



A self-adaptive multi-population differential evolution algorithm

Lin Zhu¹ · Yongjie Ma¹ · Yulong Bai¹

Published online: 26 August 2019
© Springer Nature B.V. 2019

Abstract

Differential evolution (DE) is an efficient population-based search algorithm for solving numerical optimization problems. However, the performance of DE is very sensitive to the choice of mutation strategies and their associated control parameters. In this paper, we propose a self-adaptive multi-population differential evolution algorithm, called SAMDE. The population is randomly divided into three equally sized sub-populations, each with different mutation strategies. At the end of each generation, all sub-populations are updated independently and recombined. Each sub-population uses an adaptive mechanism for selecting how current generation control parameters are generated. An improved mutation strategy, “rand assemble/1”, is proposed, its base vector is composed proportionally of three randomly selected individuals. The performance of SAMDE is evaluated on the suite of CEC 2005 benchmark functions. A comparative study is carried out with other state-of-the-art optimization techniques. The results show that SAMDE has a competitive performance compared to several other efficient DE variants.

Keywords Evolutionary algorithm · Differential evolution · Multi-population · Self-adaptive · Numerical optimization

1 Introduction

Differential evolution (DE), first proposed by Storn and Price (1997), is a simple and efficient evolutionary algorithm (EA) for solving numerical optimization problems. DE is a population-based stochastic search technique, in which mutation, crossover, and selection operators are utilized at each generation to move the population toward the global optimum (Wang et al. 2011). In the last few years, DE has been extended for handling multiobjective, constrained, large scale, dynamic and uncertain optimization problems (Das and Suganthan 2011; Mukherjee et al. 2014) and is now successfully used in various scientific and engineering fields (Plagianakos et al. 2008; Ghasemi et al. 2014; Wang and Cai 2015; Zhao et al. 2014), such as chemical engineering, engineering design, and pattern recognition (Cai and Wang 2015).

When DE is applied to a given optimization problem, there are three crucial associated parameters which

significantly affect the performance of DE (i.e., population size NP , scaling factor F , and crossover rate CR). Moreover, there are many trial vector generation strategies, and different strategies have different search capabilities for different problems in different stages of the evolutionary process. An inappropriate setting of these for a particular problem may lead to premature convergence and degrade algorithmic performance. Therefore, in order to apply DE successfully to solve optimization numerical problems, a trial and error search for the strategies and the associated control values is usually required.

Consequently, many enhanced DE variants such as jDE (with self-adapted parameters) (Brest et al. 2006), SaDE (with adapted mutation strategies and parameters) (Qin et al. 2009), JADE (with “current-to-pbest/1” mutation strategy and adaptive parameters) (Zhang and Sanderson 2009), CoDE (with composition of multiple strategies and parameter settings) (Wang et al. 2011), EPSDE (with ensemble of mutation strategies and parameters) (Malipeddi et al. 2011), SspDE (with self-adapted strategies and control parameters) (Pan et al. 2011), ESADE (with “current-to-pbest/1” mutation strategy and self-adaptive parameters) (Guo et al. 2014), MPEDE (with ensemble of multiple mutation strategies and self-adaptive parameters) (Wu et al. 2016), and DMPSADE (with self-adaptive

✉ Yongjie Ma
myjmyj@nwnu.edu.cn

¹ College of Physics and Electronic Engineering, Northwest Normal University, Lanzhou, Gansu, China

mutation strategies and parameters) (Fan and Yan 2015), have been proposed.

Although DE's effectiveness and competitive performance has been demonstrated by many experimental studies and theoretical analysis, DE still quite depends on the settings of control parameters such as scaling factor (F), crossover rate (CR), mutation/crossover strategy and population size (NP), so many variants of DE have proposed, as mentioned above, they all used a similar idea of using different mutation strategies and self-adaptive techniques to control parameter settings (Awad et al. 2016). The proposed variants of DE are classified into 3 types: DE methods with both strategy and control parameter adaptations, DE with only control parameter (F and Cr) adaptation, and DE with population size control (Das et al. 2016). As a very competitive method, the well-known EPSDE (Mallipeddi et al. 2011) is select strategy as well as values of F and Cr from the strategy pool, which is associated with the success rate of the generated trial. EPSDE is developed by a proposed variant of EPSDE where the ensemble of F and Cr values are evolved by using the optimization process of another metaheuristic algorithm called Harmony Search (HS) (Mallipeddi 2013). However, EPSDE and its variants can be computationally costlier, especially on large scale problem instances (Das et al. 2016), and the success rate of the generated trial doesn't all depends on mutation strategies.

It would be a useful attempt that DE consists of multiple populations and each sub-population is evolved using different mutation/crossover strategies and adaptive parameters (Yongjie et al. 2009), individuals may migrate among sub-populations according to certain rules.

In this paper, we propose a self-adaptive multi-population differential evolution, called SAMDE, which aims at improving the search precision and solving numerical optimization problems. The algorithm first randomly divides the population into three equally sized sub-populations. The algorithm then generates trial vectors for each sub-population using three different strategies. A modified mutation strategy uses a linear combination of randomly selected several individuals from the sub-populations. Moreover, two classical mutation strategies ("rand/1" and "best/2") are used to other two sub-populations, respectively. In addition, parameters such as scaling factor F and crossover rate CR , associated with each mutation strategy are adapted based on iterations. At the end of each generation, each sub-population is updated by random recombination. SAMDE is tested on the suite of CEC 2005 benchmark functions with 10, 30 and 50 variables, respectively. The competitive performance of SAMDE is

exhibited by extensive comparisons with several state-of-the-art DE variants.

Recently, population partitioning techniques for enhancing the performance of EAs and swarms, such as particle swarm optimization (PSO) and DE, attracted increasing attention (Wu et al. 2016). Our work is different from previous studies. In all previous works, each sub-population evolved separately for a certain number of generations and then some individuals were randomly selected migrate between sub-populations to share experiences. By contrast, each sub-population in this paper updates at the end of each generation. The application of multi-population technique in our study is aimed to maintain population diversities while enhance algorithmic performance.

The rest of this paper is structured as follows: Sect. 2 gives a brief introduction to classical DE, including its typical mutation operators, crossover, and selection operators. Section 3 reviews the related works in literature. Section 4 introduces details of the implementation of SAMDE. Experimental results and analysis are presented in Sect. 5. Section 6 concludes this paper.

2 Differential evolution

The DE firstly generates a random initial population within the scope of solution, then uses differential mutation, crossover, and selection operation to produce a new generation of population. The DE algorithm, which selects real number coding, generates random initialization population in the feasible solution space. The initial value of the j th decision variable of the i th individual at generation $G = 0$ is generated within the search space constrained by the prescribed minimum and maximum decision variable's bounds $X_{\min} = \{x_{\min}^1, \dots, x_{\min}^D\}$ and $X_{\max} = \{x_{\max}^1, \dots, x_{\max}^D\}$ by:

$$x_{i,0}^j = x_{\min}^j + \text{rand}(0, 1) \cdot (x_{\max}^j - x_{\min}^j), j = 1, 2, \dots, D \quad (1)$$

where D is the dimensions of the problem, $\text{rand}(0,1)$ represents a uniformly distributed random variable within the range $[0,1]$.

Constrained numerical optimization problems (CNOPs) is defined as:

$$\text{Min}f(X), \quad X \in [X_{\min}, X_{\max}]$$

Because of the highly competitive performance showed by DE when solving CNOPs, the researchers have been

focused on providing modifications to DE variants, such as self-adaptive parameter control in constrained search spaces, etc.

2.1 Mutation operation

In DE, the variation vector is composed of the difference vector of individual in the population after scaling and other different individuals within the population. A variety of mutation strategies are obtained according to different methods of generating the variation vector. The most commonly used mutation strategies are the followings (Mallipeddi and Lee 2015):

$$\text{“DE/rand/1”} : V_{i,G} = X_{r_1^i,G} + F \cdot (X_{r_2^i,G} - X_{r_3^i,G}) \quad (2)$$

$$\text{“DE/rand/2”} : V_{i,G} = X_{r_1^i,G} + F \cdot (X_{r_2^i,G} - X_{r_3^i,G}) + F \cdot (X_{r_4^i,G} - X_{r_5^i,G}) \quad (3)$$

$$\text{“DE/best/1”} : V_{i,G} = X_{best,G} + F \cdot (X_{r_1^i,G} - X_{r_2^i,G}) \quad (4)$$

$$\text{“DE/best/2”} : V_{i,G} = X_{best,G} + F \cdot (X_{r_1^i,G} - X_{r_2^i,G}) + F \cdot (X_{r_3^i,G} - X_{r_4^i,G}) \quad (5)$$

$$\text{“DE/rand-to-best/1”} : V_{i,G} = X_{i,G} + K \cdot (X_{best,G} - X_{i,G}) + F \cdot (X_{r_1^i,G} - X_{r_2^i,G}) \quad (6)$$

$$\text{“DE/rand-to-best/2”} : V_{i,G} = X_{i,G} + K \cdot (X_{best,G} - X_{i,G}) + F \cdot (X_{r_1^i,G} - X_{r_2^i,G} + X_{r_3^i,G} - X_{r_4^i,G}) \quad (7)$$

$$\text{“DE/current - to - rand/1”} : V_{i,G} = X_{i,G} + K \cdot (X_{r_1^i,G} - X_{i,G}) + F \cdot (X_{r_2^i,G} - X_{r_3^i,G}) \quad (8)$$

The parameters $r_1^i, r_2^i, r_3^i, r_4^i, r_5^i$ are randomly generated independent integers different from the index i and within the range of $[1, NP]$. These indexes are randomly generated in each vector. $X_{best,G}$ is the best individual which has the best fitness value in the G th generation populations. K is randomly selected in $[0, 1]$. F is the scale factor.

2.2 Crossover operation

Crossover operator aims to generate trial vectors. The DE algorithm generally employs two kinds of crossover methods: binomial crossover (bin) and exponential crossover (exp).

2.2.1 Binomial crossover

In binomial crossover, at least one component of trial vector is provided by the variation vector with the method of random selection. Operation equation is as follows:

$$u_{i,j,G} = \begin{cases} v_{i,j,G} & \text{rand}(0,1) \leq CR \text{ or } j = j_{rand} \\ x_{i,j,G} & \text{otherwise} \end{cases} \quad (9)$$

where $j = 1, 2, \dots, D$, j_{rand} is a randomly chosen integer in the range $[1, D]$, $CR \in (0, 1)$ is the crossover rate.

2.2.2 Exponential crossover

Exponential crossover operation mode is as follows:

$$u_{i,j,G} = \begin{cases} v_{i,j,G} & j = \langle l \rangle_D, \langle l+1 \rangle_D, \dots, \langle l+L-1 \rangle_D \\ x_{i,j,G} & \text{otherwise} \end{cases} \quad (10)$$

In the exponential crossover, an integer $l \in [1, D]$, which acts as a starting point in the target vector, is chosen randomly from the point at which the crossover or exchange of components with the mutant vector starts. $L \in [1, D]$ denotes the number of components that are contributed by the mutant vector to the target vector. Integer L is drawn from $[1, D]$ depending on the crossover probability (CR). Where the angular brackets $\langle l \rangle_D$ denote a modulo function with modulus D .

2.3 Selection operation

DE uses a “greed” selection strategy to choose the best individual according to the fitness value of target vector and trial vector. The selection operation can be expressed as follows:

$$X_{i,G+1} = \begin{cases} U_{i,G} & f(U_{i,G}) < f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \quad (11)$$

where $X_{i,G+1}$ is target vector of the next generation.

3 Related works

The DE algorithm has become a powerful optimizer for solving global continuous optimization problems, which is simple, high-efficiency, easy to understand and implement, and has less controlled parameters. Meanwhile, it can proceed heuristic search in the continuous space randomly, directly and concurrently (Storn and Price 1997; Qin et al. 2009). However, the performance of the conventional DE algorithm depends on the chosen mutation/crossover strategies and the associated control parameters. In addition, the performance of DE becomes more sensitive to strategies and their associated parameter values as the complexity of the problem increases (Gämperle et al. 2002). In other words, the inappropriate selection of strategies and parameters may lead to a premature convergence, stagnation, or a waste of computational resources (Wu et al. 2016; Gämperle et al. 2002; Lampinen and Zelinka 2000; Price et al. 2005; Zaharie 2003).

In order to improve the performance of DE algorithm, in recent years, some researchers have proposed many improved measures of mutation strategy. In Zhang and Sanderson (2009), a new mutation strategy, named DE/current-to-pbest with optional archive, is proposed to serve as the basis of the adaptive DE algorithm JADE. Li et al. (2012) proposed a new mutation strategy, with the best individual as a guide but not entirely dependent on the best individual, and with a certain probability to the optimal direction of evolution. In Bi and Liu (2012), the base vector selected by the mutation strategy make a compromise between the random individual and the best individual. In Ouyang et al. (2013), a random mutation strategy was proposed. The method of random choice was adopted to implement mutation and disturbance operation, which was aimed at increasing the diversity of population and balancing the local and global search. In Kong et al. (2014), a multi-strategy mutation operator based on symbolic function was designed. In Kong et al. (2014), the global acceleration operator was proposed, which can balance the global search and local search. In Bi et al. (2012), the best global solution and the best previous solution of each individual are utilized in the new mutation strategy to guide the search direction by introducing more effective directional information. This method avoids the search blindness brought by the random selection of individuals in

the difference vector. In Qiu et al. (2015), the mutation strategy of DE is proposed and divided into two parts to reflect the changes of the target population trends and their random variations. It consists of an additional mutation factor which is simulated by a different Hurst index fractal Brownian motion. In Ali et al. (2015), a modified mutation strategy was proposed, which called mms, uses a convex linear combination of randomly selected individuals from the population. This novel mutation strategy was used to produce quality solutions to balance exploration and exploitation. In Mukherjee et al. (2016), the mutation phase has been entrusted to a locality-induced operation that retains traits of Euclidean distance-based closest individuals around a potential solution. A locality based DE mutation scheme called 'DE/current-to-p-local_best/1' has been devised.

