For different NP, the mean value (Mean) of the best fitness of 10 independent run times are shown in Table 2.

It can be seen clearly from Table 2 that, when NP $= 5$, the mean optimal solution of the proposed algorithm is superior to or equal to that of others for all functions except for function F_6 and F_{16}. So we recommend the NP is set to 5.

5.2.2 Effect of CR_{max} and CR_{min} on the accuracy of solution of the proposed algorithm

In the proposed algorithm, the parameter CR is changed dynamically with the evolution process, which depends on the $\mathrm{CR}_{\max}$ and $\mathrm{CR}_{\min}$. Appropriate values of $\mathrm{CR}_{\max}$ and $\mathrm{CR}_{\min}$ are very important to the proposed algorithm. In this section, we investigate the effect of $\mathrm{CR}_{\max}$ and $\mathrm{CR}_{\min}$ on performance of the myDE algorithm.

Parameters setting: $D = 1000$; $\mathrm{CR}_{\max}$ is read from set $A = \{0.24, 0.2, 0.16, 0.12, 0.08\}(\{240/D, 200/D, 160/D, 120/D, 80/D\})$; $\mathrm{CR}_{\min}$ is read from set $B = \{0.08, 0.06, 0.04, 0.02\}$ $(\{80/D, 60/D, 40/D, 20/D\})$. Full combinatorial values of the set A and B are used to investigate the effect on the performance of algorithm. The experimental results are shown in Figs. 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12.

It can be seen from Figs. 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12 that, for most of functions, the smaller the $\mathrm{CR}_{\max}$ and $\mathrm{CR}_{\min}$ are, the higher precision the solution is. That is because the smaller the CR is, the higher the probability that new generated trial vector $U^i(t)$ is superior to $X^i(t)$ is, thus the higher the success rate that $X^i(t+1) = U^i(t)$ is. But the CR ($\mathrm{CR}_{\max}$ and $\mathrm{CR}_{\min}$) cannot be set too small value, which will lead to the decrease of convergence speed. A great number of simulation experiments show that setting $\mathrm{CR}_{\max}$ between $50/D$ and $100/D$ is an appropriate choice, and setting $\mathrm{CR}_{\min}$ between

Table 2 Comparison of the mean optimal solution with change of NP for Functions $F_1 - F_{16}$ on $D = 1000$

NP		Mean		Mean		Mean		Mean		Mean		Mean		Mean		Mean
5	F_1	*2.59E–13*	F_3	*−4.33E+02*	F_5	*3.64E–12*	F_7	*1.01E–09*	F_9	*−4.50E+02*	F_{11}	*1.05E+03*	F_{13}	*−1.80E+02*	F_{15}	*−7.36E+03*
10		3.11E–13		−2.32E+02		9.22E+01		*1.01E–09*		*−4.50E+02*		1.23E+03		*−1.80E+02*		−5.91E+03
15		1.43E–02		−2.18E+02		2.25E+03		3.53E–01		*−4.50E+02*		1.10E+03		*−1.80E+02*		−5.22E+03
20		2.02E–12		−2.13E+02		2.65E+03		1.04E–09		*−4.50E+02*		1.27E+03		*−1.80E+02*		−5.23E+03
25		2.24E–09		−2.09E+02		2.97E+03		5.38E–05		*−4.50E+02*		1.16E+03		*−1.80E+02*		−5.13E+03
30		1.76E–07		−2.05E+02		3.13E+03		2.37E+01		*−4.50E+02*		1.07E+03		*−1.80E+02*		−5.09E+03
35		2.20E–02		−2.02E+02		3.22E+03		5.34E+03		*−4.50E+02*		3.49E+03		*−1.80E+02*		−4.95E+03
5	F_2	*7.77E–16*	F_4	*6.58E–26*	F_6	6.38E+02	F_8	*3.03E–25*	F_{10}	*−4.32E+02*	F_{12}	*−3.30E+02*	F_{14}	*−1.40E+02*	F_{16}	3.51E+02
10		1.48E–13		1.28E–25		6.54E+02		5.79E–25		−4.16E+02		−3.25E+02		*−1.40E+02*		8.67E+01
15		1.71E–02		5.82E–04		6.77E+02		2.62E–03		−4.14E+02		9.12E+02		*−1.40E+02*		*1.17E+01*
20		*7.77E–16*		4.43E–21		7.32E+02		6.31E–23		−4.05E+02		1.26E+03		*−1.40E+02*		3.56E+01
25		*7.77E–16*		3.40E–13		6.02E+02		4.58E–17		−4.04E+02		1.67E+03		*−1.40E+02*		1.43E+02
30		2.04E–12		1.10E–09		5.95E+02		4.54E–13		−4.03E+02		1.91E+03		*−1.40E+02*		5.71E+02
35		4.34E–02		1.43E–03		*5.18E+02*		6.11E–03		−4.00E+02		2.19E+03		*−1.40E+02*		5.42E+02

Italics denotes the best results obtained with the corresponding NP

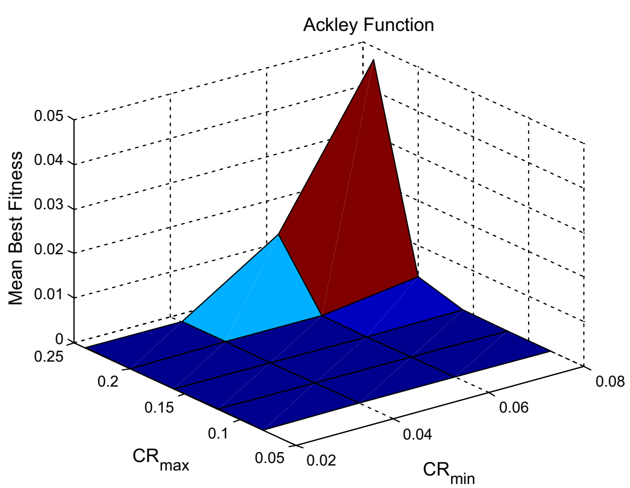

Fig. 3 The surface change chart of optimal solution of Ackley function on different CR_{max} and CR_{min} ($D = 1000$)

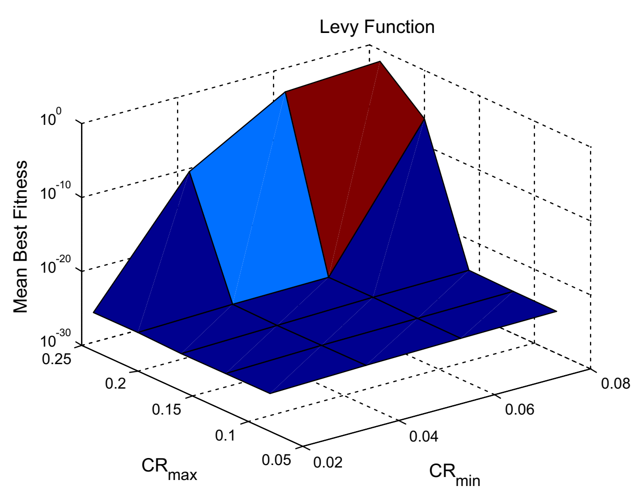

Fig. 4 The surface change chart of optimal solution of Levy function on different CR_{max} and CR_{min} ($D = 1000$)

$10/D$ and $30/D$ is recommended. In this paper, the CR_{max} is set to $100/D$ and the CR_{min} is set to $30/D$.

5.3 Comparisons of solution quality

To investigate the performance of the proposed algorithm, the following two tests are carried out.

(Test 1) All algorithms are performed on sixteen complex high-dimensional benchmark functions (shown in Table 3), and the results of proposed algorithm are compared with that of standard DE algorithm which use DE/rand/1, DE/Best/1, DE/current-to-best/1, DE/Best/2, and DE/rand/2 as the mutation operation. In 20 independent runs, the mean fitness value (Mean), standard deviation of fitness (SD) and the mean run time (CPU time) are summarized in Tables 4, 5 and 6. Figures 13, 14, 15, 16, 17 and 18 compare the convergence curves of mean optimal fitness values of ten algorithms. The statistical box plots in Figs. 19, 20, 21, 22, 23 and 24 draw the distributions of optimal solutions of the six functions in 20 independent runs for ten algorithms.

(Test 2) The DE/Best/1_2 (called *myDE* in this work) is compared with three state-of-the-art evolutionary algorithms (SaDE Qin et al. 2009, CMAES Hansen and Ostermeier 2001, CoDE Wang et al. 2011). In the test, the best fitness value (Best), Mean, the worst fitness value (Worst), SD, and the mean run time of 20 independent runs for each function are shown in Tables 7, 8 and 9. The convergence curves of mean optimal fitness values of four algorithms are shown in

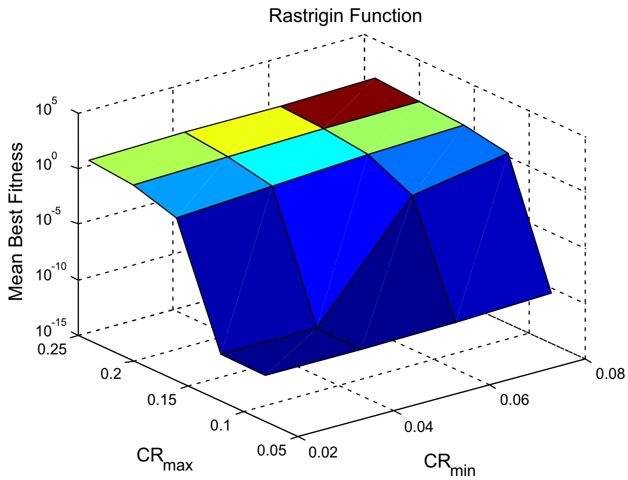

Fig. 5 The surface change chart of optimal solution of Rastrigin function on different CR_{max} and CR_{min} ($D = 1000$)

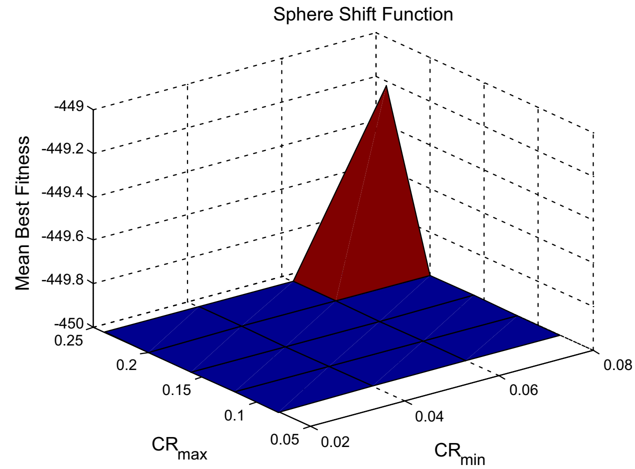

Fig. 8 The surface change chart of optimal solution of Sphere shift function on different CR_{max} and CR_{min} ($D = 1000$)

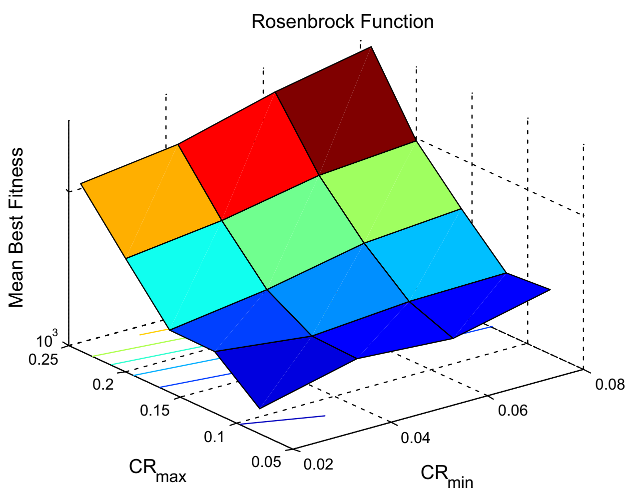

Fig. 6 The surface change chart of optimal solution of Rosenbrock function on different CR_{max} and CR_{min} ($D = 1000$)

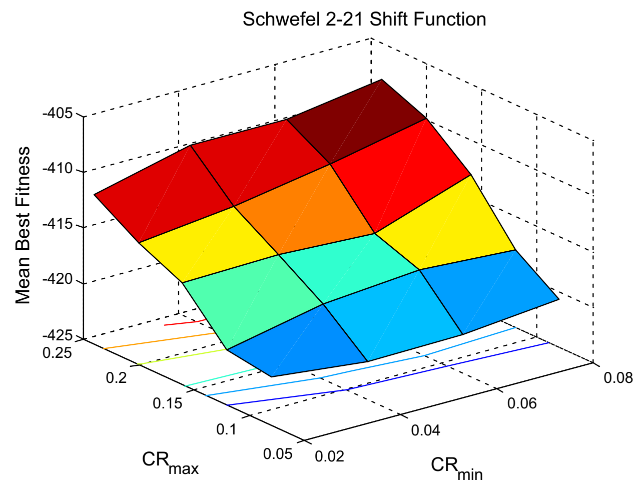

Fig. 9 The surface change chart of optimal solution of Schwefel2.21 shift function on different CR_{max} and CR_{min} ($D = 1000$)

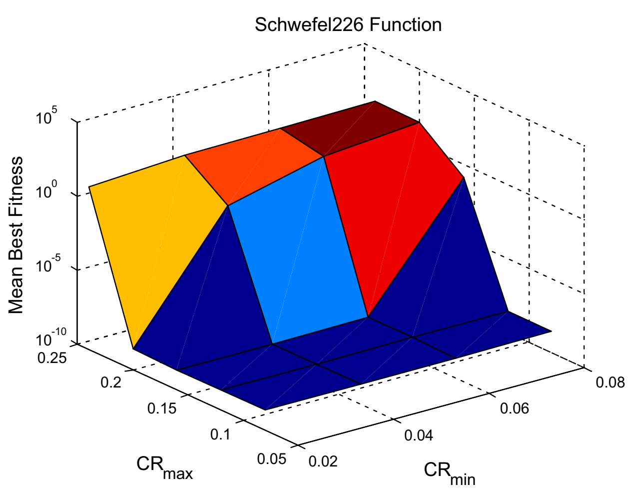

Fig. 7 The surface change chart of optimal solution of Schwefel 2.26 function on different CR_{max} and CR_{min} ($D = 1000$)

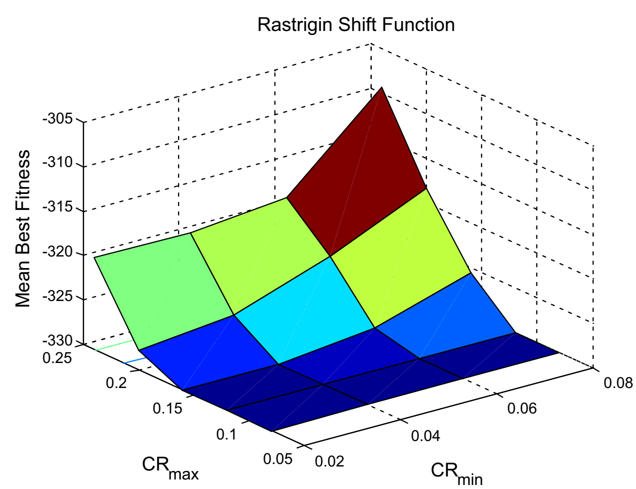

Fig. 10 The surface change chart of optimal solution of Rastrigin shift function on different CR_{max} and CR_{min} ($D = 1000$)

