Natural Computing (2020) 19:683–698
https://doi.org/10.1007/s11047-018-9692-z

A new multi-stage perturbed differential evolution with multi-parameter adaption and directional difference

Guangzhi Xu[1,2] · Rui Li[1,2] · Junling Hao[3] · Xinchao Zhao[2] · Ying Tan[4]

Published online: 20 June 2018

Abstract
A new multi-stage perturbed differential evolution (MPDE) is proposed in this paper. A new mutation strategy "multi-stage perturbation" is implemented with directivity difference information strategy and multiple parameters adaption. The DE/current-to-*p*best is introduced to increase the population diversity while remaining its elitist learning behavior in this architecture. The multi-stage perturbation-based mutation operation utilizes the Normal random distribution with adjustable variance to perturb the chosen solutions. Multiple parameters are adaptively adjusted to appropriate values to match the current search status of algorithm. It is thus helpful to enhance the performance and the robustness of algorithm. Simulation results show that the newly proposed MPDE is better than, or at least comparable to CLPSO, SPSO2011, NGHS, jDE, CoDE, SaDE and JADE algorithms in terms of optimization performance based on CEC2015 benchmark function.

Keywords Multi-stage perturbation · Multiple parameters adaption · Directional difference · Differential evolution · Evolutionary optimization

1 Introduction

Differential Evolution (DE) is a stochastic search method, which has emerged as one of the most competitive evolutionary computing algorithms and has been successfully applied to solve many numerous real-world problems from diverse domains of science and technology (Das and Suganthan 2011; Zhao et al. 2013). As other evolutionary algorithms such as PSO (Zhao et al. 2016a, b; Zhao et al. 2014; Khan et al. 2016), the performance of DE depends mainly on mutation strategy and the parameters setting. An effective mutation strategy is used for increasing the individual generation and explore the search space. In many cases main control parameters of DE algorithms (population size, scaling factor F, and crossover rate CR) also have important influence on the performance of DE (Yu et al. 2014). As we know, appropriate settings of these parameters perform well with a balanced exploratory and exploitation capability to avoid being trapped in local optimum (Das et al. 2016; Shukla et al. 2017; Suresh and Lal 2017). Therefore, it is necessary to adopt an effective mutation strategy and a well matched control parameters for the desired results. There are already many existing classical operations for three main control parameters and their changing mechanisms for a better performance. However, it is not sufficient for multi-modal problems during different stages of search process. For example, the diversity is more important than the convergence speed in the first half search stage and vice versa. So it is necessary to propose the multi-stage perturbed strategy and the multi-parameter adaption mechanism.

It is of interest to remain the large proportion of population diversity in the first half process to enhance the exploring ability for the swarm optimization algorithms. On the contrary, accelerating the convergence speed of the

✉ Xinchao Zhao
xcbupt@126.com

1 Automation School, Beijing University of Posts and Telecommunications, Beijing 100876, China

2 School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

3 School of Statistics, University of International, Business and Economics, Beijing 10029, China

4 School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China

algorithm is necessary in the latter process to improve the exploitation ability of swarm intelligence. In the middle search stage of algorithm, exploration and exploitation should be considered simultaneously. The improved DE variants with changing population diversity have shown more efficient and more reliable convergence performance than the classic DE algorithms (Qin and Suganthan 2005; Brest et al. 2006, 2007; Mohamed 2017; Zhou et al. 2016). Several adaptive algorithms (Teo 2006; Cui et al. 2016; Fan and Yan 2016; Wu et al. 2016a, b; Ali et al. 2017) are developed based on the classic DE/rand/1 which is known to be the most widely considered DE mutation strategy. Others simultaneously implement DE/rand/1 and one or more greedy DE mutations, such as DE/current-to-best/1, to dynamically update their probabilities of being used to generate offspring (Yi et al. 2016; Zhao et al. 2016a, b; Wang and Tang 2016; Cui et al. 2018; Huang et al. 2006, Brest et al. 2006). Until now, there are also some difficulties to develop a greedy DE variant (e.g. DE/current-to-best/1 and DE/best/1) that utilizes the elitist information of the best solution(s) in the current population. The reason seems to be straightforward: a greedy variant is usually less reliable and may lead to premature convergence, especially for multimodal problems (Mendes et al. 2006). However, the elitist information of the best individuals is crucial to algorithm. How to deal with this dilemma? In order to develop the currently best solution-based mutation strategy, an improved current-to-*p*best strategy is proposed by Yang et al. (2008) and Zhang and Sanderson (2009).

In view of the above considerations, a new mutation strategy, "DE/current-to-*p*-*p*best" with multi-stage perturbation, directivity difference strategy and multiple parameters adaption are presented to dynamically adjust the population diversity on the basis of the elitism learning. As a generalization of DE/current-to-best, DE/current- to-*p*best utilizes the information of both the current best solution and the other excellent solutions (Zhang and Sanderson 2009). Classic DE is incapable of adjusting the balance of exploratory and exploitation freely with one scaling factor F only in different search stages. Therefore, multiple parameters adaption strategy with two decoupled scaling factors is designed for precisely balancing of exploration and exploitation, in which one parameter mainly adjusts the population diversity and the other one mainly adjusts the convergence speed to adapt the search stage.