Extensive studies have been done on appropriate setting of the control parameters of DE, such as scaling factor F and crossover rate CR . Initially, Storn and Price (1997) said that $F = 0.5$ would be a good initial choice, and its adjustable range is from 0.4 to 1. On this basis, Liu and Lampinen (2010) mentioned that $F = 0.9$ would be a good initial choice, and $F \in (0.4, 0.95)$. Ronkkonen et al. (Ronkkonen et al. 2005) set $CR \in (0, 0.2)$ in the separable functions. But if parameters are dependent on each other, $CR \in (0.9, 1)$. Zielinski et al. (2006) pointed out that $F \geq 0.6$ and $CR \geq 0.6$ can make the algorithm get better performance. From the above, it can be observed that parameters maintain a fixed value in the process of evolution, which lack sufficient theoretical justification. Therefore, the researchers began to consider the automatic adjustment of DE parameters. Brest et al. (2006) proposed a self-adaptation scheme (jDE), in which control parameters F and CR were encoded into the individuals and are adjusted in the run of DE. Qin et al. (2009) considered allowing F to take different random values in the range $(0, 2]$ with normal distribution of mean 0.5 and standard deviation 0.3 for different individuals in the current population, and accumulating the previous learning experience within a certain generation interval so as to dynamically adapt the value of CR to a suitable range. In Zhang and Sanderson (2009), the parameter adaptation automatically updates the control parameters to appropriate values. In Zou et al. (2013), MDE adjusts scale factor F and crossover rate CR by using Gauss distribution and uniform distribution, respectively.

From the analysis of population structure of DE, the initial population is divided into multiple sub-populations that according to the topology relationship and migrate by the corresponding individual migration mechanism. This form of DE is called distributed DE. In recent years, the distributed DE algorithm has become an important branch of DE. Therefore, the researchers began to pay close attention to the distributed DE. Some researches on the present mainly partition the initial population or swarm into multiple equal smaller sub-populations. As the algorithm proceeds, information exchange among sub-populations and regrouping operators will be triggered with a certain frequency with the aim to maintain the diversity of the whole population and balance the exploitation and exploration capabilities (Wu et al. 2016). Weber et al. (2011) proposed shuffle or update parallel differential evolution (SOUPE), which is a structured population algorithm characterized by sub-populations employing a differential evolution logic. Two simple mechanisms have been integrated. The first, namely shuffling, consists of randomly rearranging the individuals over the sub-populations. The second consists of updating all the scale factors of the sub-populations. In Ali et al. (2015), the population is divided into four independent sub-groups, each with different mutation and update strategies. The size of each group is fixed and computed by dividing the population size by number of predefined groups. Each sub-group is filled randomly by picking random individuals from the entire population. In Wu et al. (2016), there are three equally sized smaller indicator sub-populations and one much larger reward sub-population. Each constituent mutation strategy has one indicator sub-population. After every certain number of generations, the current best performing mutation strategy will be determined according to the ratios between fitness improvements and consumed function evaluations. Then the reward sub-population will be allocated to the determined best performing mutation strategy dynamically. Shang et al. (2014) proposed a multi-population based cooperative coevolutionary algorithm (MPCCA) to solve the multi-objective capacitated arc routing problem. In MPCCA, population is partitioned into multiple sub-populations with respect to their different direction vectors. These sub-populations evolve separately and search different objective sub-regions simultaneously. The adjacent sub-populations are able to share their information. The differences between our research and other related studies are explained in Sect. 1.

4 Self-adaptive multi-population DE (SAMDE)

The sub-populations quickly exploit some areas of the decision space, thus drastically and quickly reducing the fitness value in the highly multi-variate fitness landscape (Weber et al. 2011). Moreover, the suitable mutation strategies choosing and the reasonable control parameter setting are important to enhance the search ability of DE algorithm. Therefore, in this study, the population is divided into three equally sized sub-populations, each with different mutation strategies. At the end of each generation, all sub-populations are updated by random recombination. Meanwhile, the SAMDE algorithm is proposed to implement the self-adaptive control parameters to ensure the search ability of algorithm.

4.1 Multiple populations

The mechanism of multi-population ensures that each sub-population is not affected by the interference of other sub-populations in the process of evolution. In this work, the formation process of the sub-populations is first initializes the entire population and then calculates fitness value of all individuals, at last, the population is randomly divided into three same size subgroups, called X1, X2, X3. Three sub-populations use different mutation strategies and evolve respectively and concurrently within each sub-population. At the end of each iteration, all individuals are randomly reordered. It is worth mentioning that each sub-population is updated at each generation, which actually share optimization experience with each other and realize the information exchange among sub-populations.

4.2 Parameter adaptation

The choice of numerical values for control parameters F , CR significantly affect the performance of the DE algorithm. Instead of fixing the values of these parameters, the current trend is to use self-adaptive parameter setting mechanisms. In this study, three sub-populations use three sets of scale factor F_k and crossover probabilities CR_k , $k = 1, 2, 3$. In SAMDE, a parameter adaptive approach is proposed that control parameters of each generation can be gradually self-adapted according to the number of iterations. This adaptive scheme can keep both local and global

search ability to generate the potential good mutant vector throughout the evolution process.

At each generation t , mutation factor Fk_i^t of each individual xi is independently generated according to a normal distribution of mean μFk^t and standard deviation 0.1 and is bounded in the interval (0, 1) as:

$$Fk_i^t = randn_i(\mu Fk^t, 0.1) \quad (12)$$

In generation t , the location parameter μFk^t of the normal distribution is updated at the end of each generation as:

$$\mu Fk^t = o\mu Fk - 0.1 \times e^{\left(\frac{-t}{Gm}\right)} \quad (13)$$

Similarly, at each generation t , the crossover probability CRk_i^t of each individual xi is independently generated according to a normal distribution of mean μCRk^t and standard deviation 0.1 and is bounded in the interval (0, 1) as:

$$CRk_i^t = randn_i(\mu CRk^t, 0.1) \quad (14)$$

In generation t , the location parameter μCRk^t of the normal distribution is updated at the end of each generation as:

$$\mu CRk^t = o\mu CRk - 0.1 \times e^{\left(\frac{-t}{Gm}\right)} \quad (15)$$

where Gm is maximum number of iterations. $o\mu Fk$ is the initial value of μFk , and $o\mu CRk$ is the initial value of μCRk . Equations (13) and (15) will complete coarse search at beginning of iterations and fine search at ending of iterations.

4.3 Choice of mutation strategies for populations in SAMDE

The choices of different mutation strategies affect the efficiency of DE and accuracy of solution. Therefore, the reasonable selection of mutation strategies is especially important. Several mutation strategies from the DE literature were presented in Sect. 2.1. Two of these strategies are

directly used in some sub-populations in SAMDE while one is a modified mutation strategy.

Two standard mutation strategies are used in X1 and X3, that are “rand/1” and “best/2”. “DE/rand/1/bin” is one of the most commonly used strategies in the DE research. It can effectively maintain the diversity of population. For this reason, “DE/rand/1/bin” mutation strategy is used in X1. In contrast, the “DE/best/2/bin” mutation strategy have a faster convergence rate and better searching ability. It can availably ensure optimization ability of the sub-population. Thus, we use “DE/best/2/bin” mutation strategy in X3.

In Ali et al. (2015), a new mutation strategy is proposed. Based on this mutation strategy, we propose a new mutation strategy, call it as “rand assemble/1”, use in X2. In “rand assemble/1”, the choice of the base vector is different from standard DE. It uses a linear combination of randomly selected three individuals from the population, which will carry more paternal genes and pass to their offspring. This modified mutation strategy will produce well distributed solutions with higher convergence rate and increase the diversity of population due to the generational interchange of sub-populations using different mutation strategies.

Mutation strategy 1: “rand/1”

$$V_{i,t} = X_{r_1^i,t} + F \cdot (X_{r_2^i,t} - X_{r_3^i,t}) \quad (16)$$

Mutation strategy 2: “rand assemble/1”

$$V_{i,t} = (aX_{r_1^i,t} + bX_{r_2^i,t} + cX_{r_3^i,t}) + F \cdot (X_{r_4^i,t} - X_{r_5^i,t}) \quad (17)$$

Mutation strategy 3: “best/2”

$$V_{i,t} = X_{best,t} + F \cdot (X_{r_1^i,t} - X_{r_2^i,t}) + F \cdot (X_{r_3^i,t} - X_{r_4^i,t}) \quad (18)$$

where r_1, r_2, r_3, r_4, r_5 are all random integers within the interval $[1, NP]$, a, b and c are random numbers selected from the interval (0, 1). Where $a + b + c = 1$.

According to three mutation strategies and parameter adaptation introduced above, we come to the framework of SAMDE as given in Algorithm 1.

Algorithm 1: Pseudo code of SAMDE

```

Initialize  $D, MAX\_FES, NP$ ;
Set  $o\mu F1, o\mu F2, o\mu F3, o\mu CR1, o\mu CR2, o\mu CR3$ ;
Set  $FES = 0$ ;
Initialize the population randomly distributed in the solution space;
For  $i=1$  to  $NP$ 
    Calculate  $f(x_i)$  for each individual  $X_i$  in population;
     $FES = FES+1$ ;
End For
Randomly partition population into  $X1, X2, X3$  with the same size;
While  $FES < MAX\_FES$ 
    For  $k=1$  to 3
        Calculate  $\mu Fk(t)$  and  $\mu CRk(t)$ ;
        For each  $x_i$  in  $Xk$  do
            Calculate  $Fk_i(t)$  and  $CRk_i(t)$  for each individual  $x_i$  in  $Xk$ ;
            If  $k = 1$  Then
                 $v_i(t) = x_{r1}(t) + Fk_i(t) \times (x_{r2}(t) - x_{r3}(t))$ ;
            Elseif  $k = 2$  Then
                 $a, b, c = \text{rand}(0,1)$  and  $a+b+c = 1$ ;
                 $v_i(t) = a \times x_{r1}(t) + b \times x_{r2}(t) + c \times x_{r3}(t) + Fk_i(t) \times (x_{r4}(t) - x_{r5}(t))$ ;
            Else
                 $v_i(t) = x_{best}(t) + Fk_i(t) \times (x_{r1}(t) - x_{r2}(t))$ ;
            End If
             $j_{rand} = [\text{rand}[0,1], D]$ ;
            For  $j = 1$  to  $D$ 
                If  $\text{rand} < CRk_i(t)$  or  $j = j_{rand}$  Then
                     $u_i(t) = v_i(t)$ ;
                Else
                     $u_i(t) = x_i(t)$ ;
                End If
            End For
            Calculate  $f(u_i)$  for each mutation individual  $u_i$  in  $Xk$ ;
             $FES = FES+NP/3$ ;
            If  $f(x_i(t)) \leq f(u_i(t))$ 
                 $x_i(t) = u_i(t)$ ;
            Else
                 $x_i(t) = u_i(t)$ ;
            End If
        End For
    End For
    Randomly repartition population into  $X1, X2, X3$  with the same size;
     $t = t+1$ ;
End While

```

5 Experimental study

5.1 Experimental settings

The proposed SAMDE algorithm was implemented and tested on a set of 25 benchmark functions which proposed in CEC 2005 (Suganthan et al. 2005). The test suite consists of 25 benchmark functions which includes unimodal functions $F1$ – $F5$, basic multimodal functions $F6$ – $F12$, expanded multimodal functions $F13$ – $F14$, and hybrid composition functions $F15$ – $F25$. The maximum number of function evaluation was set to $2000 \times D$. The population size for this algorithm was set as $NP = 60$. The initial value of location parameters were set as $o\mu F1 = 1$, $o\mu F2 = 0.75$, $o\mu F3 = 0.5$, $o\mu CR1 = 1.1$, $o\mu CR2 = 1$ and $o\mu CR3 = 0.9$. The crossover operator was set as binomial crossover. Both F and CR are taken in the range $(0, 1)$.

The SAMDE and other DE variants were coded in Matlab environment. The computations were carried out using a PC with Intel(R) Core(TM) i3 - 2350 M @ 2.3 GHz CPU and 2 GB RAM while running Matlab R2012a on 64-bit windows operating system.

5.2 Comparison with state-of-the-art evolutionary algorithms

This section mainly presents 25 real-valued benchmark functions which are used to evaluate the proposed SAMDE algorithm against other DE variants. The computational results obtained by running each of the eight comparative DE variants 25 times on each benchmark function with 10, 30 and 50 variables are reported in Tables 1, 2 and 3, respectively. These tables show statistical results of the mean error and standard deviation values obtained for all these functions. The symbols of “–”, “+”, and “=” in last three rows of each table respectively denote that the performance of the corresponding algorithm is worse than, better than and similar to that of SAMDE. α is level of significance, which is used to determine at which level the null hypothesis H_0 may be rejected, and wins = worse (–) + [similar(=)/2].

