Neurocomputing 290 (2018) 130–147

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

An improved multi-population ensemble differential evolution

Lyuyang Tong[a], Minggang Dong[a,b,*], Chao Jing[a,b]

[a] College of Information Science and Engineering, Guilin University of Technology, 12 Jiangan Road, 541004 Guilin, China
[b] Guangxi key Laboratory of Embedded Technology and Intelligent System, Guilin University of Technology, Guilin, China

ARTICLE INFO

Article history:
Received 19 July 2017
Revised 8 November 2017
Accepted 8 February 2018
Available online 13 February 2018

Communicated by Prof. Zidong Wang

MSC:
00-01
99-00

Keywords:
Differential Evolution (DE)
Multiple-population
Ensemble mutation strategies

ABSTRACT

Differential evolution (DE) is a population-based stochastic optimization technique that can be applied to solve global optimization problems. The selected mutation strategies and the control parameters can affect the performance of DE. As mutation strategies have significant effects on solving optimization problems, multiple mutation strategies of DE have been developed. Multi-population ensemble DE (MPEDE) realizes an ensemble of multiple strategies, but "DE/rand/1" may be slow at exploitation of the solutions, and the control parameter by applying arithmetic mean in "DE/current-to-pbest/1" may cause premature convergence. Address these issues, an improved multi-population ensemble DE (IMPEDE) is proposed in this paper. IMPEDE proposes a new mutation strategy "DE/pbad-to-pbest/1" instead of the mutation strategy "DE/rand/1" in MPEDE, and the new strategy utilize not only the good solution information(pbest), but also the information of the bad solution (pbad) toward the good solution to balance exploration and exploitation. Furthermore, IMPEDE employs the improved parameter adaptation approach to avoid premature convergence of the "DE/current-to-pbest/1" strategy by adding the weighted Lehmer mean strategy. Experiments have been conducted with CEC2005 and CEC2017 benchmark functions, and the results have demonstrated that IMPEDE outperforms than that of MPEDE and the other four recent proposed DE methods in obtaining the global optimum and accelerating the convergence speed.

1. Introduction

Differential evolution (DE), which was firstly proposed by Storn and Price, is population-based, stochastic optimization algorithm [1,2]. Also, DE is a simple and efficient evolutionary algorithm (EA) for global optimization that has been successfully applied to various optimizati on problems [3–8]. Despite of its simplicity, DE has been shown to be competitive with other EAs and applies mutation, crossover, and selection operations at each generation to guide its population toward the global optimum [9–12].

However, two crucial components may significantly influence the optimization performance of DE. One is mutation strategy, and the other is control parameters such as population size *NP*, mutation factor *F*, and the crossover probability *CR* [13,14]. There are many researchers paying considerable attentions to choosing the suitable mutation strategy and control parameters [15–18]. In addition, these experiences are immensely helpful in enhancing the performance of DE.

Zhang and Sanderson [19] have proposed a well-known, effective DE variant Adaptive Differential Evolution with Optional External Archive (JADE) which uses a novel mutation strategy called "DE/current-to-pbest/1" with an archive and also employs a control parameter adaptation mechanism. The "DE/current-to-pbest/1" is a mutation strategy which directs the generation of mutant vector towards the best and the other good member of the population. The archive is utilized to add the parent solutions, when the parent solutions failed in the selection process. The "DE/current-to-pbest/1" with an archive is not liable to be trapped into the local optimum and very useful in solving complex optimization problems such as the unimodal problem and the multimodal problem. Motivated by JADE, Wu et al. [20] have introduced multi-population ensemble DE(MPEDE), in which a multi-population based approach is utilized to realize a dynamic ensemble of multiple mutation strategies consisting of "DE/current-to-pbest/1" with an archive, "DE/current-to-rand/1", and "DE/rand/1". "DE/rand/1" is a mutation strategy which directs the generation of mutant vector towards the random member of the population. In addition, "DE/current-to-rand/1", a rotation-invariant strategy without the crossover operation, is extremely effective in solving rotated problems. Moreover, control parameters of MPEDE are adapted based on the mechanism of JADE. However, there are some issues in these works. "DE/rand/1" mutation strategy may be good at exploring the search space, but may be slow at exploitation of the solutions [21,22]. In the control parameter adaptation

* Corresponding author at: College of Information Science and Engineering, Guilin University of Technology, 12 Jiangan Road, 541004, Guilin, China
E-mail address: d2015mg@qq.com (M. Dong).

https://doi.org/10.1016/j.neucom.2018.02.038
0925-2312/

mechanism of "DE/current-to-pbest/1", the control parameter by applying arithmetic mean has an implicit bias to small values during self-adaptation process and causes premature convergence at the end [23,24]. Therefore, by integrating the advantages and overcoming the disadvantages of JADE and MPEDE, we have proposed an improved multi-population ensemble differential evolution algorithm (IMPEDE).

IMPEDE proposes a new mutation strategy "DE/pbad-to-pbest/1" to balance exploration and exploitation with the objective of obtaining optimal solution and accelerating convergent speed instead of the "DE/rand/1" mutation strategy in MPEDE. Meanwhile, because the "DE/current-to-pbest/1" with an archive and "DE/current-to-rand/1" are the key strategies for solving optimization problems in MPEDE, we combine these two strategies with a new mutation strategy "DE/pbad-to-pbest/1" called the improved multi-population based mutation strategy ensemble approach in IMPEDE to search the global optimal solution using MPEDE framework. Furthermore, to tackle the issue of premature convergence caused by applying arithmetic mean in the control parameter of "DE/current-to-pbest/1", IMPEDE employs the improved parameter adaptation approach to make slight modifications on the "DE/current-to-pbest/1" strategy by adding the weighted Lehmer mean strategy on the adaptation of control parameter. Experimental results have shown that the IMPEDE is more efficient than that of the previous DE algorithms. It can achieve the goals of accelerating the speed of convergence and jumping out of local optimum.

The rest of the paper is organized as follow: Section 2 introduces the original DE and its development briefly. The IMPEDE is proposed in Section 3. In Section 4, the performance of the proposed IMPEDE algorithm is evaluated using the CEC2005 and CEC2017 benchmark functions and compared with other DE algorithms. IMPEDE is also applied to solve a real-life problem. Finally, the research limitation and future works are discussed in Section 5 and conclusions are given in Section 5.

2. Related work

Differential evolution is a parallel direct search method that encodes the candidate solution, i.e.$X_{i,G} = \{x^1_{i,G}, \ldots, x^D_{i,G}\}, i = 1, \ldots, NP$, where D is the dimension of the problem and NP is the population size. DE enters an evolutionary process which contains mutation, crossover, and selection.

During the mutation process, the mutant vector $V_{i,G}$ is generated by employing one of the following mutation strategies, corresponding to each member or the target vector $X_{i,G}$.

DE/rand/1:

$$V_{i,G} = X_{r_1,G} + F \cdot (X_{r_2,G} - X_{r_3,G}) \tag{1}$$

DE/current-to-best/1:

$$V_{i,G} = X_{i,G} + F \cdot (X_{best,G} - X_{i,G}) + F(X_{r_1,G} - X_{r_2,G}) \tag{2}$$

DE/best/1:

$$V_{i,G} = X_{best,G} + F \cdot (X_{r_1,G} - X_{r_2,G}) \tag{3}$$

where the indices r_1, r_2 and r_3 are distinct integers randomly selected from the range of $[1, NP]$. $X_{best,G}$ is the best individual vector with the best fitness value in the current generation G. The mutation factor F is a positive control parameter for weighting the difference vectors within the range of [0, 1].

After the mutation phase, DE utilizes the crossover operation to each pair of the target vector $X_{i,G}$ and its corresponding mutant vector $V_{i,G}$ to generate the trail vector $U_{i,G}$. The binominal crossover is described as follow:

$$U^j_{i,G} = \begin{cases} V^j_{i,G} & if(rand_j[0,1] \le CR) or (j = j_{rand}), j = 1, 2, \ldots D \\ X^j_{i,G} & otherwise \end{cases} \tag{4}$$

where $rand_j(0, 1)$ represents a uniformly disturbed random variable within the range [0, 1] and j_{rand} is an integer randomly generated from the range of $[1, D]$. The crossover probability CR is a user-specified constraint within the range [0, 1], which controls the average fraction of vector components that are inherited from the mutant vector.

After the crossover phase, the selection operation selects the better one from the trail vector $U_{i,G}$ and the target vector $X_{i,G}$ based on the fitness value. The selection operation is described as follow:

$$X_{i,G} = \begin{cases} U_{i,G} & if(f(U_{i,G}) \le f(X_{i,G})) \\ X_{i,G} & otherwise \end{cases} \tag{5}$$

As DE has attracted much attention recently, many researchers focus on improving DE mainly by the mutation strategy and control parameter [25]. Thus, many studies have been undertaken on the mutation strategy for improved DE. Das et al. proposed a family of improved variants of the "DE/target-to-best/1" scheme, in which the best solution is chosen from a small neighborhood [16]. Islam et al. proposed a variant of "DE/current-to-best/1" that utilizes the best solution of a dynamic group of randomly selected solutions from the current population to perturb the target vector [26]. Zou et al. proposed a modified DE (MDE) [27] that adopts two mutation strategies "DE/rand/1" and "DE/best/1" to produce new solutions. During late evolution process, "DE/best/1" is more likely to be utilized based on the probability. In [28], "DE/current-to-rand/1" is a rotation-invariant strategy that can efficiently solve rotated problems. In JADE [19], "DE/current-to-pbest/1" not only utilizes the best solutions, but also takes advantage of the information of other good solution which can balance exploration and exploitation. Therefore, exploitation and exploration should be considered in designing efficient evolutionary and swarm intelligent algorithm [29]. Furthermore, it is important to obtain the balance between exploration and exploitation of the search space in the population-based evolutionary algorithms [30–32].

Additionally, amount of research work has been conducted on the control parameter, such as the mutation factor F and the crossover probability CR. Brest et al. proposed a self-adaptive approach in which the cross probability CR and the mutation factor F are randomly generated based on τ_1 and τ_2, respectively [33]. Qin et al. proposed a self-adaptive DE (SaDE) in which the mutation factor F and the cross probability CR are self-adapted by the previous experience [15]. Wang et al. proposed a composite DE (CoDE), which uses three trial vector generation strategies and three control parameter settings of CR and F. Each individual randomly chooses the strategies and parameters to generate trial vector [34]. In JADE [19], the mutation factor F and the crossover probability CR are generated using two parameters, μCR and μF, respectively. After each generation, μCR is updated as:

$$\mu CR = (1 - c) \cdot \mu CR + c \cdot mean_A(S_{CR}), \tag{6}$$

where S_{CR} is the successful crossover probability and $mean_A$ is a function calculating the arithmetic mean value of elements in S_{CR} and c is a positive constant between 0 and 1. After each generation, μF is updated as:

$$\mu F = (1 - c) \cdot \mu F + c \cdot mean_L(S_F), \tag{7}$$

where S_F is the successful mutation factor and $mean_L$ is the Lehmer mean:

$$mean_L(S_{F,j}) = \frac{\sum_{k=1}^{|S_F|} S^2_{F,k}}{\sum_{k=1}^{|S_F|} S_{F,k}} \tag{8}$$

Table 1
Comparison of IMPEDE with JADE, jDE, SaDE, CoDE and MPEDE.

Algorithm	Comparison
JADE	JADE implements a new mutation strategy "DE/current-to-pbest" with optional external archive and updates control parameters in an adaptive manner.
jDE	JDE employs a self-adaptation scheme for the DE control parameters. The range of F and CR values are [0.1, 1.0] and [0, 1], respectively. The F and CR are updated based on two constants τ_1 and τ_2, which are set to 0.1.
SaDE	SaDE utilizes the self-adaptive mutation strategies and respective control parameters based on their previous experiences of generating promising solutions. The F values are randomly generated with a mean and standard deviation of 0.5 and 0.3, respectively.
CoDE	CoDE uses three trial vector generation strategies and three control parameter settings. It randomly combines them to generate trial vectors.
MPEDE	MPEDE realizes the ensemble of multiple mutation strategies and partitions the population into the subpopulations whose sizes are unequal. Moreover, the best mutation strategy will dynamically be rewarded with large population resources after every certain number of generations in MPEDE.
IMPEDE	IMPEDE proposes a new mutation strategy "DE/pbad-to-pbest/1" instead of the mutation strategy "DE/rand/1" in MPEDE, and the new strategy utilize not only the good solution information(pbest), but also the information of the bad solution (pbad) toward the good solution to balance exploration and exploitation. Furthermore, IMPEDE employs the improved parameter adaptation approach to avoid premature convergence of the "DE/current-to-pbest/1" strategy by adding the weighted Lehmer mean strategy.

Table 2
The list of CEC2005 benchmark functions.

Function type	Function name	Initialization range	Search range	F(x^*) f_bias
Unimodal functions	F1:Shifted sphere function	$[-100, 100]^D$	$[-100, 100]^D$	−450
	F2:Shifted Schwefel's problem 1.2	$[-100, 100]^D$	$[-100, 100]^D$	−450
	F3:Shifted rotated high conditioned elliptic function	$[-100, 100]^D$	$[-100, 100]^D$	−450
	F4:Shifted Schwefel's problem 1.2 with noise in fitness	$[-100, 100]^D$	$[-100, 100]^D$	−450
	F5:Schwefel's problem 2.6 with global optimum on bounds	$[-100, 100]^D$	$[-100, 100]^D$	−310
Basic multi-modal functions	F6:Shifted Rosenbrock's function	$[-100, 100]^D$	$[-100, 100]^D$	−390
	F7:Shifted rotated Griewank's function without bounds	$[0, 600]^D$	$[-600, 600]^D$	−180
	F8:Shifted rotated Ackley's function with global optimum on bounds	$[-32, 32]^D$	$[-32, 32]^D$	−140
	F9:Shifted Rastrigin's function	$[-5, 5]^D$	$[-5, 5]^D$	−330
	F10:Shifted rotated Rastrigin's function	$[-5, 5]^D$	$[-5, 5]^D$	−330
	F11:Shifted rotated Weierstrass function	$[-0.5, 0.5]^D$	$[-0.5, 0.5]^D$	90
	F12:Schwefel's problem 2.13	$[-100, 100]^D$	$[-100, 100]^D$	−460
Expanded multi-modal functions	F13:Expanded extended Griewank's plus Rosenbrock's function(F8F2)	$[-3, 1]^D$	$[-3, 1]^D$	−130
	F14:Shifted rotated expanded Scaffer's F6	$[-100, 100]^D$	$[-100, 100]^D$	−300
Hybrid composition functions	F15:Hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	120
	F16:Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	120
	F17:Rotated hybrid composition function with noise in fitness	$[-5, 5]^D$	$[-5, 5]^D$	120
	F18:Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	10
	F19:Rotated hybrid composition function with an narrow basin for the global optimum	$[-5, 5]^D$	$[-5, 5]^D$	10
	F20:Rotated hybrid composition function with the global optimum on the bounds	$[-5, 5]^D$	$[-5, 5]^D$	10
	F21:Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	360
	F22:Rotated hybrid composition function with high condition number matrix	$[-5, 5]^D$	$[-5, 5]^D$	360
	F23:Non-continuous rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	360
	F24:Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	260
	F25:Rotated hybrid composition function without bounds	$[-2, 5]^D$	$[-5, 5]^D$	260

Table 3
The list of CEC2017 benchmark functions.