To be specific, any of the top 100p% solutions, $p \in (0, 1]$, can be chosen in DE/current-to-*p*best strategy to be a possible alternative choice besides the best solution in traditional DE/current-to-best. In addition, the random selected *p*best solution is perturbed with a normal random distribution with different variance in different search stages of algorithm, which aims to diversify the elitist neighborhood slightly and to decrease its greedy property simultaneously. The proposed multi parameters adaption strategy is also able to diversify the population during the search process so that the problems such as premature convergence can be alleviated.

The rest of the paper is organized as follows. In Sect. 2, the basic DE operations are presented. The related works on typical DE variants are described in Sect. 3. Section 4 proposes the new algorithm (MPDE) using multi-stage perturbation, directivity difference mechanism and multi parameters adaption. Section V firstly considers the effect of the parameters cooperation on algorithm. Then comprehensive empirical evaluation of MPDE is conducted with several state-of-the-art DE variants and other evolutionary algorithms based on IEEE CEC2015 benchmarks. Section 6 concludes the paper with a analytic discussion.

2 Differential evolution

This section briefly describes the fundamental operations of DE algorithm (Wu et al. 2016a, b). Similar to other evolutionary algorithms, the solution space of DE algorithm contains three elements: dimensionality, individual and population. The individual vector $X_i = [X_1, \ldots, X_D]$, $(i = 1, 2, \ldots, N)$ represents a candidate solution vector, where D is the dimensionality of the target problem, and N is the population size. At the beginning of the search, the individual vectors are usually initialized in the search space randomly. After initialization, DE enters a loop of evolutionary operations: mutation, crossover, and selection which drive the candidate solutions towards more and more competitive solutions.

2.1 Main operations of DE

At each generation G, DE generates a mutant vector for each individual X_i^G (called a target vector) in the current population.

$$V_i^G = \left[V_{i,1}^G, V_{i,1}^G, \ldots, V_{i,D}^G\right] \quad i = 1, 2, \cdots, N \tag{1}$$

Several widely used DE mutation strategies are shown as follows.

- DE/rand/1

$$V_{\mathrm{i}} = X_{\mathrm{r1}} + F(X_{\mathrm{r2}} - X_{\mathrm{r3}}) \tag{2}$$

- DE/rand/2

$$V_{\mathrm{i}} = X_{r1} + F(X_{r2} - X_{r3}) + F(X_{r4} - X_{r5}) \tag{3}$$

- DE/best/1

$$V_i = X_{best} + F(X_{r2} - X_{r3}) \tag{4}$$

- DE/current-to-best/1

$$V_i = X_i + F(X_{best} - X_i) + F(X_{r2} - X_{r3}) \quad (5)$$

- DE/rand-to-best/1

$$V_{\mathrm{i}} = X_{r1} + F(X_{best} - X_{r3}) + F(X_{r4} - X_{r5}) \quad (6)$$

The indices r_1, …, r_5 are randomly selected from [1, N] such that they differ from each other as well as i; X_{best} is the best individual in population. Parameter F = (0, 1] controls the magnitude of the differential mutation operation.

Binomial Crossover, the most commonly used crossover operator in DE, is implemented as follows:

$$U_{\mathrm{i,j}}^G = \begin{cases} V_{i,j}^G, & rand \le CR \; or \quad j = j_{rand} \\ X_{i,j}^G, & otherwise \end{cases} \quad (7)$$

where $rand$ denotes a uniformly selected random number in [0 1], and j_{rand} represents a random integer in [1, N], which ensures that the trial vector gets at least one component from the mutant vector. CR is the crossover probability, which controls the portion of mutant vector that are copied to the trial vector in process.

After all of the trial vectors U_i^G $(0 \le i \le N)$ have been generated, a selection process determines the survivors for the next generation. The fitness of each trial vector is compared with its corresponding target vector in the current population. Finally, the vector with better fitness enters the next generation. The above three basic steps are repeated until some specific termination criteria are met and a final candidate solution is obtained.

3 Related works

DE has undergone a significant progress since it is presented. A large number of DE variants have been proposed to improve the performance of traditional DE. There have been many research works which pay attention to self-adapt the selection of the mutation strategy, as well as the control parameter values (Zhang et al. 2017; Zhou et al. 2017). In this part, we briefly review several state-of-the-art DE variants.

3.1 jDE

jDE is based on the classic DE/rand/1/bin proposed by Brest et al. (2006). Similar to other schemes, jDE assigns a different set of parameter values F_i and CR_i to each X_i, which is associated with trial vectors. Initially, the parameters for all individuals i are set to F_i = 0.5, CR_i = 0.9. jDE modifies parameters for F_i and CR_i according to uniform distributions on pre-specified range, respectively. It is believed that better parameters tend to generate more competitive individuals to survive and thus these parameters should be kept as the beneficial heuristic information for the next generation.