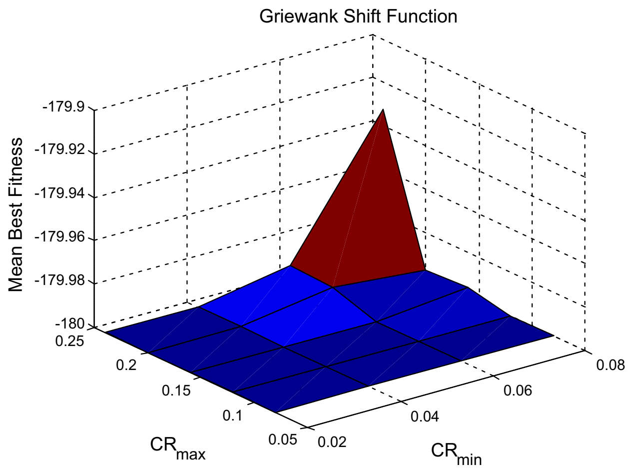

Fig. 11 The surface change chart of optimal solution of Griewank shift function on different CR_{max} and CR_{min} $(D = 1000)$

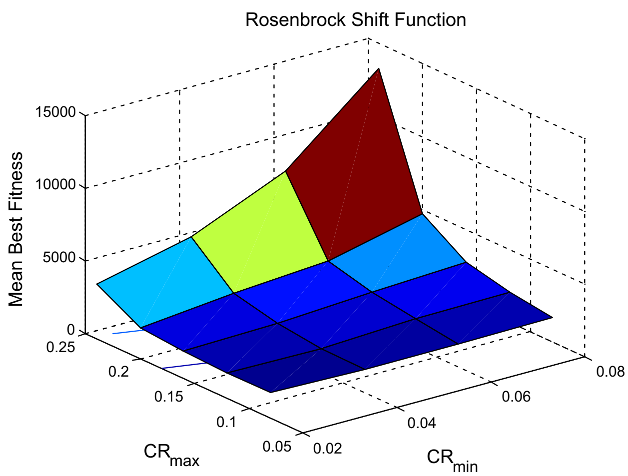

Fig. 12 The surface change chart of optimal solution of Rosenbrock shift function on different CR_{max} and CR_{min} $(D = 1000)$

Fig. 25. And the optimal solution distribution of 20 independent runs of four algorithms is drawn in Fig. 26.

(Test 3) The myDE algorithm is compared with seven state-of-the-art high-dimensional optimization algorithms (jDEdynNP-F Brest et al. 2008, EPUS-PSO Hsieh et al. 2008, UEP Macnish and Yao 2008, MTS Tseng and Chen 2008, LSEDA-gl Wang and Li 2008, DEwSAcc Zamuda et al. 2008, DMS-PSO Zhao et al. 2008) that are proposed to solve large-scale optimization problems. In the test, seven functions (F_9−F_{15}) in CEC 2008 (*which does not contains the bias*) are employed to compare with 1000 dimensions. To make a fair comparison, MaxFEs = 1000D, 25 independent runs for each problems. The experimental results (see Table 10) of the seven state-of-the-art algorithms are all from the original paper and website: http://www.ntu.edu.sg/home/EPNSugan/ (http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-08/CEC2008_SUMMARY.pdf).

5.4 Statistical pairwise test: the Wilcoxon Signed-Rank Test

To further investigate overall performance of the proposed algorithm for solving all test functions, nonparametric Wilcoxon Signed-Rank Test is employed to compare the performance with other algorithms. In the Wilcoxon Signed-Rank Test, with statistical significance value $a = 0.05$, if p value smaller than or equal to significance level (a) value is produced, null hypothesis for the test is rejected. The statistical results are shown in Table 11. "$W+$" denotes the sum of ranks of the problems in which the left algorithm outperformed the right; "$W =$" indicates the sum of ranks of the problems in which the left algorithm had the same performance with the right; "$W-$" is the sum of ranks of the functions in which the right algorithm outperformed the left (Derrac et al. 2011; Civicioglu 2013).

5.5 Analysis of experimental results

A detailed view of Tables 4, 5 and 6 shows that the improved DE algorithm outperform the standard DE algorithm on most of test functions in terms of all three metrics (Mean, SD, runtime); specially for eleven multimodal functions, the proposed algorithm is superior to standard DE algorithm with same mutation operators, and with the increasing of dimensionality (200 → 1000), its performance advantage is more apparent and it takes less CPU runtime than standard DE.

It indicates in Figs. 13, 14, 15, 16, 17 and 18 that the convergence curve of proposed algorithm is active during the evolution process. And we can find easily from Figs. 19, 20, 21, 22, 23 and 24 that the proposed DE algorithm can obtain higher precision solution than standard DE and the distribution of all the optimal solutions of the independent runs of our method is narrower than the standard DE, which means that our method is more stable than the standard DE.

Table 7, 8 and 9 show that, for most of functions, the advantages of proposed algorithm (myDE) are obvious over other three algorithms in terms of the five performance metrics (Best, Mean, Worst, SD, run time), and for others functions, its optimal solutions are almost equivalent to that of the best results of all algorithms. The CMAES algorithm has slim advantages on quality of solutions for functions F_2, F_6, F_8, F_9, F_{10} and F_{11} over the proposed algorithm; however, its CPU runtime is much more than that of myDE, and it needs much large storage space (O(D^2)).

In Fig. 25, the convergence curve of proposed algorithm falls slower than other algorithms in the early stage, but its curve keeps decreasing during the searching process, which illustrates that the proposed algorithm can avoid prematurely trapping into local search. But other three algorithms are easy to be dropped into local minimum due to the quick convergence in the early stage of search process.

Table 3 Sixteen complex benchmark functions(F_1-F_{16})

Function name	Benchmark functions expression	Search range	Optimum value	Type multimodal (M) unimodal (U)
F_1 Ackley function	$F_1(\boldsymbol{X}) = -20e^{-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)} + 20 + e$	$[-15, 30]^D$	$\boldsymbol{X}^* = (0, 0, \ldots, 0)$, $F_1(\boldsymbol{X}^*) = 0$	M
F_2 Griewank function	$F_2(\boldsymbol{X}) = \frac{1}{4000}\sum_{i=1}^{D}x_i^2 - \prod_{i=1}^{D}(\cos(\frac{x_i}{\sqrt{i}})) + 1$	$[-600, 600]^D$	$\boldsymbol{X}^* = (0, 0, \ldots, 0)$, $F_2(\boldsymbol{X}^*) = 0$	M
F_3 Michalewics function	$F_3(\boldsymbol{X}) = -\sum_{i=1}^{D}\sin(x_i)\left(\sin(ix_i^2/\pi)\right)^{2m}$; $m = 10$	$[-10, \pi]^D$	unknown	M
F_4 Levy function	$F_4(\boldsymbol{X}) = \sin^2(\pi y_1) + \sum_{i=1}^{D-1}\left[(y_i-1)^2\left(1+10\sin^2(\pi y_i+1)\right)\right] + (y_D-1)^2\left(1+10\sin^2(2\pi y_D)\right)$ $y_i = 1 + \frac{x_i-1}{4}, i = 1, 2, \ldots, D$	$[-10, 10]^D$	$\boldsymbol{X}^* = (1, 1, \ldots, 1)$, $F_4(\boldsymbol{X}^*) = 0$	M
F_5 Rastrigin function	$F_5(\boldsymbol{X}) = 10D + \sum_{i=1}^{D}\left(x_i^2 - 10\cos(2\pi x_i)\right)$	$[-5.12, 5.12]^D$	$\boldsymbol{X}^* = (0, 0, \ldots, 0)$, $F_5(\boldsymbol{X}^*) = 0$	M
F_6 Rosenbrock function	$F_6(\boldsymbol{X}) = \sum_{i=2}^{D}\left[(x_i^2 - x_{i-1})^2 + (x_i-1)^2\right]$	$[-100, 100]^D$	$\boldsymbol{X}^* = (1, 1, \ldots, 1)$, $F_6(\boldsymbol{X}^*) = 0$	U
F_7 Schwefel 2.26 function	$F_7(\boldsymbol{X}) = 418.982887272434D - \sum_{i=1}^{D}x_i^2\sin(\sqrt{\lvert x_i\rvert})$	$[-512, 512]^D$	$\boldsymbol{X}^* = (420.9687, 420.9687, \ldots, 420.9687)$, $F_7(\boldsymbol{X}^*) = 0$	M
F_8 Sphere function	$F_8(\boldsymbol{X}) = \sum_{i=1}^{D}x_i^2$	$[-100, 100]^D$	$\boldsymbol{X}^* = (0, 0, \ldots, 0)$, $F_8(\boldsymbol{X}^*) = 0$	U
F_9 Sphere shift function	$F_9(\boldsymbol{X}) = \sum_{i=1}^{D}z_i^2 + \text{bias}$, $z = \boldsymbol{X} - o$; $o = [o_1, o_2, \ldots, o_D]$; bias $= -450$	$[-100, 100]^D$	$\boldsymbol{X}^* = o$, $F_9(x^*) = -450$	U
F_{10} Schwefel_Shift function	$F_{10}(\boldsymbol{X}) = \max(\lvert z\rvert) + \text{bias}$, $z = \boldsymbol{X} - o$; $o = [o_1, o_2, \ldots, o_D]$; bias $= -450$	$[-100, 100]^D$	$\boldsymbol{X}^* = o$, $F_{10}(\boldsymbol{X}^*) = -450$	U

Table 3 continued

Function name	Benchmark functions expression	Search range	Optimum value	Type multimodal (M) unimodal (U)
$F_1 1$ Rosenbrock shift function	$F_{11}(\boldsymbol{X}) = \sum_{i=2}^{D}\left[100(z_i^2 - z_{i-1})^2 + (z_i - 1)^2\right] + \text{bias}, z = \boldsymbol{X} - o; o = [o_1, o_2, \ldots, o_D]$; bias $= 390$	$[-100, 100]^D$	$\boldsymbol{X}^* = o, F_{11}(\boldsymbol{X}^*) = 390$	U
F_{12} Rastrigin shift function	$F_{12}(\boldsymbol{X}) = \frac{1}{4000}\sum_{i=1}^{D} z_i^2 - \prod_{i=1}^{D}(\cos(\frac{z_i}{\sqrt{i}})) + 1 +$ bias$z = (\boldsymbol{X} - o) \times M, o = [o_1, o_2, \ldots, o_D]$; bias $= -330$	$[-600, 600]^D$	$\boldsymbol{X}^* = o, F_{12}(\boldsymbol{X}^*) = -330$	M
F_{13} Griewank shift function	$F_{13}(\boldsymbol{X}) = 10D + \sum_{i=1}^{D}\left(z_i^2 - 10\cos(2\pi z_i)\right) + \text{bias}$ $z = \boldsymbol{X} - o; o = [o_1, o_2, \ldots, o_D]$; bias $= -180$	$[-5.12, 5.12]^D$	$\boldsymbol{X}^* = o, F_{13}(\boldsymbol{X}^*) = -180$	M
F_{14} Ackley shift function	$F_{14}(\boldsymbol{X}) = -20e^{-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)} + 20 + e +$ bias$z = \boldsymbol{X} - o; o = [o_1, o_2, \ldots, o_D]$; bias $= -140$	$[-32, 32]^D$	$\boldsymbol{X}^* = o, F_{14}(\boldsymbol{X}^*) = -140$	M
F_{15} FastFractal 'DoubleDip' function	$F_{15}(\boldsymbol{X}) = \sum_{i=1}^{D} fractal1D(x_i + twist(x_{(i \bmod D)+1}))$ $twist(y) = 4(y^4 - 2y^3 + y_)^2$ $fractal1D(x) \approx \sum_{k=1}^{3}\sum_1^{2^{k-1}}\sum_1^{ran2(o)} doubledip(x, ran1(o), \frac{1}{2^{k-1}(2-ran1(o))})$ $doubledip(x, c, s) = \begin{cases} (-6144(x-c)^6 + 3088(x-c)^4 \\ \quad -392(x-c)^2 + 1)s \\ 0, otherwise \end{cases}$ ran1(o) : double, pseudo randomly chosen, with seed o, with equal probability from the interval[0,1]	$[-1, 1]^D$	Unknown	M

Table 3 continued

Function name	Benchmark functions expression	Search range	Optimum value	Type multimodal (M) unimodal (U)
	$\mathrm{ran2}(o)$: integer, pseudo randomly chosen, with seed o, with equal probability from the set $\{0, 1, 2\}$			
	$\mathrm{fractal1}D(x)$ is an approximation to a recursive algorithm, it does not take account of wrapping at the boundaries, or local re-seeding of the random generators.			
F_{16}: Schaffer Shift function	$F_{16}(\boldsymbol{X}) = \sum_{i=1}^{D-1} (z_i^2 + z_{i+1}^2)^{0.25} (\sin^2(50 \times (z_i^2 + z_{i+1}^2)^{0.1}) + 1), z = X - o$	$[-100, 100]^D$	$\boldsymbol{X}^* = o, F_{16}(\boldsymbol{X}^*) = 0$	M

Table 4 Experimental results (Mean, SD and mean CPU runtime) ($D = 200$)

Algorithm	F_1			F_2			F_3			F_4		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	*3.38E–14*	1.58E–15	*1.72E+01*	*1.11E–16*	*0.00E+00*	*3.61E+01*	−6.68E+01	1.15E+00	*6.25E+01*	4.54E–02	2.03E–01	*6.58E+01*
DE/rand/1_2	6.34E–14	8.02E–15	3.10E+01	4.93E–04	2.20E–03	4.53E+01	*−1.62E+02*	*1.41E+01*	8.87E+01	*5.98E–10*	*1.76E–09*	6.90E+01
DE/best/1	1.11E+01	9.78E–01	*2.07E+01*	1.40E+00	2.37E+00	*2.78E+01*	−7.44E+01	4.37E+00	*7.10E+01*	9.00E+01	1.66E+01	*5.66E+01*
DE/best/1_2	*1.65E–13*	*6.93E–15*	2.79E+01	*6.16E–04*	*2.75E–03*	3.61E+01	*−1.98E+02*	*6.97E–01*	*6.25E+01*	*2.44E–26*	*1.19E–27*	8.94E+01
DE/rand/2	1.20E–01	3.69E–01	*2.94E+01*	4.66E–03	1.25E–02	*2.91E+01*	−6.29E+01	1.80E+00	7.72E+01	1.32E+01	5.32E+00	*6.24E+01*
DE/rand/2_2	*5.54E–14*	*4.35E–15*	4.20E+01	*4.93E–04*	*2.20E–03*	4.65E+01	*−1.26E+02*	*8.17E+00*	*7.49E+01*	*3.58E–31*	*6.87E–31*	6.67E+01
DE/current-to-best/1	1.16E+01	6.45E–01	*3.28E+01*	1.80E+02	3.92E+01	*2.74E+01*	*−1.54E+02*	4.22E+00	*5.34E+01*	5.44E+01	9.84E+00	*6.29E+01*
DE/current-to-best/1_2	*9.22E–14*	*6.22E–15*	3.98E+01	*5.32E–15*	*1.37E–15*	3.23E+01	−9.92E+01	*1.88E+00*	8.19E+01	*6.36E–31*	*6.39E–31*	*6.85E+01*
DE/best/2	*3.11E–14*	*3.16E–15*	*1.86E+01*	1.36E–03	3.34E–03	*2.55E+01*	−6.28E+01	*1.18E+00*	*5.59E+01*	1.41E+01	4.41E+00	9.59E+01
DE/best/2_2(myDE)	5.58E–14	5.70E–15	3.46E+01	*5.38E–16*	*2.53E–16*	3.26E+01	*−1.27E+02*	6.13E+00	6.27E+01	*3.83E–31*	*5.67E–31*	*7.19E+01*