SAMDE was compared with eight other state-of-the-art DE variants including JADE (Zhang and Sanderson 2009), SaDE (Qin et al. 2009), EPSDE (Mallipeddi et al. 2011), CoDE (Wang et al. 2011), SOUPDE (Weber et al. 2011), SspDE (Pan et al. 2011), ESADE (Guo et al. 2014) and MPEDE (Wu et al. 2016). The reasons we choose these eight DE variants as comparative algorithms are explained as follows. First, JADE, SspDE and ESADE are three DE variants which parameters are set in an adaptive manner. Second, SaDE, EPSDE and CoDE also incorporate multiple mutation strategies as SAMDE. Third, SOUPDE and MPEDE are two DE variants which use evolution of

multiple populations as SAMDE, and MPEDE is a recently proposed DE variant that can reflect the latest progress of DE. Hence, it is meaningful to compare SAMDE with them. The settings of each algorithm are as follows:

1. JADE with $NP = 100$.
2. SADE with $NP = 50$, $LP=30$.
3. EPSDE with $NP = 50$.
4. CoDE with $NP = 30$.
5. SOUPDE with $NP = 60$, $ps = 0.5$, $pu = 0.5$, $CR = 0.9$.
6. SspDE with $NP = 100$, $BR = 0.8$.
7. ESADE with $NP = 50$, $t0 = 1000$.
8. MPEDE with $NP = 250$, $ng = 20$, $\lambda_1 = \lambda_2 = \lambda_3 = 0.2$.

Let us detail the eight algorithms according to the time order that they were published. All these algorithms have been summarized in Sect. 1.

As indicated in Table 1, SAMDE shows a statistically significant performance when compared with all the contestant algorithms in most of the functions. It is found that SAMDE has the best performance among the other eight classical state-of-the-art DE variants on the 13 test functions with 10D. Regarding the unimodal functions $F1$ – $F5$, JADE, EPSDE, ESADE and SAMDE show the better performance. SAMDE obtains significantly best results on the 4 test functions ($F1$, $F2$, $F4$ and $F5$) than all other peers. Only in function $F3$, JADE performs better than SAMDE. Whereas, compared with the other seven DE variants, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE, SAMDE exhibits better overall performance. For basic multimodal benchmark functions $F6$ – $F12$, SaDE, EPSDE, CoDE and SAMDE show the better performance. SAMDE has significantly best results on the 3 test functions ($F6$, $F8$ and $F12$) than all the other DE variants. Moreover, SAMDE is better than JADE, SspDE and MPEDE on all simple multimodal functions. SAMDE outperforms SaDE, EPSDE, CoDE, SOUPDE and ESADE on three ($F6$, $F7$ and $F12$), three ($F6$, $F7$ and $F12$), four ($F6$, $F7$, $F10$ and $F12$), five ($F6$ – $F8$, $F10$ and $F12$) and five ($F6$, $F8$ – $F10$ and $F12$) benchmark functions, respectively. As for expanded multimodal functions, SAMDE generally performs worse than other DE variants (except MPEDE) on function $F13$ while it outperforms all the other DE variants on function $F14$. With regard to the more complex hybrid composition functions $F15$ – $F25$, CoDE, SOUPDE, SspDE and SAMDE show the better performance. SAMDE performs significantly best results on the 5 test functions ($F16$, $F17$, $F19$, $F21$ and $F25$) than all other peer DE variants. It outperforms SaDE, EPSDE, CoDE, SOUPDE, SspDE and MPEDE on nine ($F16$ – $F19$ and $F21$ – $F25$), nine ($F16$, $F17$, $F19$ and $F21$ – $F25$), five ($F16$, $F17$, $F19$, $F21$ and $F25$), seven ($F15$ – $F17$, $F19$, $F21$, $F22$ and $F25$), six ($F15$ – $F17$, $F19$, $F21$ and $F25$) and ten ($F15$ – $F23$ and $F25$) benchmark functions, respectively. Moreover, SAMDE is better than

Table 1 Comparison of the average error for 25 independent runs by different optimization techniques with $D = 10$ on functions F_1 – F_{25}

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F1									
Mean	8.00E−10	3.45E−18	1.86E−17	7.43E−05	2.44E−06	1.40E−06	8.96E−15	1.24E+00	1.56E−22
Std	3.67E−10	1.73E−18	9.26E−18	3.04E−05	5.66E−06	7.48E−07	3.59E−15	2.71E−02	7.31E−23
W	–	–	–	–	–	–	–	–	–
F2									
Mean	4.86E−04	3.77E−03	1.23E−03	1.16E+00	3.39E+00	3.06E−01	2.94E−02	3.95E+01	3.37E−09
Std	2.54E−04	2.05E−03	6.65E−04	4.07E−01	2.16E+00	2.29E−01	2.57E−02	1.14E+00	1.82E−09
W	–	–	–	–	–	–	–	–	–
F3									
Mean	7.00E+01	8.64E+05	6.84E+04	1.42E+04	3.51E+06	5.35E+04	2.74E+03	2.38E+05	6.23E+02
Std	3.65E+01	5.29E+05	3.24E+04	4.35E+03	2.49E+06	4.94E+04	5.94E+01	1.03E+04	1.33E+02
W	+	–	–	–	–	–	–	–	–
F4									
Mean	7.97E−03	6.97E−02	1.20E−02	6.26E+00	1.24E+01	8.49E−01	5.82E−02	9.77E+01	7.77E−08
Std	4.15E−03	3.74E−02	6.05E−03	2.22E+00	8.69E+00	6.42E−01	4.74E−02	1.94E+00	3.71E−08
W	–	–	–	–	–	–	–	–	–
F5									
Mean	8.28E−01	3.48E−02	3.06E−03	2.68E+01	1.21E+02	1.43E+01	8.71E−04	1.95E+02	1.46E−04
Std	1.75E−01	6.54E−03	7.80E−04	5.04E+00	3.79E+01	3.08E+00	1.86E−04	2.28E+00	2.43E−05
W	–	–	–	–	–	–	–	–	–
F6									
Mean	2.06E+01	4.86E+00	2.15E+00	7.99E+01	1.73E+02	8.47E+00	7.82E+00	2.83E+03	1.21E+00
Std	6.38E+01	1.74E−01	2.31E−01	3.85E+01	1.22E+02	2.98E+00	1.57E+01	9.62E+01	6.58E−02
W	–	–	–	–	–	–	–	–	–
F7									
Mean	8.86E−01	7.53E−01	7.98E−01	9.20E−01	9.75E−01	8.78E−01	1.92E−01	1.33E+00	6.30E−01
Std	1.07E−01	1.49E−01	1.28E−01	1.06E−01	1.03E−01	1.12E−01	5.92E−03	7.26E−03	1.29E−01
W	–	–	–	–	–	–	+	–	–
F8									
Mean	2.11E+01	2.08E+01	2.08E+01	2.08E+01	2.09E+01	2.09E+01	2.16E+01	2.10E+01	2.08E+01
Std	1.61E−01	1.25E−01	1.31E−01	1.18E−01	1.32E−01	1.39E−01	2.09E−01	6.23E−03	1.31E−01
W	–	=	=	=	–	–	–	–	–
F9									
Mean	6.67E+00	9.44E+00	9.98E+00	4.84E−01	4.58E+00	3.28E+01	6.12E+01	4.46E+01	2.32E+01
Std	2.05E+00	2.71E+00	3.10E+00	2.06E−01	2.66E+00	1.29E+01	1.44E+01	6.16E−01	6.71E+00
W	+	+	+	+	+	–	–	–	–
F10									
Mean	7.11E+01	4.85E+01	5.05E+01	5.56E+01	5.67E+01	5.82E+01	1.11E+02	7.13E+01	5.15E+01
Std	1.17E+01	9.24E+00	9.62E+00	9.92E+00	1.13E+01	9.89E+00	2.11E+01	4.75E−01	8.19E+00
W	–	=	+	–	–	–	–	–	–
F11									
Mean	1.40E+01	1.12E+01	1.12E+01	1.03E+01	1.12E+01	1.19E+01	6.63E−01	1.35E+01	1.14E+01
Std	1.26E+00	1.22E+00	1.20E+00	1.16E+00	1.26E+00	1.22E+00	1.17E−01	6.86E−02	1.29E+00
W	–	=	=	+	+	–	+	–	–
F12									
Mean	2.09E+03	5.97E+02	3.25E+02	3.35E+03	3.04E+03	2.74E+02	3.54E+02	9.37E+03	1.69E+02
Std	2.80E+03	5.25E+02	1.18E+02	1.45E+03	1.51E+03	4.35E+02	1.26E+03	2.25E+02	1.20E−03
W	–	–	–	–	–	–	–	–	–

Table 1 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F13									
Mean	3.77E+00	3.09E+00	2.86E+00	2.75E+00	2.53E+00	3.91E+00	1.14E+01	5.38E+00	4.11E+00
Std	8.69E-01	6.98E-01	7.41E-01	7.37E-01	7.38E-01	1.03E+00	4.37E+00	5.53E-02	8.03E-01
W	+	+	+	+	+	+	+	-	
F14									
Mean	4.58E+00	4.22E+00	4.22E+00	4.19E+00	4.29E+00	4.33E+00	4.92E+00	4.50E+00	4.18E+00
Std	1.63E-01	1.84E-01	1.98E-01	1.93E-01	1.99E-01	1.90E-01	1.31E-01	8.62E-03	1.86E-01
W	-	-	-	-	-	-	-	-	
F15									
Mean	3.94E+02	2.20E+02	2.20E+02	1.54E+02	2.97E+02	4.34E+02	3.18E+02	5.86E+02	2.95E+02
Std	1.48E+02	8.52E+01	5.93E+01	3.95E+01	1.33E+02	1.26E+02	2.81E+01	2.50E+00	2.63E+01
W	-	+	+	+	-	-	-	-	
F16									
Mean	2.60E+02	2.09E+02	2.09E+02	2.19E+02	2.34E+02	2.25E+02	3.03E+02	2.60E+02	2.04E+02
Std	3.94E+01	2.79E+01	2.27E+01	2.48E+01	3.15E+01	2.54E+01	4.42E+01	1.64E+00	2.24E+01
W	-	-	-	-	-	-	-	-	
F17									
Mean	2.99E+02	2.33E+02	2.39E+02	2.59E+02	3.01E+02	2.53E+02	3.68E+02	2.96E+02	2.32E+02
Std	4.19E+01	2.93E+01	2.60E+01	3.24E+01	6.34E+01	2.71E+01	5.94E+01	1.72E+00	2.42E+01
W	-	-	-	-	-	-	-	-	
F18									
Mean	7.75E+02	7.96E+02	6.35E+02	5.40E+02	6.59E+02	6.78E+02	8.32E+02	8.20E+02	6.85E+02
Std	5.78E+01	2.23E+01	2.93E+00	2.55E+01	4.36E+01	3.10E+01	1.19E-01	1.81E+00	6.83E-04
W	-	-	+	+	+	+	-	-	
F19									
Mean	7.53E+02	7.15E+02	7.78E+02	6.69E+02	6.67E+02	7.01E+02	8.07E+02	8.14E+02	6.65E+02
Std	5.50E+01	2.12E+01	5.79E+00	2.37E+01	5.73E+01	2.12E+01	1.04E-01	1.56E+00	4.81E-04
W	-	-	-	-	-	-	-	-	
F20									
Mean	7.84E+02	6.51E+02	6.96E+02	5.86E+02	6.70E+02	7.22E+02	8.23E+02	8.29E+02	7.47E+02
Std	2.65E+01	2.66E+01	2.03E+00	3.07E+01	4.15E+01	6.16E+00	1.24E-01	3.11E+00	2.90E-03
W	-	+	+	+	+	+	-	-	
F21									
Mean	5.01E+02	7.22E+02	4.85E+02	5.26E+02	5.76E+02	5.02E+02	6.90E+02	8.17E+02	4.83E+02
Std	1.25E+02	2.51E+01	7.63E-08	2.25E+00	5.81E+00	5.66E+00	1.58E-03	1.27E+00	1.37E-06
W	-	-	-	-	-	-	-	-	
F22									
Mean	8.03E+02	7.91E+02	7.73E+02	7.33E+02	7.93E+02	7.38E+02	8.05E+02	8.12E+02	7.70E+02
Std	1.07E+01	6.54E+00	5.82E+00	1.31E+01	1.94E+01	1.13E+01	1.20E+01	4.31E-01	5.39E+00
W	-	-	-	+	-	+	-	-	
F23									
Mean	8.06E+02	7.73E+02	7.54E+02	6.20E+02	6.52E+02	6.56E+02	9.58E+02	8.43E+02	7.18E+02
Std	7.85E+01	2.93E+01	4.19E+00	6.03E+00	1.42E+01	1.95E+01	3.45E+01	2.21E+00	6.65E-08
W	-	-	-	+	+	+	-	-	
F24									
Mean	2.15E+02	2.13E+02	2.36E+02	2.00E+02	2.01E+02	2.00E+02	2.48E+02	2.05E+02	2.12E+02
Std	2.81E+01	2.63E+00	3.65E-10	9.82E-05	2.74E+00	9.85E-06	1.27E-09	1.11E-01	2.94E-16
W	-	-	-	+	+	+	-	+	

Table 1 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F25									
Mean	4.25E+02	3.99E+02	4.17E+02	4.01E+02	4.45E+02	4.01E+02	4.40E+02	4.11E+02	3.94E+02
Std	1.17E+01	5.01E+00	4.60E+00	5.58E+00	1.74E+01	6.50E+00	3.17E+01	4.44E-01	4.57E+00
W	–	–	–	–	–	–	–	–	–
–	22	18	17	15	17	19	22	24	
+	3	4	6	9	7	6	3	1	
=	0	3	2	1	1	0	0	0	
α	0.05	0.05	0.05	–	0.1	0.05	0.05	0.05	

The better values of Mean and Std among SAMDE and eight other state-of-the-art DE variants JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE, are highlighted in bold

JADE and ESADE on all hybrid composition functions. The performance of MPEDE is better than that of SAMDE only on function *F24*. However, SAMDE obtains worse performance on functions *F18*, *F20* and *F24*. And CoDE is superior to SAMDE on functions *F15*, *F18*, *F20*, *F22* and *F23*.