Function type	Function name	F(x*) *f_bias*
Unimodal functions	F1:Shifted and rotated bent cigar function	100
	F2:Shifted and rotated sum of different power function	200
	F3:Shifted and rotated Zakharov function	300
Simple multi-modal functions	F4:Shifted and rotated Rosenbrock's function	400
	F5:Shifted and rotated Rastrigin's function	500
	F6:Shifted and rotated expanded Scaffer's F6 funciton	600
	F7:Shifted and rotated Lunacek Bi-Rastrigin function	700
	F8:Shifted and rotated non-continuous Rastrigin's function	800
	F9:Shifted and rotated Levy function	900
	F10:Shifted and rotated Schwefel's function	1000
Hybrid functions	F11:Hybrid function 1($N = 3$)	1100
	F12:Hybrid function 2($N = 3$)	1200
	F13:Hybrid function 3($N = 3$)	1300
	F14:Hybrid function 4($N = 4$)	1400
	F15:Hybrid function 5($N = 4$)	1500
	F16:Hybrid function 6($N = 4$)	1600
	F17:Hybrid function 6($N = 5$)	1700
	F18:Hybrid function 6($N = 5$)	1800
	F19:Hybrid function 6($N = 5$)	1900
	F20:Hybrid function 6($N = 6$)	2000
Composition functions	F21:Composition function 1($N = 3$)	2100
	F22:Composition function 2($N = 3$)	2200
	F23:Composition function 3($N = 4$)	2300
	F24:Composition function 4($N = 4$)	2400
	F25:Composition function 5($N = 5$)	2500
	F26:Composition function 6($N = 5$)	2600
	F27:Composition function 7($N = 6$)	2700
	F28:Composition function 8($N = 6$)	2800
	F29:Composition function 9($N = 3$)	2900
	F30:Composition function 10($N = 3$)	3000
	Search range:$[-100, 100]^D$	

As the mutation strategy and the control parameter of DE have been developed, population partitioning techniques for enhancing evolutionary algorithm attracted increasing attention recently [35–40]. In MPEDE [20], the whole population is dynamically partitioned into small sized multiple indicator subpopulations and one large sized reward subpopulation. Let *pop* be the overall population, and *pop* is defined by:

$$pop = \bigcup_{j=1,2,3,4} pop_j \tag{9}$$

Let *NP* be the size of *pop*, NP_j be the size of pop_j, and λ_j be the portion between pop_j and *pop*.

$$NP_j = \lambda_j \cdot NP,\ j = 1, 2, 3, 4 \tag{10}$$

where j is the index of the mutation strategies. Set $\lambda_1 = \lambda_2 = \lambda_3$ and $\sum_{j=1,2,3,4} \lambda_j = 1$. The reward subpopulation is assigned to the mutation strategy that performed the best by the metric for evaluating the performance of the jth mutation strategy during the previous *ng* generations. The metric is defined as follow:

$$k = \arg\left(\max_{1 \le j \le 3}\left(\frac{\Delta f_j}{ng \cdot NP_j}\right)\right) \tag{11}$$

where k is the indices of the best mutation strategy and Δf_j is the accumulated fitness improvement brought by the jth mutation strategy during the previous *ng* generations.

The dynamic partitioning technique ensures that the best mutation strategy consumes the most computational resources. However, in MPEDE, the mutation strategy "DE/rand/1" can lead to low convergence rate and the parameter adaptation based on the mechanism of JADE also can cause the premature convergence. So we not only take advantage of the dynamic partitioning technique, but also should have solved these issues of MPEDE. Inspired by this, we have proposed IMPEDE, which utilizes a new mutation strategy "DE/pbad-to-pbest/1" in the improved multi-population based mutation strategy ensemble approach to balance exploration and exploitation and also adopts the improved parameter adaptation approach to deal with the issues of the premature convergence.

3. An improved multi-population ensemble differential evolution (IMPEDE)

3.1. Overview

IMPEDE adopts the improved Multi-population based mutation strategy ensemble approach which combines mutation strategies of "DE/current-to-pbest/1" with an archive and "DE/current-to-rand/1", and "DE/pbad-to-pbest/1", a new mutation strategy of which we propose instead of the mutation strategy "DE/rand/1" in MPEDE algorithm. Due to less efficient of "DE/rand/1" in terms of convergence rate, we propose a new mutation strategy "DE/pbad-to-pbest/1" which utilizes not only the good solution information but also the information of the bad solution (*pbad*) toward the good solution (*pbest*) to balance exploration and exploitation. Furthermore, IMPEDE adopts the improved parameter adaptation approach to deal with the issues of premature convergence caused by applying arithmetic mean in the control parameter of the "DE/current-to-pbest/1" strategy. To avoid the premature, we make slight modifications on the "DE/current-to-pbest/1" strategy by adding the weighted

Algorithm 1 Pseudo code of IMPEDE.

Set μCR_j=0.5, μF_j=0.5, $\Delta f_j = 0$ and $\Delta Fes_j = 0$ for each j=1,...4;
Initialize NP, ng;
Initialize the pop randomly distributed in the solution space;
Initialize λ_j and set $NP_j = \lambda_j \cdot NP$;
Set g=0 and FES=0;
while FES<Max_FES **do**
  g=g+1;
  if mod(g,ng)==0 **then**
    $k = \arg(\max_{1\le j\le 3}(\frac{\Delta f_j}{ng \cdot NP_j}))$;
    $\Delta f_j = 0$;
  end if
  Randomly partition pop into pop_1, pop_2, pop_3 and pop_4;
  Let $pop_k = pop_k \cup pop_4$ and $NP_k = NP_k + NP_4$;
  Set $S_{CR,1} = \phi$ and $S_{F,1} = \phi$;
  for $i = 1$ to NP_1 **do**
    Calculate $CR_{i,1}$ and $F_{i,1}$ for each individual X_i in pop_1;
    Perform the first mutation strategy "DE/current-to-pbest/1" and related crossover operations over subpopulation pop_1;
    if $f(X_{i,g}) \le f(U_{i,g})$ **then**
      $X_{i,g+1} = X_{i,g}$;
    else
      $X_{i,g+1} = U_{i,g}$; $\Delta f_1 = \Delta f_1 + f(X_{i,g}) - f(U_{i,g})$;
      $CR_{i,1} \to S_{CR,1}$; $F_{i,1} \to S_{F,1}$; $|f(X_{i,g}) - f(U_{i,g})| \to \Delta f_{k,g}$;
    end if
  end for
  Calculate μCR_1 and μF_1;
  Set $S_{CR,2} = \phi$ and $S_{F,2} = \phi$;
  for $i = 1$ to NP_2 **do**
    Calculate for each individual X_i in pop_2;
    Perform the second mutation strategy "DE/current-to-rand/1" over subpopulation pop_2;
    if $f(X_{i,g}) \le f(U_{i,g})$ **then**
      $X_{i,g+1} = X_{i,g}$;
    else
      $X_{i,g+1} = U_{i,g}$; $\Delta f_2 = \Delta f_2 + f(X_{i,g}) - f(U_{i,g})$;
      $F_{i,2} \to S_{F,2}$;
    end if
  end for
  Calculate μF_2;
  Set $S_{CR,3} = \phi$ and $S_{F,3} = \phi$;
  for $i = 1$ to NP_3 **do**
    Calculate $CR_{i,3}$ and $F_{i,3}$ for each individual X_i in pop_3;
    Perform the third mutation strategy "DE/pbad-to-pbest/1" and related crossover operations over subpopulation pop_3;
    if $f(X_{i,g}) \le f(U_{i,g})$ **then**
      $X_{i,g+1} = X_{i,g}$;
    else
      $X_{i,g+1} = U_{i,g}$; $\Delta f_3 = \Delta f_3 + f(X_{i,g}) - f(U_{i,g})$;
      $CR_{i,3} \to S_{CR,3}$; $F_{i,3} \to S_{F,3}$;
    end if
  end for
  Calculate μCR_3 and μF_3;
  $FES = FES + NP$;
end while

Lehmer mean strategy on the adaptation of control parameter. Weight updating of weighted Lehmer mean strategy is based on the quantity of fitness improvement in order to affect self-adaptation process, so it can overcome the premature convergence.

3.2. Algorithm details

The general idea of IMPEDE is: first, each subpopulation pop_1, pop_2 and pop_3 can be allocated to the relative mutation strategies and pop_4 is randomly distributed to one of strategies mentioned. During the evolutionary process, we revaluate the best performance of mutation strategies and assign the reward subpopulation pop_4 to the current best performing mutation strategy after the following ng generations. Then, we employ the improved multi-population based mutation strategy ensemble approach and execute the corresponding operations of mutation, crossover, and selection. Third, during the selection, the trail vector $U_{i,G}$ fitness is calculated and compared to the target vector $X_{i,G}$ fitness. If the trail vector fitness is better than the target vector fitness, the target vector will be replaced, and the mutation factor F and the crossover probability CR will be stored in the successful mutation factor S_F and the successful crossover probability S_{CR}, respectively; otherwise not. So we can utilize the improved adaptation approach by using the information of S_F and S_{CR}. Fourth, if the number of the fitness evaluations (FES) reach the maximum, the IMPEDE stops and outputs the best solution. The framework of the IMPEDE can be given in Algorithm 1.

Compared to the previous work, IMPEDE has the improvements in the improved multi-population based mutation strategy ensemble approach and the improved parameter adaptation approach, which are going to detail in the next two subsections.

3.2.1. Improved multi-population based mutation strategy ensemble approach

DE which employs different mutation strategies usually performs differently at solving different optimization problems. In a word, selecting mutation strategies is significant for designing ensemble DE. Our principle aims at selecting suitable mutation strategies which can perform well on different optimization problem. In this paper, we adopt the improved multi-population based mutation strategy ensemble approach which employs three mutation strategies including "DE/current-to-pbest/1" with an archive, "DE/current-to-rand/1" and "DE/pbad-to- pbest /1". The "DE/current-to-pbest/1" with an archive, which is not liable to be trapped into the local optimum, is very useful in solving complex optimization problems such as the unimodal problem and the multimodal problem. In addition, "DE/current-to-rand/1", which is a rotation-invariant strategy without the crossover operation, is extremely effective in solving rotated problems. Inspired by a greedy mutation strategy "DE/current-to-pbest/1" in JADE algorithm, we propose a new mutation strategy "DE/pbad-to-pbest/1" which utilizes not only the good solution information but also the information of the bad solution ($pbad$) toward the good solution ($pbest$) to balance exploration and exploitation. In the exploration process, the new mutation strategy takes full advantage of the good information to accelerate the convergence rate. On the other hand, it applies the information of the bad solution toward the good solution to enhance the diversity and increase the probability of finding the global optimum and jumping out of the local optimum in the exploitation process. The employed mutation strategies are listed as below:

Mutation strategy 1: "DE/current-to-pbest/1" with an archive

$$V_{i,G} = X_{i,G} + F \cdot (X_{pbest,G} - X_{i,G} + X_{r_1^i,G} - \tilde{X}_{r_2^i,G}) \tag{12}$$

Mutation strategy 2: "DE/current-to-rand/1"

$$U_{i,G} = X_{i,G} + K \cdot (X_{r_1^i,G} - X_{i,G}) + F \cdot (X_{r_2^i,G} - X_{r_3^i,G}) \tag{13}$$

Mutation strategy 3: "DE/pbad-to-pbest/1"

$$V_{i,G} = X_{i,G} + F \cdot (X_{pbest,G} - X_{pbad,G}) \tag{14}$$

Table 4
Testing results for the 30 dimensional CEC2005Benchmark functions.

Functions		JADE	jDE	SaDE	CoDE	MPEDE	IMPEDE
Unimodal functions	F1	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)
	F2	2.17E−28 (1.11E−28)≈	1.42E−06 (2.11E−06)−	1.01E−05 (2.45E−05)−	1.26E−15 (1.98E−15)−	1.63E−27 (1.72E−27)−	2.38E−28 (1.85E−28)
	F3	1.18E+04 (5.72E+03)−	1.75E+05 (1.50E+05)−	5.97E+05 (2.45E−05)−	9.89E+04 (6.84E+04)−	2.74E+02 (1.07E+03)−	1.99E−03 (8.84E−03)
	F4	1.23E−09 (3.51E−09)−	1.60E−01 (4.79E−01)−	2.07E+02 (4.24E+02)−	5.60E−03 (1.47E−02)−	2.54E−17 (6.02E−17)−	9.17E−20 (4.25E−19)
	F5	5.86E−02 (2.79E−01)−	3.73E+02 (3.65E+02)−	3.24E+03 (6.58E+02)−	3.72E+02 (3.29E+02)−	2.50E−06 (5.75E−06)−	1.81E−07 (8.46E−07)
Basic multi-modal functions	F6	5.91E+00 (2.54E+01)−	1.96E+01 (2.33E+01)−	4.66E+01 (3.21E+01)−	3.24E−08 (1.59E−07)−	4.41E+00 (1.36E+01)−	3.18E−12 (1.47E−11)
	F7	4.69E+03 (2.11E-12)−	4.69E+03 (2.59E−12)−	4.69E+03 (2.76E−12)−	5.41E−03 (7.91E−03)−	2.56E−03 (4.23E−03)≈	6.90E−04 (2.41E−03)
	F8	2.09E+01 (1.97E-01)≈	2.09E+01 (4.29E−02)≈	2.09E+01 (6.21E−02)≈	2.01E+01 (1.74E−01)+	2.09E+01 (5.21E−02)≈	2.09E+01 (1.93E−01)
	F9	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	7.95E−02 (2.75E−01)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)≈	0.00E+00 (0.00E+00)
	F10	2.57E+01 (4.23E+00)≈	5.64E+01 (1.25E+01)−	4.88E+01 (1.15E+01)−	4.25E+01 (1.40E+01)−	2.84E+01 (1.33E+01)≈	2.40E+01 (7.13E+00)
	F11	2.50E+01 (1.45E+00)−	2.77E+01 (1.84E+00)−	1.71E+01 (1.78E+00)−	1.20E+01 (2.46E+00)≈	2.31E+01 (4.95E+00)−	1.37E+01 (7.62E+00)
	F12	6.53E+03 (5.95E+03)−	8.08E+03 (8.81E+03)−	3.62E+03 (3.02E+03)−	2.07E+03 (2.47E+03)≈	1.51E+03 (1.54E+03)≈	1.23E+03 (2.11E+03)
Expanded multimodal functions	F13	1.46E+00 (1.09E-01)+	1.69E+00 (1.33E−01)≈	3.98E+00 (3.87E−01)−	1.73E+00 (3.95E−01)≈	1.94E+00 (2.98E−01)−	1.70E+00 (2.10E−01)
	F14	1.22E+01 (4.28E-01)≈	1.29E+01 (2.59E−01)−	1.26E+01 (2.24E−01)−	1.24E+01 (3.27E−01)≈	1.24E+01 (2.31E−01)≈	1.23E+01 (2.19E−01)
Hybrid composition functions	F15	3.52E+02 (1.04E+02)≈	3.80E+02 (7.07E+01)≈	3.60E+02 (8.71E+01)≈	3.84E+02 (8.98E+01)≈	3.80E+02 (9.57E+01)≈	3.92E+02 (8.13E+01)
	F16	1.14E+02 (1.42E+02)≈	7.40E+01 (1.19E+01)−	1.22E+02 (1.36E+02)−	6.62E+01 (2.09E+01)+	6.56E+01 (9.24E+01)≈	7.06E+01 (7.53E+01)
	F17	1.39E+02 (1.53E+02)−	1.32E+02 (2.11E+01)−	6.65E+01 (8.93E+00)+	6.43E+01 (1.13E+01)+	6.40E+01 (7.58E+01)≈	7.95E+01 (1.03E+02)
	F18	9.04E+02 (1.00E+00)−	9.04E+02 (1.26E+00)−	8.63E+02 (6.21E+01)≈	9.05E+02 (1.35E+00)−	9.03E+02 (2.58E−01)≈	9.03E+02 (5.64E−01)
	F19	9.04E+02 (1.15E+00)−	9.04E+02 (1.17E+00)−	8.67E+02 (6.11E+01)≈	9.05E+02 (1.33E+00)−	9.04E+02 (8.71E−01)−	9.03E+02 (4.13E−01)
	F20	9.04E+02 (1.30E+00)−	9.04E+02 (1.22E+00)−	8.98E+02 (5.05E+01)+	9.04E+02 (7.85E−01)−	9.03E+02 (6.13E−01)≈	9.03E+02 (2.94E−01)
	F21	5.00E+02 (8.12E-14)≈	5.12E+02 (6.00E+01)≈	5.00E+02 (1.70E−13)≈	5.00E+02 (5.80E−14)+	5.00E+02 (6.46E−14)≈	5.00E+02 (7.60E−14)
	F22	8.61E+02 (2.29E+01)≈	8.80E+02 (2.00E+01)−	9.39E+02 (2.36E+01)−	8.62E+02 (2.29E+01)≈	8.59E+02 (2.23E+01)≈	8.63E+02 (1.85E+01)
	F23	5.49E+02 (7.61E+01)−	5.34E+02 (2.59E-04)−	5.34E+02 (2.71E−03)−	5.34E+02 (4.75E−04)−	5.34E+02 (4.08E−13)+	5.34E+02 (5.31E−13)
	F24	2.00E+02 (2.90E-14)≈	2.00E+02 (2.90E-14)≈	2.00E+02 (2.90E−14)≈	2.00E+02 (2.90E−14)≈	2.00E+02 (2.90E−14)≈	2.00E+02 (2.90E−14)
	F25	1.63E+03 (4.00E+00)−	1.63E+03 (4.09E+00)−	1.62E+03 (5.30E+00)−	2.10E+02 (8.71E−01)−	2.09E+02 (4.92E−01)≈	2.09E+02 (4.50E−01)
-		13	18	15	12	8	
+		1	0	2	4	1	
≈		11	7	8	9	16	

where $X_{pbest,G}$ is randomly chosen as one of the best top 100p% of individuals in the current population with $p \in (0, 1]$. In contrast, $X_{pbad,\ G}$ is randomly chosen as one of the bad top 100p'% of individuals in the current population with $p' \in (0, 1]$. $\tilde{X}_{r_2^i,G}$ is randomly chosen from the union set of the current population and the archive. The mutation factor F is a positive control parameter for weighting the difference vectors. F and K are two uniformly distributed random numbers ranged from 0 to 1.