Algorithm	F_5			F_6			F_7			F_8		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	1.53E+03	3.32E+01	*2.51E+01*	*1.92E+02*	*1.09E+01*	*1.56E+01*	5.13E+04	1.01E+03	*3.67E+01*	*2.27E–37*	*1.77E–37*	*1.23E+01*
DE/rand/1_2	*3.13E–02*	*4.91E–02*	4.32E+01	2.48E+02	5.53E+01	2.62E+01	*3.73E–10*	*1.79E–11*	5.46E+01	2.68E–35	1.53E–36	2.18E+01
DE/best/1	5.54E+02	9.55E+01	*3.09E+01*	8.71E+03	2.03E+04	*1.71E+01*	2.68E+04	1.66E+03	*4.23E+01*	*3.23E–51*	*1.31E–50*	*1.67E+01*
DE/best/1_2	*3.07E–13*	*1.69E–13*	4.07E+01	*2.57E+02*	*5.02E+01*	3.19E+01	*3.51E–10*	*6.51E–12*	5.26E+01	1.11E–25	8.69E–27	3.09E+01
DE/rand/2	1.49E+03	5.11E+01	*2.19E+01*	2.67E+02	*4.98E+01*	*2.55E+01*	2.04E+04	1.89E+03	*3.75E+01*	*1.95E–54*	*3.02E–54*	*1.98E+01*
DE/rand/2_2	*3.55E–01*	*7.47E–01*	2.66E+01	*2.66E+02*	5.86E+01	*3.49E+01*	*1.55E+01*	*3.34E+01*	4.59E+01	1.06E–35	8.01E–37	2.79E+01
DE/current-to-best/1	*3.89E+02*	*4.76E+01*	*1.98E+01*	2.37E+05	6.49E+04	*2.56E+01*	*3.42E+04*	*1.45E+03*	*4.10E+01*	5.24E+02	8.28E+01	*2.35E+01*
DE/current-to-best/1_2	4.41E+02	8.61E+01	2.70E+01	*2.26E+02*	*4.06E+01*	3.29E+01	*3.81E+04*	4.54E+03	4.91E+01	*1.08E–35*	*5.08E–37*	3.18E+01
DE/best/2	*1.51E+03*	5.79E+01	3.28E+01	3.05E+02	5.52E+01	*2.61E+01*	2.15E+04	1.99E+03	*4.26E+01*	*3.15E–54*	*2.53E–54*	*2.37E+01*
DE/best/2_2(myDE)	*3.40E–01*	*7.59E–01*	*2.90E+01*	*2.23E+02*	*3.05E+01*	3.36E+01	*1.39E+01*	*1.67E+01*	4.98E+01	1.08E–35	6.27E–37	3.36E+01

Table 4 continued

Algorithm	F_9			F_{10}			F_{11}			F_{12}		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	*−4.50E+02*	*5.83E–14*	*1.88E+01*	−4.15E+02	8.22E+00	*2.38E+01*	1.27E+06	5.69E+06	*2.21E+01*	9.73E+02	2.84E+01	*2.42E+01*
DE/rand/1_2	*−4.50E+02*	1.49E–13	3.10E+01	*−4.45E+02*	*5.24E–01*	4.22E+01	*2.71E+03*	*3.64E+03*	3.29E+01	*−3.30E+02*	*7.04E–01*	3.32E+01
DE/best/1	−2.31E+02	4.27E+02	*2.19E+01*	−3.45E+02	4.25E+00	*3.00E+01*	1.28E+08	3.32E+08	*2.43E+01*	3.93E+02	9.97E+01	*2.39E+01*
DE/best/1_2	*−4.50E+02*	*1.46E–13*	3.01E+01	*−4.44E+02*	*5.52E–01*	3.06E+01	*6.85E+02*	*9.10E+01*	3.10E+01	*−3.30E+02*	*5.05E–14*	3.15E+01
DE/rand/2	*−4.50E+02*	*7.02E–14*	*2.28E+01*	−3.94E+02	5.44E+00	*2.22E+01*	6.99E+02	6.87E+01	*2.35E+01*	8.39E+02	6.40E+01	*2.61E+01*
DE/rand/2_2	*−4.50E+02*	1.33E–13	2.94E+01	*−4.46E+02*	*3.59E–01*	2.76E+01	*6.31E+02*	*5.78E+01*	3.10E+01	*−3.29E+02*	*1.42E+00*	3.14E+01
DE/current-to-best/1	1.40E+05	2.36E+04	*2.28E+01*	−3.57E+02	3.89E+00	*2.25E+01*	3.04E+10	7.19E+09	*2.36E+01*	5.96E+02	6.42E+01	*2.44E+01*
DE/current-to-best/1_2	*−4.50E+02*	*1.19E–13*	2.95E+01	*−4.47E+02*	*6.12E–01*	2.86E+01	*6.30E+02*	*7.59E+01*	3.05E+01	*2.01E+02*	*6.79E+01*	3.24E+01
DE/best/2	*−4.50E+02*	*5.98E–14*	*2.24E+01*	−3.88E+02	1.18E+01	*2.27E+01*	7.35E+02	1.35E+02	*2.40E+01*	8.59E+02	7.66E+01	*2.60E+01*
DE/best/2_2(myDE)	*−4.50E+02*	1.20E–13	2.88E+01	*−4.46E+02*	*4.03E–01*	2.85E+01	*6.14E+02*	*5.40E+01*	3.18E+01	*−3.30E+02*	*8.76E–01*	3.25E+01

Algorithm	F_{13}			F_{14}			F_{15}			F_{16}		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	*−1.80E+02*	*5.83E–14*	2.61E+01	*−1.40E+02*	*4.75E–14*	1.96E+01	−1.77E+03	1.47E+01	95.68667169	9.778E–01	2.441E–02	9.928E+01
DE/rand/1_2	*−1.80E+02*	4.34E–03	3.78E+01	*−1.40E+02*	*5.09E–14*	3.17E+01	*−2.43E+03*	3.85E+02	112.574106	5.541E+01	1.108E+01	1.052E+02
DE/best/1	−1.74E+02	9.52E+00	2.77E+01	−1.25E+02	1.38E+00	2.31E+01	−2.74E+03	3.55E+01	92.34984997	8.085E+02	6.808E+01	1.013E+02
DE/best/1_2	*−1.80E+02*	*4.28E–03*	3.61E+01	*−1.40E+02*	*3.77E–02*	3.01E+01	*−3.02E+03*	5.73E+00	101.7870566	9.700E+01	8.039E+00	1.050E+02
DE/rand/2	*−1.80E+02*	3.09E–01	2.81E+01	*−1.40E+02*	3.85E–01	2.19E+01	−1.82E+03	2.34E+01	97.68260068	1.431E+03	2.867E+01	1.026E+02
DE/rand/2_2	*−1.80E+02*	*2.75E–03*	3.57E+01	*−1.40E+02*	*4.07E–14*	2.99E+01	*−2.70E+03*	7.94E+01	105.221965	8.330E+01	6.753E+00	1.084E+02
DE/current-to-best/1	9.46E+02	1.21E+02	2.97E+01	−1.23E+02	4.57E–01	2.36E+01	−1.83E+03	5.70E+01	96.60881951	7.476E+02	4.991E+01	1.018E+02
DE/current-to-best/1_2	*−1.80E+02*	*3.14E–03*	3.56E+01	*−1.40E+02*	*3.80E–14*	2.99E+01	*−2.05E+03*	2.78E+01	108.064947	1.096E+02	8.962E+00	1.048E+02
DE/best/2	*−1.80E+02*	2.39E–02	2.81E+01	*−1.40E+02*	4.90E–01	2.17E+01	−1.80E+03	3.02E+01	97.78035343	9.104E+00	1.030E+01	9.892E+01
DE/best/2_2(myDE)	*−1.80E+02*	*3.70E–03*	3.68E+01	*−1.40E+02*	*4.02E–14*	3.04E+01	*−2.70E+03*	8.14E+01	106.4930098	8.625E+01	1.201E+01	1.068E+02

Italics represents the best results obtained with the corresponding mutation operator

Table 5 Experimental results (Mean, SD and mean CPU runtime) ($D = 500$)

Algorithm	F_1			F_2			F_3			F_4		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	1.19E–07	2.16E–08	7.72E+01	1.78E–12	7.29E–13	1.16E+02	−1.20E+02	*1.43E+00*	2.99E+02	1.25E+00	1.39E+00	2.84E+02
DE/rand/1_2	*1.64E–13*	*1.25E–14*	9.14E+01	*1.47E–13*	*5.83E–13*	1.29E+02	*−2.11E+02*	8.48E+00	3.17E+02	*2.09E–11*	*3.91E–11*	3.43E+02
DE/best/1	1.52E+01	4.96E–01	8.82E+01	1.09E+01	2.77E+01	1.17E+02	−1.27E+02	*3.86E+00*	3.42E+02	3.53E+02	2.97E+01	3.54E+02
DE/best/1_2	*2.55E–13*	*7.77E–15*	9.11E+01	*6.16E–04*	*2.75E–03*	1.29E+02	*−4.08E+02*	2.22E+01	3.94E+02	*6.52E–26*	*2.84E–27*	3.69E+02
DE/rand/2	2.37E+00	3.91E–01	8.72E+01	5.15E–02	1.90E–01	1.20E+02	−1.13E+02	*2.86E+00*	3.32E+02	1.27E+02	2.00E+01	2.56E+02
DE/rand/2_2	*1.45E–13*	*6.36E–15*	8.74E+01	*1.25E–15*	*2.07E–16*	1.29E+02	*−2.50E+02*	5.73E+00	3.34E+02	*4.03E–31*	*6.28E–31*	2.58E+02
DE/current-to-best/1	1.53E+01	3.22E–01	9.13E+01	1.57E+03	1.71E+02	1.40E+02	−3.11E+02	6.88E+00	3.45E+02	3.70E+02	5.66E+01	2.50E+02
DE/current-to-best/1_2	*2.25E–13*	*1.05E–14*	8.60E+01	*4.93E–04*	*2.20E–03*	1.29E+02	*−2.45E+02*	*3.06E+00*	3.35E+02	*1.22E–30*	*1.64E–30*	2.57E+02
DE/best/2	2.42E+00	3.89E–01	8.66E+01	2.87E–02	6.85E–02	1.16E+02	−1.14E+02	*1.93E+00*	3.94E+02	1.28E+02	2.22E+01	3.60E+02
DE/best/2_2(myDE)	*1.45E–13*	*6.26E–15*	8.65E+01	*1.20E–15*	*2.34E–16*	1.27E+02	*−2.50E+02*	4.89E+00	4.01E+02	*2.93E–31*	*7.07E–31*	3.21E+02

Algorithm	F_5			F_6			F_7			F_8		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	5.00E+03	7.38E+01	9.42E+01	*5.03E+02*	*2.43E+01*	7.75E+01	1.57E+05	1.12E+03	1.30E+02	2.97E–13	1.22E–13	7.39E+01
DE/rand/1_2	*3.28E+00*	*2.15E+00*	1.00E+02	5.84E+02	7.90E+01	9.29E+01	*1.13E–09*	*5.71E–11*	1.54E+02	*8.33E–35*	*5.09E–36*	9.05E+01
DE/best/1	2.14E+03	2.25E+02	9.19E+01	4.62E+04	*4.97E+04*	8.40E+01	7.43E+04	3.82E+03	1.31E+02	5.40E+01	9.42E+01	7.96E+01
DE/best/1_2	*3.77E–12*	*6.78E–13*	9.52E+01	*6.34E+02*	7.55E+01	9.36E+01	*1.02E–09*	*1.61E–11*	1.52E+02	*2.92E–25*	*1.36E–26*	8.82E+01
DE/rand/2	3.09E+03	1.01E+03	1.01E+02	8.71E+02	1.21E+02	1.05E+02	5.25E+04	2.96E+03	1.22E+02	*1.59E–39*	*1.71E–39*	8.24E+01
DE/rand/2_2	*5.61E+00*	*6.58E+00*	9.58E+01	*5.36E+02*	*5.01E+01*	1.05E+02	*7.62E+01*	*9.11E+01*	1.20E+02	2.78E–35	1.10E–36	8.48E+01
DE/current-to-best/1	1.86E+03	1.38E+02	9.32E+01	2.60E+06	2.25E+05	8.82E+01	1.12E+05	*4.01E+03*	1.35E+02	3.99E+03	3.28E+02	8.22E+01
DE/current-to-best/1_2	*1.02E+03*	*7.40E+01*	9.99E+01	*5.45E+02*	*7.17E+01*	1.05E+02	*1.07E+05*	6.56E+03	1.45E+02	*2.78E–35*	*8.99E–37*	8.35E+01
DE/best/2	3.31E+03	8.22E+02	1.01E+02	8.37E+02	1.45E+02	8.60E+01	5.15E+04	3.10E+03	1.35E+02	*2.48E–39*	*6.07E–39*	8.19E+01
DE/best/2_2(myDE)	*5.69E+00*	*5.67E+00*	9.51E+01	*5.49E+02*	*4.77E+01*	8.69E+01	*6.14E+01*	*8.17E+01*	1.47E+02	2.70E–35	9.84E–37	8.37E+01