In summary, SAMDE has the best overall performance compared with other eight competitors, namely JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on all the 25 benchmark functions with 10 variables. Actually, the results of Wilcoxon’s rank sum tests reported in the last three rows indicate that SAMDE is significantly better than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 22, 18, 17, 15, 17, 19, 22 and 24 functions, respectively. It is significantly worse than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 3, 4, 6, 9, 7, 6, 3 and 1 functions and similar to them on 0, 3, 2, 1, 1, 0, 0 and 0 functions, respectively. As the Table 1 states, SAMDE shows a significant improvement over JADE, SaDE, EPSDE, SspDE, ESADE and MPEDE, with a level of significance $\alpha = 0.05$, and over SOUPDE, with $\alpha = 0.1$.

Applying the Multiple Sign test, suppose level of significance $\alpha = 0.1$, the critical value of R_j is 6 for $m = 8(m = k - 1)$ and $n = 25$ (Derrac et al. 2011), and losses = better(–)+[similar(=)/2], since the number of minuses in the pairwise comparison between the control algorithm SAMDE and JADE, SaDE, SspDE, ESADE, MPEDE is equal to 3, 6, 6, 3, 1 in Table 1, respectively, we may conclude that SAMDE has a significantly better performance than them.

As indicated in Table 2, it is found that SAMDE has the best performance among the other eight classical state-of-the-art DE variants on the 9 test functions with 30D. SAMDE shows a statistically significant performance when compared with all the contestant algorithms in most of the functions.

Regarding the unimodal functions *F1–F5*, JADE and SAMDE show the better performance. SAMDE obtains

significantly best results on function *F5* than all other peers. JADE performs better than SAMDE on three functions (*F2–F4*). EPSDE, CoDE and SOUPDE do not show better performance than SAMDE on any unimodal functions.

For basic multimodal benchmark functions *F6–F12*, SaDE, EPSDE and SspDE show the better performance. SAMDE has significantly best results on the 2 test functions (*F8* and *F12*) than all the other DE variants. Moreover, SAMDE outperforms JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on five (*F6*, *F8* and *F10–F12*), four (*F6*, *F7*, *F9* and *F12*), three (*F9*, *F10* and *F12*), five (*F6*, *F7*, *F9*, *F10* and *F12*), four (*F6*, *F7*, *F10* and *F12*), five (*F6*, *F7*, *F9*, *F10* and *F12*), six (*F6–F9*, *F11* and *F12*) and five (*F6–F9* and *F12*) benchmark functions, respectively. As for expanded multimodal functions, SAMDE is inferior to JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on function *F13*. In contrast, it outperforms all the other DE variants on function *F14*.

With regard to the more complex hybrid composition functions *F15–F25*, ESADE, MPEDE and SAMDE show the better performance. SAMDE performs significantly best results on the 5 test functions (*F15*, *F18*, *F19*, *F24* and *F25*) than all other peer DE variants. It outperforms JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on nine (*F15–F20*, *F22*, *F23* and *F25*), nine (*F15–F20*, *F22*, *F23* and *F25*), nine (*F15–F20*, *F22*, *F23* and *F25*), seven (*F15*, *F17–F20*, *F22* and *F25*), eight (*F15–F20*, *F22* and *F25*), nine (*F15*, *F17–F23* and *F25*), six (*F15*, *F17*, *F19*, *F21*, *F24* and *F25*) and six (*F15*, *F18–F20*, *F23* and *F25*) benchmark functions, respectively. However, SAMDE obtains worse performance on functions *F16* and *F21*. And CoDE, SOUPDE, ESADE and MPEDE are superior to SAMDE on three functions (*F16*, *F21* and *F23*), two functions (*F21* and *F23*), four functions (*F16*, *F20*, *F22* and *F23*) and four functions (*F16*, *F17*, *F21* and *F22*), respectively.

In summary, SAMDE has the best overall performance compared with other eight competitors, namely JADE,

Table 2 Comparison of the average error for 25 independent runs by different optimization techniques with $D = 30$ on functions F_1 – F_{25}

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F1									
Mean	1.16E–23	1.57E–26	4.92E–23	1.69E–02	9.27E–06	2.52E–10	5.78E+00	4.27E–05	5.97E–24
Std	3.20E–24	4.24E–27	1.36E–23	3.64E–03	1.95E–05	8.26E–11	1.54E+00	7.22E–07	8.68E–25
W	–	+	–	–	–	–	–	–	–
F2									
Mean	9.40E–03	2.02E+02	1.36E+02	1.41E+02	6.28E+02	3.49E+01	5.77E+00	1.51E+01	6.71E–01
Std	7.21E–04	1.96E+01	2.21E+01	1.17E+01	2.46E+02	1.04E+01	1.52E+00	1.30E–01	6.66E–02
W	+	–	–	–	–	–	–	–	–
F3									
Mean	3.06E+05	9.77E+06	6.50E+06	2.25E+06	1.45E+08	1.47E+06	8.76E+05	1.04E+05	1.93E+06
Std	4.95E+03	1.91E+06	1.20E+06	3.93E+04	6.06E+07	2.75E+05	7.86E+03	4.54E+02	1.30E+05
W	+	–	–	–	–	+	+	+	–
F4									
Mean	1.94E+01	3.32E+03	1.44E+03	2.74E+03	8.91E+03	9.05E+02	9.05E+02	2.41E+02	1.54E+02
Std	1.86E+00	4.23E+02	2.54E+02	2.66E+02	4.09E+03	1.88E+02	3.52E+01	2.32E+00	1.53E+01
W	+	–	–	–	–	–	–	–	–
F5									
Mean	1.61E+03	2.71E+03	1.88E+03	2.78E+03	8.26E+03	2.16E+03	2.65E+03	1.58E+03	1.52E+03
Std	1.18E+01	4.16E+01	6.84E+01	1.06E+02	1.59E+03	6.50E+01	8.68E+00	3.54E+00	6.87E+00
W	–	–	–	–	–	–	–	–	–
F6									
Mean	1.16E+02	7.72E+01	3.23E+01	4.74E+02	3.50E+02	8.40E+01	6.76E+01	1.37E+02	3.75E+01
Std	2.11E+02	5.18E–01	1.94E–01	1.43E+02	7.02E+02	8.86E–01	3.54E+00	7.08E–01	2.27E–02
W	–	–	+	–	–	–	–	–	–
F7									
Mean	1.28E–02	2.49E–02	1.39E–02	1.15E+00	1.28E+00	1.47E–01	5.16E+00	5.04E–01	1.73E–02
Std	4.14E–06	3.02E–04	2.65E–07	3.30E–02	1.13E–01	1.61E–02	1.19E+00	3.95E–03	6.09E–06
W	+	–	+	–	–	–	–	–	–
F8									
Mean	2.14E+01	2.12E+01	2.12E+01	2.12E+01	2.12E+01	2.12E+01	2.16E+01	2.13E+01	2.12E+01
Std	7.97E–02	6.41E–02	6.25E–02	5.87E–02	6.26E–02	6.37E–02	1.20E–01	4.26E–03	6.54E–02
W	–	=	=	=	=	=	–	–	–
F9									
Mean	2.88E+01	3.47E+01	1.01E+02	3.43E+01	2.35E+01	1.43E+02	8.82E+01	1.09E+02	2.91E+01
Std	4.65E+00	5.16E+00	1.38E+01	4.80E+00	5.59E+00	3.95E+01	1.39E+01	1.12E+00	8.09E–01
W	+	–	–	–	+	–	–	–	–
F10									
Mean	2.94E+02	2.20E+02	2.34E+02	2.53E+02	2.64E+02	2.33E+02	2.14E+02	2.10E+02	2.25E+02
Std	2.40E+01	1.90E+01	1.90E+01	2.15E+01	3.43E+01	1.98E+01	2.51E+01	8.81E–01	1.64E+01
W	–	+	–	–	–	–	+	+	–
F11									
Mean	5.02E+01	4.35E+01	4.26E+01	3.91E+01	4.04E+01	4.38E+01	5.17E+01	4.44E+01	4.52E+01
Std	2.13E+00	2.04E+00	2.20E+00	2.45E+00	2.67E+00	2.29E+00	3.67E+00	1.21E–01	1.79E+00
W	–	+	+	+	+	+	–	+	–
F12									
Mean	6.53E+04	1.53E+04	6.57E+04	1.27E+05	2.18E+05	7.36E+03	1.46E+04	9.96E+03	6.17E+03
Std	3.34E+04	1.13E+04	1.55E+04	2.57E+04	6.12E+04	6.45E+02	5.95E+03	3.44E+01	2.50E+00
W	–	–	–	–	–	–	–	–	–

Table 2 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F13									
Mean	1.54E+01	1.50E+01	1.40E+01	1.46E+01	8.02E+00	1.58E+01	2.66E+01	1.28E+01	1.78E+01
Std	1.86E+00	1.46E+00	1.76E+00	1.88E+00	1.25E+00	2.97E+00	3.64E+00	1.34E-01	1.62E+00
W	+	+	+	+	+	+	-	+	
F14									
Mean	1.45E+01	1.40E+01	1.40E+01	1.40E+01	1.40E+01	1.41E+01	1.49E+01	1.43E+01	1.40E+01
Std	1.70E-01	1.86E-01	1.86E-01	2.09E-01	2.04E-01	1.85E-01	1.48E-01	1.12E-02	1.76E-01
W	-	=	=	=	=	-	-	-	
F15									
Mean	3.50E+02	3.59E+02	3.88E+02	3.94E+02	4.17E+02	4.12E+02	3.83E+02	3.97E+02	3.49E+02
Std	1.21E+01	2.08E+01	3.23E-08	3.09E-01	3.46E+01	3.89E-04	1.83E+00	5.47E-03	5.36E-13
W	-	-	-	-	-	-	-	-	
F16									
Mean	3.48E+02	2.86E+02	2.94E+02	2.69E+02	3.59E+02	2.56E+02	2.82E+02	2.50E+02	2.84E+02
Std	3.56E+01	2.92E+01	1.89E+01	2.33E+01	6.89E+01	2.12E+01	2.23E+01	1.63E+00	1.45E+01
W	-	-	-	+	-	+	+	+	
F17									
Mean	4.00E+02	3.02E+02	3.38E+02	3.37E+02	4.26E+02	3.08E+02	3.37E+02	2.87E+02	3.00E+02
Std	3.84E+01	3.98E+01	1.88E+01	3.33E+01	7.31E+01	2.35E+01	2.32E+01	1.15E+00	1.70E+01
W	-	-	-	-	-	-	-	+	
F18									
Mean	9.04E+02	9.04E+02	9.04E+02	9.11E+02	9.18E+02	9.09E+02	9.03E+02	9.07E+02	9.03E+02
Std	4.41E-01	5.30E-01	2.76E-01	6.20E-01	2.34E+00	3.44E-01	1.61E+00	1.32E-02	7.53E-04
W	-	-	-	-	-	-	=	-	
F19									
Mean	9.01E+02	9.03E+02	9.09E+02	9.11E+02	9.19E+02	9.10E+02	9.12E+02	9.07E+02	8.99E+02
Std	4.14E-01	2.45E+00	2.96E-01	6.00E-01	2.30E+00	3.60E-01	1.74E+00	1.17E-02	6.79E-04
W	-	-	-	-	-	-	-	-	
F20									
Mean	9.08E+02	9.07E+02	9.10E+02	9.11E+02	9.18E+02	9.05E+02	8.90E+02	9.07E+02	8.99E+02
Std	6.58E-01	5.35E-01	3.34E-01	6.09E-01	2.28E+00	2.85E-01	1.58E+00	1.17E-02	6.87E-04
W	-	-	-	-	-	-	+	-	
F21									
Mean	5.00E+02	5.00E+02	5.16E+02	5.00E+02	5.00E+02	5.38E+02	5.30E+02	5.00E+02	5.24E+02
Std	6.08E-13	3.07E-13	2.67E-13	1.39E-03	1.41E-05	4.10E-03	1.51E+00	2.81E-07	5.22E-13
W	+	+	+	+	+	-	-	+	
F22									
Mean	9.47E+02	9.49E+02	9.43E+02	9.70E+02	1.11E+03	9.43E+02	9.14E+02	9.19E+02	9.29E+02
Std	8.35E+00	7.99E+00	8.08E+00	9.28E+00	5.68E+01	8.62E+00	7.56E+00	3.37E-01	5.49E+00
W	-	-	-	-	-	-	+	+	
F23									
Mean	5.50E+02	5.50E+02	5.66E+02	5.34E+02	5.34E+02	5.66E+02	5.34E+02	5.38E+02	5.37E+02
Std	1.20E-03	1.89E+00	1.28E-04	1.10E-03	2.15E-03	6.17E-04	1.02E+00	3.14E-05	7.08E-13
W	-	-	-	+	+	-	+	-	
F24									
Mean	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.05E+02	2.00E+02	2.00E+02
Std	7.61E-13	1.34E-13	8.61E-13	3.56E-03	1.22E-04	8.81E-11	1.40E+00	7.27E-07	2.21E-13
W	=	=	=	=	=	=	-	=	

Table 2 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F25									
Mean	2.18E+02	2.16E+02	2.13E+02	2.14E+02	2.36E+02	2.13E+02	2.13E+02	2.13E+02	2.12E+02
Std	3.95E-01	3.93E-01	2.43E-01	2.14E-01	4.81E+00	2.75E-01	1.44E+00	6.04E-03	1.40E-01
W	–	–	–	–	–	–	–	–	–
–	17	17	18	17	17	19	18	16	
+	7	5	5	5	5	4	6	8	
=	1	3	2	3	3	2	1	1	
α	0.1	0.05	0.05	0.05	0.05	0.05	0.05	–	

The better values of Mean and Std among SAMDE and eight other state-of-the-art DE variants JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE, are highlighted in bold

SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on benchmark functions with 30 variables. Actually, the results of Wilcoxon's rank sum tests reported in the last three rows indicate that SAMDE is significantly better than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 17, 17, 18, 17, 17, 19, 18 and 16 functions, respectively. It is significantly worse than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 7, 5, 5, 5, 5, 4, 6 and 8 functions and similar to them on 1, 3, 2, 3, 3, 2, 1 and 1 functions, respectively. As the Table 2 states, SAMDE shows a significant improvement over SaDE, EPSDE, CoDE, SOUPDE, SspDE and ESADE, with a level of significance $\alpha = 0.05$, and over JADE, with $\alpha = 0.1$.