3.2.2. Improved parameter adaptation approach

The effective parameter adaptation leads the better performance of a DE variant, but different mutation strategies may require different parameter settings. Recently, a good volume of studies focuses on guided adaptation of the two primary control parameter of DE called the crossover probability $CR_{i,\ j}$ and the mutation factor $F_{i,j}$. We have found the parameter adaptation approach in JADE is more effective after tying different adaptation techniques. However, we also have found the "DE/current-to-pbest/1" strategy by applying arithmetic mean in JADE has an implicit bias to small values during self-adaptation process and causes premature convergence at the end. In order to prevent this bias and avoid the premature, we employ the improved parameter adaptation approach, which makes slight modifications on the "DE/current-to-pbest/1" strategy by adding the weighted Lehmer mean strategy on the adaptation of control parameter and remains the adapted parameter mechanism of JADE on the other mutation strategies.

The crossover probability $CR_{i,\ j}$ is updated according to a normal distribution of mean μCR_j and standard deviation 0.1, and then truncated to [0, 1] at each generation as follow:

$$CR_{i,j} = randn_{i,j}(\mu CR_j, 0.1) \tag{15}$$

where $CR_{i,\ j}$ denotes the crossover probability of individual X_i that uses jth mutation strategy.

Table 5
Statistic results of the 10 dimensional CEC2017 benchmark functions, average over 51 independent runs.

Functions	Best	Worst	Median	Mean	Std
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F5	0.00E+00	5.15E+00	2.98E+00	2.88E+00	1.32E+00
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	1.15E+01	1.57E+01	1.33E+01	1.33E+01	9.71E−01
F8	9.95E-01	7.96E+00	3.03E+00	3.34E+00	1.42E+00
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	4.87E+00	2.44E+02	2.88E+01	7.20E+01	7.28E+01
F11	0.00E+00	9.95E−01	0.00E+00	3.90E−02	1.95E−01
F12	0.00E+00	1.31E+02	4.16E−01	1.50E+01	3.95E+01
F13	0.00E+00	6.89E+00	9.95E−01	1.89E+00	2.37E+00
F14	0.00E+00	1.99E+00	1.50E−03	1.82E−01	4.11E−01
F15	9.89E−04	4.96E−01	1.29E−02	9.19E−02	1.66E−01
F16	1.29E−01	1.24E+00	6.07E−01	6.18E−01	2.53E−01
F17	3.33E−02	2.69E+00	5.33E−01	7.78E−01	6.04E−01
F18	1.18E−03	5.00E−01	5.72E−02	1.38E−01	1.71E−01
F19	2.14E−03	3.66E−01	4.08E−02	6.70E−02	7.05E−02
F20	0.00E+00	3.12E−01	5.77E−08	3.06E−02	9.38E−02
F21	1.00E+02	2.10E+02	2.03E+02	1.60E+02	5.26E+01
F22	0.00E+00	1.00E+02	1.00E+02	8.85E+01	3.19E+01
F23	3.00E+02	3.08E+02	3.05E+02	3.04E+02	2.13E+00
F24	1.00E+02	3.38E+02	3.32E+02	2.87E+02	9.35E+01
F25	3.98E+02	4.46E+02	3.98E+02	4.09E+02	1.99E+01
F26	3.00E+02	3.00E+02	3.00E+02	3.00E+02	0.00E+00
F27	3.89E+02	3.90E+02	3.90E+02	3.89E+02	2.02E−01
F28	3.00E+02	6.12E+02	3.00E+02	3.72E+02	1.31E+02
F29	2.30E+02	2.46E+02	2.38E+02	2.38E+02	4.33E+00
F30	3.95E+02	8.18E+05	3.95E+02	3.24E+04	1.60E+05

Similarly, the mutation factor $F_{i,j}$ is generated according to a Cauchy distribution of the local parameter μF_j and the scale parameter 0.1 at each generation as follow:

$$F_{i,j} = randc_{i,j}(\mu F_j, 0.1) \tag{16}$$

where $F_{i,j}$ denotes the mutation factor of individual X_i that uses jth mutation strategy. When $F_{i,j} \geq 1$, $F_{i,j}$ is truncated to 1, and when $F_{i,j} \leq 0$, $F_{i,j}$ is regenerated to a valid value. At the beginning of the evolutionary process, μCR_1 and μF_1 are both initialized to 0.5, and adapted during the evolutionary process as follow.

After each generation, μCR_1 and μF_1 are updated on the "DE/current-to-pbest/1" strategy as follow:

$$\mu CR_1 = (1-c)\cdot \mu CR_1 + c \cdot mean_{WL}(S_{CR,1}) \tag{17}$$

$$\mu F_1 = (1-c)\cdot \mu F_1 + c \cdot mean_{WL}(S_{F,1}) \tag{18}$$

where $mean_{WL}$ is the weighted Lehmer mean and c is a positive constant between 0 and 1. The weighted Lehmer mean is computed as:

$$mean_{WL}(S_{CR,1}) = \frac{\sum_{k=1}^{|S_{CR}|} \omega_{CR,k} \cdot S_{CR,k}^2}{\sum_{k=1}^{|S_{CR}|} \omega_{CR,k} \cdot S_{CR,k}} \tag{19}$$

$$mean_{WL}(S_{F,1}) = \frac{\sum_{k=1}^{|S_F|} \omega_{F,k} \cdot S_{F,k}^2}{\sum_{k=1}^{|S_F|} \omega_{F,k} \cdot S_{F,k}} \tag{20}$$

Weight is updated like that:

$$\omega_{CR,k} = \frac{\Delta f_{k,g}}{\sum_{k=1}^{|S_{CR}|} \Delta f_{k,g}} \tag{21}$$

$$\omega_{F,k} = \frac{\Delta f_{k,g}}{\sum_{k=1}^{|S_F|} \Delta f_{k,g}} \tag{22}$$

$$\Delta f_{k,g} = |f(U_{k,G}) - f(X_{k,G})| \tag{23}$$

where the quantity of fitness improvement $\Delta f_{k,g}$ is used in order to affect the parameter adaptation.

3.3. Complexity analysis

The runtime complexity of a classic DE is $O(G_{\max} \cdot NP \cdot D)$ [16], where $G_{\max}$ is maximum number of generations, and NP is the number of population, and D is the dimension of the problem. According to Zhang and Sanderson [41], the total complexity of JADE is $O(G_{\max} \cdot NP \cdot [D + \log(NP)])$. Considering that JADE, MPEDE and IMPEDE share the same framework of algorithm, MPEDE and IMPEDE just extend JADE in terms of the mutation strategies and parameter adaptation. So MPEDE and IMPEDE do not increase in the overall algorithm complexity. Similarly, the total complexity of MPEDE and IMPEDE is $O(G_{\max} \cdot NP \cdot [D + \log(NP)])$. Hence, the complexity of IMPEDE does not increase significantly. As the problem dimension D increases, D has a significant impact on time complexity and the population size NP setting is related to the problem dimension D. Compared with a classic DE, IMPEDE is approximately equal in the runtime complexity and does not impose any serious burden on the runtime complexity.

4. Performance evaluation

4.1. Experimental parameters setting

The performance of the proposed algorithm is evaluated and compared with the other DE algorithms such as JADE [19]; jDE [33]; SaDE [15]; CoDE [34] and MPEDE [20]. JADE implements a new mutation strategy "DE/current-to-pbest" with optional external archive and updates control parameters in an adaptive manner. The jDE algorithm employs a self-adaptation scheme for the DE control parameters. SaDE utilizes the self-adaptive mutation strategies and respective control parameters based on their previous experiences of generating promising solutions. In CoDE, a trail vector is chosen from a set of candidates generated by using different DE strategies and control parameter settings. MPEDE realizes the ensemble of multiple mutation strategies and partitions the population into the subpopulations whose sizes are unequal. Moreover, the best mutation strategy will dynamically be rewarded with large population resources after every certain number of generations in MPEDE. The comparison of IMPEDE with JADE, jDE, SaDE, CoDE and MPEDE is shown in Table 1.

To test performance of IMPEDE, we utilize 25 benchmarked optimization functions proposed in the CEC2005 special session on real-parameter optimization [42]. When implementing IMPEDE, corresponding parameters are set as: $ng = 20$, λ_1=λ_2=λ_3= 0.2 and $NP = 250$. All the algorithms are evaluated with 25 benchmark functions and run 25 times with 30 variables. The allowed maximum function evaluations (FES) for the benchmark with 30 decision variables are set to 300,000. Furthermore, we compared IMPEDE against those five competitive methods on the 10-D and 50-D from IEEE CEC2017 [43] to its performance. In IMPEDE, NP are set to 125 and 400 on the 10-D and 50-D. The allowed maximum function evaluations (FES) for the benchmark with 10 and 50 decision variables are set to 100,000 and 500,000, respectively. To provide a comprehensive comparison, we run each comparative algorithm 51 times over the CEC2017 test functions.

Table 6
Testing results of DE algorithms for 10 dimensional CEC2017 benchmark functions.

Functions	JADE	jDE	SaDE	CoDE	MPEDE	IMPEDE
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.33E−09	0.00E+00
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(5.49E−09)≈	(0.00E+00)
F2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)
F4	0.00E+00	1.97E−02	1.04E+00	0.00E+00	1.46E−08	0.00E+00
	(0.00E+00)≈	(3.63E−02)−	(7.38E−01)−	(0.00E+00)≈	(4.37E−08)−	(0.00E+00)
F5	3.11E+00	5.95E+00	3.63E+00	3.73E+00	6.16E+00	2.88E+00
	(7.71E−01)≈	(1.23E+00)−	(1.37E+00)≈	(1.17E+00)≈	(1.59E+00)−	(1.32E+00)
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.24E−05	0.00E+00
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(6.82E−06)−	(0.00E+00)
F7	1.34E+01	1.76E+01	1.38E+01	1.38E+01	1.79E+01	1.33E+01
	(7.37E−01)≈	(1.70E+00)−	(1.86E+00)≈	(1.98E+00)−	(1.70E+00)−	(9.71E−01)
F8	3.55E+00	6.77E+00	3.25E+00	4.53E+00	6.10E+00	3.34E+00
	(6.79E−01)≈	(1.36E+00)−	(1.55E+00)≈	(1.86E+00)−	(1.85E+00)−	(1.42E+00)
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)
F10	8.92E+01	2.82E+02	2.00E+02	1.12E+02	2.72E+02	7.20E+01
	(6.26E+01)−	(8.78E+01)−	(1.50E+02)−	(8.95E+01)≈	(1.30E+02)−	(7.28E+01)
F11	2.37E+00	1.99E+00	5.48E−01	3.51E−01	2.33E+00	3.90E−02
	(7.43E−01)−	(1.00E+00)−	(6.06E−01)−	(5.91E−01)−	(3.82E−01)−	(1.95E−01)
F12	6.33E+01	4.89E+01	1.63E+02	5.21E−01	1.21E+01	1.50E+01
	(7.00E+01)−	(6.13E+01)−	(1.34E+02)−	(1.56E+00)≈	(1.09E+01)+	(3.95E+01)
F13	3.80E+00	4.94E+00	4.13E+00	2.38E+00	5.25E+00	1.89E+00
	(2.67E+00)−	(2.20E+00)−	(2.95E+00)−	(2.17E+00)≈	(2.13E+00)−	(2.37E+00)
F14	7.31E−01	5.10E−01	3.12E−01	2.73E−01	4.65E+00	1.82E−01
	(4.76E−01)−	(3.99E−01)−	(4.66E−01)−	(4.91E−01)−	(1.32E+00)−	(4.11E−01)
F15	4.52E−01	1.79E−01	3.97E−01	9.83E−02	7.04E−01	9.19E−02
	(1.68E−01)−	(2.10E−01)−	(5.25E−01)≈	(2.81E−01)−	(2.11E−01)−	(1.66E−01)
F16	1.32E+00	7.56E−01	4.81E−01	2.07E−01	2.41E+00	6.18E−01
	(6.41E−01)−	(2.87E−01)−	(2.55E−01)+	(1.91E−01)+	(7.60E−01)−	(2.53E−01)
F17	4.92E−01	1.06E+00	4.90E−01	1.15E−01	7.57E+00	7.78E−01
	(3.14E−01)+	(4.65E−01)−	(4.93E−01)+	(2.18E−01)+	(2.81E+00)−	(6.04E−01)
F18	4.19E−01	5.49E−02	7.18E−01	7.01E−02	2.70E+00	1.38E−01
	(4.83E−01)−	(8.96E−02)≈	(7.06E−01)−	(1.76E−01)+	(1.38E+00)−	(1.71E−01)
F19	4.60E−02	2.76E−02	2.97E−03	1.79E−03	5.74E−01	6.70E−02
	(1.97E−02)≈	(1.94E−02)+	(7.36E−03)+	(5.35E−03)+	(1.82E−01)−	(7.05E−02)
F20	1.22E−02	6.12E−03	0.00E+00	6.12E−03	1.02E+00	3.06E−02
	(6.12E−02)−	(4.37E−02)+	(0.00E+00)+	(4.37E−02)+	(7.08E−01)−	(9.38E−02)
F21	1.63E+02	1.21E+02	1.39E+02	1.55E+02	1.17E+02	1.60E+02
	(4.87E+01)≈	(4.36E+01)≈	(5.14E+01)≈	(5.46E+01)≈	(3.96E+01)≈	(5.26E+01)
F22	9.91E+01	9.88E+01	9.83E+01	8.32E+01	9.02E+01	8.85E+01
	(6.75E+00)−	(8.44E+00)−	(1.24E+01)−	(3.77E+01)+	(3.00E+01)−	(3.19E+01)
F23	3.05E+02	3.07E+02	3.04E+02	3.05E+02	3.06E+02	3.04E+02
	(1.32E+00)−	(1.49E+00)−	(2.46E+00)≈	(2.26E+00)≈	(1.90E+00)−	(2.13E+00)
F24	2.97E+02	2.13E+02	2.41E+02	3.03E+02	2.47E+02	2.87E+02
	(7.75E+01)≈	(1.21E+02)≈	(1.14E+02)+	(8.18E+01)−	(1.14E+02)≈	(9.35E+01)
F25	4.17E+02	4.08E+02	4.14E+02	4.05E+02	4.05E+02	4.09E+02
	(2.29E+01)≈	(1.90E+01)≈	(2.16E+01)−	(1.71E+01)≈	(1.67E+01)≈	(1.99E+01)
F26	3.00E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02
	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(0.00E+00)≈	(1.79E−08)−	(0.00E+00)
F27	3.89E+02	3.90E+02	3.94E+02	3.89E+02	3.89E+02	3.89E+02
	(5.85E−01)−	(1.53E+00)≈	(1.66E+00)−	(1.49E+00)−	(2.43E−01)−	(2.02E−01)
F28	3.84E+02	3.46E+02	3.17E+02	3.11E+02	3.12E+02	3.72E+02
	(1.32E+02)≈	(1.08E+02)≈	(6.74E+01)+	(5.56E+01)+	(6.11E+01)+	(1.31E+02)
F29	2.43E+02	2.54E+02	2.42E+02	2.31E+02	2.50E+02	2.38E+02
	(5.58E+00)−	(6.13E+00)−	(7.87E+00)−	(2.57E+00)+	(5.07E+00)−	(4.33E+00)
F30	2.00E+04	4.28E+02	8.70E+02	4.01E+02	3.95E+02	3.24E+04
	(1.16E+05)≈	(4.54E+01)+	(1.19E+03)+	(1.67E+01)+	(8.72E−01)+	(1.60E+05)
+/-/≈	1/13/16	3/15/12	7/11/12	9/7/14	3/20/7	