Table 5 continued

Algorithm	F_9 Mean	SD	Runtime	F_{10} Mean	SD	Runtime	F_{11} Mean	SD	Runtime	F_{12} Mean	SD	Runtime
DE/rand/1	*−4.50E+02*	3.00E−13	8.99E+01	−3.70E+02	1.50E+01	1.14E+02	*9.23E+02*	*4.84E+01*	9.16E+01	3.97E+03	7.15E+01	1.20E+02
DE/rand/1_2	*−4.50E+02*	*9.22E−14*	1.08E+02	*−4.34E+02*	*8.90E−01*	1.31E+02	1.76E+03	1.88E+03	1.13E+02	*−3.29E+02*	*7.04E−01*	1.32E+02
DE/best/1	4.06E+03	7.91E+03	9.88E+01	−3.18E+02	3.39E+00	1.21E+02	2.67E+09	4.54E+09	9.82E+01	2.93E+03	2.62E+02	1.29E+02
DE/best/1_2	*−4.50E+02*	*5.53E−14*	1.08E+02	*−4.32E+02*	*7.87E−01*	1.32E+02	*1.07E+03*	*1.50E+02*	1.09E+02	*−3.30E+02*	*3.91E−14*	1.62E+02
DE/rand/2	*−4.50E+02*	2.63E−13	1.24E+02	−3.46E+02	4.51E+00	1.14E+02	1.25E+03	1.31E+02	1.06E+02	8.14E+02	3.57E+02	1.52E+02
DE/rand/2_2	*−4.50E+02*	*6.52E−14*	1.21E+02	*−4.35E+02*	*7.74E−01*	1.02E+02	*9.52E+02*	*5.70E+01*	1.06E+02	*−3.29E+02*	*1.71E+00*	1.61E+02
DE/current-to-best/1	8.64E+05	6.39E+04	1.08E+02	−3.38E+02	4.03E+00	1.23E+02	3.05E+11	2.94E+10	1.03E+02	3.71E+03	1.50E+02	1.48E+02
DE/current-to-best/1_2	*−4.50E+02*	*6.52E−14*	1.20E+02	*−4.34E+02*	*7.12E−01*	1.24E+02	*9.10E+02*	*4.31E+01*	1.05E+02	*9.60E+02*	*6.92E+01*	1.64E+02
DE/best/2	*−4.50E+02*	9.67E−13	1.03E+02	−3.48E+02	4.20E+00	1.23E+02	1.28E+03	1.36E+02	1.03E+02	8.34E+02	4.08E+02	1.54E+02
DE/best/2_2(myDE)	*−4.50E+02*	*6.78E−14*	1.02E+02	*−4.35E+02*	*8.07E−01*	1.25E+02	*9.64E+02*	*5.63E+01*	1.05E+02	*−3.29E+02*	*1.58E+00*	1.62E+02

Algorithm	F_{13} Mean	SD	Runtime	F_{14} Mean	SD	Runtime	F_{15} Mean	SD	Runtime	F_{16} Mean	SD	Runtime
DE/rand/1	*−1.80E+02*	*1.18E−13*	1.75E+02	*−1.40E+02*	2.18E−08	1.33E+02	−3.961E+03	2.200E+01	7.891E+02	2.74E+02	3.57E+01	5.55E+02
DE/rand/1_2	*−1.80E+02*	2.28E−03	1.85E+02	*−1.40E+02*	*5.98E−14*	1.56E+02	*−7.149E+03*	*1.121E+02*	*7.860E+02*	*1.86E+02*	*8.40E+00*	5.71E+02
DE/best/1	−1.32E+02	7.89E+01	1.64E+02	−1.21E+02	3.50E−01	1.44E+02	−6.353E+03	5.469E+01	8.099E+02	3.03E+03	1.51E+02	5.61E+02
DE/best/1_2	*−1.80E+02*	*2.76E−03*	1.73E+02	*−1.40E+02*	*4.37E−14*	1.50E+02	*−7.398E+03*	*1.586E+01*	*7.790E+02*	*2.15E+02*	*2.14E+01*	5.48E+02
DE/rand/2	*−1.80E+02*	5.00E−01	1.67E+02	−1.37E+02	7.94E−01	1.45E+02	−3.952E+03	1.558E+01	8.251E+02	5.65E+02	7.46E+01	5.17E+02
DE/rand/2_2	*−1.80E+02*	*5.34E−14*	1.71E+02	*−1.40E+02*	*5.90E−14*	1.45E+02	*−5.416E+03*	*4.804E+01*	*7.997E+02*	*2.00E+02*	*1.89E+01*	5.20E+02
DE/current-to-best/1	6.72E+03	3.39E+02	1.80E+02	−1.21E+02	1.84E−01	1.45E+02	−5.054E+03	6.384E+02	8.116E+02	2.98E+03	4.45E+01	5.12E+02
DE/current-to-best/1_2	*−1.80E+02*	*3.14E−03*	1.71E+02	*−1.40E+02*	*7.74E−14*	1.44E+02	*−5.133E+03*	*3.559E+01*	*7.894E+02*	*5.43E+02*	*3.70E+01*	5.15E+02
DE/best/2	−1.79E+02	4.17E+00	1.66E+02	−1.36E+02	2.34E+00	1.46E+02	−4.058E+03	5.433E+01	8.112E+02	5.57E+02	3.82E+01	5.21E+02
DE/best/2_2(myDE)	*−1.80E+02*	*3.80E−14*	1.70E+02	*−1.40E+02*	*7.02E−14*	1.46E+02	*−6.342E+03*	*1.836E+02*	*7.836E+02*	*2.07E+02*	*1.62E+01*	5.29E+02

Italics represents the best results obtained with the corresponding mutation operator

Table 6 Experimental results (Mean, SD and mean CPU runtime) ($D = 1000$)

Algorithm	F_1			F_2			F_3			F_4		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	2.47E–04	5.78E–05	3.78E+02	8.73E–06	4.42E–06	5.54E+02	−1.99E+02	2.29E+00	1.16E+03	1.84E+01	2.81E+00	8.39E+02
DE/rand/1_2	*2.64E–13*	*8.85E–15*	*3.51E+02*	*4.31E–15*	*4.47E–16*	*5.21E+02*	−3.88E+02	*5.37E+00*	*1.16E+03*	*1.11E–09*	*2.47E–09*	*8.22E+02*
DE/best/1	1.68E+01	3.08E–01	4.46E+02	1.96E+02	7.72E+01	5.65E+02	−2.07E+02	2.14E+00	1.21E+03	8.05E+02	8.21E+01	8.46E+02
DE/best/1_2	*1.23E–08*	*9.47E–11*	*3.47E+02*	*7.73E–15*	*3.56E–16*	*5.20E+02*	−6.34E+02	*1.66E+01*	*1.17E+03*	*1.33E–25*	*2.76E–27*	*8.13E+02*
DE/rand/2	7.47E+00	1.19E+00	4.51E+02	1.34E–01	2.33E–01	6.58E+02	−1.96E+02	4.13E+00	1.24E+03	4.21E+02	1.07E+01	8.68E+02
DE/rand/2_2	*2.89E–13*	*4.63E–15*	*3.40E+02*	*3.31E–15*	*1.65E–16*	*5.67E+02*	−4.77E+02	*8.07E+00*	*1.17E+03*	*3.44E–31*	*2.83E–31*	*8.10E+02*
DE/current-to-best/1	1.69E+01	1.91E–01	4.71E+02	4.52E+03	1.89E+02	6.93E+02	−5.13E+02	9.65E+00	1.23E+03	1.38E+03	8.88E+01	8.57E+02
DE/current-to-best/1_2	*4.29E–13*	*8.18E–15*	*3.40E+02*	*7.22E–15*	*1.15E–15*	*5.73E+02*	−5.85E+02	*3.96E+00*	*1.17E+03*	*2.72E–30*	*1.97E–30*	*8.06E+02*
DE/best/2	8.88E+00	6.11E–01	4.54E+02	2.66E–01	4.34E–01	5.71E+02	−1.99E+02	3.36E+00	1.24E+03	4.51E+02	5.13E+01	8.69E+02
DE/best/2_2(myDE)	*2.79E–13*	*6.36E–15*	*3.42E+02*	*3.46E–15*	*3.08E–16*	*4.96E+02*	−4.75E+02	*3.98E+00*	*1.17E+03*	*1.77E–31*	*1.50E–31*	*8.08E+02*
Algorithm	F_5			F_6			F_7			F_8		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	1.10E+04	1.11E+02	3.71E+02	1.23E+03	8.70E+01	2.92E+02	3.44E+05	1.34E+03	3.45E+02	2.09E–06	1.21E–06	2.47E+02
DE/rand/1_2	*8.86E+00*	*3.95E+00*	*3.22E+02*	*1.24E+03*	*6.29E+01*	*2.67E+02*	*2.32E–09*	*7.59E–11*	*3.08E+02*	*1.70E–34*	*5.49E–36*	*2.31E+02*
DE/best/1	5.14E+03	2.14E+02	3.58E+02	3.48E+05	2.12E+05	3.03E+02	1.62E+05	1.40E+03	3.45E+02	6.23E+02	2.83E+02	2.66E+02
DE/best/1_2	*1.96E–11*	*8.13E–13*	*3.25E+02*	*1.16E+03*	*4.30E+01*	*2.60E+02*	*2.40E–09*	*6.38E–11*	*3.04E+02*	*6.06E–25*	*1.82E–26*	*2.26E+02*
DE/rand/2	2.63E+03	1.30E+02	4.15E+02	1.80E+03	2.41E+02	3.24E+02	1.08E+05	4.28E+03	3.61E+02	8.62E–22	1.82E–21	2.85E+02
DE/rand/2_2	*1.05E+01*	*6.03E+00*	*3.40E+02*	*1.25E+03*	*1.64E+02*	*2.54E+02*	*1.22E+02*	*5.82E+01*	*2.90E+02*	*5.71E–35*	*1.61E–36*	*2.21E+02*
DE/current-to-best/1	5.64E+03	2.40E+02	4.26E+02	9.91E+06	6.75E+05	3.24E+02	2.64E+05	4.26E+03	3.57E+02	1.28E+04	1.01E+03	2.83E+02
DE/current-to-best/1_2	*2.05E+03*	*1.04E+02*	*3.66E+02*	*1.30E+03*	*4.63E+01*	*2.54E+02*	*2.21E+05*	*2.98E+03*	*2.91E+02*	*5.68E–35*	*4.02E–37*	*2.19E+02*
DE/best/2	2.64E+03	3.31E+02	4.27E+02	1.96E+03	2.37E+02	3.27E+02	1.03E+05	4.20E+03	3.70E+02	1.67E–22	2.14E–22	2.85E+02
DE/best/2_2(myDE)	*8.83E+00*	*4.52E+00*	*3.37E+02*	*1.15E+03*	*8.79E+01*	*2.54E+02*	*7.09E+01*	*4.34E+01*	*2.95E+02*	*5.67E–35*	*1.63E–36*	*2.20E+02*

Table 6 continued

Algorithm	F_9			F_{10}			F_{11}			F_{12}		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	*−4.50E+02*	1.89E–05	2.78E+02	−3.34E+02	9.35E+00	2.82E+02	1.78E+04	6.53E+03	4.13E+02	9.33E+03	1.67E+02	4.08E+02
DE/rand/1_2	*−4.50E+02*	*8.04E–14*	*2.54E+02*	*−4.23E+02*	*8.93E–01*	*2.55E+02*	*1.74E+03*	*1.15E+02*	*3.79E+02*	*−3.28E+02*	*1.48E+00*	*3.69E+02*
DE/best/1	4.37E+04	2.62E+04	2.99E+02	−3.02E+02	3.44E+00	3.07E+02	3.66E+10	2.56E+10	4.34E+02	8.48E+03	5.32E+02	3.98E+02
DE/best/1_2	*−4.50E+02*	*5.68E–14*	*2.48E+02*	*−4.20E+02*	*7.67E–01*	*2.49E+02*	*1.70E+03*	*1.25E+02*	*3.65E+02*	*−3.30E+02*	*4.92E–14*	*3.55E+02*
DE/rand/2	−4.43E+02	1.64E+01	3.11E+02	−3.12E+02	4.12E+00	3.50E+02	2.61E+06	2.67E+06	4.23E+02	2.79E+03	2.76E+02	4.43E+02
DE/rand/2_2	*−4.50E+02*	*8.99E–14*	*2.44E+02*	*−4.26E+02*	*6.74E–01*	*3.09E+02*	*1.57E+03*	*8.32E+01*	*3.43E+02*	*−3.24E+02*	*4.70E+00*	*3.74E+02*
DE/current-to-best/1	2.53E+06	6.59E+04	3.09E+02	−3.23E+02	3.05E+00	3.12E+02	1.19E+12	8.36E+10	4.35E+02	1.10E+04	3.88E+02	4.53E+02
DE/current-to-best/1_2	*−4.50E+02*	*1.02E–13*	*2.43E+02*	*−4.26E+02*	*4.98E–01*	*2.41E+02*	*1.41E+03*	*5.87E+01*	*3.42E+02*	*2.24E+03*	*3.65E+01*	*3.94E+02*
DE/best/2	3.22E+01	6.62E+02	3.13E+02	−3.14E+02	3.86E+00	3.15E+02	2.68E+07	5.99E+07	4.43E+02	2.78E+03	5.34E+01	4.49E+02
DE/best/2_2(myDE)	*−4.50E+02*	*4.92E–14*	*2.42E+02*	*−4.26E+02*	*1.02E+00*	*2.43E+02*	*1.59E+03*	*4.39E+01*	*3.60E+02*	*−3.21E+02*	*8.78E+00*	*3.76E+02*

Algorithm	F_{13}			F_{14}			F_{15}			F_{16}		
	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime	Mean	SD	Runtime
DE/rand/1	−1.78E+02	5.23E+00	6.07E+02	*−1.40E+02*	7.53E–05	3.86E+02	−7.20E+03	2.31E+01	1.86E+03	1.37E+03	5.04E+02	2.62E+03
DE/rand/1_2	*−1.80E+02*	*3.31E–03*	*5.88E+02*	*−1.40E+02*	*1.21E–13*	*3.66E+02*	*−9.35E+03*	1.11E+02	1.89E+03	*3.43E+02*	*3.73E+00*	*2.91E+03*
DE/best/1	2.32E+02	2.42E+02	5.49E+02	−1.20E+02	4.08E–02	4.20E+02	−1.21E+04	4.17E+01	1.85E+03	4.33E+03	1.66E+02	2.43E+03
DE/best/1_2	*−1.80E+02*	*4.41E–03*	*4.94E+02*	*−1.40E+02*	*1.02E–13*	*3.60E+02*	*−1.42E+04*	1.96E+02	1.59E+03	*7.66E+01*	*1.36E+01*	*2.61E+03*
DE/rand/2	*−1.80E+02*	4.84E–01	5.52E+02	−1.27E+02	1.81E+00	4.52E+02	−7.40E+03	8.50E+01	1.69E+03	1.61E+04	1.50E+02	2.53E+03
DE/rand/2_2	*−1.80E+02*	*3.48E–14*	*4.85E+02*	*−1.40E+02*	*5.68E–14*	*3.48E+02*	*−1.14E+04*	2.74E+02	1.79E+03	*8.32E+01*	*1.42E+01*	*2.41E+03*
DE/current-to-best/1	2.18E+04	4.80E+02	6.13E+02	−1.20E+02	1.05E–01	4.56E+02	−9.27E+03	7.20E+02	1.85E+03	7.23E+03	1.61E+02	2.22E+03
DE/current-to-best/1_2	*−1.80E+02*	*6.19E–14*	*4.88E+02*	*−1.40E+02*	*1.52E–13*	*3.59E+02*	*−9.93E+03*	5.10E+01	1.72E+03	*2.60E+02*	*8.05E+00*	*2.11E+03*
DE/best/2	−1.54E+02	3.46E+01	5.51E+02	−1.27E+02	3.24E+00	4.54E+02	−7.33E+03	4.81E+01	1.67E+03	3.14E+03	2.33E+02	2.72E+03
DE/best/2_2(myDE)	*−1.80E+02*	*6.19E–14*	*4.85E+02*	*−1.40E+02*	*8.76E–14*	*3.57E+02*	*−1.16E+04*	1.23E+02	1.71E+03	*4.51E+02*	*8.05E+00*	*2.52E+03*