Applying the Multiple Sign test, as mentioned above, since the number of minuses in the pairwise comparison between the control algorithm SAMDE and EPSDE, SspDE is equal to 6, 5 in Table 2, respectively, we may conclude that SAMDE has a significantly better performance than them.

As indicated in Table 3, it is found that SAMDE has the best performance among the other eight classical state-of-the-art DE variants on the 4 test functions with 50D.

Regarding the unimodal functions $F1-F5$, JADE and MPEDE show the better performance. JADE performs better than SAMDE on three functions ($F1-F3$). CoDE, SOUPDE and SspDE do not show better performance than SAMDE on any unimodal functions. SAMDE performs better than SaDE, EPSDE, ESADE and MPEDE on four functions ($F2-F5$), four functions ($F2-F5$), four functions ($F1, F2, F4$ and $F5$), one functions ($F1$), respectively.

For basic multimodal benchmark functions $F6-F12$, EPSDE and ESADE show the better performance. SAMDE outperforms JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on four ($F8-F11$), three ($F7, F9$ and $F12$), two ($F9$ and $F10$), five ($F6, F7, F9, F10$ and $F12$), four ($F6, F7, F10$ and $F12$), five ($F7-F10$ and $F12$), three ($F7, F8$ and $F12$) and four ($F7-F9$ and $F12$)

benchmark functions, respectively. As for expanded multimodal functions, SAMDE is superior to JADE, SaDE, EPSDE, CoDE, SspDE and MPEDE on function $F13$. meanwhile, it outperforms all the other DE variants on function $F14$.

With regard to the more complex hybrid composition functions $F15-F25$, CoDE, ESADE and MPEDE show the better performance. SAMDE outperforms JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on ten ($F15-F19$ and $F21-F25$), seven ($F15, F16, F18-F20, F22$ and $F25$), seven ($F15-F19, F22$ and $F25$), three ($F15, F16$ and $F22$), eight ($F15-F20, F22$ and $F25$), four ($F15, F16, F22$ and $F25$), three ($F15, F21$ and $F25$) and five ($F15, F19, F21$ and $F25$) benchmark functions, respectively. SAMDE outperforms all the other DE variants on function $F14$. However, SAMDE obtains worse performance on functions $F20, F21$ and $F23$. And CoDE, SspDE and ESADE are superior to SAMDE on seven functions ($F17-F21, F23$ and $F25$), six functions ($F17-F21$ and $F23$) and seven functions ($F16-F20, F22$ and $F23$), respectively.

In summary, the results of Wilcoxon's rank sum tests reported in the last three rows indicate that SAMDE is significantly better than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 18, 16, 15, 15, 18, 16, 11 and 11 functions, respectively. It is significantly worse than JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE on 7, 7, 8, 8, 5, 8, 13 and 13 functions and similar to them on 0, 2, 2, 2, 2, 1, 1 and 1 functions, respectively. As the Table 3 states, SAMDE shows a significant improvement over other 8 algorithms, with a level of significance $\alpha = 0.05$.

Applying the Multiple Sign test, as mentioned above, since the number of minuses in the pairwise comparison between the control algorithm SAMDE and JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE is equal to 4, 6, 6, 6, 5, 5 in Table 3, respectively, we may conclude that SAMDE has a significantly better performance than them.

Table 3 Comparison of the average error for 25 independent runs by different optimization techniques with $D = 50$ on functions F_1-F_{25}

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F1									
Mean	9.79E-28	3.35E-28	5.89E-26	1.60E-02	8.06E-06	7.19E-12	8.30E-02	5.27E-08	2.60E-18
Std	1.78E-28	7.07E-29	1.09E-26	3.60E-03	1.52E-05	1.47E-12	1.86E-02	5.79E-10	1.47E-19
W	+	+	+	-	-	-	-	-	-
F2									
Mean	1.89E+00	2.34E+03	2.47E+03	8.91E+02	3.83E+03	4.39E+02	1.88E+02	5.09E+01	1.26E+02
Std	7.39E-02	8.22E+01	2.03E+02	3.15E+01	1.24E+03	5.15E+01	2.98E+00	1.63E-01	4.61E+00
W	+	-	-	-	-	-	-	+	-
F3									
Mean	8.28E+05	4.74E+06	1.12E+07	5.48E+06	3.20E+08	3.51E+06	2.57E+06	9.69E+05	2.80E+06
Std	5.39E+03	4.40E+04	8.54E+05	8.92E+04	1.04E+08	1.94E+05	7.23E+03	6.64E+02	5.25E+04
W	+	-	-	-	-	-	+	+	-
F4									
Mean	6.95E+03	2.14E+04	1.87E+04	1.67E+04	9.34E+04	9.74E+03	1.38E+04	5.89E+03	6.55E+03
Std	2.75E+02	1.73E+03	2.32E+03	8.20E+02	3.49E+04	1.05E+03	3.14E+02	1.73E+01	4.20E+02
W	-	-	-	-	-	-	-	+	-
F5									
Mean	4.61E+03	6.08E+03	4.45E+03	5.46E+03	2.33E+04	5.21E+03	6.90E+03	3.43E+03	4.35E+03
Std	6.57E+00	2.81E+01	4.76E+01	6.42E+01	3.77E+03	5.54E+01	1.48E+00	1.44E+00	6.45E+00
W	-	-	-	-	-	-	-	+	-
F6									
Mean	8.45E+01	1.15E+02	6.54E+01	1.99E+02	3.80E+02	1.44E+02	1.27E+02	1.07E+02	1.60E+02
Std	4.13E-01	1.13E-01	9.62E-02	1.95E+01	7.28E+02	1.55E+00	3.14E+00	1.40E-01	1.20E-01
W	+	+	+	-	-	+	+	+	-
F7									
Mean	6.06E-03	2.99E-02	7.00E-03	1.13E+00	1.05E+00	4.86E-01	1.09E+00	5.63E-02	2.50E-02
Std	5.83E-05	4.42E-04	1.12E-05	2.11E-02	1.87E-02	1.67E-02	2.13E-02	2.72E-04	2.38E-04
W	+	-	+	-	-	-	-	-	-
F8									
Mean	2.15E+01	2.13E+01	2.13E+01	2.13E+01	2.13E+01	2.13E+01	2.15E+01	2.14E+01	2.13E+01
Std	5.83E-02	4.45E-02	4.38E-02	4.00E-02	4.70E-02	4.53E-02	8.91E-02	1.85E-03	4.56E-02
W	-	=	=	=	=	-	-	-	-
F9									
Mean	1.03E+02	5.90E+01	2.24E+02	1.05E+02	4.36E+01	2.74E+02	2.08E+01	1.68E+02	5.14E+01
Std	1.11E+01	4.14E+00	2.07E+01	1.07E+01	7.42E+00	6.12E+01	3.60E+00	1.28E+00	1.80E-08
W	-	-	-	-	+	-	+	-	-
F10									
Mean	5.02E+02	4.08E+02	4.35E+02	4.27E+02	5.12E+02	4.22E+02	2.54E+02	3.09E+02	4.14E+02
Std	3.33E+01	2.72E+01	2.63E+01	2.95E+01	6.17E+01	2.92E+01	3.29E+01	2.09E+00	2.54E+01
W	-	+	-	-	-	-	+	+	-
F11									
Mean	8.71E+01	7.74E+01	7.69E+01	6.90E+01	7.17E+01	7.79E+01	7.29E+01	7.63E+01	8.02E+01
Std	2.77E+00	2.69E+00	2.77E+00	3.15E+00	3.85E+00	3.01E+00	3.96E+00	1.60E-01	2.34E+00
W	-	+	+	+	+	+	+	+	-
F12									
Mean	2.36E+04	3.53E+04	2.32E+04	3.78E+05	6.44E+05	3.54E+04	4.50E+04	3.69E+04	3.48E+04
Std	9.63E+02	3.58E+01	5.43E+01	6.13E+04	1.44E+05	1.31E+02	3.97E+01	1.89E+01	7.99E+00
W	+	-	+	-	-	-	-	-	-

Table 3 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F13									
Mean	2.65E+01	2.90E+01	2.76E+01	3.01E+01	1.34E+01	2.95E+01	1.26E+01	2.01E+01	2.00E+01
Std	2.52E+00	2.11E+00	2.39E+00	2.76E+00	1.65E+00	4.82E+00	1.59E+00	1.22E-01	1.45E+00
W	-	-	-	-	+	-	+	-	
F14									
Mean	2.44E+01	2.38E+01	2.38E+01	2.39E+01	2.38E+01	2.38E+01	2.41E+01	2.40E+01	2.38E+01
Std	1.91E-01	2.02E-01	2.09E-01	2.00E-01	2.47E-01	2.06E-01	2.26E-01	8.48E-03	1.83E-01
W	-	-	-	-	-	-	-	-	
F15									
Mean	3.29E+02	3.73E+02	3.42E+02	4.09E+02	3.55E+02	3.56E+02	3.42E+02	3.52E+02	3.21E+02
Std	2.62E+00	1.38E+01	2.45E-13	1.47E-01	6.83E+01	9.13E-05	9.09E-02	1.52E-04	1.14E-11
W	-	-	-	-	-	-	-	-	
F16									
Mean	3.62E+02	2.31E+02	3.36E+02	2.93E+02	4.45E+02	2.96E+02	2.76E+02	2.16E+02	2.89E+02
Std	2.19E+01	2.35E+01	1.50E+01	2.38E+01	4.95E+01	1.93E+01	1.49E+01	7.57E-01	1.59E+01
W	-	-	-	-	-	-	+	+	
F17									
Mean	4.21E+02	3.43E+02	3.70E+02	3.36E+02	5.71E+02	3.39E+02	3.62E+02	2.93E+02	3.67E+02
Std	2.83E+01	1.97E+01	1.54E+01	3.37E+01	8.49E+01	1.94E+01	2.19E+01	1.02E+00	1.12E+01
W	-	+	-	+	-	+	+	+	
F18									
Mean	9.32E+02	9.42E+02	9.35E+02	9.11E+02	9.76E+02	9.17E+02	7.87E+02	9.18E+02	9.30E+02
Std	2.24E+00	1.37E+00	8.39E-01	5.74E-01	1.67E+01	2.41E-01	2.15E-01	5.90E-03	3.12E-05
W	-	-	-	+	-	+	+	+	
F19									
Mean	9.36E+02	9.45E+02	9.29E+02	9.02E+02	9.62E+02	9.10E+02	7.44E+02	9.30E+02	9.27E+02
Std	1.51E+00	2.22E+00	3.21E-01	5.07E-01	6.94E+00	3.30E-01	2.24E-01	4.11E-03	8.08E-05
W	-	-	-	+	-	+	+	-	
F20									
Mean	9.28E+02	9.43E+02	9.26E+02	9.10E+02	9.77E+02	9.01E+02	8.09E+02	9.17E+02	9.32E+02
Std	1.12E+00	1.10E+00	5.52E-01	5.49E-01	1.72E+01	1.53E-01	2.51E-01	7.15E-03	1.89E-05
W	+	-	+	+	-	+	+	+	
F21									
Mean	5.40E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.12E+02	5.40E+02	5.61E+02	5.39E+02
Std	1.47E-12	4.93E-13	4.03E-13	1.80E-03	5.35E-06	7.85E-08	1.89E-02	8.27E-04	1.07E-10
W	-	+	+	+	+	+	-	-	
F22									
Mean	1.00E+03	9.88E+02	9.76E+02	9.68E+02	1.09E+03	9.86E+02	9.49E+02	9.34E+02	9.64E+02
Std	7.80E+00	5.50E+00	6.04E+00	9.19E+00	3.00E+01	5.97E+00	6.64E+00	1.65E-01	4.29E+00
W	-	-	-	-	-	-	+	+	
F23									
Mean	5.95E+02	5.67E+02	5.42E+02	5.34E+02	5.39E+02	5.55E+02	5.67E+02	5.58E+02	5.83E+02
Std	2.28E-01	4.23E-02	6.14E-08	1.00E-03	1.75E-02	5.63E-02	1.21E-02	1.61E-04	2.02E-09
W	-	+	+	+	+	+	+	+	
F24									
Mean	8.80E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
Std	8.63E+00	2.62E-11	1.05E-13	2.70E-03	9.40E-03	2.12E-11	2.03E-02	2.64E-09	2.86E-13
W	-	=	=	=	=	=	=	=	

Table 3 (continued)

	JADE	SaDE	EPSDE	CoDE	SOUPDE	SspDE	ESADE	MPEDE	SAMDE
F25									
Mean	4.92E+02	2.43E+02	2.24E+02	2.14E+02	3.31E+02	2.26E+02	2.23E+02	2.28E+02	2.22E+02
Std	3.82E+00	9.72E−01	2.46E−01	2.36E−01	1.33E+01	7.79E−01	1.35E−01	6.83E−03	1.46E−01
W	–	–	–	+	–	–	–	–	–
–	18	16	15	15	18	16	11	11	–
+	0	2	2	2	2	1	1	1	–
=	7	7	8	8	5	8	13	13	–
α	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	–

The better values of Mean and Std among SAMDE and eight other state-of-the-art DE variants JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE, are highlighted in bold

SAMDE outperforms all the other DE variants on functions $F8$ and $F14$ for both 10, 30 and 50 variables. SAMDE obtains best performance on functions $F8$, $F14$, $F15$ and $F24$ for both 30 and 50 variables that compares with all other peer algorithms. SAMDE outperforms all other peer algorithms on functions $F5$, $F8$, $F12$, $F14$, $F19$ and $F25$ for both 10 and 30 variables.