4.2. Benchmark functions

In this subsection, the proposed IMPEDE is applied to minimize a set of 25 CEC2005 benchmark functions [42] as shown in Table 2. These 25 benchmark functions were proposed in the CEC 2005 special session on real-parameter optimization, which are used to the performance of the proposed IMPEDE algorithm. All benchmark functions are listed with search range, initialization range and bias values in Table 2. These 25 benchmark functions can be classified into four types: unimodal functions F_1–F_5; basic multimodal functions F_6–F_{12}; expanded multimodal functions F_{13}–F_{14}; hybrid composition functions F_{15}–F_{25}. Furthermore, a set of 30 CEC2017 benchmark functions [43] is utilized to evaluate the performance of IMPEDE. From Table 3, these 30 test functions can be classified into four types: unimodal functions F_1–F_3; basic multimodal functions F_4–F_{10}; hybrid functions F_{11}–F_{20}; composition functions F_{21}–F_{30}.

Table 7
Statistic results of the 50 dimensional benchmark functions, average over 51 independent runs

Functions	Best	Worst	Median	Mean	Std
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	6.00E+00	0.00E+00	6.08E−01	1.08E+00
F3	0.00E+00	2.22E−01	0.00E+00	4.73E−03	3.11E−02
F4	0.00E+00	1.44E+02	6.68E+01	6.59E+01	4.60E+01
F5	2.49E+01	8.06E+01	4.97E+01	5.20E+01	1.32E+01
F6	2.23E−06	1.20E−02	3.64E−05	8.47E−04	2.23E−03
F7	8.00E+01	1.31E+02	1.03E+02	1.03E+02	1.24E+01
F8	2.29E+01	9.25E+01	4.88E+01	5.15E+01	1.33E+01
F9	0.00E+00	1.36E+00	8.95E−02	3.36E−01	4.12E−01
F10	3.71E+03	6.48E+03	5.37E+03	5.32E+03	6.50E+02
F11	4.97E+01	1.17E+02	7.90E+01	7.99E+01	1.58E+01
F12	1.11E+03	9.33E+03	2.32E+03	3.00E+03	1.83E+03
F13	8.95E+00	2.04E+02	6.75E+01	7.43E+01	3.79E+01
F14	2.86E+01	8.10E+01	4.62E+01	4.75E+01	9.72E+00
F15	1.43E+01	8.57E+01	4.70E+01	4.84E+01	1.52E+01
F16	1.42E+02	1.57E+03	8.04E+02	7.94E+02	3.27E+02
F17	8.23E+01	8.04E+02	4.98E+02	5.10E+02	1.69E+02
F18	2.91E+01	1.88E+02	4.16E+01	5.35E+01	3.18E+01
F19	1.68E+01	7.26E+01	3.45E+01	3.54E+01	1.15E+01
F20	4.07E+01	7.10E+02	4.11E+02	4.18E+02	1.73E+02
F21	2.25E+02	2.72E+02	2.51E+02	2.51E+02	1.10E+01
F22	1.00E+02	7.07E+03	1.00E+02	2.35E+03	2.86E+03
F23	4.56E+02	5.21E+02	4.74E+02	4.77E+02	1.34E+01
F24	5.10E+02	5.76E+02	5.34E+02	5.36E+02	1.33E+01
F25	4.61E+02	5.65E+02	4.92E+02	5.05E+02	2.97E+01
F26	1.25E+03	1.91E+03	1.55E+03	1.58E+03	1.55E+02
F27	5.20E+02	6.33E+02	5.42E+02	5.53E+02	2.95E+01
F28	4.59E+02	5.08E+02	4.99E+02	4.83E+02	2.44E+01
F29	2.99E+02	7.19E+02	3.58E+02	3.95E+02	9.99E+01
F30	5.79E+05	9.60E+05	6.86E+05	6.94E+05	8.54E+04

4.3. Experiments on the 25 benchmark test functions with 30-D designed in IEEE CEC2005

In this subsection, the proposed IMPEDE is compared with other DE algorithms. The performance metrics are measured in terms of the average and standard deviation of the function error value $(f(x) - f(x^*))$, where x is the best solution found by the algorithm in a run and x^* is the global optimum of the best function. In order to have statistically sound conclusion, Wilcoxon's rank sum test at a 0.05 significance level is conducted on the experimental results. Notations "−", "+", and "≈" indicate that the corresponding comparative DE variant is significantly worse than, better than, and similar to IMPEDE, respectively.

The experimental results have been listed in Table 4. Unimodal Functions F_1–F_5: Clearly, all algorithms perform well on the shifted sphere function F_1. IMPEDE is the best among the six algorithms on these five unimodal functions. IMPEDE performs better than JADE, jDE, SaDE, CoDE and MPEDE on three $(F_3 - F_5)$, four $(F_2 - F_5)$, four $(F_2 - F_5)$, four $(F_2 - F_5)$, and four $(F_2 - F_5)$ test functions, respectively. Overall, IMPEDE provides outstanding and robust performance over other DE algorithms. Basic Multimodal Functions F_6–F_{12}: On these seven test functions, IMPEDE is significantly better than JADE, jDE, SaDE, CoDE and MPEDE on four $(F_6 - F_7$ and $F_{11} - F_{12})$, five $(F_6 - F_7$ and $F_{10} - F_{12})$, five $(F_6 - F_7$ and $F_{10} - F_{12})$, three $(F_6 - F_7$ and $F_{10})$ and two $(F_6$ and $F_{11})$ test functions, respectively. CoDE outperforms IMPEDE on one (F_8) test function, and JADE, jDE, SaDE, and MPEDE cannot be significantly better than IMPEDE on these seven test functions. It shows that IMPEDE could balance exploration and exploitation on these test function by combing the improved multi-population based mutation strategy ensemble approach with the improved parameter adaptation approach. Expanded Multimodal Functions F_{13}–F_{14}: On these two functions, JADE is the best among the six algorithms. IMPEDE is the second best and similar to CoDE, and outperform three other algorithms.

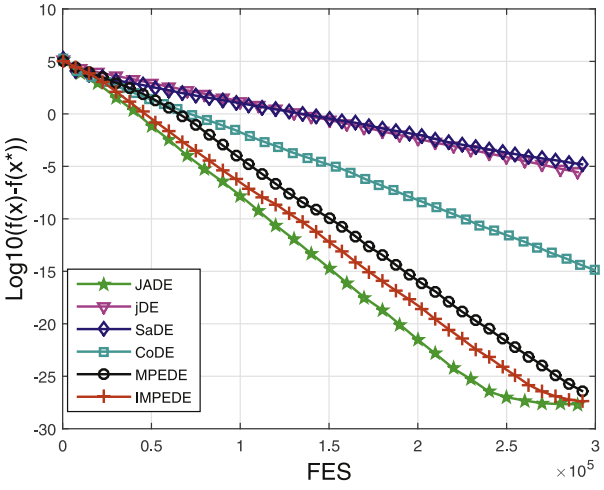

Fig. 1. The results on f_2.

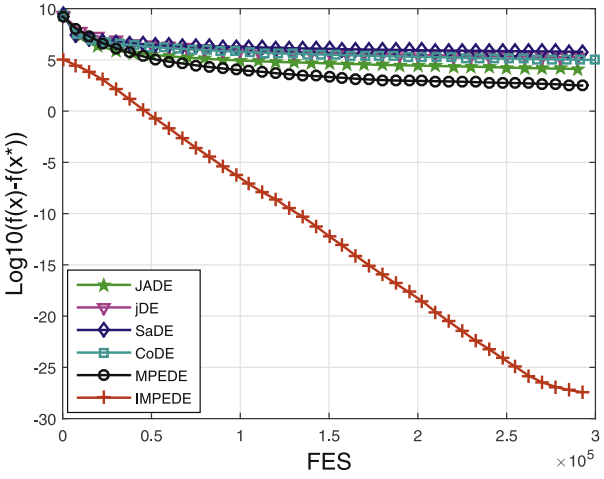

Fig. 2. The results on f_3.

Hybrid Composition Functions F_{15}–F_{25}: On these eleven functions, IMPEDE is significantly better than JADE, jDE, SaDE, CoDE and MPEDE on six $(F_{17} - F_{20}$, F_{23} and $F_{25})$, eight $(F_{16} - F_{20}$, $F_{22} - F_{23}$ and $F_{25})$, four $(F_{16}$, $F_{22} - F_{23}$ and $F_{25})$, five $(F_{18} - F_{20}$, F_{23} and $F_{25})$ and one (F_{19}) test functions, respectively. In contrast, SaDE, CoDE, and MPEDE are better than IMPEDE on two $(F_{17}$ and $F_{20})$, three $(F_{16} - F_{17}$ and $F_{21})$ and one (F_{23}) test functions, respectively. The last three rows of Wilcoxon's rank sum test in Table 4 show that IMPEDE is significantly better than JADE, jDE, SaDE, CoDE and MPEDE on 13 $(F_3 - F_7$, $F_{11} - F_{12}$, $F_{17} - F_{20}$, F_{23} and $F_{25})$, 18 $(F_2 - F_7$, $F_{10} - F_{12}$, F_{14}, $F_{16} - F_{20}$, $F_{22} - F_{23}$ and $F_{25})$, 15 $(F_2 - F_7$, $F_{10} - F_{14}$, F_{16}, $F_{22} - F_{23}$ and $F_{25})$, 12 $(F_2 - F_7$, F_{10}, $F_{18} - F_{20}$, F_{23} and $F_{25})$ and 8 $(F_2 - F_6$, F_{11}, F_{13} and $F_{19})$ functions, respectively. It is worse than JADE, SaDE, CoDE and MPEDE on 1 (F_{13}), 2 $(F_{17}$ and $F_{20})$, 4 $(F_8$, $F_{16} - F_{17}$ and F_{21})and 1 (F_{23}) functions, and similar to JADE, jDE, SaDE, CoDE and MPEDE on 11 $(F_1 - F_2$, $F_8 - F_{10}$, $F_{14} - F_{16}$, $F_{21} - F_{22}$, and $F_{24})$, 7 $(F_1$, $F_8 - F_9$, F_{13}, F_{15}, F_{21}, $F_{24})$, 8 $(F_1$, $F_8 - F_9$, F_{15}, $F_{18} - F_{19}$, F_{21} and $F_{24})$, 9 $(F_1$, F_9, $F_{11} - F_{15}$, F_{22} and $F_{24})$ and 16 $(F_1$, $F_7 - F_{10}$, F_{12}, $F_{14} - F_{18}$, $F_{20} - F_{22}$ and $F_{24} - F_{25})$ functions, respectively. IMPEDE performs better than other DE algorithms on most of the 30 dimensional problems.

Since the convergent speed is also key metric for these algorithms, we have evaluated the convergent speed from Figs. 1–18. It can be seen that X axial stands for the total number of function evaluations (FES) and Y axial represents for the logarithm of the average function error value. The convergence of IMPEDE is

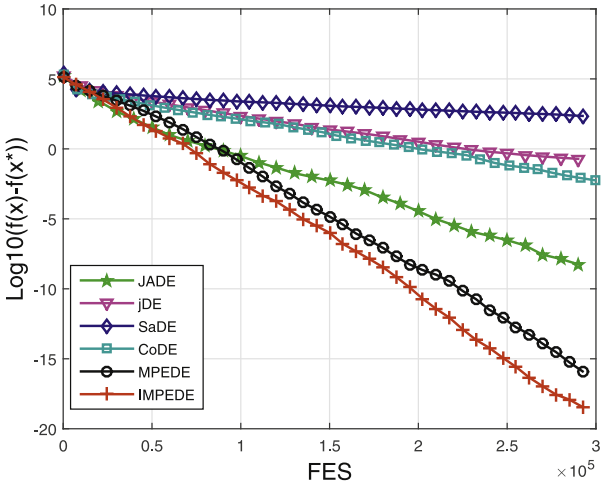

Fig. 3. The results on f_4.

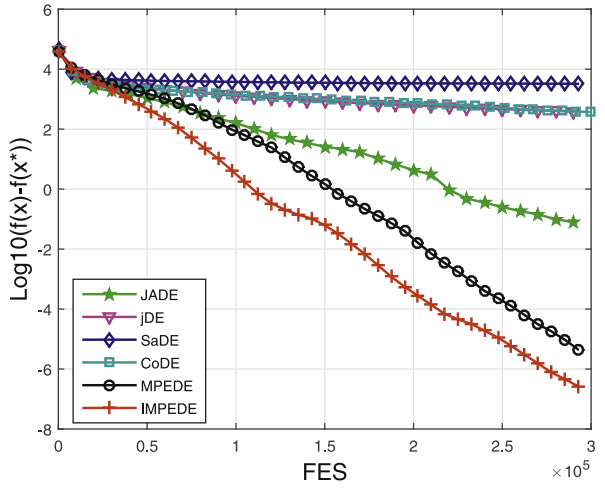

Fig. 4. The results on f_5.

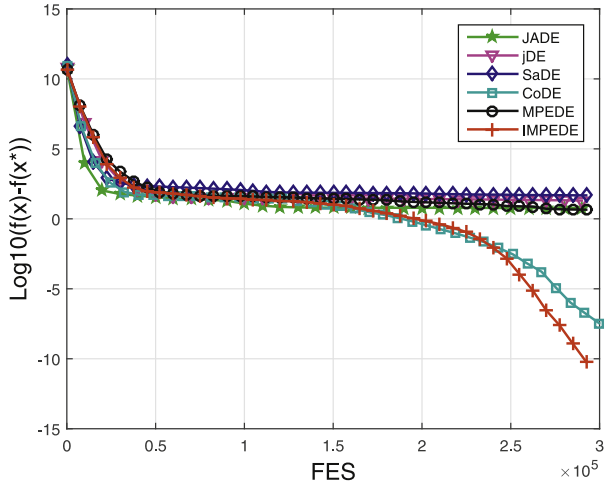

Fig. 5. The results on f_6.