Italics represents the best results obtained with the corresponding mutation operator

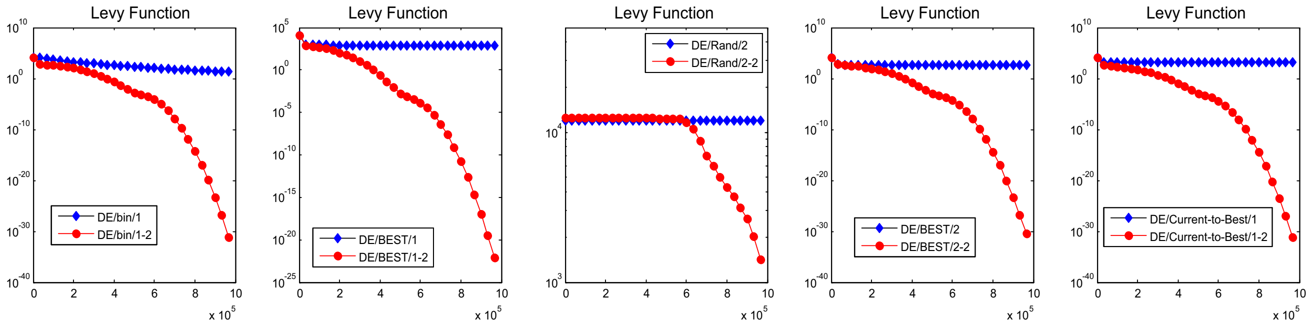

Fig. 13 Convergence curves of Levy function for comparison between five standard DE and five improved DE on $D = 1000$

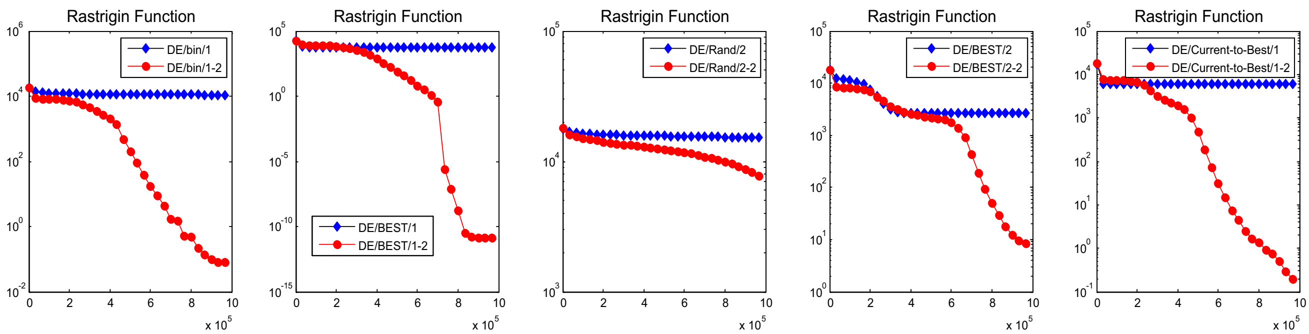

Fig. 14 Convergence curves of Rastrigin function for comparison between five standard DE and five improved DE on $D = 1000$

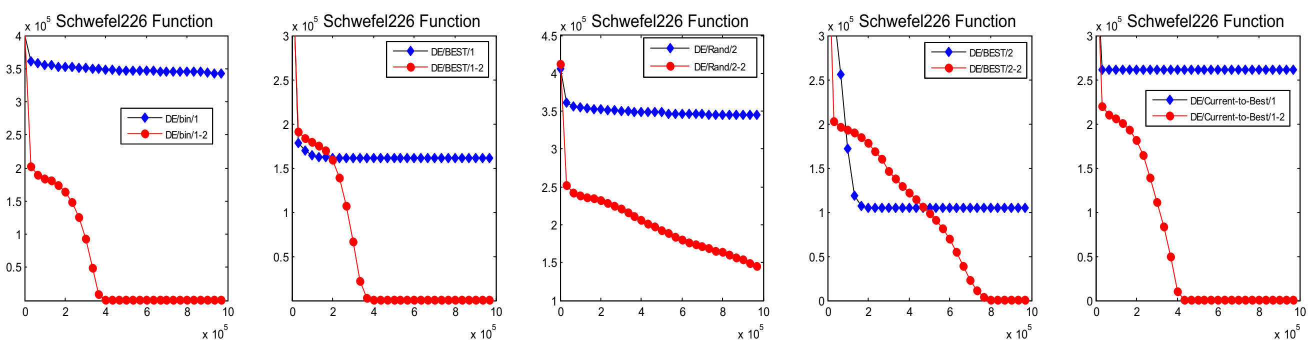

Fig. 15 Convergence curves of Schwefel 2.26 function for comparison between five standard DE and five improved DE on $D = 1000$

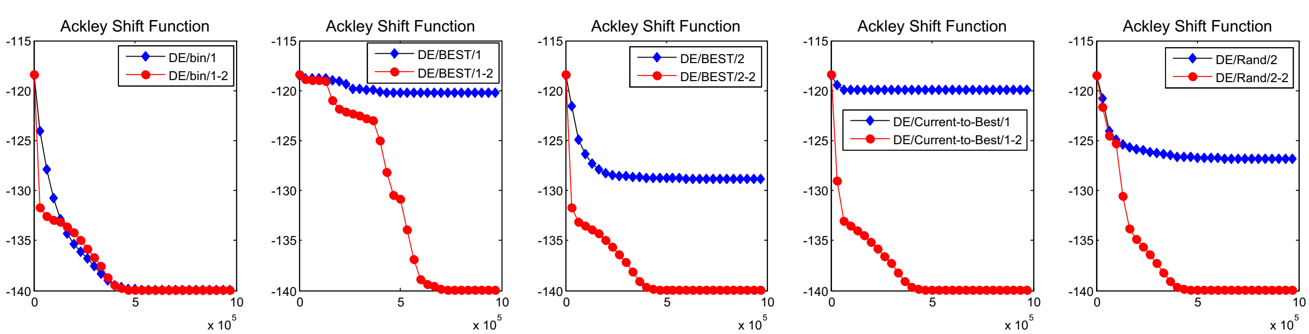

Fig. 16 Convergence curves of Ackley Shift function for comparison between five standard DE and five improved DE on $D = 1000$

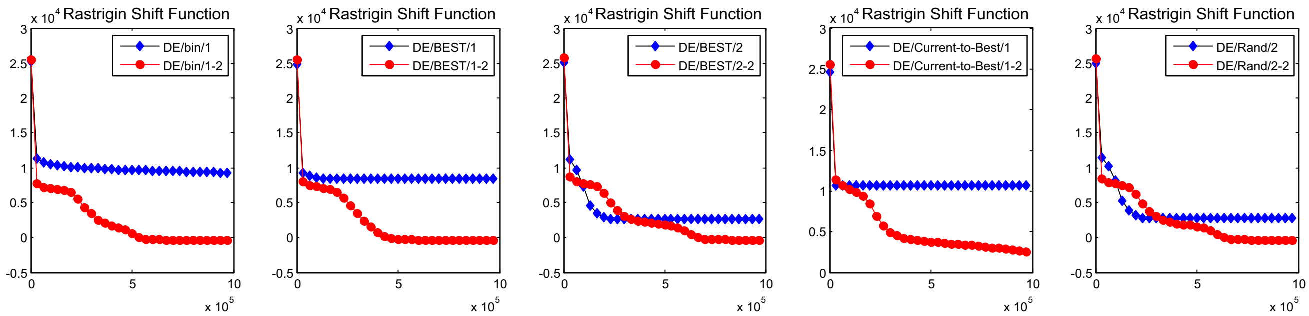

Fig. 17 Convergence curves of Rastrigin Shift function for comparison between five standard DE and five improved DE on $D = 1000$

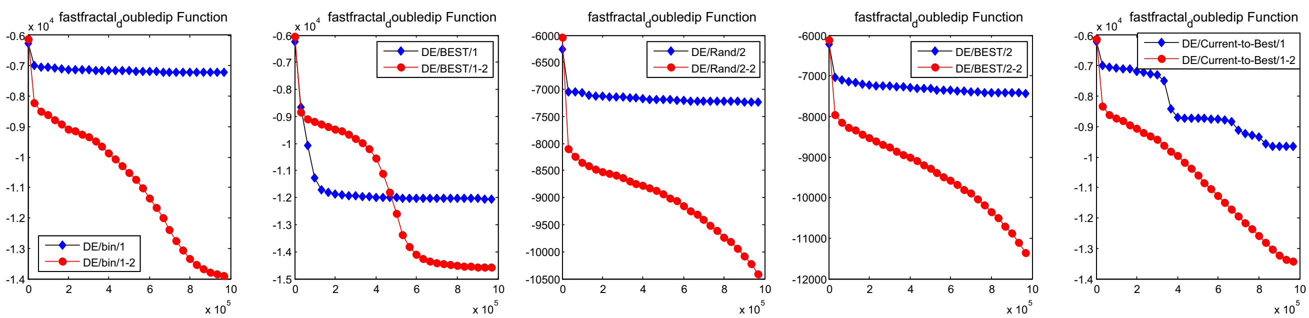

Fig. 18 Convergence curves of Fastfractal doubledip function for comparison between five standard DE and five improved DE on $D = 1000$

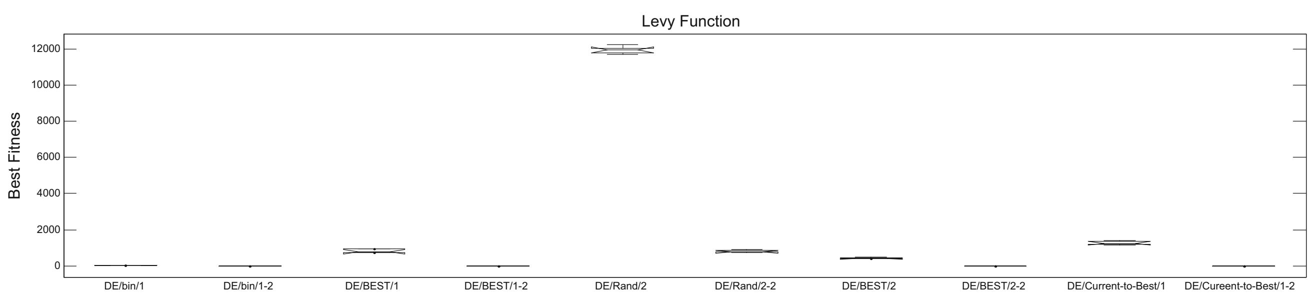

Fig. 19 Distribution boxplot of Levy function for comparison between five standard DE and five improved DE on $D = 1000$

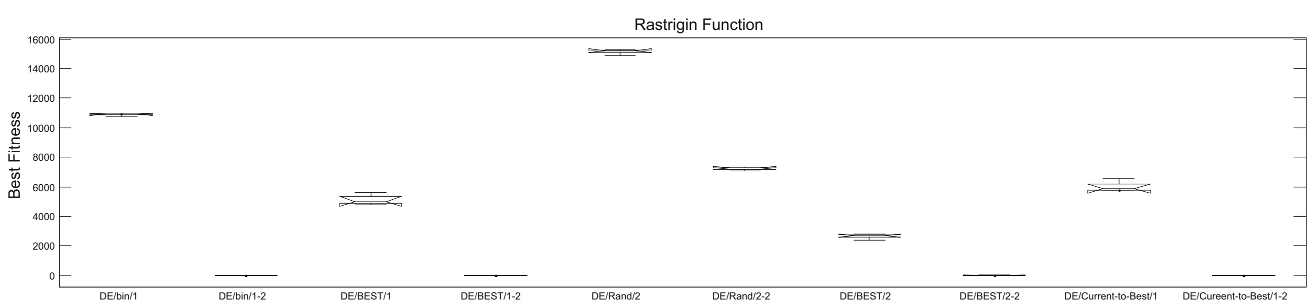

Fig. 20 Distribution boxplot of Rastrigin function for comparison between five standard DE and five improved DE on $D = 1000$

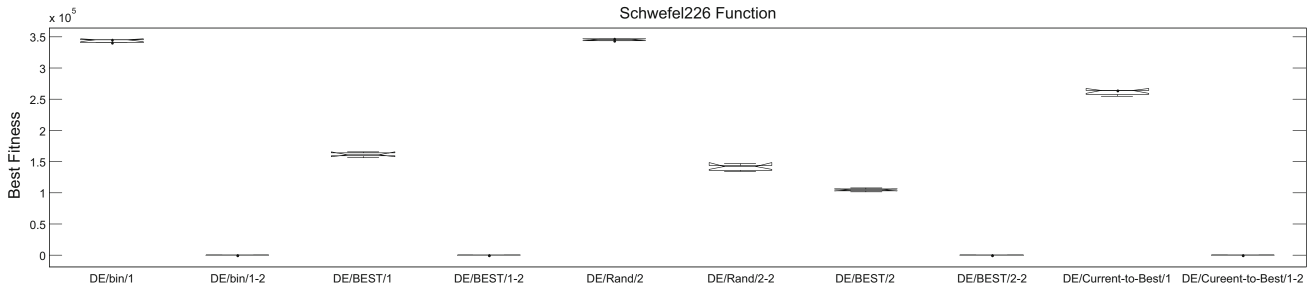

Fig. 21 Distribution boxplot of Schwefel 2.26 function for comparison between five standard DE and five improved DE on $D = 1000$

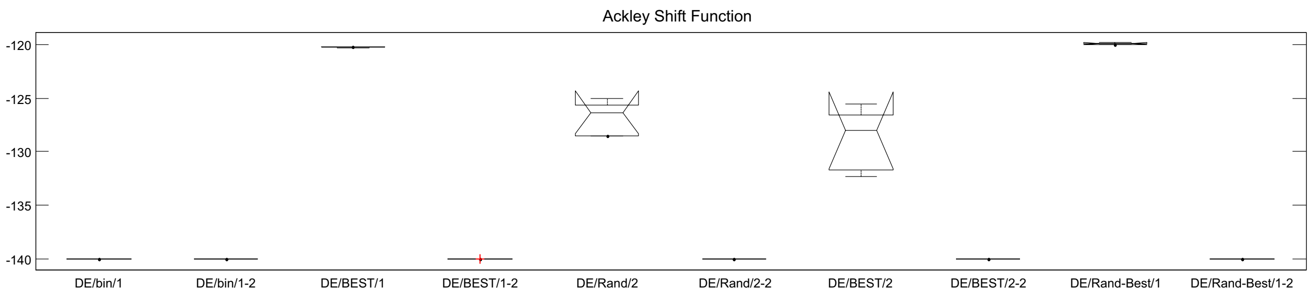

Fig. 22 Distribution boxplot of Ackley shift function for comparison between five standard DE and five improved DE on $D = 1000$