5.3 Sensitivity analysis

5.3.1 Sensitiveness to the NP values

In this subsection, an experiment is conducted to investigate the sensitivity of the SAMDE algorithm to variations in population size. The population size of all the DE algorithms is set as $NP = 30, 60, 90, 120, 150, 180$ and 300 , respectively. We use those algorithms to solve the 30-dimensional functions and the maximum number of function evaluations was fixed at $2000 \times D$. Table 4 shows the results of this sensitivity tests. Results highlighted in boldface show the best results from the best setting for each parameter. We performed 25 independent runs for every parameters.

In Table 4, it can be observed that SAMDE is sensitive to parameter NP on many of the benchmark functions. For the unimodal functions $F1$ – $F5$, SAMDE shows better performance with population size decrease in solving problems $F1$ – $F4$. However, SAMDE is not sensitive to parameter NP in solving problem $F5$. With regard to the basic multimodal benchmark functions $F6$ – $F12$, the better performance of SAMDE concentrate in $NP = 60$. As for expanded multimodal functions, SAMDE shows better performance with population size decrease. For the hybrid composition functions $F15$ – $F25$, SAMDE shows better performance with $NP=30$ in solving problems $F16$ and $F17$. And SAMDE is not sensitive to parameter NP in solving problem $F18$ – $F25$. When NP was equal to 60, the algorithm gave the best results on most of selected

benchmarks. So we choose the value of the population size as 60.

5.3.2 Sensitiveness to the F and CR values

In the traditional DE algorithm, the control parameter F and CR will be tuned by a time-consuming trial-and-error procedure for different optimization problems (Zhao et al. 2016). In SAMDE, F is adaptively adjusted by the adaptive parameter μF . μF according to the number of iteration as the search process continues. And three sub-populations have different values of μF . Each μF is decided by the corresponding initial value $o\mu F$. In order to study their effects on the performance of SAMDE, some experimental studies are conducted in Fig. 1.

We choose four benchmark functions as representative ones. They are functions $F1$, $F7$, $F13$ and $F20$ which respectively as unimodal function, basic multimodal function, expanded multimodal function and hybrid composition function. The horizontal axis is the numbers of iteration, and the vertical axis is the average error values of fitness. From Fig. 1, it can be observed that the effects of the different value of $o\mu F$ for the evolution results were significant when solving some optimization problem, such as functions $F1$ and $F7$. In addition, mutation strategy “best/2” is relatively sensitive to F value with other two mutation strategies. In contrast, the effects of the different value of $o\mu F$ for the evolution results were not sensitive when solving some optimization problem, such as functions $F13$ and $F20$. We can find that the best $o\mu F$ values for one sub-population are different when solving different optimization problems. For instance, in sub-population X2 (Fig. 1 a2, b2, c2, d2), the best $o\mu F2$ values are 0.7 for solving function $F1$, 1 for solving function $F7$, 0.5 for solving function $F13$ and 0.6 for solving function $F20$. And the best $o\mu F$ values for different sub-populations are different when solving one optimization problem. For

Table 4 Sensitivity analysis with respect to population size (NP)

NP	30	90	120	150	180	300	60
F1	3.93E-17-	2.38E-19-	4.19E-15-	3.16E-11-	3.50E-09-	4.38E-05-	5.97E-24
F2	1.17E+01-	4.67E+01-	1.11E+02-	1.50E+02-	8.75E+02-	4.70E+03-	6.71E-01
F3	1.17E+06+	4.94E+06-	9.31E+06-	1.51E+07-	2.53E+07-	3.00E+07-	1.93E+06
F4	4.05E+02-	3.22E+02-	2.62E+02-	3.01E+03-	2.76E+03-	1.60E+04-	1.54E+02
F5	2.16E+03-	1.48E+03+	1.98E+03-	1.26E+03+	1.64E+03-	3.61E+03-	1.52E+03
F6	6.60E+03-	2.63E+01+	2.43E+01+	2.46E+01+	2.55E+01+	6.02E+01-	3.75E+01
F7	6.19E-01-	1.98E-02-	3.68E-02-	4.15E-02-	1.17E-01-	1.04E+00-	1.73E-02
F8	2.12E+01=	2.12E+01=	2.12E+01=	2.13E+01-	2.13E+01-	2.13E+01-	2.12E+01
F9	5.07E+01-	1.89E+01+	2.33E+01+	1.57E+02-	3.63E+01-	1.90E+02-	2.91E+01
F10	2.98E+01-	2.33E+02-	2.28E+02-	2.36E+02-	2.50E+02-	2.61E+02-	2.25E+02
F11	1.72E+01+	4.63E+01-	4.63E+01-	4.67E+01-	4.69E+01-	4.74E+01-	4.52E+01
F12	6.11E+03+	5.37E+03+	7.57E+03-	3.02E+04-	3.53E+03+	3.26E+03+	6.17E+03
F13	2.58E+00+	1.79E+01-	1.84E+01-	2.02E+01-	2.05E+01-	2.18E+01-	1.78E+01
F14	1.35E+01+	1.41E+01-	1.41E+01-	1.42E+01-	1.42E+01-	1.43E+01-	1.40E+01
F15	5.10E+02-	3.00E+02+	4.00E+02-	2.00E+02+	3.00E+02+	5.00E+02-	3.49E+02
F16	6.64E+01+	4.02E+02-	3.54E+02-	2.64E+02+	2.71E+02+	3.72E+02-	2.84E+02
F17	1.83E+02+	2.71E+02+	2.90E+02+	5.53E+02-	3.10E+02-	4.48E+02-	3.00E+02
F18	9.47E+02-	9.11E+02-	9.09E+02-	9.05E+02-	9.10E+02-	9.12E+02-	9.03E+02
F19	9.29E+02-	9.11E+02-	9.05E+02-	9.04E+02-	9.10E+02-	9.08E+02-	8.99E+02
F20	9.48E+02-	9.07E+02-	9.06E+02-	9.11E+02-	9.05E+02-	9.09E+02-	8.99E+02
F21	5.38E+02-	5.46E+02-	5.40E+02-	5.43E+02-	5.36E+02-	5.29E+02-	5.24E+02
F22	9.55E+02-	9.39E+02-	9.43E+02-	9.42E+02-	9.30E+02-	9.59E+02-	9.29E+02
F23	9.65E+02-	5.34E+02=	5.34E+02=	5.34E+02=	5.34E+02=	5.34E+02=	5.34E+02
F24	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02
F25	1.28E+03-	2.12E+02=	2.12E+02=	2.13E+02-	2.13E+02-	2.16E+02-	2.12E+02
-	15	15	18	19	19	22	
+	7	6	3	4	4	1	
=	2	4	4	2	2	2	

example, when solving function $F1$ (Fig. 1 a1, a2, a3), the best $\sigma\mu F1$ is 0.9, the best $\sigma\mu F2$ is 0.7, the best $\sigma\mu F3$ is 0.5.

In SAMDE, CR is adaptively adjusted by the adaptive parameter μCR . μCR according to iteration number as the search process continues. And three sub-populations have different values of μCR . Each μCR is decided by the corresponding initial value $\sigma\mu CR$. In order to study their effects on the performance of SAMDE, some experimental studies are conducted in Fig. 2.

We plot the average values of fitness changes for different $\sigma\mu CR$ in each sub-population. We select four benchmark functions which same as the setting of experimental studies for $\sigma\mu F$. For $F1$ and $F7$, the effects of the different value of $\sigma\mu CR$ for the evolution results were significant. In particular, this phenomenon is more obvious in sub-population X3. In contrast, the average fitness values is not sensitive to parameter $\sigma\mu F$ in solving problems $F13$ and $F20$. From Fig. 2, it can be observed that two point,

one is the best $\sigma\mu CR$ values for one sub-population are different when solving different optimization problems. The other is the best $\sigma\mu CR$ values for different sub-populations are different when solving one optimization problem.

In summary, the select of appropriate parameters are more important. According to these experimental studies, we choose proper $\sigma\mu CR$ and $\sigma\mu F$ to solve 25 numerical optimization problems.

5.3.3 Comparisons between SAMDE and its variants

In this experiment, comparisons between twelve variants of SAMDE and their corresponding SAMDE algorithms are made. They are divided into three groups. The first group is comparison between SAMDE and its variants derived by different setting of update for sub-populations. The second group is comparison between SAMDE and its variants

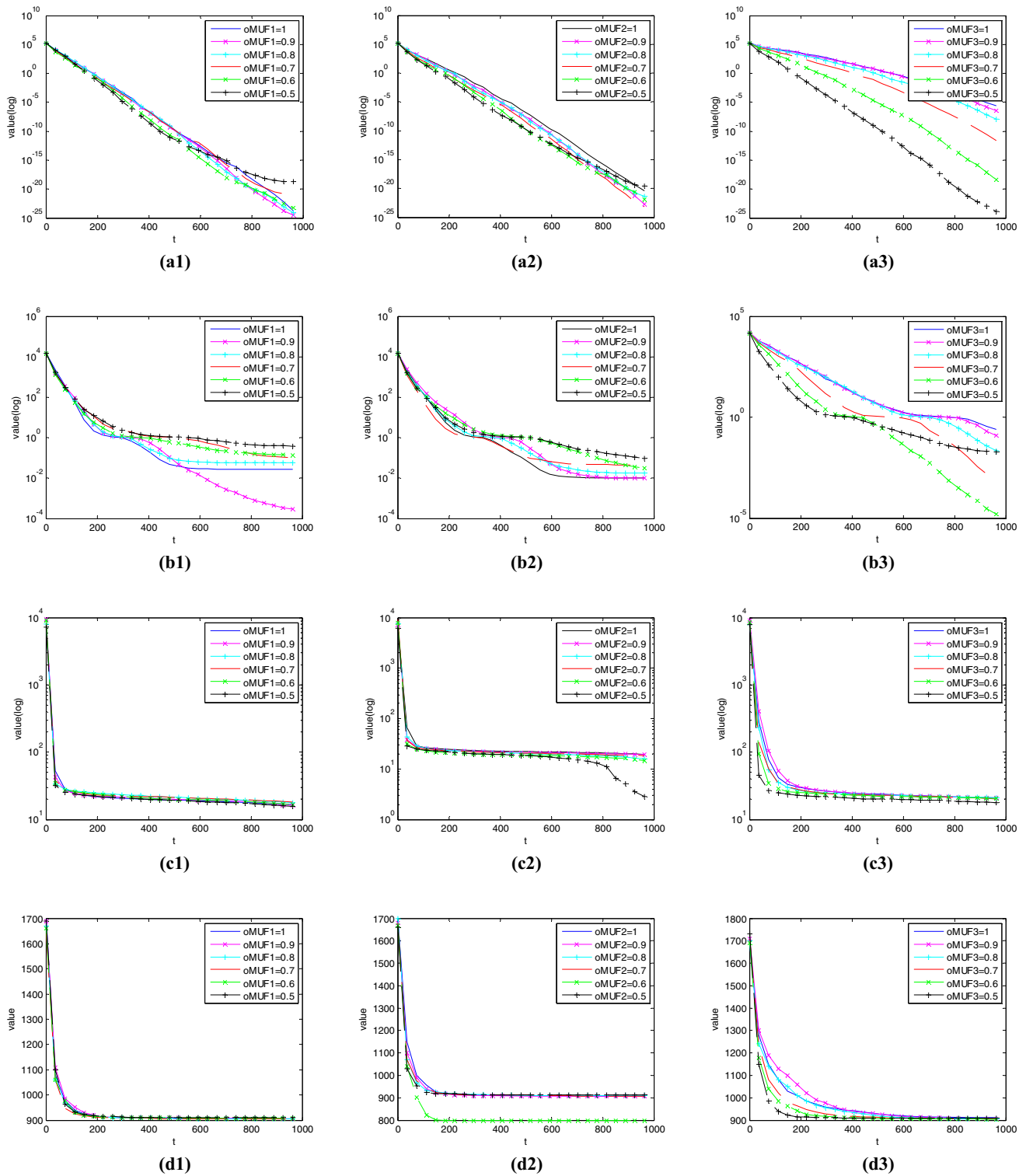


Fig. 1 The influence of $\sigma\mu F$ on evolution of four representative functions. **a1–a3** are the effect of $\sigma\mu F1$, $\sigma\mu F2$ and $\sigma\mu F3$ take different values on function $F1$, respectively; **b1–b3** are related to function $F7$; **c1–c3** are related to function $F13$; **d1–d3** are related to function $F20$

derived by different setting of mutation strategies. The third group is comparison between SAMDE and its variants derived by different setting of evolution methods.