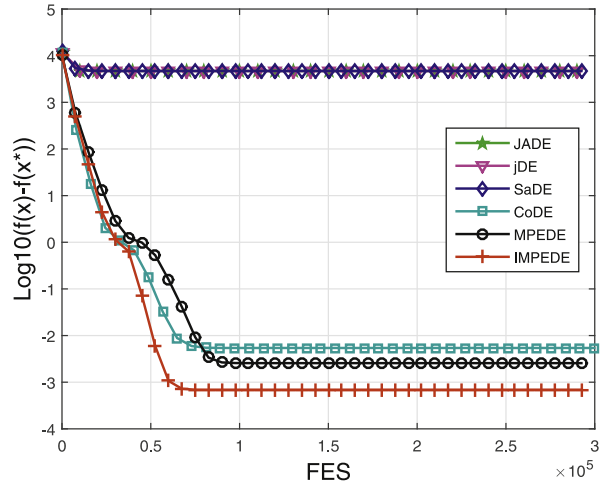

Fig. 6. The results on f_7.

compared with JADE, jDE, SaDE, CoDE and MPEDE. Among unimodal functions, the convergence graph of Shifted Schwefel's problem1.2 with noise in fitness (F_4) is presented in Fig. 3. It can be clearly observed that the proposed IMPEDE obtains better performance than other DE algorithms. Among multimodal functions, the convergence graph of Shifted Rosenbrock's function (F_6) is presented in Fig. 6. The proposed IMPEDE is able to jump out of local minimum and achieves the best performance. Among the hybrid composite functions, the convergence graph of Rotated hybrid composition function with high condition number matrix (F_{22}) is presented in Fig. 15. It shows that the proposed IMPEDE achieves the outstanding performance.

To sum up, from Table 4 and Figs. 1–18, due to the improvements we have made in IMPEDE, it outperforms than that of the other well-known algorithms. Also it demonstrates that IMPEDE can achieve the objectives of obtaining the global optimum solution and fastening the convergent speed.

4.4. Experiments on the 30 benchmark test functions with 10-D and 50-D designed in IEEE CEC2017

In this section, each algorithm runs 51 times for each test problem with a number of function evaluations equals to 10,000*D. IMPEDE gives the best, worst, median, mean and the standard deviation over the 51 runs of the error value between the best fitness values found in each run and the optimal value in Tables 5 and 7. Moreover, we compare IMPEDE against five competitive methods on the 30 test functions with 10-D and 50-D from IEEE CEC2017 to validate its performance: JADE, jDE, SaDE, CoDE, and MPEDE. From the mean error and standard deviation of the function error values given in Tables 6 and 8, the Wilcoxon's rank sum test is used to judge the significance of the results. For each of the competitive functions in those tables, "−", "+" and "≈" denote that the performance of the corresponding method is worse than, better than, and similar to that of IMPEDE, respectively.

In the case of $D = 10$, Table 5 shows that IMPEDE is successfully able to obtain the optimal solution in $F_1 - F_4$, F_6, and F_9. From the data given in Table 6, we can make several observations. Firstly, for unimodal functions $F_1 - F_3$, IMPEDE obtains significantly better results similar to all other five competitive methods. However, all methods can seek the optimal solution except for MPEDE on function F_1. Secondly, considering multimodal functions $F_4 - F_{10}$, IMPEDE outperforms JADE, jDE, SaDE, CoDE and MPEDE on one (F_{10}), five (F_4, F_5, F_7, F_8 and F_{10}), two (F_4 and F_{10}), two (F_7 and F_8) and six ($F_4 - F_8$ and F_{10}) benchmark functions, respectively. Thirdly, in regard to hybrid functions $F_{11} - F_{20}$, IMPEDE, SaDE and CoDE exhibit better performance than other comparative ones. IMPEDE is superior to SaDE and CoDE on five ($F_{11} - F_{14}$ and F_{18}) and three (F_{11} and $F_{14} - F_{15}$) functions while inferior to SaDE and CoDE on four ($F_{16} - F_{17}$ and $F_{19} - F_{20}$) and five ($F_{16} - F_{20}$) functions, respectively. Although JADE, jDE and MPEDE are better than IM-

Table 8
Testing results of DE algorithms for 50 dimensional CEC2017 benchmark functions.

Functions	JADE	jDE	SaDE	CoDE	MPEDE	IMPEDE
F1	0.00E+00	3.58E−08	2.79E+03	7.78E+01	0.00E+00	0.00E+00
	(0.00E+00)≈	(9.11E−08)−	(2.95E+03)−	(1.63E+02)−	(0.00E+00)≈	(0.00E+00)
F2	2.30E+14	8.80E+10	1.02E+02	2.22E+02	3.63E+08	6.08E−01
	(1.60E+15)−	(6.27E+11)−	(4.22E+01)−	(7.08E+02)−	(1.70E+09)−	(1.08E+00)
F3	2.73E+04	8.75E+00	1.46E+02	1.12E−09	3.93E−04	4.73E−03
	(4.37E+04)−	(5.49E+01)−	(2.71E+02)−	(4.69E−09)≈	(1.25E−03)+	(3.11E−02)
F4	4.03E+01	6.08E+01	9.90E+01	4.60E+01	4.30E+01	6.59E+01
	(4.63E+01)+	(4.71E+01)≈	(4.25E+01)−	(4.32E+01)+	(4.74E+01)+	(4.60E+01)
F5	5.46E+01	9.48E+01	9.50E+01	7.81E+01	5.60E+01	5.20E+01
	(8.73E+00)≈	(1.20E+01)−	(1.51E+01)−	(2.01E+01)−	(1.23E+01)≈	(1.32E+01)
F6	0.00E+00	0.00E+00	6.18E−03	3.48E−07	8.01E−04	8.47E−04
	(0.00E+00)+	(0.00E+00)+	(1.60E−02)−	(2.43E−06)+	(2.33E−03)+	(2.23E−03)
F7	1.01E+02	1.49E+02	1.45E+02	1.32E+02	1.10E+02	1.03E+02
	(7.04E+00)≈	(1.05E+01)−	(1.77E+01)−	(1.94E+01)−	(1.08E+01)−	(1.24E+01)
F8	5.40E+01	9.49E+01	9.60E+01	8.37E+01	5.30E+01	5.15E+01
	(7.74E+00)−	(1.05E+01)−	(2.01E+01)−	(2.15E+01)−	(1.21E+01)≈	(1.33E+01)
F9	9.61E−01	5.14E−02	5.53E+01	1.05E+01	8.78E−01	3.36E−01
	(1.16E+00)−	(1.32E−01)+	(8.19E+01)−	(1.43E+01)−	(8.18E−01)−	(4.12E−01)
F10	3.71E+03	5.17E+03	6.36E+03	4.21E+03	4.94E+03	5.32E+03
	(3.48E+02)+	(3.56E+02)≈	(1.41E+03)−	(7.32E+02)+	(8.14E+02)+	(6.50E+02)
F11	1.31E+02	5.44E+01	1.14E+02	5.61E+01	9.69E+01	7.99E+01
	(3.72E+01)−	(1.70E+01)+	(2.88E+01)−	(1.76E+01)+	(2.21E+01)−	(1.58E+01)
F12	5.19E+03	4.51E+04	1.19E+05	3.67E+04	8.78E+03	3.00E+03
	(3.26E+03)−	(3.64E+04)−	(7.56E+04)−	(2.29E+04)−	(7.55E+03)−	(1.83E+03)
F13	2.80E+02	1.59E+03	1.54E+03	3.04E+03	9.16E+01	7.43E+01
	(2.13E+02)−	(1.88E+03)−	(1.48E+03)−	(4.16E+03)−	(3.64E+01)−	(3.79E+01)
F14	1.46E+04	6.15E+01	2.18E+03	5.82E+01	6.26E+01	4.75E+01
	(4.28E+04)−	(2.06E+01)−	(3.15E+03)−	(1.67E+01)−	(1.70E+01)−	(9.72E+00)
F15	3.23E+02	8.53E+01	3.38E+03	1.67E+02	7.42E+01	4.84E+01
	(1.42E+02)−	(9.81E+01)−	(2.70E+03)−	(2.72E+02)−	(3.24E+01)−	(1.52E+01)
F16	8.25E+02	9.37E+02	8.20E+02	1.04E+03	8.22E+02	7.94E+02
	(1.67E+02)≈	(1.97E+02)−	(2.24E+02)≈	(2.84E+02)−	(3.38E+02)≈	(3.27E+02)
F17	6.07E+02	7.11E+02	4.88E+02	7.53E+02	5.87E+02	5.10E+02
	(1.54E+02)−	(1.36E+02)−	(1.71E+02)≈	(2.48E+02)−	(1.70E+02)≈	(1.69E+02)
F18	2.41E+04	2.12E+03	3.52E+04	3.66E+03	1.14E+02	5.35E+01
	(1.71E+05)−	(2.30E+03)−	(2.61E+04)−	(4.88E+03)−	(9.61E+01)−	(3.18E+01)
F19	6.15E+02	3.12E+01	1.47E+04	3.13E+01	4.63E+01	3.54E+01
	(2.03E+03)−	(1.38E+01)+	(5.68E+03)−	(2.00E+01)+	(2.40E+01)−	(1.15E+01)
F20	4.75E+02	5.68E+02	3.57E+02	5.28E+02	3.66E+02	4.18E+02
	(1.31E+02)≈	(1.06E+02)−	(1.31E+02)+	(2.09E+02)−	(1.76E+02)≈	(1.73E+02)
F21	2.52E+02	2.94E+02	2.84E+02	2.75E+02	2.51E+02	2.51E+02
	(8.89E+00)≈	(1.01E+01)−	(1.85E+01)−	(1.48E+01)−	(1.22E+01)≈	(1.10E+01)
F22	3.42E+03	4.43E+03	3.89E+03	4.92E+03	3.03E+03	2.35E+03
	(1.76E+03)−	(2.45E+03)−	(3.21E+03)≈	(8.53E+02)−	(2.73E+03)≈	(2.86E+03)
F23	4.80E+02	5.14E+02	5.20E+02	5.06E+02	4.81E+02	4.77E+02
	(1.08E+01)≈	(1.34E+01)−	(2.22E+01)−	(1.61E+01)−	(1.38E+01)≈	(1.34E+01)
F24	5.40E+02	5.77E+02	5.92E+02	5.73E+02	5.40E+02	5.36E+02
	(9.74E+00)−	(1.05E+01)−	(2.55E+01)−	(1.76E+01)−	(1.29E+01)≈	(1.33E+01)
F25	5.23E+02	5.21E+02	5.51E+02	5.25E+02	5.22E+02	5.05E+02
	(3.18E+01)−	(4.03E+01)−	(3.81E+01)−	(3.41E+01)−	(3.73E+01)−	(2.97E+01)
F26	1.62E+03	1.97E+03	2.55E+03	1.97E+03	1.61E+03	1.58E+03
	(1.12E+02)≈	(1.22E+02)−	(2.82E+02)−	(1.91E+02)−	(1.60E+02)≈	(1.55E+02)
F27	5.56E+02	5.35E+02	7.24E+02	5.50E+02	5.45E+02	5.53E+02
	(3.15E+01)≈	(1.79E+01)+	(6.10E+01)−	(2.75E+01)≈	(2.50E+01)≈	(2.95E+01)
F28	4.91E+02	4.85E+02	5.04E+02	4.86E+02	4.89E+02	4.83E+02
	(2.22E+01)−	(2.34E+01)−	(1.74E+01)−	(2.23E+01)≈	(2.40E+01)−	(2.44E+01)
F29	4.72E+02	5.13E+02	5.15E+02	5.30E+02	4.40E+02	3.95E+02
	(7.24E+01)−	(6.56E+01)−	(1.24E+02)−	(1.63E+02)−	(1.14E+02)−	(9.99E+01)
F30	6.66E+05	5.99E+05	8.01E+05	5.95E+05	6.68E+05	6.94E+05
	(9.74E+04)+	(2.80E+04)+	(9.27E+04)−	(2.06E+04)+	(7.79E+04)≈	(8.54E+04)
+/-/≈	4/17/9	6/22/2	1/26/3	6/21/3	4/13/13	

PEDE on one (F_{17}), two ($F_{19} - F_{20}$) and one (F_{12})functions, IMPEDE outperforms JADE, jDE and MPEDE on eight ($F_{11} - F_{16}$, F_{18} and F_{20}), seven ($F_{11} - F_{17}$) and nine (F_{11} and $F_{13} - F_{20}$) benchmark functions, respectively. Finally, considering the composition functions $F_{21} - F_{30}$, CoDE shows the best performance. Although CoDE outperforms IMPEDE on functions F_{22} and $F_{28} - F_{30}$, IMPEDE is capable of finding the better solutions for these functions at the cost of a few more function evaluations. IMPEDE is also very competitive. Actually, IMPEDE is superior to CoDE on functions F_{24} and F_{27}. In addition, IMPEDE outperforms JADE, jDE, SaDE and MPEDE on four (F_{22}, F_{23}, F_{27} and F_{29}), three(F_{22}, F_{23} and F_{29}), four(F_{22}, F_{25}, F_{27} and F_{29}) and five (F_{22}, F_{23}, F_{26}, F_{27} and F_{29}) benchmark functions, respectively.

In summary, the last row results of Wilcoxon's rank sum tests in the Table 5 indicate that IMPEDE is significantly better than JADE, jDE, SaDE, CoDE and MPEDE on 13, 15 11, 7 and 20 functions, respectively. It is worse than JADE, jDE, SaDE, CoDE and MPEDE on 1, 3, 7, 9 and 3 functions, and similar to them on 16, 12, 12, 14 and 7 functions, respectively. Actually, IMPEDE has the best performance compared with other four competitors, namely JADE, jDE,

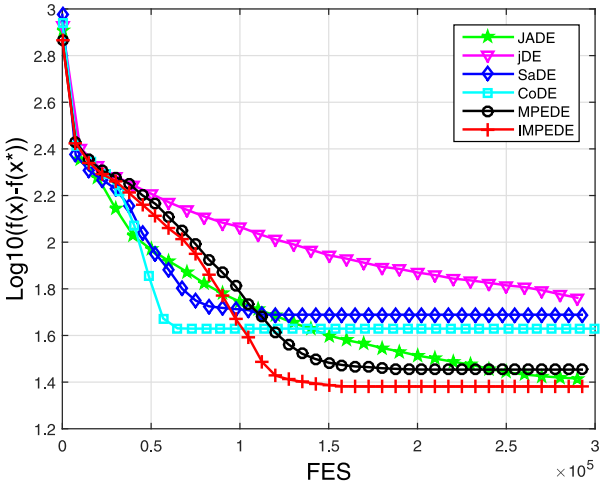

Fig. 7. The results on f_{10}.

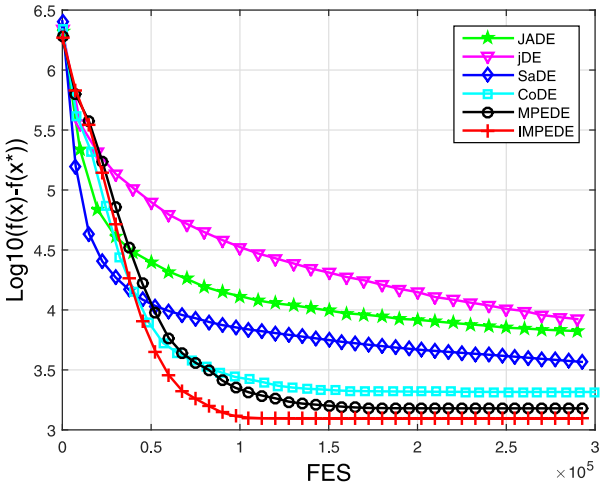

Fig. 8. The results on f_{12}.