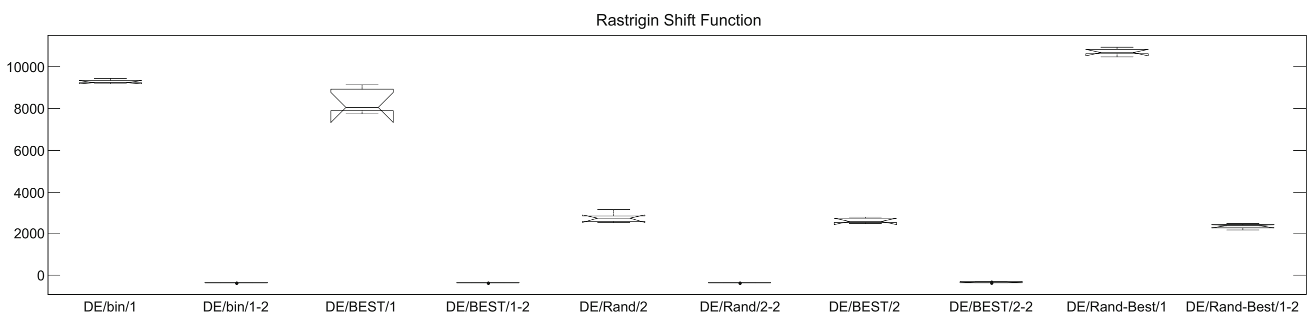

Fig. 23 Distribution boxplot of Rastrigin function for comparison between five standard DE and five improved DE on $D = 1000$

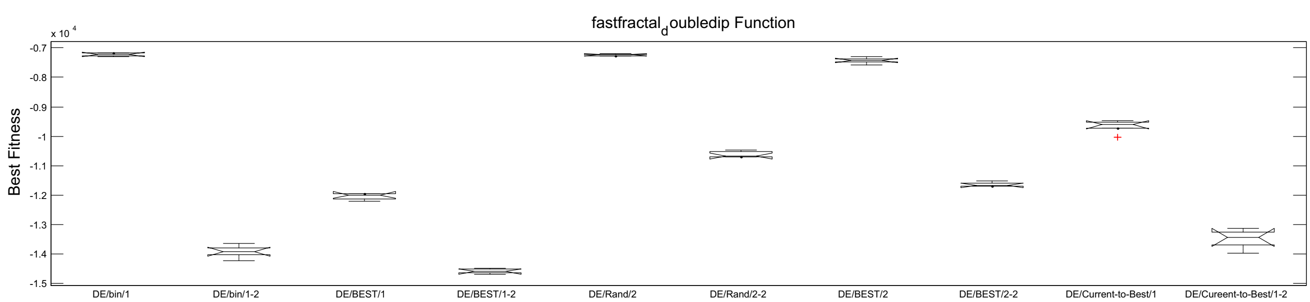

Fig. 24 Distribution boxplot of Fastfractal doubledip function for comparison between five standard DE and five improved DE on $D = 1000$

It indicates from the distribution box plots in Fig. 26 that the distribution of optimal solutions of the myDE algorithm is narrowest among all algorithms, which demonstrates that myDE is more stable than other algorithms.

As shown in Table 10, myDE algorithm has nearly the same advantages as MTS algorithm. MTS which uses complex multiple agents to search the solution space concurrently; each agent does an iterated local search using one of three candidate local search methods. Compared with two DE variants (jDEdynNP-F, DEwSAcc) and other four algorithms, myDE algorithm has very obvious advantages on most of functions.

Table 7 Experimental results (Best, Mean, Worst, SD and mean CPU run time) of four algorithm (myDE, CMAES, SaDE, and CoDE) ($D = 200$)

Algorithm	Fun	Best	Mean	Worst	SD	Run time	Fun	Best	Mean	Worst	SD	Run time
myDE	F_1	*1.50E–13*	*1.66E–13*	*1.86E–13*	*7.77E–15*	*2.01E+01*	F_5	*0.00E+00*	*2.61E–13*	*4.55E–13*	*1.53E–13*	*2.59E+01*
CMAES		1.87E+01	1.90E+01	1.91E+01	1.38E–01	4.17E+01		1.12E+03	1.34E+03	1.61E+03	1.09E+02	3.65E+01
CoDE		1.07E–13	2.60E–13	7.93E–13	1.68E–13	2.55E+01		3.34E+02	4.24E+02	5.21E+02	4.61E+01	3.34E+01
SaDE		4.01E+00	5.79E+00	6.93E+00	8.51E–01	1.32E+02		3.98E+01	5.75E+01	6.77E+01	7.00E+00	1.26E+02
myDE	F_2	1.11E–16	*7.40E–04*	*7.40E–03*	*2.28E–03*	*3.16E+01*	F_6	1.87E+02	2.57E+02	3.91E+02	5.66E+01	2.22E+01
CMAES		1.11E–16	1.36E–03	1.23E–02	3.44E–03	4.24E+01		*9.51E+01*	*9.88E+01*	*1.14E+02*	*3.90E+00*	3.82E+01
CoDE		*0.00E+00*	8.61E–04	1.72E–02	3.85E–03	3.80E+01		1.64E+02	2.04E+02	2.81E+02	3.61E+01	2.87E+01
SaDE		*0.00E+00*	1.04E–01	4.58E–01	1.23E–01	1.78E+02		1.75E+02	3.21E+02	4.58E+02	7.54E+01	1.31E+02
myDE	F_3	*−1.99E+02*	*−1.98E+02*	*−1.95E+02*	*1.06E+00*	*5.70E+01*	F_7	*3.35E–10*	*3.47E–10*	*3.64E–10*	*7.12E–12*	*3.47E+01*
CMAES		−1.45E+02	−1.40E+02	−1.34E+02	3.03E+00	1.11E+02		3.24E+04	3.48E+04	3.91E+04	1.71E+03	3.85E+01
CoDE		−1.24E+02	−1.14E+02	−1.05E+02	4.63E+00	8.23E+01		9.58E+03	1.25E+04	1.51E+04	1.44E+03	3.85E+01
SaDE		−1.25E+02	−1.22E+02	−1.18E+02	1.97E+00	1.69E+02		1.07E+03	2.00E+03	3.51E+03	6.05E+02	1.47E+02
myDE	F_4	*2.17E–26*	*2.42E–26*	*2.73E–26*	*1.44E–27*	*8.12E+01*	F_8	9.85E–26	1.15E–25	1.28E–25	8.48E–27	*2.07E+01*
CMAES		3.45E+02	3.84E+02	4.47E+02	2.90E+01	9.21E+01		5.35E–28	6.59E–28	*8.41E–28*	*8.06E–29*	3.04E+01
CoDE		3.58E–01	1.73E+00	3.62E+00	8.78E–01	6.70E+01		1.40E–26	7.47E–26	2.79E–25	6.61E–26	1.98E+01
SaDE		5.29E+00	9.41E+00	1.46E+01	2.53E+00	1.74E+02		*6.16E–35*	*1.09E–28*	1.88E–27	4.19E–28	1.12E+02
myDE	F_9	*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	1.50E–13	*2.72E+01*	F_{13}	*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	3.87E–03	*3.90E+01*
CMAES		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	2.06E–13	3.81E+01		*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	6.96E–14	9.88E+01
CoDE		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	*4.33E–14*	2.55E+01		*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	*1.13E–14*	7.27E+01
SaDE		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	6.65E–14	1.18E+02		*−1.80E+02*	*−1.80E+02*	−1.78E+02	4.04E–01	1.76E+02
myDE	F_{10}	−4.45E+02	−4.44E+02	−4.43E+02	4.81E–01	3.24E+01	F_{14}	*−1.40E+02*	*−1.40E+02*	*−1.40E+02*	*4.24E–03*	*4.35E+01*
CMAES		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	*6.20E–12*	3.98E+01		−1.20E+02	−1.20E+02	−1.20E+02	3.33E–02	5.98E+01
CoDE		−4.42E+02	−4.24E+02	−4.09E+02	9.59E+00	4.47E+01		*−1.40E+02*	*−1.40E+02*	*−1.40E+02*	*9.05E–13*	3.35E+01
SaDE		−4.42E+02	−4.36E+02	−4.25E+02	5.16E+00	1.77E+02		−1.37E+02	−1.36E+02	−1.34E+02	8.27E–01	1.53E+02
myDE	F_{11}	5.80E+02	6.64E+02	8.11E+02	7.68E+01	3.33E+01	F_{15}	*−3.03E+03*	*−3.02E+03*	*−3.01E+03*	*4.86E+00*	*1.50E+02*
CMAES		*4.87E+02*	*5.01E+02*	*5.34E+02*	*1.59E+01*	4.55E+01		−2.77E+03	−2.67E+03	−2.58E+03	4.46E+01	1.58E+02
CoDE		5.81E+02	7.06E+02	9.32E+02	8.81E+01	3.09E+01		−2.23E+03	−2.02E+03	−1.92E+03	7.52E+01	1.63E+02
SaDE		7.85E+02	9.80E+02	2.04E+03	2.74E+02	1.25E+02		−2.20E+03	−2.16E+03	−2.13E+03	1.87E+01	2.59E+02
myDE	F_{12}	*−3.30E+02*	*−3.30E+02*	*−3.30E+02*	*1.06E–13*	*3.00E+01*	F_{16}	*5.84E+01*	*9.56E+01*	*1.11E+02*	*9.15E+00*	1.07E+02
CMAES		1.85E+03	2.25E+03	3.04E+03	2.61E+02	4.57E+01		1.72E+03	1.82E+03	1.88E+03	3.59E+01	1.61E+02
CoDE		6.17E+01	9.89E+01	1.46E+02	2.10E+01	4.12E+01		6.51E+01	9.58E+01	1.51E+02	2.52E+01	1.00E+02
SaDE		−2.04E+02	−1.44E+02	−9.42E+01	2.25E+01	1.75E+02		2.30E+02	3.01E+02	3.60E+02	3.65E+01	1.63E+02

Italics represents the best results obtained by the corresponding algorithm for each function

Table 8 Experimental results (Best, Mean, Worst, SD and mean CPU run time) of four algorithm (myDE, CMAES, SaDE, and CoDE) ($D = 500$)

Algorithm	Fun	Best	Mean	Worst	SD	Run time	Fun	Best	Mean	Worst	SD	Run time
myDE	F_1	*2.53E–13*	*2.59E–13*	*2.71E–13*	*6.71E–15*	*9.95E+01*	F_5	*2.73E–12*	*4.09E–12*	*4.55E–12*	*6.43E–13*	*1.16E+02*
CMAES		1.88E+01	1.90E+01	1.91E+01	8.89E–02	3.26E+02		2.40E+03	2.64E+03	3.03E+03	1.87E+02	3.14E+02
CoDE		2.05E+00	2.39E+00	3.15E+00	3.42E–01	1.05E+02		5.37E+02	6.32E+02	7.98E+02	8.61E+01	1.49E+02
SaDE		8.99E+00	9.65E+00	1.03E+01	4.19E–01	3.76E+02		2.32E+02	2.70E+02	2.95E+02	2.08E+01	3.16E+02
myDE	F_2	7.77E–16	8.66E–16	9.99E–16	8.76E–17	*1.60E+02*	F_6	4.89E+02	6.46E+02	7.96E+02	1.12E+02	8.69E+01
CMAES		6.66E–16	*7.55E–16*	*8.88E–16*	*1.02E–16*	4.01E+02		*2.55E+02*	*3.51E+02*	*3.76E+02*	*3.45E+01*	3.88E+02
CoDE		*1.11E–16*	3.17E–02	1.67E–01	6.14E–02	1.36E+02		5.70E+02	6.70E+02	7.67E+02	6.60E+01	1.05E+02
SaDE		1.34E–11	1.15E–01	4.02E–01	1.52E–01	3.59E+02		7.85E+02	9.26E+02	1.06E+03	1.08E+02	3.57E+02
myDE	F_3	*−4.43E+02*	*−4.23E+02*	*−3.95E+02*	*1.52E+01*	*3.20E+02*	F_7	*9.90E–10*	*1.02E–09*	*1.05E–09*	*1.65E–11*	*1.38E+02*
CMAES		−3.44E+02	−3.33E+02	−3.06E+02	1.09E+01	5.54E+02		7.94E+04	8.55E+04	9.02E+04	2.96E+03	3.41E+02
CoDE		−3.11E+02	−2.85E+02	−2.61E+02	1.80E+01	3.70E+02		1.23E+04	1.48E+04	2.40E+04	3.36E+03	1.91E+02
SaDE		−2.41E+02	−2.38E+02	−2.33E+02	2.89E+00	6.28E+02		2.05E+04	2.41E+04	2.88E+04	2.71E+03	3.78E+02
myDE	F_4	*6.35E–26*	*6.52E–26*	*6.76E–26*	*1.39E–27*	*2.83E+02*	F_8	2.79E–25	3.01E–25	3.22E–25	1.45E–26	*8.50E+01*
CMAES		9.28E+02	9.76E+02	1.05E+03	4.12E+01	5.06E+02		*2.11E–27*	*2.51E–27*	*2.77E–27*	*2.06E–28*	3.15E+02
CoDE		1.24E+01	1.54E+01	2.03E+01	3.04E+00	2.83E+02		1.46E–19	6.64E–18	5.29E–17	1.64E–17	8.72E+01
SaDE		1.87E+01	3.19E+01	4.34E+01	7.38E+00	4.94E+02		9.83E–13	1.25E–09	8.63E–09	2.64E–09	3.30E+02
myDE	F_9	*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	6.28E–14	*1.25E+02*	F_{13}	*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	*3.42E–14*	*1.67E+02*
CMAES		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	1.78E–13	3.52E+02		*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	2.34E–03	4.37E+02
CoDE		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	*5.99E–14*	1.03E+02		*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	3.32E–02	1.94E+02
SaDE		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	1.06E–06	3.55E+02		*−1.80E+02*	*−1.80E+02*	*−1.80E+02*	1.34E–01	4.46E+02
myDE	F_{10}	−4.36E+02	−4.35E+02	−4.34E+02	8.32E–01	*1.03E+02*	F_{14}	*−1.40E+02*	*−1.40E+02*	*−1.40E+02*	*4.92E–14*	*1.49E+02*
CMAES		*−4.50E+02*	*−4.50E+02*	*−4.50E+02*	*7.61E–07*	3.43E+02		−1.20E+02	−1.20E+02	−1.20E+02	1.27E–02	3.55E+02
CoDE		−3.96E+02	−3.68E+02	−3.59E+02	1.23E+01	1.41E+02		−1.38E+02	−1.38E+02	−1.37E+02	2.13E–01	1.38E+02
SaDE		−4.15E+02	−4.06E+02	−3.93E+02	6.40E+00	4.11E+02		−1.31E+02	−1.30E+02	−1.29E+02	4.10E–01	4.16E+02
myDE	F_{11}	9.26E+02	1.07E+03	1.32E+03	1.14E+02	*1.18E+02*	F_{15}	*−7.41E+03*	*−7.38E+03*	*−7.31E+03*	*4.04E+01*	*8.78E+02*
CMAES		*7.50E+02*	*7.60E+02*	*7.87E+02*	*1.39E+01*	7.22E+02		−6.48E+03	−6.37E+03	−6.30E+03	8.20E+01	1.86E+03
CoDE		1.30E+03	1.48E+03	1.76E+03	1.28E+02	2.01E+02		−6.91E+03	−6.85E+03	−6.78E+03	5.57E+01	7.97E+02
SaDE		1.62E+03	1.93E+03	2.59E+03	2.84E+02	4.36E+02		−5.03E+03	−4.99E+03	−4.97E+03	2.45E+01	1.08E+03
myDE	F_{12}	*−3.30E+02*	*−3.30E+02*	*−3.30E+02*	*6.83E–14*	*1.88E+02*	F_{16}	2.02E+02	2.25E+02	2.49E+02	2.04E+01	6.07E+02
CMAES		3.38E+03	4.11E+03	4.92E+03	4.91E+02	5.94E+02		4.47E+03	4.56E+03	4.64E+03	7.80E+01	8.79E+02
CoDE		4.18E+02	5.43E+02	7.31E+02	8.96E+01	2.54E+02		4.66E+02	4.96E+02	5.26E+02	2.67E+01	6.26E+02
SaDE		8.62E+02	1.12E+03	1.39E+03	1.89E+02	6.43E+02		6.00E+02	7.53E+02	8.88E+02	1.32E+02	8.82E+02