3.2 SaDE

Qin and Suganthan (2005) proposed SaDE with two mutation strategies "DE/rand/1" and "DE/current-to-best/1." SaDE uses a memory of past search behaviors in order to self-adapt the crossover strategies and parameters. In each generation, the value of F for each individual X_i is randomly assigned by a normal distribution randn(0.5 0.3), and does not adapt during the search. In contrast, the crossover probabilities are randomly generated according to an independent normal distribution randn(CR_{mk} 0.1). On the contrary of F_i, CR_i values remain fixed to adapt to proper values for five generations before updating.

3.3 JADE

JADE (Zhang and Sanderson 2009) is a very competitive DE variant which employs well-adapted parameters. In addition to parameter adaptation, JADE also uses a novel mutation strategy called current-to-*p*best/1 and an external archive for storing previously generated individuals. Instead of a static crossover rate CR and scaling factor F, JADE has two corresponding, adaptive variables, μCR and μF. The crossover rate and scaling factor associated with each individual are generated according to a normal/Cauchy distribution with means μCR and μF. At the end of each generation, the values of μCR and μF are updated according to the CR and F pairs in those generations of the successful trial vectors being obtained. During the search progress, μCR and μF are possible to gradually adapt to the most competitive parameters setup.

3.4 CoDE

CoDE (Wang et al. 2011), another competitive self-adaptive DE variant, does not adapt its parameters setting, but randomly selects parameters from a predefined set. More specifically, at each generation, CoDE randomly combines 3 hand-selected mutation strategies (rand/1/bin, rand/2/bin, current-to-rand/1) with 3 hand-selected [F; CR] pairs ([1:0; 0:1], [1:0; 0:9], [0:8; 0:2]). Three trial vectors are generated for each individual and the best one remained.

3.5 EPSDE

EPSDE (Mallipeddi et al. 2011) takes advantage of 3 separate pools to achieve self-adaptation, including the mutation strategy, parameters F, and CR. The mutation strategy pool contains rand/1/bin, best/2/bin and current-to-rand/1. The parameter F pool stores the values between [0.4; 0.9] in 0.1 increment, and the CR pool includes the values between [0.1; 0.9] in 0.1 increment. At the beginning of search, each individual is randomly assigned values from F and CR pools and selected a mutation strategy from the strategies pool. During search, successful parameter sets that result better individuals are inherited and used in the next generations. On the contrary, those parameter sets that fail to obtain better offspring are reinitialized.

3.6 SHADE

Success-History based Adaptive DE (SHADE), proposed by Tanabe and Fukunaga (2013), is an enhancement to JADE which uses a successful history-based parameter adaptation scheme. Instead of generating new control parameters based on some distribution around a single pair of parameters μCR and μF, it uses a historical memory MCR and MF which stores a set of CR and F values with better performance in the past. Then new CR and F pairs are generated by directly sampling from the parameter space of those stored pairs.

4 Multi-stage perturbed DE

In this section, mutation strategy "DE/current-to-$p$best" is modified with multi-stage perturbation, directivity difference strategy and multiple parameters adaption.

4.1 DE/Current-to-*p*best

The mutation strategy DE/current-to-$p$best aims to keep the fast convergence but less premature (Zhang and Sanderson 2009). Compared with mutation strategies DE/rand, greedy strategies, such as DE/current-to-best and DE/best, benefit from their fast convergence by incorporating best solution information during the evolutionary search. However, excessive best solution information may also cause such problems as premature convergence due to the resultant reduced population diversity.

In DE/current-to-$p$best/1, a mutant vector is generated in the following manner:

$$V_i^G = X_i^G + F\left(X_{pbest}^G - X_i^G\right) + F\left(X_{r1}^G - X_{r2}^G\right) \tag{8}$$

where X_{pbest}^G is randomly chosen from one of the top 100p% individuals in the current population with $p \in (0, 1]$, and F is the scale factor that is associated with FES. It is regenerated according to the searching process requirement introduced later in Section C. DE/current-to-$p$best is a generalization of DE/current-to-best. Any of the top 100p % solutions can be randomly chosen to play the role of the single best solution in DE/current-to-best.

4.2 Multi-stage perturbed and directivity difference strategy

The multi-stage perturbed and directivity difference strategy operations are simply designed to avoid the significant premature convergence due to the resultant reducing population diversity. The new mutation strategy is shown as follows.

$$p{-}pbest = N(pbest, \sigma) \tag{9}$$

$$V_i^G = X_i^G + F\left(X_{p-pbest}^G - X_i^G\right) + F\left(X_{r1}^G - X_{r2}^G\right) \tag{10}$$

where σ is a function of iteration/generation number G. It is used to measure the perturbing uncertainty around the *pbest* individual. The following formula describes the update model of σ:

Algorithm I: Directivity difference vector construction
$temp1 = X_{r1};\quad temp2 = X_{r2};$
If $f(temp1) > f(temp2)$
$\quad temp = temp1; temp1 = temp2; temp2 = temp;$
End If
If $G \geq 0.8 \cdot \max G$
$\quad V_i^G = X_i^G + F\cdot(X_{p-pbest}^G - X_i^G) + F\cdot(temp1 - temp2)$
End If

$$\sigma = \begin{cases} \sigma_1, & G < \alpha_1 \cdot \max G \\ \sigma_2, & \alpha_1 \cdot \max G < G < \alpha_2 \cdot \max G \\ \sigma_3, & G > \alpha_2 \cdot \max G \end{cases} \tag{11}$$

where $\sigma_1 > \sigma_2 > \sigma_3$ are the variance parameters of the Gaussian disturbance amplitude. $\alpha_1 < \alpha_2$ are the parameters of controlling radius. Parameter *maxG* represents the maximal generation number. Normal random distribution is more suitable for the exploitation stage owing to its narrow sampling range, while the introduced parameter σ suffices to balance exploitation and exploration. At the early process of offspring generation, large value of σ will promote the individuals to perturb around the *pbest* for exploration with a larger range. This operation provides a simple and efficient parallel search on multiple path. When evolution process arrives at the middle stage, the perturbed strategy will ensure algorithm search around a smaller neighborhood of *pbest* with small value σ, making the current solution almost no longer jump out from a promising area. At the end evolution stage σ is set very small value (tends to zero) to make the current solution learn from the *pbest* and exploration at a very local neighborhood. It is probably to guarantee the algorithm a good convergence.

Difference vector ($X_{r1}^G - X_{r2}^G$) is the core idea strategy of DE (Cui et al. 2016; Fan and Yan 2016), and is the most widely used scheme in the literature (Zhang and Sanderson 2009). However, the directivity of difference vector between X_{r1}^G and X_{r2}^G doesn't explicitly indicate. In fact, the directivity difference between X_{r1}^G and X_{r2}^G does have important effects on algorithm in theory own to the potential learning mechanism from the better examples in a certain range. It is well-known that difference vector ($X_{r1}^G - X_{r2}^G$) is good at strengthening the population diversity. However, the beneficial heuristic information from the better solution needs to draw more attention, especially at the last evolutionary search stage. Based on the analysis, it is essential to introduce a directivity difference vector based mutation strategy. Firstly, the fitness values of X_{r1}^G and X_{r2}^G are compared and then the difference vector is constructed whose direction is from the worse solution to the better one. This operation only occurs at the latter search stage according to Algorithm I.

4.3 Analysis of multi-parameter settings

As we know the performance of DE not only associates with three basic operations but also the control parameters. Choosing suitable control parameter values is usually a crucial task. In this section, a new parameter tuning approach is proposed. It can be observed that PID (proportion, integral, derivative) and DE have something in common (Moharam et al. 2016). The proportion part represents the current particle X_i^G. The integration in PID and the mutation vector ($X_{p\text{-}pbest}^G - X_i^G$) are equivalent which means the error reducing to minimum with time going. The derivative item in PID is used for predicting the change trend of error, which can restrain beyond the domain. However, the difference vector ($X_{r1}^G - X_{r2}^G$) of DE also predicts the search direction of solutions. In theory, population diversity should be paid more attention at the first half of search. But at the second half stage the directivity vector is more crucial to predict the optimal direction and weaken the effect of diversity. For this reason, Eq. (12) is used for mutation strategy. Two parameters F_1 and F_2 imitating PID control method are used to adjust the mutation operation, which is shown as follows.

$$V_i^G = X_i^G + F_1\left(X_{p-pbest}^G - X_i^G\right) + F_2\left(X_{r1}^G - X_{r2}^G\right) \tag{12}$$

F_1, F_2 and CR are calculated according to the following equations.

$$\begin{cases} F_1 = \lambda_1 - \varphi(G, \eta_1) \\ F_2 = \lambda_2 - \varphi(G, \eta_2) \\ CR = \lambda_3 - \varphi(G, \eta_3) \end{cases} \tag{13}$$

where $\varphi(G,\eta) = 1 - \eta^{\left(\frac{G}{\max G}\right)}$, η is used for adjusting the changing range of parameters. λ_i (i = 1,2,3) are the initial values of parameter F and CR. The difference vector ($X_{p\text{-}pbest}{}^G - X_i^G$) is described as a learning direction from the current solution to the perturbed *pbest* solution. The larger value of parameter F_1 contributes to the quicker convergence. The difference vector ($X_{r1}^G - X_{r2}^G$) is introduced for adapting the search trajectories diversity. The large value of F_2 benefits to the global exploration in the search space at the first stage of algorithm. On the contrary, with the evolution of algorithm going on, the value of F_2 is reduced by probability which promise the searching around the current relative better neighborhood. At the end stage of algorithm, the small value of F_2 can guarantee the current promising solutions to continue search around the competitive neighborhood. The algorithm can compromise the fast convergence rate and population diversity at the same time when tuning F_1 and F_2 cooperatively. Both of them are executed at the individual level. The appropriate choice of parameters and cooperation lead to more competitive individuals and even better performance of algorithm as a result. So, it is necessary to introduce an adaptive perturbation operation for the balanced exploration and exploitation from the multiple perturbation operation.