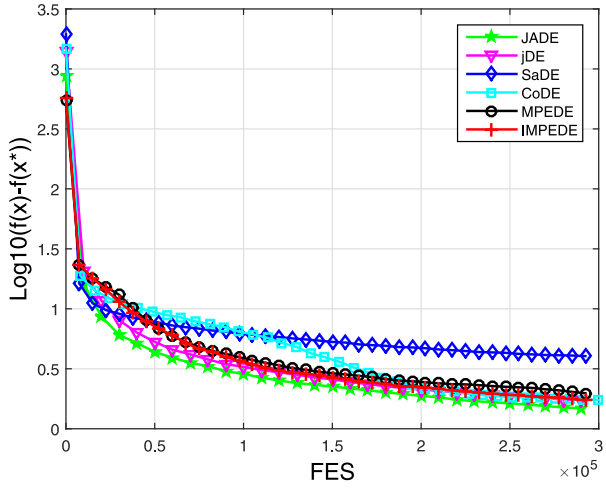

Fig. 9. The results on f_{13}.

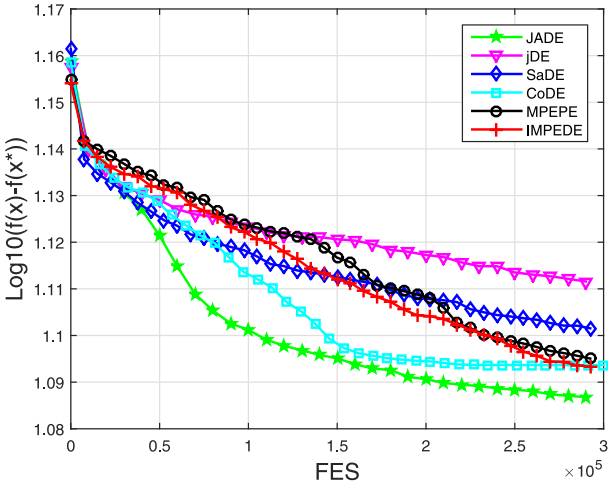

Fig. 10. The results on f_{14}.

SaDE and MPEDE (except CoDE) on all the 30 benchmark functions with 10 variables. Although CoDE is slightly better than IMPEDE, IMPEDE is also very competitive.

In the case of $D = 50$, we can observe from Table 7 that it is very difficult to obtain the optimal solution when D increases. IMPEDE is able to obtain the optimal solution for F_1. The results of DE algorithms for benchmark functions with 50 variables are given in Table 8. For unimodal functions, IMPEDE and MPEDE show the best performance. IMPEDE is superior to MPEDE on function (F_2) while inferior to MPEDE on function (F_3). Compared with JADE, jDE, SaDE and CoDE, IMPEDE outperforms these DE variants on two ($F_2 - F_3$), three ($F_1 - F_3$), three ($F_1 - F_3$) and two ($F_1 - F_2$) benchmark functions, respectively. For multimodal functions, JADE and MPEDE show the best performance. IMPEDE is also competitive and outperforms JADE and MPEDE on two ($F_8 - F_9$) and two (F_7 and F_9) functions, respectively. In addition, IMPEDE is comparable to JADE and MPEDE on functions (F_5 and F_7) and (F_5 and F_8). JDE, SaDE and CoDE are worse than IMPEDE on three (F_5 and $F_7 - F_8$), seven ($F_4 - F_{10}$) and four (F_5 and $F_7 - F_9$) functions. For hybrid functions, the overall performance of IMPEDE is superior to all comparative DE variants. Actually, IMPEDE is superior to JADE, jDE, SaDE, CoDE and MPEDE on eight ($F_{11} - F_{15}$ and $F_{17} - F_{19}$), eight ($F_{12} - F_{18}$ and F_{20}), seven($F_{11} - F_{15}$ and $F_{18} - F_{19}$), eight ($F_{12} - F_{18}$ and F_{20}) and seven ($F_{11} - F_{15}$ and $F_{18} - F_{19}$) functions, respectively. In contrast, IMPEDE is inferior to jDE, SaDE and CoDE on two (F_{11} and F_{19}), one (F_{20}) and two (F_{11} and F_{19}) functions, respectively. For composition functions, IMPEDE exhibits the best performance. In fact, IMPEDE surpasses JADE, jDE, SaDE, CoDE and MPEDE on five (F_{22}, $F_{24} - F_{25}$ and $F_{28} - F_{29}$), eight ($F_{21} - F_{26}$ and $F_{28} - F_{29}$), nine(F_{21} and $F_{23} - F_{30}$), seven ($F_{21} - F_{26}$ and F_{29}) and three (F_{25} and $F_{28} - F_{29}$) functions, respectively.

In summary, the last row results of Wilcoxon's rank sum tests reported in Table 7 indicate that IMPEDE is superior to JADE, jDE, SaDE, CoDE and MPEDE on 17, 22, 26, 21 and 13 functions, respectively. It is inferior to JADE, jDE, SaDE, CoDE and MPEDE on 4, 6, 1, 6 and 4 functions, while similar to them on 9, 2, 3, 3 and 13 functions, respectively. The superiority of IMPEDE to other comparative algorithms is more apparent when the number of decision variables in the test functions increases to 50. It is worth noting that the overall performance of IMPEDE is the best in our experiments.

4.4.1. Parameter analysis

We analyze the impacts of parameter *NP* (population size) on the performance of IMPEDE. To study how the performance of IMPEDE is sensitive to this parameter, we have tried different values of *NP*. We compare IMPEDE against other IMPEDE versions with different values on the 30 test functions with 10-D and 50-D from IEEE CEC2017. The mean error of the function error values is used to judge the results. "–", "+" and "≈" denote that the performance of the corresponding IMPEDE is worse than, better than, and

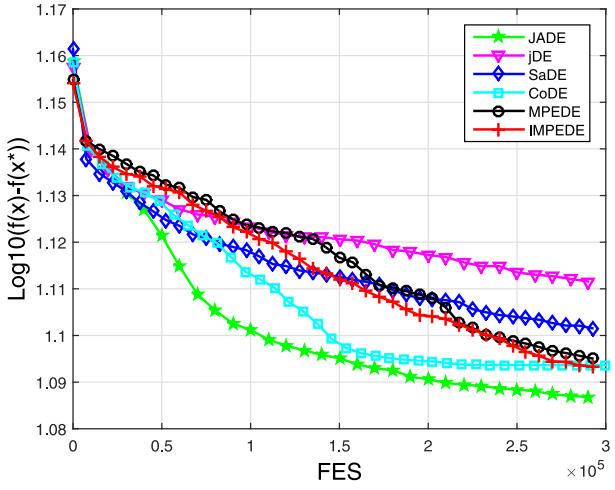

Fig. 11. The results on f_{15}.

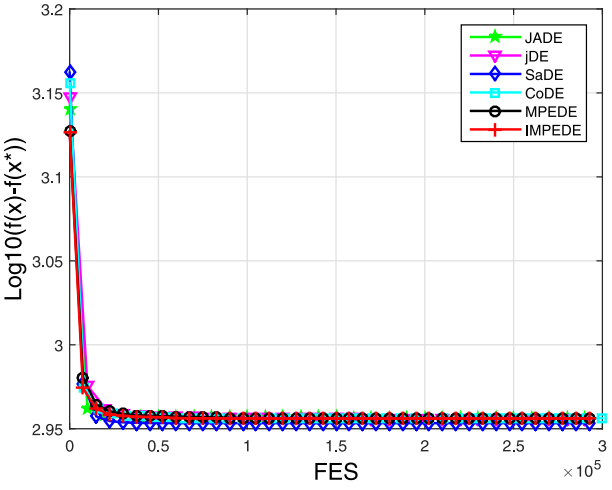

Fig. 13. The results on f_{20}.

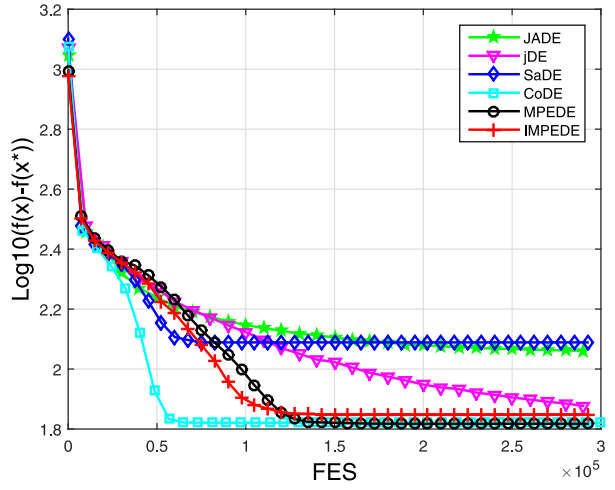

Fig. 12. The results on f_{16}.

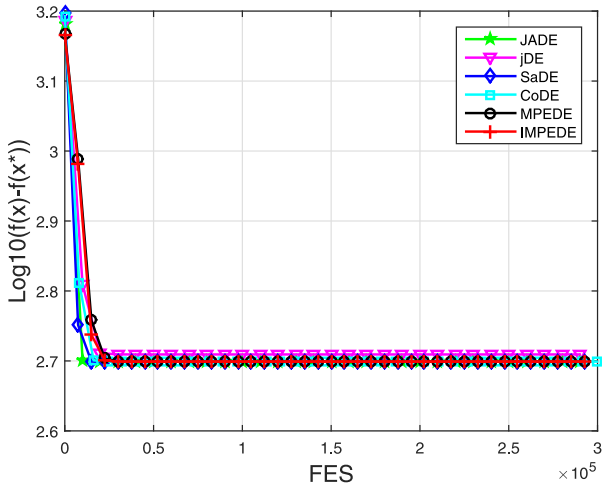

Fig. 14. The results on f_{21}.

similar to that of the default IMPEDE, respectively. The sensitivity analysis results of parameter *NP* are listed in Tables 9 and 10.

From the analysis results of parameter *NP* in Tables 9 and 10, we can find that parameter *NP* has great impacts on the performance of IMPEDE. Table 9 shows that IMPEDE with $NP = 125$ obtains the overall best performance. On 10-D CEC2017 functions, IMPEDE with $NP = 125$ is considered as the default IMPEDE and superior to IMPEDE with $NP = 50$, $NP = 100$, $NP = 200$ and $NP = 250$, respectively. As the dimension increases to 50 dimensions, Table 10 shows that IMPEDE with $NP = 400$ obtains the overall best performance. On 50-D CEC2017 functions, IMPEDE with $NP = 400$ is considered as the default IMPEDE and superior to IMPEDE with $NP = 50$, $NP = 100$, $NP = 200$ and $NP = 250$, respectively.

4.5. Discussion

In this section, the additional experiments are carried out on the 10 benchmark test functions ($F_1 - F_{10}$) with 10-D from IEEE CEC2017. For all the experiments, 51 runs are executed and the allowed maximum function evaluations (*FES*) are set to 100,000. For each of the competitive functions in those tables, "−", "+" and "≈" denote that the performance of the corresponding method is worse than, better than, and similar to that of IMPEDE, respectively.

The comparison of the mean functional evaluations for each benchmark function is shown in the Table 11. IMPEDE is superior to JADE, jDE, CoDE, and MPEDE, while inferior to SaDE. Furthermore, the mean computation time of each algorithm for each benchmark functions is shown in Table 12. JADE and jDE show the shortest mean computation time. IMPEDE is also competitive. IMPEDE is superior to SaDE, CoDE, and MPEDE. However, in order to further test the performance, we are more interested in the percentage of problems that can be solved with given function evaluations. We use data profiles [44] to have an insight into this aspect. The percentage of problems is described as follow:

$$d_s(\alpha) = \frac{1}{|P|} size\{p \in P; t_{p,s} \leq \alpha\} \tag{24}$$

where $d_s(\alpha)$ is the percentage of problems that can be solved with α function evaluations by s solver and $|P|$ denotes the cardinality of the set P benchmark problems and $t_{p,s}$ is the number of function evaluation required to finding the best values on the p problem by s solver. In Fig. 19, it can be seen that X axial stands for the total number of function evaluations (*FES*) and Y axial represents for the percentage of problems. As *FES* increases, IMPEDE displays strong extensibility. Fig. 19 shows that IMPEDE exhibits the best performance than JADE, jDE, SaDE, CoDE and MPEDE. Overall, IMPEDE shows the better performance on the mean computation time, the mean functional evaluations and the percentage of problems. In addition, we have tested the efficiency of the improved multi-population based mutation strategy ensemble approach and the improved parameter adaptation approach. We

Table 9
Computational results of IMPEDE with different population sizes over 10 dimensional CEC2017 benchmark functions.

Functions	NP = 50	NP = 100	NP = 200	NP = 250	NP = 125 (standard)
F1	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F2	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F3	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F4	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F5	5.91E+00(−)	3.29E+00(−)	4.39E+00(−)	5.37E+00(−)	2.88E+00
F6	2.79E−08(−)	0.00E+00(≈)	8.41E−07(−)	2.35E−05(−)	0.00E+00
F7	1.55E+01(−)	1.31E+01(+)	1.61E+01(−)	1.66E+01(−)	1.33E+01
F8	5.89E+00(−)	3.75E+00(−)	4.20E+00(−)	5.57E+00(−)	3.34E+00
F9	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F10	2.01E+02(−)	7.40E+01(−)	1.50E+02(−)	2.17E+02(−)	7.20E+01
F11	1.29E+00(−)	1.56E−01(−)	5.91E−01(−)	1.96E+00(−)	3.90E−02
F12	1.51E+02(−)	1.53E+01(−)	4.45E+00(+)	1.20E+01(+)	1.50E+01
F13	3.60E+00(−)	2.11E+00(−)	3.90E+00(−)	5.36E+00(−)	1.89E+00
F14	2.81E+00(−)	2.17E−01(−)	3.26E+00(−)	5.93E+00(−)	1.82E−01
F15	3.10E−01(−)	8.35E−02(+)	1.93E−01(−)	6.08E−01(−)	9.19E−02
F16	1.21E+01(−)	5.07E−01(+)	1.65E+00(−)	2.58E+00(−)	6.18E−01
F17	2.35E+00(−)	3.75E−01(+)	5.45E+00(−)	9.99E+00(−)	7.78E−01
F18	3.72E+00(−)	2.19E−01(−)	3.88E−01(−)	2.39E+00(−)	1.38E−01
F19	8.76E−02(−)	2.42E−02(+)	5.45E−01(−)	9.58E−01(−)	6.70E−02
F20	6.78E−01(−)	2.45E−02(+)	8.89E−01(−)	4.16E+00(−)	3.06E−02
F21	1.86E+02(−)	1.64E+02(−)	1.35E+02(+)	1.33E+02(+)	1.60E+02
F22	9.85E+01(−)	9.61E+01(−)	9.02E+01(−)	9.61E+01(−)	8.85E+01
F23	3.07E+02(−)	3.05E+02(−)	3.05E+02(−)	3.00E+02(+)	3.04E+02
F24	3.19E+02(−)	3.16E+02(−)	3.00E+02(−)	2.80E+02(+)	2.87E+02
F25	4.21E+02(−)	4.18E+02(−)	4.03E+02(+)	4.05E+02(+)	4.09E+02
F26	3.19E+02(−)	3.00E+02(≈)	3.00E+02(≈)	3.00E+02(≈)	3.00E+02
F27	3.90E+02(−)	3.89E+02(≈)	3.89E+02(≈)	3.89E+02(≈)	3.89E+02
F28	4.18E+02(−)	3.97E+02(−)	3.27E+02(+)	3.18E+02(+)	3.72E+02
F29	2.41E+02(−)	2.37E+02(+)	2.46E+02(−)	2.47E+02(−)	2.38E+02
F30	1.46E+05(−)	1.64E+04(+)	3.95E+02(+)	3.95E+02(+)	3.24E+04
+/−/≈	0/25/5	8/14/8	5/18/7	7/16/7	

Table 10
Computational results of IMPEDE with different population sizes over 50 dimensional CEC2017 benchmark functions.