Italics represents the best results obtained by the corresponding algorithm for each function

Table 9 Experimental results (Best, Mean, Worst, SD and mean CPU run time) of four algorithm (myDE, CMAES, SaDE, and CoDE) ($D = 1000$)

Algorithm	Fun	Best	Mean	Worst	SD	Run time	Fun	Best	Mean	Worst	SD	Run time
myDE	F_1	*4.166E–13*	*4.301E–13*	*4.414E–13*	*9.858E–15*	*5.907E+02*	F_5	*1.273E–11*	*1.564E–11*	*1.819E–11*	*2.074E–12*	*6.962E+02*
CMAES		1.880E+01	1.889E+01	1.897E+01	6.424E–02	2.679E+03		4.639E+03	4.761E+03	4.931E+03	1.079E+02	5.290E+03
CoDE		6.759E+00	7.187E+00	7.519E+00	3.171E–01	6.571E+02		8.935E+02	9.270E+02	9.641E+02	2.888E+01	7.459E+02
SaDE		1.225E+01	1.249E+01	1.284E+01	2.134E–01	1.161E+03		6.308E+02	6.780E+02	7.333E+02	4.064E+01	1.331E+03
myDE	F_2	3.109E–15	3.175E–15	3.220E–15	*6.081E–17*	*6.731E+02*	F_6	1.088E+03	1.231E+03	1.296E+03	8.992E+01	*5.684E+02*
CMAES		*2.331E–15*	*2.465E–15*	*2.554E–15*	9.289E–17	3.195E+03		*6.347E+02*	*7.538E+02*	*8.183E+02*	*8.891E+01*	4.765E+03
CoDE		1.804E–05	2.260E–01	1.027E+00	4.479E–01	7.201E+02		1.927E+03	1.996E+03	2.057E+03	5.227E+01	7.336E+02
SaDE		1.343E–05	2.196E–01	6.676E–01	2.717E–01	1.176E+03		1.887E+03	2.055E+03	2.262E+03	1.813E+02	1.332E+03
myDE	F_3	*−6.703E+02*	*−6.358E+02*	*−6.147E+02*	*2.154E+01*	*1.715E+03*	F_7	*2.328E–09*	*2.367E–09*	*2.387E–09*	*3.361E–11*	*4.160E+02*
CMAES		−6.415E+02	−6.239E+02	−5.863E+02	2.186E+01	3.521E+03		1.619E+05	1.675E+05	1.717E+05	5.066E+03	5.849E+03
CoDE		−6.073E+02	−5.914E+02	−5.740E+02	1.344E+01	1.827E+03		5.552E+04	5.751E+04	5.958E+04	2.033E+03	5.155E+02
SaDE		−4.158E+02	−4.103E+02	−4.069E+02	4.019E+00	2.548E+03		8.816E+04	9.152E+04	9.440E+04	3.145E+03	8.786E+02
myDE	F_4	*1.302E–25*	*1.333E–25*	*1.364E–25*	*2.377E–27*	*2.401E+03*	F_8	5.764E–25	5.923E–25	6.105E–25	1.162E–26	*3.719E+02*
CMAES		1.775E+03	1.836E+03	1.924E+03	5.688E+01	6.619E+03		*5.407E–27*	*6.871E–27*	*7.723E–27*	*6.233E–28*	2.559E+03
CoDE		4.332E+01	5.117E+01	6.236E+01	7.847E+00	2.832E+03		3.888E–08	6.393E–07	3.699E–06	1.112E–06	4.401E+02
SaDE		8.074E+01	8.989E+01	1.084E+02	1.078E+01	3.760E+03		2.400E–06	1.992E–04	1.246E–03	3.783E–04	8.700E+02
myDE	F_9	*−4.500E+02*	*−4.500E+02*	*−4.500E+02*	*5.684E–14*	*4.364E+02*	F_{13}	*−1.800E+02*	*−1.800E+02*	*−1.800E+02*	*4.019E–14*	*1.698E+03*
CMAES		*−4.500E+02*	*−4.500E+02*	*−4.500E+02*	2.836E–13	2.645E+03		*−1.800E+02*	*−1.800E+02*	*−1.800E+02*	2.339E–03	3.853E+03
CoDE		*−4.500E+02*	*−4.500E+02*	*−4.500E+02*	1.233E–04	4.995E+02		*−1.800E+02*	−1.781E+02	−1.622E+02	5.592E+00	1.695E+03
SaDE		*−4.500E+02*	−4.488E+02	−4.396E+02	3.265E+00	9.269E+02		*−1.800E+02*	−1.799E+02	−1.793E+02	2.335E–01	2.418E+03
myDE	F_{10}	−4.221E+02	−4.202E+02	−4.187E+02	1.004E+00	4.538E+02	F_{14}	*−1.400E+02*	*−1.400E+02*	*−1.400E+02*	*4.494E–14*	*7.466E+02*
CMAES		*−4.500E+02*	*−4.500E+02*	*−4.500E+02*	*8.550E–04*	2.519E+03		−1.201E+02	−1.201E+02	−1.201E+02	1.322E–02	5.291E+03
CoDE		−3.489E+02	−3.448E+02	−3.413E+02	2.632E+00	6.076E+02		−1.331E+02	−1.319E+02	−1.308E+02	9.079E–01	7.677E+02
SaDE		−3.861E+02	−3.816E+02	−3.760E+02	3.779E+00	1.061E+03		−1.277E+02	−1.272E+02	−1.270E+02	3.367E–01	1.363E+03
myDE	F_{11}	1.587E+03	1.742E+03	2.026E+03	1.386E+02	*6.261E+02*	F_{15}	*−1.431E+04*	*−1.414E+04*	*−1.392E+04*	1.644E+02	*2.346E+03*
CMAES		*1.066E+03*	*1.187E+03*	*1.232E+03*	*5.748E+01*	2.812E+03		−1.259E+04	−1.252E+04	−1.240E+04	*7.295E+01*	5.046E+03
CoDE		3.006E+03	1.516E+04	1.028E+05	3.143E+04	7.174E+02		−1.082E+04	−1.058E+04	−1.046E+04	1.418E+02	2.508E+03
SaDE		3.531E+03	4.841E+03	1.059E+04	2.185E+03	1.199E+03		−8.939E+03	−8.923E+03	−8.892E+03	1.829E+01	3.762E+03
myDE	F_{12}	*−3.300E+02*	*−3.300E+02*	*−3.300E+02*	*5.684E–14*	*5.259E+02*	F_{16}	*3.085E+02*	*3.314E+02*	*3.495E+02*	*1.467E+01*	*2.079E+03*
CMAES		5.611E+03	6.175E+03	6.737E+03	3.842E+02	2.800E+03		9.091E+03	9.237E+03	9.311E+03	8.625E+01	7.353E+03
CoDE		1.661E+03	1.848E+03	2.068E+03	1.170E+02	6.493E+02		4.124E+03	4.345E+03	4.530E+03	1.522E+02	2.151E+03
SaDE		4.202E+03	4.528E+03	5.053E+03	2.723E+02	1.034E+03		4.061E+03	4.699E+03	5.125E+03	3.888E+02	2.709E+03

Italics represents the best results obtained by the corresponding algorithm for each function

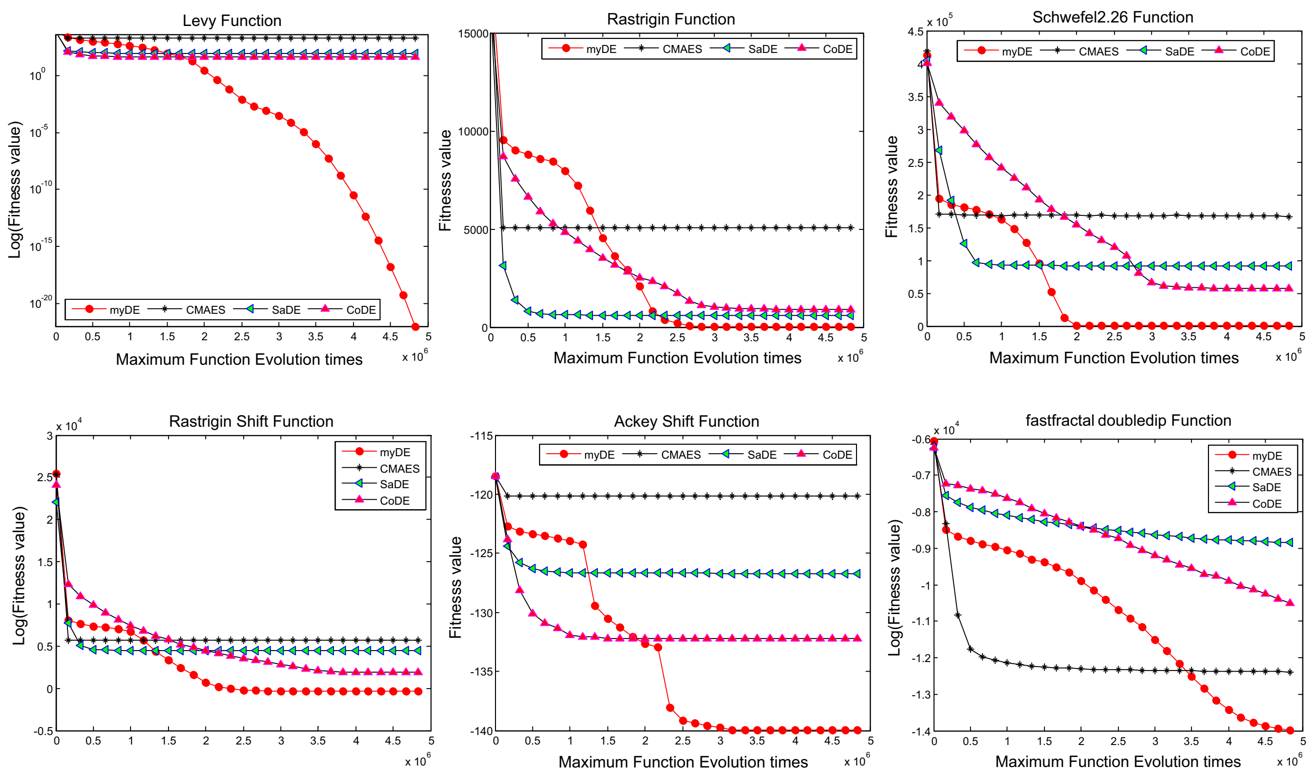

Fig. 25 Convergence curves of six functions (Levy, Rastrigin, Schwefel 2.26, Ackey Shift, Rastrigin Shift and Fastfractal doubledip) for comparison of four algorithms on $D = 1000$

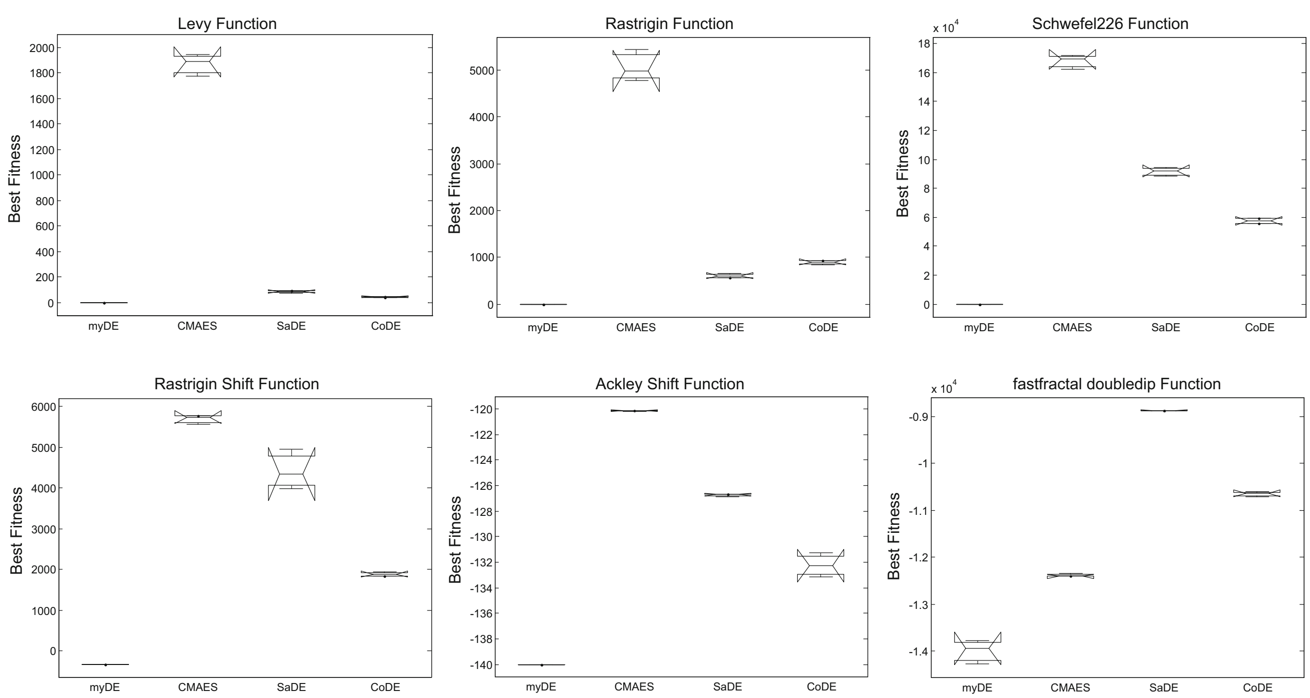

Fig. 26 Optimal solution distribution boxplot of six functions (Levy, Rastrigin, Schwefel 2.26, Ackey Shift, Rastrigin Shift and Fastfractal doubledip) on $D = 1000$

Table 10 Results comparison with seven state-of-the-art high-dimensional optimization algorithms on 1000D problems

Algorithm	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
jDEdynNP-F	1.14E–13	1.95E+01	1.31E+03	3.98E–14	2.17E–04	1.47E–11	−1.35E+04
EPUS-PSO	5.53E+02	4.66E+01	8.37E+05	5.89E+00	7.58E+03	1.89E+01	−6.62E+03
UEP	5.37E–12	1.05E+02	1.96E+03	8.87E–04	1.03E+04	1.99E+01	−1.18E+04
MTS	*0.00E+00*	4.72E–02	*3.41E–04*	*0.00E+00*	*0.00E+00*	1.21E–11	*−1.40E+04*
LSEDA-gl	3.22E–13	*1.04E–05*	1.73E+03	1.71E–13	5.45E+02	4.26E–13	−1.35E+04
DEwSAcc	8.79E–03	9.61E+01	9.15E+03	3.58E–03	1.82E+03	2.30E+00	−1.06E+04
DMS-PSO	*0.00E+00*	9.15E+01	8.98E+09	*0.00E+00*	3.84E+03	7.76E+00	−7.51E+03
myDE	*0.00E+00*	2.91E+01	1.59E+03	*0.00E+00*	*0.00E+00*	*1.09E–13*	*−1.400E+04*