5 Experimental results and analysis

In this section, fifteen benchmark functions proposed in the CEC 2015 special session on real-parameter optimization were used to study the performance of the algorithms. A detailed description of the test instances can be found in Liang et al. (2014), which can be divided into four classes:

1. unimodal functions f_1–f_2;
2. simple multimodal functions f_3–f_5;
3. hybrid functions f_6–f_8;
4. composition functions f_9–f_{15}.

5.1 Parameter analysis of multi-parameter DE

Multi-stage perturbed strategy with multi-parameter adaption introduces two parameters λ and η to F_1, F_2 and CR which elaborate determine the greediness of between the mutation strategy and the population diversity. λ is the initial value and η controls the changing range of parameters which dynamically are adjusted with the algorithm going on. It is believed that both groups of parameters are

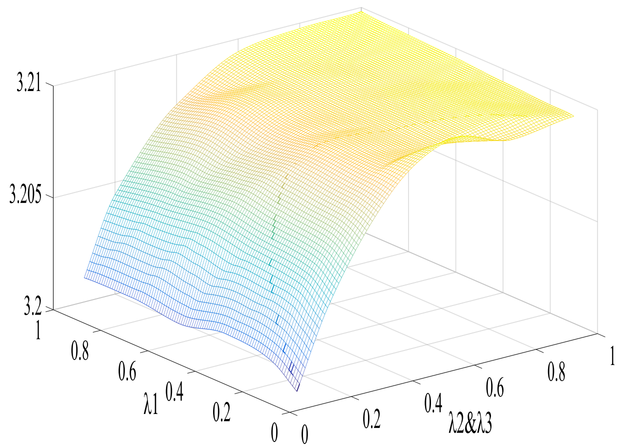

Fig. 1 Mean of the best-so-far function values of f3 with parameters combination

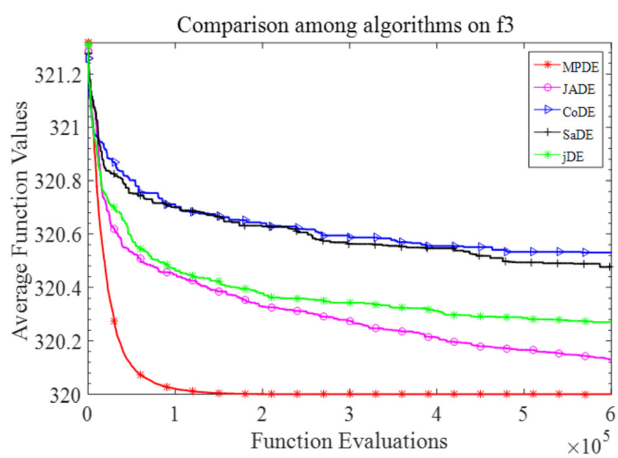

Fig. 2 Comparison results of five algorithms on f3

ingenious controlling the compromise of diversity and greediness according to their roles in multi-stage perturbation with multi-parameter adaption. It is thus an advantage over the only one parameter selection (F) in the classic DE. However, it is still interesting to find a reasonable range for these parameters which is appropriate for different problems (Fig. 1).

As shown in Fig. 1, the mean best values are plotted for multi-stage perturbed and multi-parameter DE with different parameters combinations: λ_1, λ_2,λ_3∈{0.1, 0.2, 0.4, 0.5, 0.7, 0.9, 1} with $\eta_{1,2,3}$ = 0.9. As expected, a small value of λ_1 (e.g., $\lambda_1 = 0.1 \sim 0.3$) or λ_2, λ_3 (e.g., λ_2, $\lambda_3 = 0.1 \sim 0.3$) may lead to less satisfactory results in some cases. The small value of λ_1 causes slow convergence due to the insufficient best solution information to influence the mutant vector, while small values of λ_2 and λ_3 are possible to result in the scarce of population diversity. On the other hand, large initial value of λ_1 is good at improving the convergence. In the same way large values of λ_2 and λ_3 enhance population diversity. It hopes that MPDE performs better than or competitive to other algorithms in a compromise range for λ_1 and λ_2, λ_3. Specifically, it shows that multi-stage perturbed DE with multi-parameter works best when $\lambda_1 = 0.81$ with $\eta_1 = 0.99$, λ_2, λ_3 = [0.75, 0.95] with $\eta_{2,3}$ = [0.25, 0.95] and $\sigma 1 = 0.0001$, $\sigma 2 = 0.000095$, $\sigma 3 = 0.000002$ after empirical analysis. For example, the function f_3 is a simple multimodal and non-separable function with only one peak valley. It doesn't need maintain too much greedy diversity of the population to find the best solution. As shown in Fig. 2 and Table 1 small values of λ_2, $\lambda_3 \in [0.01, 0.8 \sim 0.95]$ with $\eta_{2,3}$ = [0.99, 0.99] can achieve best solution superior to other four methods. Compared with jDE, it uses one parameter F only, so tuning parameter F in jDE may not be cooperative to improve the performance. These considered parameters scale the difference vectors and accordingly adjust the diversity of the search trajectories to some extent.