Functions	NP = 50	NP = 100	NP = 200	NP = 250	NP = 400 (stanadard)
F1	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F2	2.55E+29(−)	5.03E+21(−)	3.04E+06(−)	3.24E+10(−)	6.08E−01
F3	4.35E−07(+)	4.67E−04(+)	1.08E−03(+)	2.63E−03(+)	4.73E−03
F4	5.26E+01(+)	4.05E+01(+)	6.10E+01(+)	4.84E+01(+)	6.59E+01
F5	1.09E+02(−)	7.18E+01(−)	5.36E+01(−)	5.72E+01(−)	5.20E+01
F6	1.43E+00(−)	1.46E−01(−)	1.09E−02(−)	6.82E−03(−)	8.47E−04
F7	2.08E+02(−)	1.41E+02(−)	1.16E+02(−)	1.09E+02(−)	1.03E+02
F8	1.06E+02(−)	7.12E+01(−)	5.91E+01(−)	5.59E+01(−)	5.15E+01
F9	4.88E+02(−)	3.63E+01(−)	3.93E+00(−)	2.06E+00(−)	3.36E−01
F10	4.95E+03(+)	4.57E+03(+)	4.10E+03(+)	4.31E+03(+)	5.32E+03
F11	2.08E+02(−)	1.58E+02(−)	1.17E+02(−)	1.09E+02(−)	7.99E+01
F12	9.01E+03(−)	9.86E+03(−)	1.14E+04(−)	1.11E+04(−)	3.00E+03
F13	9.27E+02(−)	3.22E+02(−)	1.29E+02(−)	9.60E+01(−)	7.43E+01
F14	2.77E+02(−)	1.69E+02(−)	7.67E+01(−)	6.31E+01(−)	4.75E+01
F15	5.39E+02(−)	2.93E+02(−)	1.11E+02(−)	7.69E+01(−)	4.84E+01
F16	1.21E+03(−)	1.05E+03(−)	9.37E+02(−)	8.81E+02(−)	7.94E+02
F17	1.00E+03(−)	7.32E+02(−)	5.82E+02(−)	5.57E+02(−)	5.10E+02
F18	1.03E+03(−)	2.33E+02(−)	1.81E+02(−)	1.37E+02(−)	5.35E+01
F19	1.68E+02(−)	1.41E+02(−)	8.41E+01(−)	5.57E+01(−)	3.54E+01
F20	7.57E+02(−)	5.29E+02(−)	4.44E+02(−)	3.77E+02(+)	4.18E+02
F21	2.91E+02(−)	2.64E+02(−)	2.60E+02(−)	2.54E+02(−)	2.51E+02
F22	5.16E+03(−)	4.45E+03(−)	3.45E+03(−)	3.95E+03(−)	2.35E+03
F23	5.40E+02(−)	5.02E+02(−)	4.86E+02(−)	4.83E+02(−)	4.77E+02
F24	5.90E+02(−)	5.62E+02(−)	5.44E+02(−)	5.41E+02(−)	5.36E+02
F25	5.35E+02(−)	5.26E+02(−)	5.15E+02(−)	5.09E+02(−)	5.05E+02
F26	2.36E+03(−)	1.85E+03(−)	1.65E+03(−)	1.61E+03(−)	1.58E+03
F27	6.94E+02(−)	6.15E+02(−)	5.64E+02(−)	5.51E+02(+)	5.53E+02
F28	4.86E+02(−)	4.93E+02(−)	4.89E+02(−)	4.83E+02(≈)	4.83E+02
F29	1.17E+03(−)	8.08E+02(−)	5.47E+02(−)	4.52E+02(−)	3.95E+02
F30	7.73E+05(−)	7.72E+05(−)	7.40E+05(−)	7.19E+05(−)	6.94E+05
+/−/≈	3/26/1	3/26/1	3/26/1	5/23/2	

Table 11
The mean functional evaluations of DE algorithms for 10 benchmark functions.

Functions	JADE	jDE	SaDE	CoDE	MPEDE	IMPEDE
F1	3.69E+04(+)	7.39E+04(−)	3.03E+04(+)	4.17E+04(−)	9.12E+04(−)	4.15E+04
F2	1.59E+04(−)	2.30E+04(−)	9.14E+03(+)	1.25E+04(+)	3.26E+04(−)	1.42E+04
F3	2.72E+04(+)	5.88E+04(−)	2.29E+04(+)	3.45E+04(−)	7.10E+04(−)	3.25E+04
F4	3.68E+04(+)	1.00E+05(−)	1.00E+05(−)	4.57E+04(−)	9.47E+04(−)	3.87E+04
F5	1.00E+05(−)	1.00E+05(−)	1.00E+05(−)	1.00E+05(−)	1.00E+05(−)	9.99E+04
F6	7.02E+04(−)	4.75E+04(+)	2.57E+04(+)	4.95E+04(+)	1.00E+05(−)	6.96E+04
F7	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05
F8	1.00E+05(≈)	1.00E+05(≈)	9.94E+04(+)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05
F9	3.26E+04(−)	3.49E+04(−)	1.39E+04(+)	2.90E+04(+)	6.93E+04(−)	3.01E+04
F10	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05(≈)	1.00E+05
+/−/≈	3/4/3	1/6/3	6/2/2	3/4/3	0/7/3	

Table 12
The mean computation time of DE algorithms for 10 benchmark functions.

Functions	JADE	jDE	SaDE	CoDE	MPEDE	IMPEDE
F1	1.47E−01(+)	2.91E−01(−)	2.03E+00(−)	8.73E−01(−)	3.36E−01(−)	2.67E−01
F2	7.23E−02(+)	9.40E−02(+)	5.94E−01(−)	3.06E−01(−)	1.40E−01(−)	9.57E−02
F3	1.07E−01(+)	2.33E−01(−)	1.59E+00(−)	7.22E−01(−)	2.69E−01(−)	2.04E−01
F4	1.47E−01(+)	3.62E−01(−)	6.98E+00(−)	9.55E−01(−)	3.56E−01(−)	2.35E−01
F5	4.30E−01(+)	4.35E−01(+)	6.96E+00(−)	2.27E+00(−)	4.40E−01(+)	6.55E−01
F6	3.98E−01(+)	2.67E−01(+)	1.67E+00(−)	1.11E+00(−)	5.56E−01(−)	5.36E−01
F7	4.43E−01(+)	4.17E−01(+)	6.49E+00(−)	2.11E+00(−)	4.23E−01(+)	6.47E−01
F8	4.37E−01(+)	4.21E−01(+)	6.93E+00(−)	2.22E+00(−)	4.18E−01(+)	6.44E−01
F9	1.54E−01(+)	1.58E−01(+)	8.80E−01(−)	6.37E−01(−)	2.93E−01(−)	2.09E−01
F10	4.87E−01(+)	4.70E−01(+)	6.58E+00(−)	2.42E+00(−)	4.60E−01(+)	7.23E−01
+/−/≈	10/0/0	7/3/0	0/10/0	0/10/0	4/6/0	

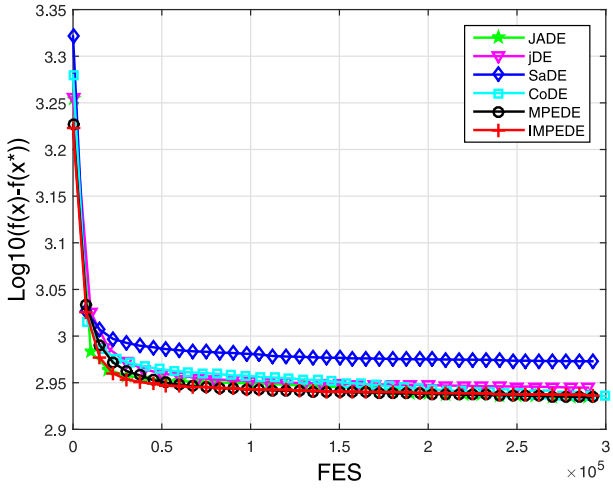

Fig. 15. The results on f_{22}.

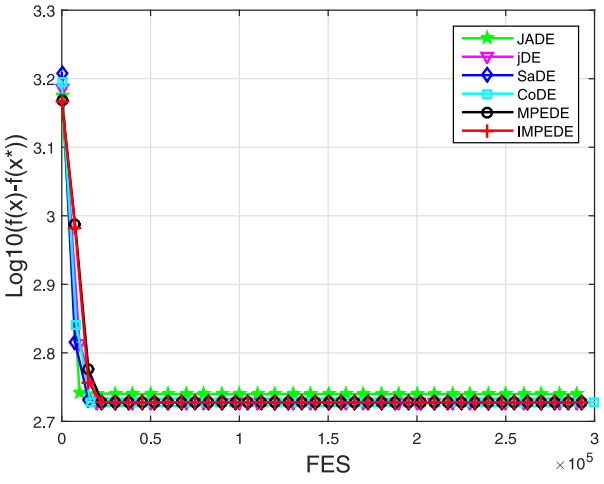

Fig. 16. The results on f_{23}.

implement a variant of IMPEDE, called IMPEDE(M), in which the improved multi-population based mutation strategy ensemble approach is only utilized. We also consider another variant of IMPEDE, called IMPEDE(P), in which the improved parameter adaptation approach is only employed. The mean error of the function error values obtained from MPEDE, IMPEDE(M), IMPEDE(P) and IMPEDE has been given in the Table 13. IMPEDE performs better than MPEDE, IMPEDE(M) and IMPEDE(P). Based on the above comparison, the improved multi-population based mutation strategy ensemble approach and the improved parameter adaptation approach do play a crucial role in IMPEDE.

4.6. An application to the hydrothermal scheduling problem

In order to test the efficiency of MPEDE in dealing with real-world optimization problems, we apply IMPEDE to the hydrothermal scheduling problem from the CEC2011 benchmark [45].

4.6.1. Formulation of the hydrothermal scheduling problem

Generally, the objective function to be minimized in a hydrothermal scheduling problem is the overall fuel cost of thermal units for the given short term. The total fuel cost for operation of the thermal system so as to meet the load demands in the scheduling period is given by *F*. The objective function is represented as:

$$Minimize : \mathrm{F} = \sum_{\mathrm{i}=1}^{\mathrm{M}} f_i(P_{Ti}) \tag{25}$$

where f_i is the cost function corresponding to the equivalent thermal unit's power generation P_{Ti} at *i*th interval. *M* is the total number of intervals considered for the short term schedule. The cost function f_i is formulated as:

$$f_i(P_{Ti}) = a_i {P_{Ti}}^2 + \mathrm{b}_i P_{Ti} + c_i + |e_i \sin(d_i(P_{Ti}^{\min} - P_{Ti}))| \tag{26}$$

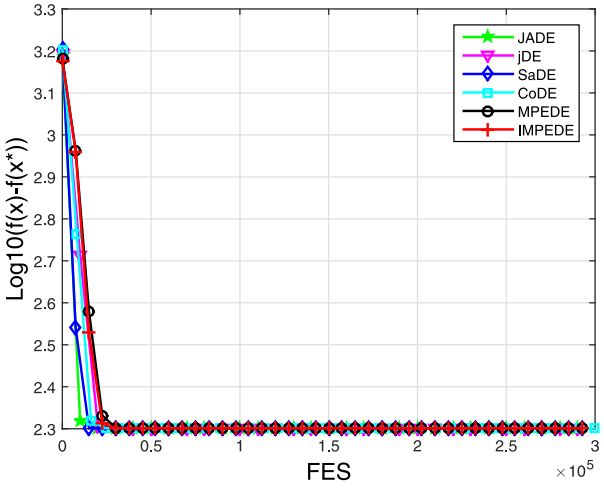

Fig. 17. The results on f_{24}.

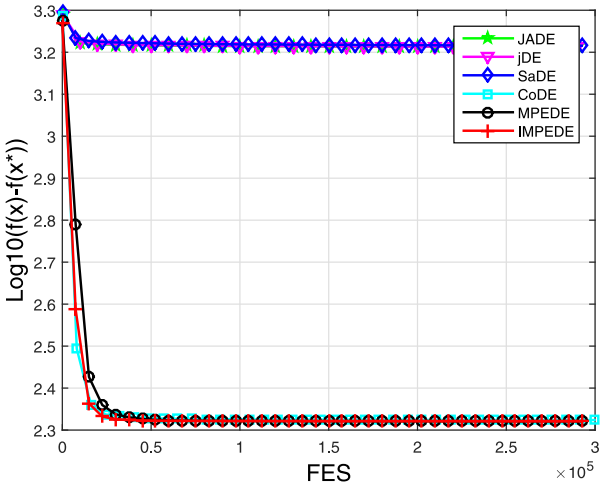

Fig. 18. The results on f_{25}.

Table 13
Test results of IMPEDE variants for 10 benchmark functions.

Functions	MPEDE	IMPEDE(M)	IMPEDE(P)	IMPEDE
F1	1.33E−09(−)	3.37E−09(−)	0.00E+00(≈)	0.00E+00
F2	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F3	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F4	1.46E−08(−)	1.87E−05(−)	0.00E+00(≈)	0.00E+00
F5	6.16E+00(−)	6.08E+00(−)	5.81E+00(−)	2.88E+00
F6	2.24E−05(−)	2.89E−05(−)	2.05E−05(−)	0.00E+00
F7	1.79E+01(−)	1.75E+01(−)	1.71E+01(−)	1.33E+01
F8	6.10E+00(−)	6.27E+00(−)	5.48E+00(−)	3.34E+00
F9	0.00E+00(≈)	0.00E+00(≈)	0.00E+00(≈)	0.00E+00
F10	2.72E+02(−)	2.41E+02(−)	2.15E+02(−)	7.20E+01
+/−/≈	0/7/3	0/7/3	0/5/5	

where a_i, b_i, c_i, e_i and d_i are the cost coefficients. Various system constraints need to be satisfied which are provided in [45].

4.6.2. Experimental results

Three hydrothermal scheduling instances are solved by IMPEDE, respectively. The computational results of IMPEDE are compared with those of algorithms in recent literature. GA-MPC [46] is the winner of the CEC2011 competition and SAMODE [47] is also competitive for the CEC2011 competition. PAL, AL, and TLBO [48] are proposed in the latest publication. The results of the comparison are shown in the Tables 14–16. The symbol "NA" in tables denotes that the related data are not reported in the reference. From these

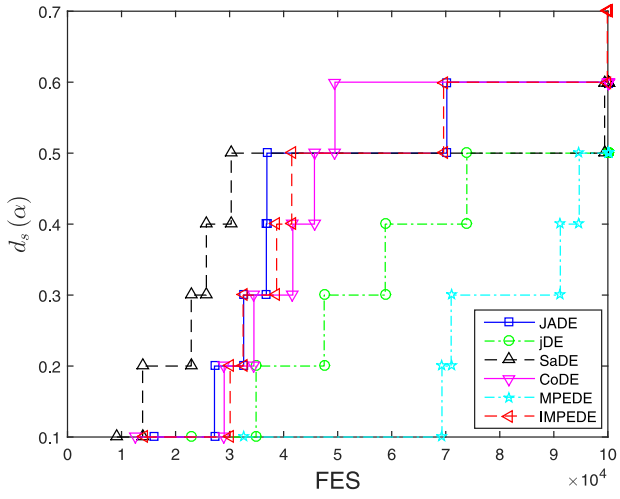

Fig. 19. The results of the percentage of problems.