Italics represents the best results obtained by the corresponding algorithm for each function

Table 11 Multi-problem-based statistical pairwise comparison of myDE and other algorithms. ($\alpha = 0.05$)

Proposed algorithm versus other algorithms	$D = 200$					$D = 500$					$D = 1000$				
	p value	$W+$	$W=$	$W-$	Winner	p value	$W+$	$W=$	$W-$	Winner	p value	$W+$	$W=$	$W-$	Winner
DE/rand/1-2 versus DE/rand/1	1.76E–01	81	10	45	DE/rand/1-2	6.10E–03	116	6	14	DE/rand/1-2	2.40E–03	136	0	0	DE/rand/1-2
DE/best/1-2 versus DE/best/1	5.31E–04	135	0	1	DE/best/1-2	4.38E–04	136	0	0	DE/best/1-2	4.38E–04	136	0	0	DE/best/1-2
DE/rand/2-2 versus DE/rand/2	4.38E–04	136	0	0	DE/rand/2	4.38E–04	136	0	0	DE/rand/2	4.38E–04	136	0	0	DE/rand/2
DE/current-to-best/1-2 versus DE/current-to-best/1	7.45E–04	123	0	3	DE/current-to-best/1-2	1.10E–03	131	0	5	DE/current-to-best/1-2	7.76E–04	133	0	3	DE/current-to-best/1-2
DE/best/2-2 versus DE/best/2	6.70E–03	116	3	17	DE/best/2-2	1.22E–04	133	1	2	DE/best/2-2	6.10E–05	136	0	0	DE/best/2-2
myDE versus CMAES	4.94E–02	102	3	31	myDE	4.94E–02	102	3	31	myDE	7.85E–02	99	3	34	myDE
myDE versus CoDE	1.07E–02	116	3	17	myDE	6.70E–03	122	1	13	myDE	1.00E–04	135	1	0	myDE
myDE versus SaDE	4.00E–03	122	1	13	myDE	4.00E–03	122	1	13	myDE	1.00E–04	136	0	0	myDE

In Table 11, it can be seen from the results of Wilcoxon Signed-Rank Test that the proposed algorithm is the winner among all algorithms.

5.6 Population diversity analysis

In order to further investigate performance of the proposed algorithm, in this work, we keep track the population diversity of five algorithms (myDE, DE/Best/1, SaDE, CoDE, and CMAES); the population diversity is defined as follows:

$$\text{Diversity} = \frac{1}{D}\sum_{i=1}^{D}\text{SD}_i$$

$$\text{SD}_i = \sqrt{\frac{1}{\text{NP}}\sum_{j=1}^{\text{NP}}\left(x_i^j - \bar{x}_i\right)^2}, \quad i = 1, 2, \ldots, D$$

where $\bar{x}_i$ is the mean value of ith decision variable in population.

Figure 27 shows the changing curves of population diversity in evolution process for two complex functions (Schwefel 2.26 function and Fastfractal Doubledip function). It indicates that the population diversities of other four algorithms (DE/Best/1, SaDE, CoDE and CMAES) falls very quickly, likely resulting in premature trapping into local minimums, accordingly the global optimal solution may be lost. On the contrary, the population diversity of myDE falls more slowly than that of other algorithms and its curve can keep a progressive decreasing. This illustrates that myDE can keep strong search ability during the evolution process and it would not prematurely fall into local search region, so it can obtained higher precision global solution (as population individuals constant gathering) than other four algorithms for multimodal optimization problem with high dimensionality.

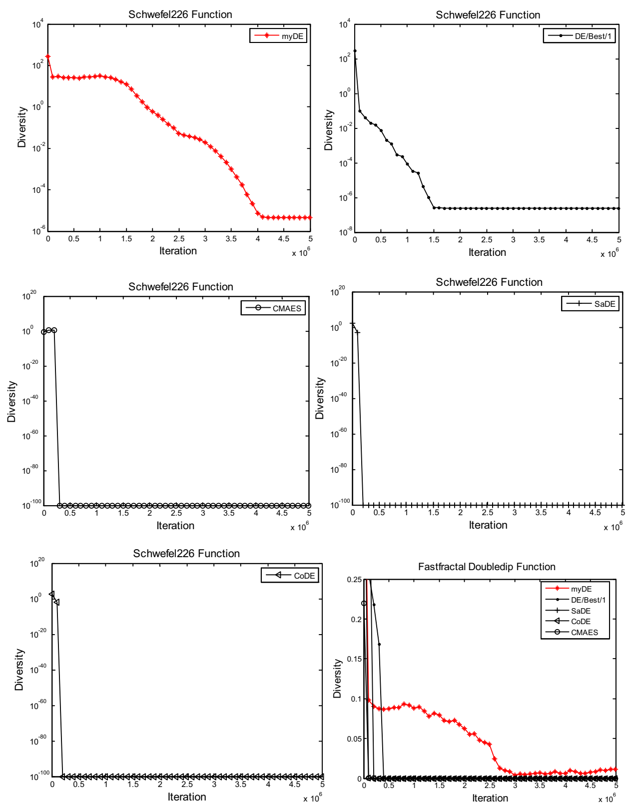

Fig. 27 The changed curves of population diversity ($D = 1000$)

6 Conclusion

Differential evolution algorithm has already developed very high level. Many of state-of-the-art variants of DE either have complex structure or need expensive time consumption and large storage space. To find the high-precision global optimal solution with small population size and low calculation consume, a new DE algorithm is proposed in this study, in which new dynamic crossover operator and novel local adjustment strategy are presented. In addition, mutation operation and the dynamic crossover operation are integrated together and are combined with local adjustment strategy to decrease the CPU time. To verify the performance of the proposed algorithm, numerical experiments have been tested on sixteen benchmark functions. Five frequently referred mutation operators, i.e., DE/rand/1, DE/Best/1, DE/current-to-best/1, DE/Best/2, DE/rand/2, are employed as mutation operation in the proposed DE; then, it is compared with standard DE and three state-of-the-art EA (SaDE, CoDE, CMAES). Wilcoxon Signed-Rank Test is used to analyze comprehensively the performance for all test functions.

Experimental results show that the proposed algorithm has a better capacity and better cost performance to solve complicated multimodal optimization problems with high dimensionality.

Acknowledgements This work was supported by the Natural Science Foundation of China under Grants 61571341, 61201312, 91530113 and 11401357, Research Fund for the Doctoral Program of Higher Education of China (No. 2013 0203110017), the Fundamental Research Funds for the Central Universities of China (Nos. BDY171416 and JB140306), the Natural Science Foundation of Shaanxi Province in China (2015JM6275), the project of Youth Star in Science and Technology of Shaanxi Province (2016KJXX-95), the Scientific Research Program funded by Shaanxi Provincial Education Department (No. 16JK1157) and the Scientific Research Program funded by the Projects Program of Academician Workstation of Shaanxi University of Technology (No. fckt201509).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interests regarding the publication of this paper. Shouheng Tuo proposed the improved DE algorithm firstly, did all experiments and written the article; Junying Zhang puts forward many constructive ideas and revised the manuscript in detail; Xiguo Yuan and Longquan Yong give some good ideas for this work.

References

Bhowmik P, Das S, Konar A, Das S, Nagar AK (2010) A new differential evolution with improved mutation strategy. In: Proceedings of the 2010 IEEE congress on evolutionary computation, pp 1–8

Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 10(6):646–657

Brest J, Zamuda A, Boskovic B, Maucec MS (2008) High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction. Evol Comput 47:2032–2039

Cai Z, Gong W, Ling CX, Zhang H (2011) A clustering-based differential evolution for global optimization. Appl Soft Comput 11(1):1363–1379

Civicioglu P (2013) Backtracking search optimization algorithm for numerical optimization problems. Appl Math Comput 219(15):8121–8144

Das S, Abraham A, Chakraborty UK, Konar A (2009) Differential evolution using a neighborhood-based mutation operator. IEEE Trans Evol Comput 13(3):526–553

Das S, Konar A, Chakraborty UK, (2005) Two improved differential evolution schemes for faster global search. In: Genetic and evolutionary computation conference, GECCO, 2005 proceedings. Washington DC, USA, June, pp 991–998

Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18

Fan HY, Lampinen J (2003) A trigonometric mutation operation to differential evolution. J Global Optim 27(1):105–129

Gamperle R, Muller S, Koumoutsakos P (2002) A parameter study for differential evolution[C]. In: Proceedings WSEAS international conference on advances in intelligent systems, fuzzy systems, evolutionary computation, pp 293–298

Ghosh A, Das S, Chowdhury A, Giri R (2011) An improved differential evolution algorithm with fitness-based adaptation of the control parameters. Inf Sci 181(18):3749–3765

Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195

Hedar AR (2013) Test functions for unconstrained global optimization. http://www.optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar-files/TestGO-files/Page364.htm Accessed 08 Dec 2010
Herrera F, Lozano M, Molina D (2010) Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other meta-heuristics for large scale continuous optimization problems. Technical report, SCI2S, University of Granada, Spain. http://sci2s.ugr.es/eamhco/CFP.php
Hsieh ST, Sun TY, Liu CC, Tsai SJ (2008) Solving large scale global optimization using improved particle swarm optimizer. In: IEEE congress on evolutionary computation, CEC 2008, June 1–6, 2008, Hong Kong, China, pp 1777–1784
Islam SM, Das S, Ghosh S, Roy S, Suganthan PN (2012) An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. IEEE Trans Syst Man Cybern B (Cybern) 42(2):482–500
Jia D, Zheng G, Khan MK (2011) An effective memetic differential evolution algorithm based on chaotic local search. Inf Sci 181(15):3175–3187
Joshi R, Sanderson AC (1997) Minimal representation multisensory fusion using differential evolution. IEEE Trans Syst Man Cybern 38:266–273
Kitayama S, Arakawa M, Yamazaki K (2011) Differential evolution as the global optimization technique and its application to structural optimization. Appl Soft Comput 11(4):3792–3803
Kovačević D, Mladenović N, Petrović B, Milošević P (2014) DE-VNS: self-adaptive differential evolution with crossover neighborhood search for continuous global optimization. Comput Oper Res 52:157–169
Liao T Warren (2010) Two hybrid differential evolution algorithms for engineering design optimization. Appl Soft Comput 10(4):1188–1199
Liu J, Lampinen J (2005) A fuzzy adaptive differential evolution algorithm. Soft Comput 9(6):448–462
Liu W, Song H, Liang JJ, Qu B, Qin AK (2014) Neural network based on self-adaptive differential evolution for ultra-short-term power load forecasting. In: International conference on intelligent computing, pp 403–412. Springer International Publishing
Macnish C, Yao X (2008) Direction matters in high-dimensional optimisation. In: IEEE congress on evolutionary computation, CEC 2008, June 1–6, 2008, Hong Kong, China, pp 2372–2379
Neri F, Iacca G, Mininno E (2011) Disturbed exploitation compact differential evolution for limited memory optimization problems. Inf Sci 181(12):2469–2487
Noman N, Iba H (2008) Accelerating differential evolution using an adaptive local search. IEEE Trans Evol Comput 12(1):107–125
Pan QK, Wang L, Gao L, Li WD (2011) An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers. Inf Sci 181(3):668–685
Price K, Storn RM, Lampinen JA (2006) Differential evolution: a practical approach to global optimization. Springer Science & Business Media, Berlin
Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417 (**SaDE**)
Rahnamayan S, Tizhoosh HR, Salama MMA (2008) Opposition-based differential evolution. IEEE Trans Evol Comput 12(1):64–79
Storn R, Price KV (1996) Minimizing the real functions of the ICEC'96 contest by differential evolution. In: International conference on evolutionary computation, pp 842–844
Storn R, Price K (1995) Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces, vol 3. ICSI, Berkeley
Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
Tang K, Yáo X, Suganthan PN, MacNish C, Chen YP, Chen CM, Yang Z (2007) Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. Nature Inspired Computation and Applications Laboratory, USTC, pp 153–177. http://nical.ustc.edu.cn/cec10ss.php
Tang K, Yáo X, Suganthan PN, MacNish C, Chen YP, Chen CM, Yang Z (2007) Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. Nature Inspired Computation and Applications Laboratory, USTC, pp 153–177. http://www.ntu.edu.sg/home/EPNSugan/, 2008
Tasgetiren MF, Suganthan PN (2006) A multi-populated differential evolution algorithm for solving constrained optimization problem. In: Proceedings of the 2006 IEEE congress on evolutionary computation (CEC'2006). IEEE Press, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, pp 33–40
Tseng LY, Chen C (2008) Multiple trajectory search for large scale global optimization. Evolutionary computation, 2008. CEC 2008 (IEEE world congress on computational intelligence). IEEE congress on, pp 3052–3059
Wang FS, Su TL, Jang HJ (2001) Hybrid differential evolution for problems of kinetic parameter estimation and dynamic optimization of an ethanol fermentation process. Ind Eng Chem Res 40(13):2876–2885
Wang Y, Liu H, Cai Z, Zhou Y (2007) An orthogonal design based constrained evolutionary optimization algorithm. Eng Optim 39(6):715–736
Wang L, Pan QK, Suganthan PN, Wang WH, Wang YM (2010) A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems. Comput Oper Res 37(3):509–520
Wang Y, Cai Z, Zhang Q (2011) Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans Evol Comput 15(1):55–66 (**CoDE**)
Wang Y, Cai Z, Zhang Q (2012) Enhancing the search ability of differential evolution through orthogonal crossover. Inf Sci 185(1):153–177 (OXDE)
Wang Y, Dang C (2007) An evolutionary algorithm for global optimization based on level set evolution and Latin squares. IEEE Trans Evol Comput 11(5):579–595
Wang Y, Li B (2008) A restart univariate estimation of distribution algorithm: sampling under mixed Gaussian and Lévy probability distribution. In: Evolutionary computation, pp 3917–3924
Weber M, Neri F, Tirronen V (2011) A study on scale factor in distributed differential evolution. Inf Sci 181(12):2488–2511
Xu Rui, Venayagamoorthy Ganesh K, Wunsch II Donald C (2007) Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization. Neural Netw 20(8):917–927
Zaharie D, Petcu D (2003) Parallel implementation of multi-population differential evolution. In: Proceedings of the NATO advanced research workshop on concurrent information processing and computing, pp 223–232
Zamuda A, Brest J, Boskovic B, Zumer V (2008) Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution. Evol Comput 268:3718–3725
Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958
Zhao SZ, Liang JJ, Suganthan PN, Tasgetiren MF(2008) Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization. In: Evolutionary computation, pp 3845–3852
Zhou Y, Li X, Gao L (2013) A differential evolution algorithm with intersect mutation operator. Appl Soft Comput 13(1):390–401

View publication stats