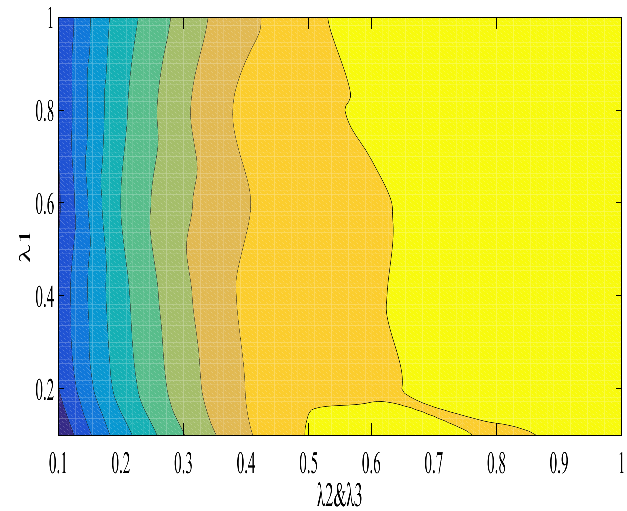

Fig. 3 Contour map of f3 with parameters variation

The effect of the initial values of F_2 and CR is investigated during the parameter tuning process. According to

Table 1 Comparison results on function f3

		MPDE	JADE	CoDE	SaDE	jDE
f3	Best	**3.2000E + 02**	3.2008E + 02	3.2040E + 02	3.2040E + 02	3.2021E + 02
	Mean	**3.2000E + 02**	3.2013E + 02	3.2052E + 02	3.2050E + 02	3.2026E + 02
	Std	**2.6593E − 05**	2.3645E − 02	4.5601E − 02	3.9075E − 02	2.4778E − 02

Table 2 Experimental results of MPDE, JADE, CoDE, SaDE, and jDE for CEC'15 benchmarks

	Item	MPDE	JADE	CoDE	SaDE	jDE
f1	Best	**1.0000E + 02**	1.0000E + 02	1.0185E + 02	3.8865E + 03	5.6309E + 02
	Mean	**1.0000E + 02**	1.0000E + 02	2.5714E + 02	3.7006E + 04	7.2428E + 03
	Std	**3.1481E − 01**	3.9564E − 01	2.3501E + 02	3.3851E + 04	8.2850E + 03
	W-test		≈	+	+	+
f2	Best	**2.0000E + 02**	2.0000E + 02	2.0000E + 02	2.0372E + 02	2.0000E + 02
	Mean	**2.0000E + 02**	2.0000E + 02	2.0000E + 02	1.4052E + 03	2.0000E + 02
	Std	9.0000E − 06	3.0583E − 14	1.8280E − 05	1.2662E + 03	6.5203E − 15
	W-test		≈	≈	+	≈
f3	Best	3.2019E + 02	**3.2008E + 02**	3.2040E + 02	3.2040E + 02	3.2021E + 02
	Mean	3.2053E + 02	**3.2013E + 02**	3.2052E + 02	3.2050E + 02	3.2026E + 02
	Std	3.5908E − 02	**2.3645E − 02**	4.5601E − 02	3.9075E − 02	2.4778E − 02
	W-test		–	≈	≈	–
f4	Best	**4.0895E + 02**	4.1525E + 02	4.7171E + 02	4.1790E + 02	4.2130E + 02
	Mean	**4.2049E + 02**	4.2162E + 02	5.0498E + 02	4.3054E + 02	4.2944E + 02
	Std	**3.0868E + 00**	4.1943E + 00	1.2764E + 01	5.8644E + 00	4.7115E + 00
	W-test		+	+	+	+
f5	Best	1.3266E + 03	**1.2634E + 03**	3.7496E + 03	3.0574E + 03	2.1805E + 03
	Mean	2.5381E + 03	**1.8929E + 03**	4.7614E + 03	3.7024E + 03	2.5162E + 03
	Std	3.4846E + 02	**2.4744E + 02**	3.4892E + 02	2.8152E + 02	2.1718E + 02
	W-test		**–**	+	+	–
f6	Best	**6.2309E + 02**	8.4604E + 02	8.1341E + 02	6.8086E + 02	6.7700E + 02
	Mean	**7.0704E + 02**	1.7058E + 03	1.4448E + 03	1.5588E + 03	1.1160E + 03
	Std	**7.7469E + 01**	4.4118E + 02	2.7397E + 02	5.8407E + 02	2.9419E + 02
	W-test		+	+	+	+
f7	Best	**7.0110E + 02**	7.0579E + 02	7.0631E + 02	7.0435E + 02	7.0595E + 02
	Mean	**7.0313E + 02**	7.0683E + 02	7.0846E + 02	7.0708E + 02	7.0677E + 02
	Std	**6.1828E − 01**	5.8256E − 01	1.2109E + 00	1.2245E + 00	4.8659E − 01
	W-test		+	+	+	+
f8	Best	**8.0170E + 02**	8.4659E + 02	8.3089E + 02	8.1363E + 02	8.1547E + 02
	Mean	**8.2735E + 02**	9.3794E + 03	1.0240E + 03	9.1250E + 02	8.9346E + 02
	Std	**2.9952E + 01**	2.7392E + 04	1.2837E + 02	7.4167E + 01	6.5809E + 01
	W-test		+	+	+	+
f9	Best	**1.0018E + 03**	1.0021E + 03	1.0032E + 03	1.0022E + 03	1.0022E + 03
	Mean	**1.0021E + 03**	1.0030E + 03	1.0035E + 03	1.0025E + 03	1.0025E + 03
	Std	**1.6344E − 01**	2.5411E − 01	1.7392E − 01	1.6386E − 01	1.6379E − 01
	W-test		+	+	≈	≈
f10	Best	**1.1601E + 03**	1.4972E + 04	1.1892E + 03	1.2627E + 03	1.4381E + 03
	Mean	**1.2984E + 03**	1.3602E + 04	1.3605E + 03	1.7324E + 03	1.6954E + 03
	Std	**1.0296E + 02**	3.8508E + 04	9.7925E + 01	5.3201E + 02	8.8426E + 01
	W-test		+	+	+	+
f11	Best	1.4002E + 03	1.4011E + 03	**1.4000E + 03**	1.4011E + 03	1.4007E + 03
	Mean	1.5202E + 03	1.5244E + 03	**1.4107E + 03**	1.5234E + 03	1.4554E + 03
	Std	4.2235E + 01	6.1728E + 01	**3.0786E + 01**	1.0274E + 02	5.0561E + 01
			+	–	+	–
f12	best	**1.2933E + 03**	1.3043E + 03	1.3068E + 03	1.3039E + 03	1.3045E + 03
	mean	**1.3033E + 03**	1.3059E + 03	1.3076E + 03	1.3048E + 03	1.3055E + 03
	std	**4.5728E − 01**	5.0511E − 01	6.0840E − 01	4.9849E − 01	6.0840E − 01
	W-test		+	+	+	+