Table 14
Comparison of optimization results of the hydrothermal scheduling instance 1.

Methods	Best	Worst	Median	Mean	Std
IMPEDE	**9.31E+05**	**9.40E+05**	**9.35E+05**	**9.35E+05**	**2.08E+03**
GA-MPC [46]	9.50E+05	9.95E+05	9.70E+05	9.71E+05	1.04E+04
SAMODE [47]	9.43E+05	9.57E+05	9.49E+05	9.49E+05	3.91E+03
PAL [48]	9.39E+05	NA	NA	9.40E+05	1.66E+03
AL [48]	9.42E+05	NA	NA	1.08E+06	2.75E+05
TLBO [48]	9.42E+05	NA	NA	1.28E+06	8.49E+05

Table 15
Comparison of optimization results of the hydrothermal scheduling instance 2.

Methods	Best	Worst	Median	Mean	Std
IMPEDE	**9.39E+05**	**9.50E+05**	**9.43E+05**	**9.43E+05**	**2.93E+03**
GA-MPC [46]	9.72E+05	1.21E+06	1.05E+06	1.06E+06	5.70E+04
SAMODE [47]	1.01E+06	1.38E+06	1.18E+06	1.21E+06	9.95E+04
PAL [48]	1.01E+06	NA	NA	1.08E+06	6.52E+04
AL [48]	1.09E+06	NA	NA	1.66E+06	4.02E+05
TLBO [48]	1.01E+06	NA	NA	1.54E+06	7.93E+05

Table 16
Comparison of optimization results of the hydrothermal scheduling instance 3.

Methods	Best	Worst	Median	Mean	Std
IMPEDE	**9.31E+05**	**9.41E+05**	**9.36E+05**	**9.36E+05**	**2.64E+03**
GA-MPC [46]	9.47E+05	9.95E+05	9.76E+05	9.75E+05	1.18E+04
SAMODE [47]	9.48E+05	9.70E+05	9.56E+05	9.59E+05	5.99E+03
PAL [48]	9.41E+05	NA	NA	9.46E+05	2.44E+03
AL [48]	9.42E+05	NA	NA	1.08E+06	2.75E+05
TLBO [48]	9.42E+05	NA	NA	1.28E+06	8.49E+05

tables, IMPEDE is able to find the best results for the hydrothermal scheduling instances. It can be pointed out that IMPEDE is a good alternative approach for solving the hydrothermal scheduling problems and also has potential in dealing with other real-world optimization problems.

5. Conclusion and future work

In this paper, we have proposed an improved Multi-population ensemble differential evolution algorithm (IMPEDE). Compare to the previous work, IMPEDE has shown the strong capability of jumping out of the local optimum and fasten the speed of the convergence. Because the "DE/current-to-pbest/1" with an archive and "DE/current-to-rand/1" are the key strategies for solving optimization problems in MPEDE, IMPEDE combine these two strategies with a new mutation strategy "DE/pbad-to-pbest/1" called the

improved multi-population based mutation strategy ensemble approach to search the global optimal solution using MPEDE framework. We propose a new mutation strategy "DE/pbad-to-pbest/1" which utilizes not only the good solution information but also the information of the bad solution (*pbad*) toward the good solution (*pbest*) to balance exploration and exploitation. In the exploration process, the new mutation strategy takes full advantage of the good information to accelerate the convergence rate. On the other hand, it applies the information of the bad solution toward the good solution to enhance the diversity and increase the probability of finding the global optimum and jumping out of the local optimum in the exploitation process. Furthermore, to tackle the issue of premature convergence caused by applying arithmetic mean in the control parameter of "DE/current-to-pbest/1", IMPEDE employs the improved parameter adaptation approach to make slight modifications on the "DE/current-to-pbest/1" strategy by adding the weighted Lehmer mean strategy on the adaptation of the control parameter. The simulation is conducted and the proposed algorithm is tested via the CEC2005 and CEC2017 benchmark functions. The experimental results have demonstrated that the proposed IMPEDE algorithm outperforms than of other alternative algorithm with the goals of obtaining the global optimum and accelerating the convergence speed.

From the experimental results, it is evident that the proposed approach has substantial potential in handing unconstrained optimization problems, and its performance remarkably outperforms other algorithms presented in recent literature. Meanwhile, it should be noted that there are fewer parameters to be tuned by user. These characters are extremely important merits for unconstrained optimization problems. However, how to extend IMPEDE to more complex optimization problems should be investigated. In the future work, we plan to apply IMPEDE to solve more complex optimization problems to further test its performance, such as constrained optimization, multi-objective optimization and resource scheduling.

Acknowledgments

The authors would like to express their sincere thanks to the Associate Editor and the anonymous reviewers for their valuable suggesti ons and comments on this paper. This work is supported by the National Natural Science Foundation of China (61563012, 61203109), Guangxi Natural Science Foundation (2014GXNSFAA118371, 2015GXNSFBA139260).

References

[1] R. Storn, K. Price, Differential Evolution-a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, International Computer Science Institute, Berkeley, CA, 1995.
[2] R. Storn, K. Price, Differential Evolution. A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Kluwer Academic Publishers, 1997.
[3] W. Gong, Z. Cai, D. Liang, Adaptive ranking mutation operator based differential evolution for constrained optimization, IEEE Trans. Cybern. 45 (4) (2015) 716–727, doi:10.1109/TCYB.2014.2334692.
[4] Y. Wang, Z. Cai, Combining multiobjective optimization with differential evolution to solve constrained optimization problems, IEEE Trans. Evolut. Comput. 16 (1) (2012) 117–134, doi:10.1109/TEVC.2010.2093582.
[5] X. Qiu, J.X. Xu, K.C. Tan, H.A. Abbass, Adaptive cross-generation differential evolution operators for multiobjective optimization, IEEE Trans. Evolut. Comput. 20 (2) (2016) 232–244, doi:10.1109/TEVC.2015.2433672.
[6] J. Wang, W. Zhang, J. Zhang, Cooperative differential evolution with multiple populations for multiobjective optimization, IEEE Trans. Cybern. 46 (12) (2016) 2848–2861, doi:10.1109/TCYB.2015.2490669.
[7] S. Das, A. Mandal, R. Mukherjee, An adaptive differential evolution algorithm for global optimization in dynamic environments, IEEE Trans. Cybern. 44 (6) (2014) 966–978, doi:10.1109/TCYB.2013.2278188.
[8] J.-T. Tsai, Improved differential evolution algorithm for nonlinear programming and engineering design problems, Neurocomputing 148 (2015) 628–640, doi:10.1016/j.neucom.2014.07.001.
[9] L. Tong, W. Wong, C. Kwong, Differential evolution-based optimal Gabor filter model for fabric inspection, Neurocomputing 173 (2016) 1386–1401, doi:10.1016/j.neucom.2015.09.011.
[10] Y.-J. Zheng, X.-L. Xu, H.-F. Ling, S.-Y. Chen, A hybrid fireworks optimization method with differential evolution operators, Neurocomputing 148 (2015) 75–82, doi:10.1016/j.neucom.2012.08.075.
[11] W. Li Xiang, N. Zhu, S. Feng Ma, X. Lei Meng, M. Qing An, A dynamic shuffled differential evolution algorithm for data clustering, Neurocomputing 158 (2015) 144–154, doi:10.1016/j.neucom.2015.01.058.
[12] W.J. Yu, M. Shen, W.N. Chen, Z.H. Zhan, Y.J. Gong, Y. Lin, O. Liu, J. Zhang, Differential evolution with two-level parameter adaptation, IEEE Trans. Cybern. 44 (7) (2014) 1080–1099, doi:10.1109/TCYB.2013.2279211.
[13] S. Das, S.S. Mullick, P. Suganthan, Recent advances in differential evolution – an updated survey, Swarm Evolut. Comput. 27 (2016) 1–30, doi:10.1016/j.swevo.2016.01.004.
[14] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evolut. Comput. 15 (1) (2011) 4–31, doi:10.1109/TEVC.2010.2059031.
[15] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evolut. Comput. 13 (2) (2009) 398–417, doi:10.1109/TEVC.2008.927706.
[16] S. Das, A. Abraham, U.K. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE Trans. Evolut. Comput. 13 (3) (2009) 526–553, doi:10.1109/TEVC.2008.2009457.
[17] S. Rahnamayan, H.R. Tizhoosh, M.M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evolut. Comput. 12 (1) (2008) 64–79, doi:10.1109/TEVC.2007.894200.
[18] J. Ronkkonen, S. Kukkonen, K.V. Price, Real-parameter optimization with differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 1, 2005, pp. 506–513Vol.1, doi:10.1109/CEC.2005.1554725.
[19] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evolut. Comput. 13 (5) (2009) 945–958, doi:10.1109/TEVC.2009.2014613.
[20] G. Wu, R. Mallipeddi, P. Suganthan, R. Wang, H. Chen, Differential evolution with multi-population based ensemble of mutation strategies, Inf. Sci. 329 (2016) 329–345, doi:10.1016/j.ins.2015.09.009. Special issue on Discovery Science
[21] N. Noman, H. Iba, Accelerating differential evolution using an adaptive local search, IEEE Trans. Evolut. Comput. 12 (1) (2008) 107–125, doi:10.1109/TEVC.2007.895272.
[22] W. Gong, Z. Cai, Differential evolution with ranking-based mutation operators, IEEE Trans. Cybern. 43 (6) (2013) 2066–2081, doi:10.1109/TCYB.2013.2239988.
[23] F. Peng, K. Tang, G. Chen, X. Yao, Multi-start jade with knowledge transfer for numerical optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2009, pp. 1889–1895, doi:10.1109/CEC.2009.4983171.
[24] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2013, pp. 71–78, doi:10.1109/CEC.2013.6557555.
[25] R. Mallipeddi, P. Suganthan, Q. Pan, M. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679–1696, doi:10.1016/j.asoc.2010.04.024. The Impact of Soft Computing for the Progress of Artificial Intelligence
[26] S.M. Islam, S. Das, S. Ghosh, S. Roy, P.N. Suganthan, An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization, IEEE Trans. Syst. Man Cybern. Part B (Cybern.) 42 (2) (2012) 482–500.
[27] D. Zou, J. Wu, L. Gao, S. Li, A modified differential evolution algorithm for unconstrained optimization problems, Neurocomputing 120 (2013) 469–481.
[28] A.W. Iorio, X. Li, Solving rotated multi-objective optimization problems using differential evolution, in: Proceedings of the Seventeenth Australian Joint Conference on Advances in Artificial Intelligence, in: AI'04, Springer-Verlag, Berlin, Heidelberg, 2004, pp. 861–872, doi:10.1007/978-3-540-30549-1_74.
[29] G. Wu, D. Qiu, Y. Yu, W. Pedrycz, M. Ma, H. Li, Superior solution guided particle swarm optimization combined with local search techniques, Expert Syst. Appl. 41 (16) (2014) 7536–7548, doi:10.1016/j.eswa.2014.06.005.
[30] A.E. Eiben, C.A. Schippers, On evolutionary exploration and exploitation, Fundam. Inf. 35 (1–4) (1998) 35–50.
[31] M. Črepinšek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: a survey, ACM Comput. Surv. 45 (3) (2013) 35:1–35:33, doi:10.1145/2480741.2480752.
[32] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm Evolut. Comput. 24 (2015) 11–24, doi:10.1016/j.swevo.2015.05.002.
[33] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evolut. Comput. 10 (6) (2006) 646–657, doi:10.1109/TEVC.2006.872133.
[34] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evolut. Comput. 15 (1) (2011) 55–66, doi:10.1109/TEVC.2010.2087271.
[35] M. Ali, M. Pant, A. Abraham, Improved differential evolution algorithm with decentralisation of population, Int. J. Bio Inspir. Comput. 3 (1) (2011) 17–30, doi:10.1504/IJBIC.2011.038701.
[36] Y.-l. Li, J. Zhang, A new differential evolution algorithm with dynamic population partition and local restart, in: Proceedings of the Thirteenth Annual Con-

ference on Genetic and Evolutionary Computation, in: GECCO '11, ACM, New York, NY, USA, 2011, pp. 1085–1092, doi:10.1145/2001576.2001723.
[37] P. Novoa-Hernández, C.C. Corona, D.A. Pelta, Self-adaptive, multipopulation differential evolution in dynamic environments, Soft Comput. 17 (10) (2013) 1861–1881.
[38] W.-j. Yu, J. Zhang, Multi-population differential evolution with adaptive parameter control for global optimization, in: Proceedings of the Thirteenth Annual Conference on Genetic and Evolutionary Computation, in: GECCO '11, ACM, New York, NY, USA, 2011, pp. 1093–1098, doi:10.1145/2001576.2001724.
[39] J. Zhang, X. Ding, A multi-swarm self-adaptive and cooperative particle swarm optimization, Eng. Appl. Artif. Intell. 24 (6) (2011) 958–967, doi:10.1016/j.engappai.2011.05.010.
[40] S.-Z. Zhao, P. Suganthan, Q.-K. Pan, M.F. Tasgetiren, Dynamic multi-swarm particle swarm optimizer with harmony search, Expert Syst. Appl. 38 (4) (2011) 3735–3742, doi:10.1016/j.eswa.2010.09.032.
[41] J. Zhang, A.C. Sanderson, Adaptive Differential Evolution: A Robust Approach to Multimodal Problem Optimization, Springer Publishing Company, Incorporated, 2009.
[42] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, Technical Report, Nanyang Technological University, Singapore, 2005.
[43] N.H. Awad, M.Z. Ali, J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective bound constrained real-parameter numerical optimization, Technical Report, Nanyang Technological University, Singapore, 2016.
[44] J.J. Moré, S.M. Wild, Benchmarking derivative-free optimization algorithms, SIAM J. Optim. 20 (1) (2009) 172–191.
[45] S. Das, P.N. Suganthan, Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems, Jadavpur University, Nanyang Technological University, Kolkata, 2010.
[46] S.M. Elsayed, R.A. Sarker, D.L. Essam, Ga with a new multi-parent crossover for solving IEEE-CEC2011 competition problems, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2011a, pp. 1034–1040.
[47] S.M. Elsayed, R.A. Sarker, D.L. Essam, Differential evolution with multiple strategies for solving CEC 2011 real-world numerical optimization problems, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2011b, pp. 1041–1048.
[48] J. Patel, V. Savsani, V. Patel, R. Patel, Layout optimization of a wind farm to maximize the power output using enhanced teaching learning based optimization technique, J. Clean. Prod. 158 (Supplement C) (2017) 81–94, doi:10.1016/j.jclepro.2017.04.132.

Lyuyang Tong was born in Hubei, China, in 1992. He received the B.E. degree in Computer Science and Technology and also the B.A. degree in Japanese Language from Changchun University of Technology, China, in 2015 and 2017, respectively. Now he is pursuing the M.S. degree in Software engineering from Guilin University of Technology, China. His current research interests include evolutionary computation, multi-objective evolutionary optimization and machine learning.

Minggang Dong received the B.S. degree from Naval University of Engineering, China, in 2000, the M.S. degree from Guangxi University, China, in 2006, and the Ph.D. degree from Zhejiang University, China, in 2012. He is currently a Professor in the College of Information Science and Engineering at Guilin University of Technology, China. His current research interests include machine learning, evolutionary computation, parallel and distributed computation.

Chao Jing received his Ph.D. degree in Computer Science from Shanghai Jiao Tong Univeristy, China 2014. From July 2014, he is an Assistant Professor with School of Information Science and Engineering at Guilin University of Technology. Since Sept 2017, he has been worked as a visiting scholar at the School of Engineering, Brown Univeristy. His research interests include Big Data & Cloud Computing, Energy-efficient Computing and Optimzation scheduling techniques.