The results for the functions at 30D are shown in Tables 5, 6 and 7. The better values in terms of mean solution error compared between SAMDE and its variants algorithm are highlighted in boldface in Tables 5, 6 and 7.

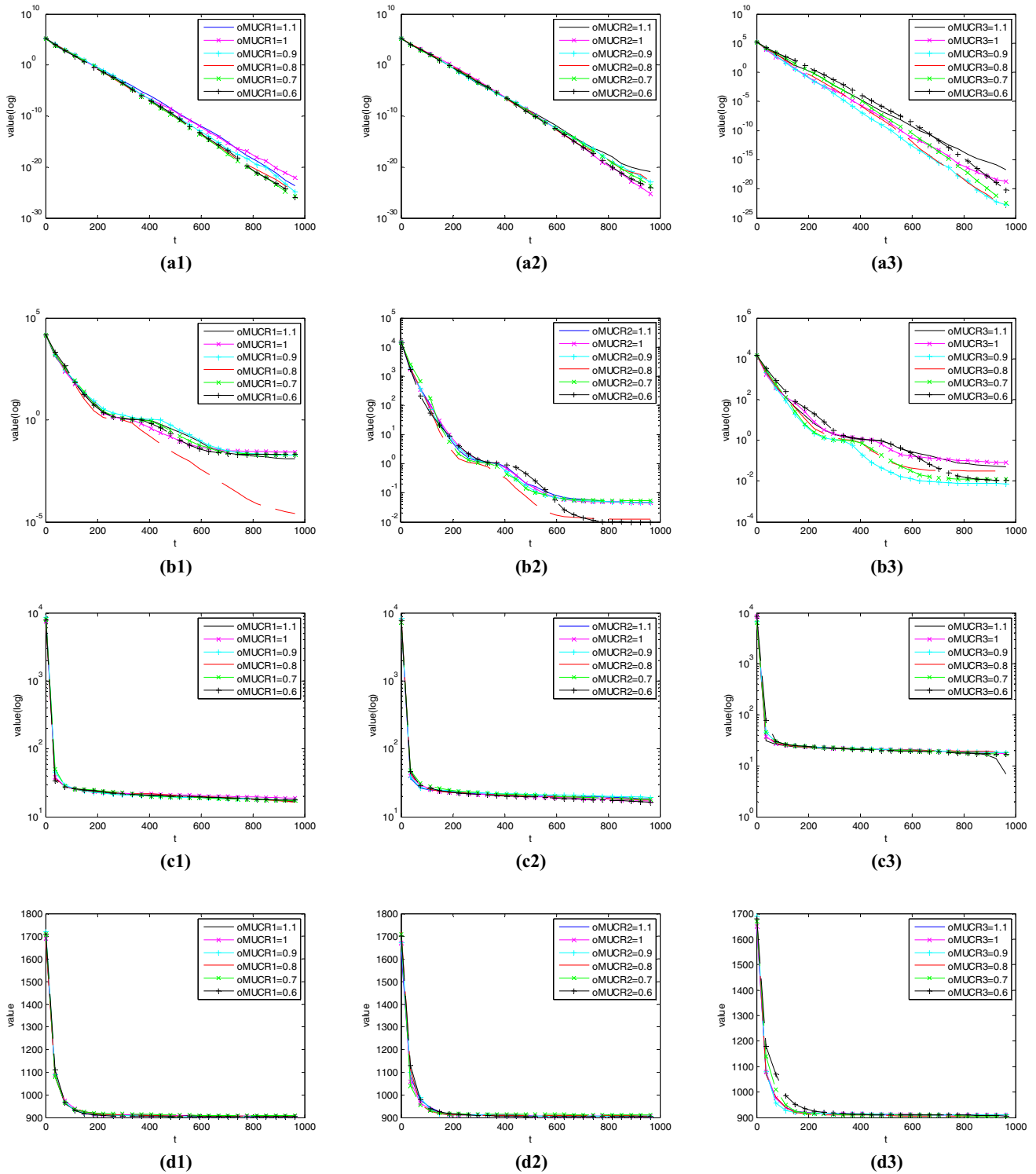


Fig. 2 The influence of $o\mu CR$ on evolution of four representative functions. **a1–a3** are the effect of $o\mu CR1$, $o\mu CR2$ and $o\mu CR3$ take different values on function $F1$, respectively; **b1–b3** are related to function $F7$; **c1–c3** are related to function $F13$; **d1–d3** are related to function $F20$

SAMDE_T1 indicates that three sub-populations are updated every 10 generations. SAMDE_T2 indicates that three sub-populations are updated every 20 generations. SAMDE_T3 indicates that three sub-populations are

updated every 50 generations. SAMDE_T4 indicates that three sub-populations are updated every 100 generations.

From Table 5, we can find that SAMDE significantly outperforms the corresponding variants with respect to the

Table 5 Comparison results between SAMDE and its variants derived by different setting of updation for sub-populations

	SAMDE_T1	SAMDE_T2	SAMDE_T3	SAMDE_T4	SAMDE
F1	2.23E−23−	6.03E−22−	1.04E−15−	4.71E−10−	5.97E−24
F2	9.62E−02+	1.01E+00−	2.04E+00−	1.79E+01−	6.71E−01
F3	3.03E+06−	3.21E+06−	1.47E+06+	1.63E+06+	1.93E+06
F4	1.40E+02+	4.60E+02−	3.16E+02−	1.25E+03−	1.54E+02
F5	2.11E+03−	3.05E+03−	4.10E+03−	2.71E+03−	1.52E+03
F6	7.18E+01−	2.11E+01+	1.43E+02−	7.98E+01−	3.75E+01
F7	8.15E−02−	2.14E−02−	9.45E−01−	7.13E−01−	1.73E−02
F8	2.12E+01−	2.12E+01−	2.12E+01−	2.12E+01−	2.12E+01
F9	2.98E+01−	2.99E+01−	3.38E+01−	6.67E+01−	2.91E+01
F10	2.26E+02−	2.23E+02+	1.15E+02+	2.27E+02−	2.25E+02
F11	4.52E+01=	4.54E+01−	4.54E+01−	2.03E+01+	4.52E+01
F12	9.96E+03−	2.66E+04−	6.75E+03−	1.85E+04−	6.17E+03
F13	1.58E+01+	1.82E+01−	1.72E+01+	5.73E+00+	1.78E+01
F14	1.40E+01=	1.40E+01=	1.40E+01=	1.40E+01=	1.40E+01
F15	3.17E+02+	5.00E+02−	4.05E+02−	2.33E+02+	3.49E+02
F16	5.01E+02−	4.01E+02−	2.66E+02−	1.84E+02+	2.84E+02
F17	2.68E+02+	4.36E+02−	2.60E+02+	2.94E+02+	3.00E+02
F18	9.11E+02−	9.08E+02−	9.17E+02−	9.47E+02−	9.03E+02
F19	9.12E+02−	9.13E+02−	9.27E+02−	9.19E+02−	8.99E+02
F20	8.00E+02+	9.20E+02−	9.13E+02−	9.26E+02−	8.99E+02
F21	5.00E+02=	8.33E+02−	5.00E+02=	5.00E+02=	5.00E+02
F22	9.33E+02−	9.51E+02−	9.37E+02−	9.39E+02−	9.29E+02
F23	5.57E+02−	5.96E+02−	5.34E+02=	5.34E+02=	5.34E+02
F24	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02
F25	2.12E+02=	2.14E+02−	2.12E+02=	2.14E+02−	2.12E+02
−	14	21	16	15	
+	6	2	4	6	
=	5	2	5	4	

Best results of each function are highlighted

overall performance. Specifically, according to the Wilcoxon's test, SAMDE significantly improves the performance of SAMDE_T2 on 21 out of 25 functions and losses on only two functions. SAMDE is significantly better than SAMDE_T1, SAMDE_T3 and SAMDE_T4 on 14, 16 and 15 functions, respectively.

SAMDE consists of three different mutation strategies. To show the significant differences between SAMDE and its variants derived by different setting of mutation strategies, the results are shown in Table 6. SAMDE_M1 SAMDE_M2, SAMDE_M3 are SAMDE with “rand/1”, “rand assemble/1” and “best/2”, respectively. They can be thought as special SAMDE variants with a single mutation strategy. SAMDE_MR denotes an SAMDE variant in which three mutation strategies are randomly selected. In this test, we keep all the other parameters fixed at values discussed in earlier subsections.

It can be noted from the Table 6 that the SAMDE outperforms its other variants. For SAMDE_M1, SAMDE is

significantly better on 19 functions and worse on only three functions. SAMDE is significantly better than SAMDE_M2 and SAMDE_M3 on 13 and 13 functions and worse than SAMDE_M2 and SAMDE_M3 on 7 and 8 functions, respectively. SAMDE significantly improves the performance of SAMDE_MR on 17 out of 25 functions and losses on five functions. These results clearly show the advantage of SAMDE caused by incorporating three mutation strategies, and by individuals' migrating among sub-populations based on the fitness value rank. It seems that different mutation strategies and self-adaptive parameter settings were used in different sub-populations, but some individuals were migrated at every iteration among sub-populations to share different mutation strategies and adaptive parameter settings. It is worth noting that SAMDE-M2 shows competitive performance on two unimodal functions *F3* and *F4* and three hybrid composition functions *F15*–*F17*. SAMDE-M3 shows competitive performance on one unimodal function *F1*, two multimodal

Table 6 Comparison results between SAMDE and its variants derived by different setting of mutation strategies

	SAMDE_M1	SAMDE_M2	SAMDE_M3	SAMDE_MR	SAMDE
F1	5.59E-09-	3.27E-21-	4.82E-27+	1.38E-21-	5.97E-24
F2	1.37E+03-	6.34E+00-	1.10E+00-	1.22E+01-	6.71E-01
F3	1.84E+07-	1.77E+06+	2.09E+06-	3.70E+06-	1.93E+06
F4	4.21E+03-	8.19E+01+	1.37E+02+	2.90E+02-	1.54E+02
F5	1.96E+03-	1.74E+03-	2.05E+03-	1.39E+03+	1.52E+03
F6	2.47E+01+	8.00E+01-	1.37E+01+	2.60E+01+	3.75E+01
F7	6.63E-02-	4.14E-01-	4.67E-02-	1.97E-02-	1.73E-02
F8	2.12E+01-	2.12E+01-	2.12E+01-	2.12E+01-	2.12E+01
F9	1.41E+02-	1.71E+02-	1.39E+01+	1.27E+02-	2.91E+01
F10	2.36E+02-	2.27E+02-	2.29E+02-	2.33E+02-	2.25E+02
F11	4.52E+01-	4.58E+01-	4.52E+01-	4.55E+01-	4.52E+01
F12	9.87E+03-	3.21E+03+	3.58E+03+	1.81E+03+	6.17E+03
F13	1.98E+01-	1.79E+01-	1.81E+01-	1.82E+01-	1.78E+01
F14	1.41E+01-	1.40E+01=	1.41E+01-	1.41E+01-	1.40E+01
F15	4.00E+02-	3.00E+02+	4.00E+02-	4.00E+02-	3.49E+02
F16	2.63E+02+	2.42E+02+	2.56E+02+	3.26E+02-	2.84E+02
F17	2.91E+02+	2.60E+02+	2.87E+02+	2.83E+02+	3.00E+02
F18	9.07E+02-	9.08E+02-	9.05E+02-	9.05E+02-	9.03E+02
F19	9.11E+02-	9.07E+02-	9.10E+02-	9.07E+02-	8.99E+02
F20	9.07E+02-	9.09E+02-	9.07E+02-	9.04E+02-	8.99E+02
F21	5.00E+02=	5.00E+02=	5.00E+02=	5.00E+02=	5.00E+02
F22	9.38E+02-	9.26E+02+	9.25E+02+	9.28E+02+	9.29E+02
F23	5.34E+02=	5.34E+02=	5.34E+02=	9.37E+02-	5.34E+02
F24	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02
F25	2.15E+02-	2.12E+02=	2.12E+02=	2.12E+02=	2.12E+02
-	19	13	13	17	
+	3	7	8	5	
=	3	5	4	3	

Best results of each function are highlighted

functions *F6* and *F9* and one hybrid composition function *F22*. SAMDE_MR obtains much better results for *F5* and *F12* than the other variants.

In this subsection, we compare four different schemes of the proposed SAMDE algorithm. All the algorithms in this section use the same population size and base on the same stopping rule. We tests how adding or changing one component at a time influences the performance.