Table 2 (continued)

	Item	MPDE	JADE	CoDE	SaDE	jDE
f13	Best	**1.3790E + 03**	1.3902E + 03	1.4073E + 03	1.4046E + 03	1.3878E + 03
	Mean	**1.3840E + 03**	1.3941E + 03	1.4119E + 03	1.4078E + 03	1.3950E + 03
	Std	2.5494E + 00	2.7899E + 00	2.0853E + 00	2.0000E + 00	2.7890E + 00
	W-test		+	+	+	+
f14	Best	**3.2479E + 04**	3.2580E + 04	**3.2479E + 04**	3.2976E + 04	3.2563E + 04
	Mean	3.3141E + 04	3.3378E + 04	**3.2586E + 04**	3.3583E + 04	3.3548E + 04
	Std	**3.0934E + 03**	8.7476E + 03	4.6641E + 03	5.5556E + 03	9.4051 E + 03
	W-test		+	–	+	+
f15	Best	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03
	Mean	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03
	Std	7.2803E − 09	0	3.4566E − 10	1.8069E − 13	0
	W-test		≈	≈	≈	≈
	+		10	10	12	9
	–		2	2	0	3
	≈		3	3	3	3

experimental studies, the initial value of F_2 has effect on the performance of the algorithm, and a moderate to large initial value is recommended for *CR*, especially for non-separable functions. For example, the success rate of JADE, CoDE, SaDE and jDE are less satisfactory when optimizing f_6, f_7 and f_8. It implies that they are incapable of maintaining sufficiently high population diversity due to the hybrid benchmark functions contain different properties for different variables subcomponents. Comparatively speaking, MPDE achieves remarkably better performance in terms of optimization and reliability. It indicates a mutually beneficial cooperation between the greedy strategy "multi-stage perturbed" and the multi-parameter tuning. In fact, a greedy mutation strategy may affect the population diversity in two opposite directions: it tends to decrease the diversity by moving individuals closer to a few best solutions. A greedy mutation strategy usually leads to premature convergence in classic DE algorithms without multi-parameter tuning, because the result of diversity decreasing of the former strategy plays the important role. However, the multi-parameter tuning scheme is able to tune different parameters to appropriate values and thus improve the optimizing progress along the promising direction. Thus, the effect of the latter strategy is capable of balancing the former strategy on diversity decreasing and the performance of the algorithm is improved (Fig. 3).

5.2 Comparison with the state-of-the-art DEs

In the experimental studies, MPDE is compared with four classical adaptive DE variants JADE, CoDE, SaDE and jDE. For fair comparison, for each algorithm and each test function, 30 independent runs are conducted with 600,000 function evaluations (FES) as the termination criterion. Dimension *D* is 30. The population size is 100, the parameters $p = 0.05$ for top 100% best individuals chosen in all simulations. For clarity, the best results of algorithms are marked boldface.