The first version is the population composed of three sub-populations that based on the fitness value, no use of randomly assigned to individuals. At the end of each generation, all sub-populations are updated based on the new fitness values of its individuals. This version uses the same parameter adaptive strategy as SAMDE. This version is called SAMDE_1. The second scheme (SAMDE_2) of this work is first initializes the entire population and then subdivides the population into three sub-populations that based on the fitness value rank, the first sub-population is called X1 which have better individuals. Then we take X1

as an outstanding sub-population and it no longer involved in later evolution. The other two sub-populations are updated in the same way as SAMDE. The third variant of this work (SAMDE_3) is same as SAMDE_1 in the first 100 generations, after 100 iterations, we take X1 as an outstanding sub-population. At the end of each generation, all sub-populations are updated based on the new fitness values of its individuals. But sub-population X1 is only updated without evolution. The fourth scheme (SAMDE_4) is same as SAMDE in the first 100 generations, after 100 iterations, there are first better nine individuals were extracted and constitute X4. The other individuals are randomly divided into three sub-populations which have the same size. And sub-population X4 is only updated without evolution.

From Table 7, the solution quality of the proposed algorithm SAMDE outperforms all other variants of the proposed work at 30D. For SAMDE_1, SAMDE is significantly better on 17 functions and worse on only four

Table 7 Comparison results between SAMDE and its variants

	SAMDE_1	SAMDE_2	SAMDE_3	SAMDE_4	SAMDE
F1	1.84E−12−	2.08E−11−	5.38E−11−	2.70E−15−	5.97E−24
F2	5.94E+02−	1.63E+03−	1.60E+03−	1.77E+00−	6.71E−01
F3	4.68E+06−	8.05E+06−	1.16E+07−	2.51E+06−	1.93E+06
F4	3.60E+03−	7.25E+03−	4.30E+03−	6.63E+02−	1.54E+02
F5	3.29E+03−	3.35E+03−	3.10E+03−	3.37E+03−	1.52E+03
F6	8.01E+01−	4.03E+05−	9.46E+06−	1.65E+02−	3.75E+01
F7	6.42E−01−	4.70E+03−	4.70E+03−	2.91E−02−	1.73E−02
F8	2.12E+01=	2.12E+01=	2.12E+01=	2.12E+01=	2.12E+01
F9	1.82E+02−	5.08E+01−	8.31E+01−	9.65E+01−	2.91E+01
F10	2.38E+02−	2.21E+02+	2.23E+02+	2.45E+02−	2.25E+02
F11	4.51E+01+	4.45E+01+	4.46E+01+	1.96E+01+	4.52E+01
F12	5.84E+03+	7.91E+04−	2.19E+04−	8.06E+03−	6.17E+03
F13	1.96E+01−	1.61E+01+	1.65E+01+	1.74E+01+	1.78E+01
F14	1.41E+01−	1.40E+01=	1.40E+01=	1.40E+01=	1.40E+01
F15	4.00E+02−	4.00E+02−	4.65E+02−	4.00E+02−	3.49E+02
F16	3.82E+02−	2.49E+02+	3.52E+02−	2.12E+02+	2.84E+02
F17	2.92E+02+	3.74E+02−	3.31E+02−	2.69E+02+	3.00E+02
F18	9.08E+02−	9.12E+02−	9.10E+02−	8.00E+02+	9.03E+02
F19	9.11E+02−	8.38E+02+	9.10E+02−	9.21E+02−	8.99E+02
F20	8.74E+02+	9.12E+02−	8.38E+02+	9.13E+02−	8.99E+02
F21	5.00E+02=	5.00E+02=	5.00E+02=	5.00E+02=	5.00E+02
F22	9.62E+02−	9.69E+02−	9.83E+02−	9.30E+02−	9.29E+02
F23	5.34E+02=	5.34E+02=	5.34E+02=	5.34E+02=	5.34E+02
F24	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02=	2.00E+02
F25	2.17E+02−	1.68E+03−	1.68E+03−	2.13E+02−	2.12E+02
−	17	15	16	15	
+	4	5	4	5	
=	4	5	5	5	

Best results of each function are highlighted

functions. SAMDE is significantly better than SAMDE_2 on 15 functions and worse than SAMDE_2 on 5 functions. SAMDE significantly improves the performance of SAMDE_3 on 16 out of 25 functions and losses on four functions. For SAMDE_4, SAMDE is significantly better on 15 functions out of 25 functions and losses on five functions.

6 Conclusions

This research proposes a self-adaptive multi-population differential evolution algorithm for solving numerical optimization problem. The algorithm divides the population into three sub-populations, each of which employs a certain mutation strategy. These mutation strategies are DE/rand/1/bin, DE/best/2/bin and a our customized mutation strategy that uses a new approach to generate base vector. Because the appropriate control parameters are very

important to the performance of DE in the DE algorithm, an adaptive mechanism is also proposed that control parameters $Fk_i(t)$ and $CRk_i(t)$ of each generation can be gradually self-adapted according to the number of iterations. In this paper, the experimental studies on CEC 2005 benchmark suite proved the good performance of SAMDE, SAMDE was compared with other eight state-of-the-art DE variants including JADE, SaDE, EPSDE, CoDE, SOUPDE, SspDE, ESADE and MPEDE. The experimental results show that the overall performance of SAMDE was better than the eight competitors. The parameter sensitivity of SAMDE is also analysed by some experimental studies. We experimentally show that the multiple mutation strategies and adaptive parameters mechanism in this study have obvious advantages and competitiveness compared with other algorithms.

In the future, we intend to further improve the SAMDE to solve high dimensional optimization problems. In addition, multi-populations is still an effective way to

improve DE efficiency, we will try to use more, different size sub-populations for parallel computing, and test them on sets of competitive benchmarks like those presented in the CEC 2013 (Li et al. 2013) and CEC 2015 (Chen et al. 2014; García-Martínez et al. 2017) special sessions and competitions.

Acknowledgements We are very grateful to the three anonymous reviewers for their constructive suggestions and useful advices. This work is supported by the NSFC (National Natural Science Foundation of China) Project (Grant Nos. 41861047, 41461078), The authors would also like to thank Dr. Wu for providing the source code of MPEDE.

References

- Ali MZ, Awad NH, Suganthan PN (2015) Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization. *Appl Soft Comput* 33:304–327
- Awad Noor H, Ali Mostafa Z, Ponnuthurai N et al (2016) A decremental stochastic fractal differential evolution for global numerical optimization. *Inform Sci* 372:470–491
- Bi X, Liu G (2012) A cloud differential evolutionary algorithm for constrained multi-objective optimization. *J Harbin Eng Univ* 33:1022–1031 (in Chinese)
- Bi X, Liu G, Xiao J (2012) Dynamic adaptive differential evolution based on novel mutation strategy. *J Comput Res Dev* 49:1288–1297 (in Chinese)
- Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. *IEEE Trans Evol Comput* 10:646–657
- Cai Y, Wang J (2015) Differential evolution with hybrid linkage crossover. *Inform Sci* 320(C):244–287
- Chen Q, Liu B, Zhang Q, Liang JJ, Suganthan PN, Qu BY (2014) Problem definition and evaluation criteria for CEC 2015 special session and competition on bound constrained single-objective computationally expensive numerical optimization. Technical report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Nanyang Technological University, Singapore
- Das S, Suganthan P (2011) Differential evolution: a survey of the state-of-the-art. *IEEE Trans Evol Comput* 15:4–31
- Das Swagatam, Mullick SS, Suganthan PN (2016) Recent advances in differential evolution—an updated survey. *Swarm Evol Comput* 27:1–30
- Derrac Joaquín et al (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol Comput* 1:3–18
- Fan Q, Yan X (2015) Self-adaptive differential evolution algorithm with discrete mutation control parameters. *Expert Syst Appl* 42:1551–1572
- Gämperle R, Müller SD, Koumoutsakos P (2002) A parameter study for differential evolution, advances in intelligent systems, fuzzy systems, evolutionary computation. WSEAS Press, Interlaken, pp 293–298
- García-Martínez Carlos et al (2017) Since CEC 2005 competition on real-parameter optimisation: a decade of research, progress and comparative analysis’s weakness. *Soft Comput* 21:1–11
- Ghasemi M, Ghanbarian MM, Ghavidel S, Rahmani S, Moghaddam EM (2014) Modified teaching learning algorithm and double differential evolution algorithm for optimal reactive power dispatch problem: a comparative study. *Inform Sci* 278:231–249
- Guo H, Li Y, Li J, Sun H, Wang D, Chen X (2014) Differential evolution improved with self-adaptive control parameters based on simulated annealing. *Swarm Evol Comput* 19:52–67
- Kong X, Gao L, Ouyang H, Shao Y (2014a) Adaptive differential evolution algorithm with bidirectional randomly multi-mutation strategy. *Comput Integr Manuf Syst* 20:1948–1958 (in Chinese)
- Kong X, Gao L, Ouyang H, Ge Y (2014b) An improved differential evolution algorithm for large scale reliability problems. *J Northeast Univ* 35:328–332 (in Chinese)
- Lampinen J, Zelinka I (2000) On stagnation of the differential evolution algorithm. In: 6th international mendel conference on soft computing, pp 76–83
- Li R, Da F, Ouyang H, Gao L (2012) An improved differential evolution algorithm for reliability problems. *J Northeast Univ* 33:182–186 (in Chinese)
- Li X, Tang K, Omidvar M, Yang Z, Qin K (2013) Benchmark functions for the CEC’2013 special session and competition on large scale global optimization. Technical report, evolutionary computation and machine learning group, RMIT University, Australia
- Liu B, Lampinen J (2010) On setting the control parameters of the differential evolution method. In: Proceedings of Mendel 2002, 8th international conference on soft computing
- Mallipeddi R (2013) Harmony search based parameter ensemble adaptation for differential evolution. *J Appl Math*. Article ID 750819
- Mallipeddi R, Lee M (2015) An evolving surrogate model-based differential evolution algorithm. *Appl Soft Comput* 34:770–787
- Mallipeddi R, Suganthan PN, Pan Q-K, Tasgetiren MF (2011) Differential evolution algorithm with ensemble of parameters and mutation strategies. *Appl Soft Comput* 11:1679–1696
- Mukherjee R, Patra GR, Kundu R, Das S (2014) Cluster-based differential evolution with crowding archive for niching in dynamic environments. *Inform Sci* 267:58–82
- Mukherjee R, Debchoudhury S, Das S (2016) Modified differential evolution with locality induced genetic operators for dynamic optimization. *Eur J Oper Res* 253:337–355
- Ouyang H, Gao L, Kong X (2013) Random mutation differential evolution algorithm. *J Northeast Univ* 34:330–333 (in Chinese)
- Pan QK, Suganthan PN, Wang L, Gao L, Mallipeddi R (2011) A differential evolution algorithm with self-adapting strategy and control parameters. *Comput Oper Res* 38:394–408
- Plagianakos V, Tasoulis D, Vrahatis M (2008) A review of major application areas of differential evolution. In: Chakraborty U (ed) *Advances in differential evolution*. Springer, Berlin, pp 197–238
- Price KV, Storn RM, Lampinen JA (2005) *Differential evolution: a practical approach to global optimization*. Springer, Berlin
- Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput* 13:398–417
- Qiu X, Jiang Y, Li B (2015) Fractal mutation factor correcting differential evolution algorithm. *Pattern Recognit Artif Intell* 28:132–138 (in Chinese)
- Ronkkonen J, Kukkonen S, Price KV (2005) Real-parameter optimization with differential evolution. In: Proceedings of the IEEE congress on evolutionary computation. IEEE Press, pp 506–513
- Shang R, Wang Y, Wang J, Jiao L, Wang S, Qi L (2014) A multi-population cooperative coevolutionary algorithm for multi-objective capacitated arc routing problem. *Inform Sci* 277:609–642

- Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 11:341–359
- Suganthan PN, Hansen N, Liang JJ, Deb K, Chen Y-P, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, KanGAL report, 2005005
- Wang J, Cai Y (2015) Multiobjective evolutionary algorithm for frequency assignment problem in satellite communications. *Soft Comput* 19:1229–1253
- Wang Y, Cai Z, Zhang Q (2011a) Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Trans Evol Comput* 15:55–66
- Wang Y, Cai Z, Zhang Q (2011b) Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Trans Evol Comput* 15:55–66
- Weber M, Neri F, Tirronen V (2011) Shuffle or update parallel differential evolution for large-scale optimization. *Soft Comput* 15:2089–2107
- Wu G, Mallipeddi R, Suganthan PN, Wang R, Chen H (2016) Differential evolution with multi-population based ensemble of mutation strategies. *Inform Sci* 329:329–345
- Yongjie MA, Yulong BAI, Zhaoyuan JIANG (2009) Fast multi-objective constrained evolutionary algorithm and its convergence. *Syst Eng-Theory Pract* 29:149–157
- Zaharie D (2003) Control of population diversity and adaptation in differential evolution algorithms. In: Proceedings of the 9th international conference on soft computing, Brno, pp 41–46
- Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. *IEEE Trans Evol Comput* 13:945–958
- Zhao J, Xu Y, Luo F, Dong Z, Peng Y (2014) Power system fault diagnosis based on history driven differential evolution and stochastic time domain simulation. *Inform Sci* 275:13–29
- Zhao Z, Yang J, Hu Z, Che H (2016) A differential evolution algorithm with self-adaptive strategy and control parameters based on symmetric Latin hypercube design for unconstrained optimization problems. *Eur J Oper Res* 250:30–45
- Zielinski K, Weitkemper P, Laur R, Kammeyer KD (2006) Parameter study for differential evolution using a power allocation problem including interference cancellation. In: IEEE congress on evolutionary computation, pp 1857–1864
- Zou D, Wu J, Gao L, Li S (2013) A modified differential evolution algorithm for unconstrained optimization problems. *Neurocomputing* 120:469–481

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.