The mean and the standard deviation of the results obtained by each algorithm for f_1–f_{15} are summarized in Table 2. The '+', '−' *and* '≈' indicate whether MPDE performs significantly better (+), significantly worse (−), or comparably (≈) when comparing with the competitors according to the Wilcoxon rank-sum test (significance level $\alpha = 0.05$). For convenience of illustration, the evolutionary convergence curves of the mean best values for f_1–f_{15} are plotted in Fig. 4, because these curves provide more online information than best and media values. It is helpful to illustrate the spread of results over 30 independent runs.

It can be observed from Fig. 4 and Table 2 that the best solutions have high convergence rate and robust reliability. First, these results suggest that the ability of finding the best solutions of the presented MPDE ranks the first or the second highest for most of the CEC2015 benchmarks. f_3 is a simple multimodal and non-separable function with only one peak valley and all four methods almost obtain the same final results. The parameters F_1, F_2 and *CR* are more compromise for obtaining well performance for all different functions. It is clear that parameter F_1 being 0.8 is not sufficient for learning from the best particle. At the same time parameters *CR* and F_2 afford much more diversity which affect the particle's search behaviors. The similar phenomenon occurs to f_{11} for the performance of

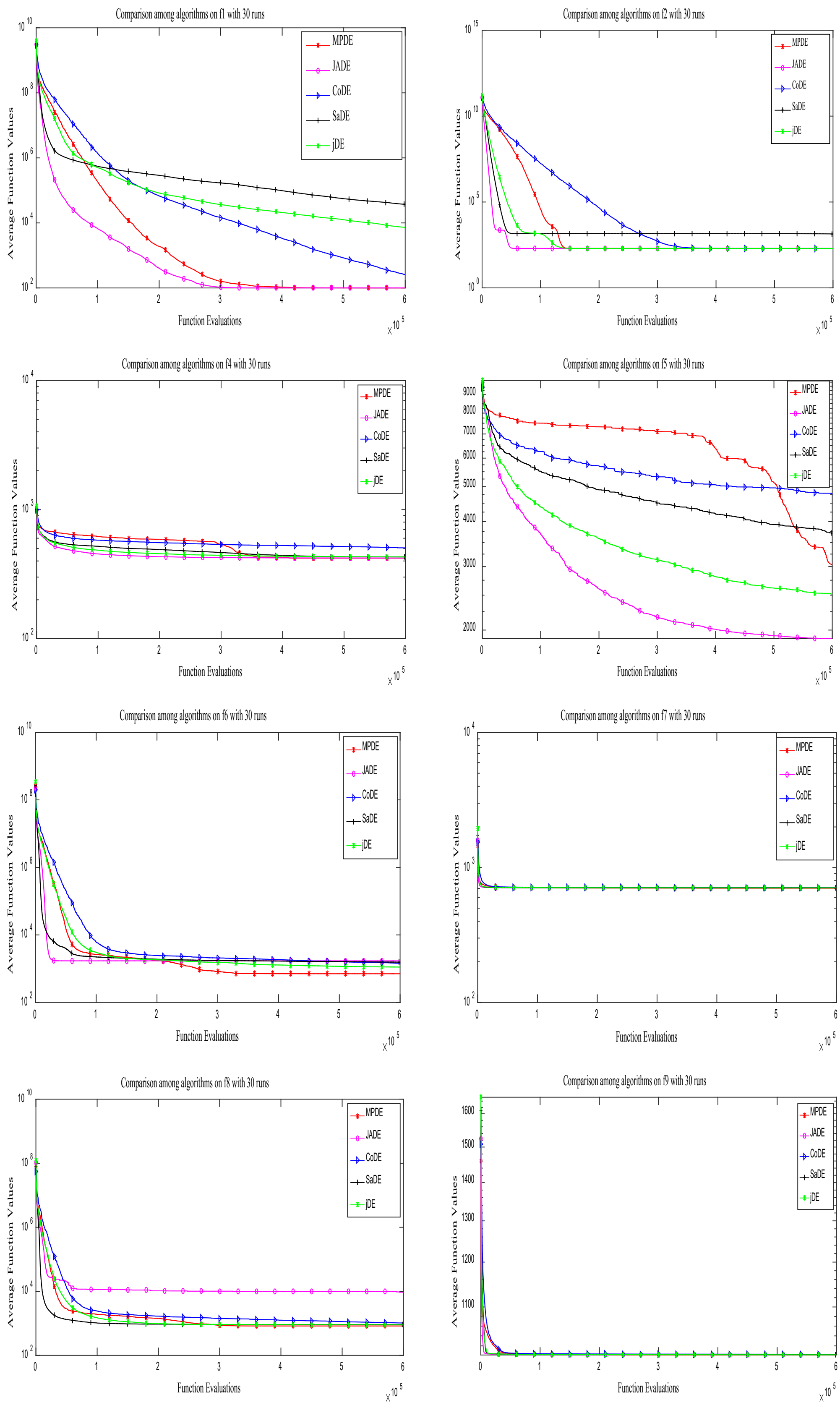

Fig. 4 Evolutionary performance comparison among state-of-the-art DEs for CEC'15 benchmarks

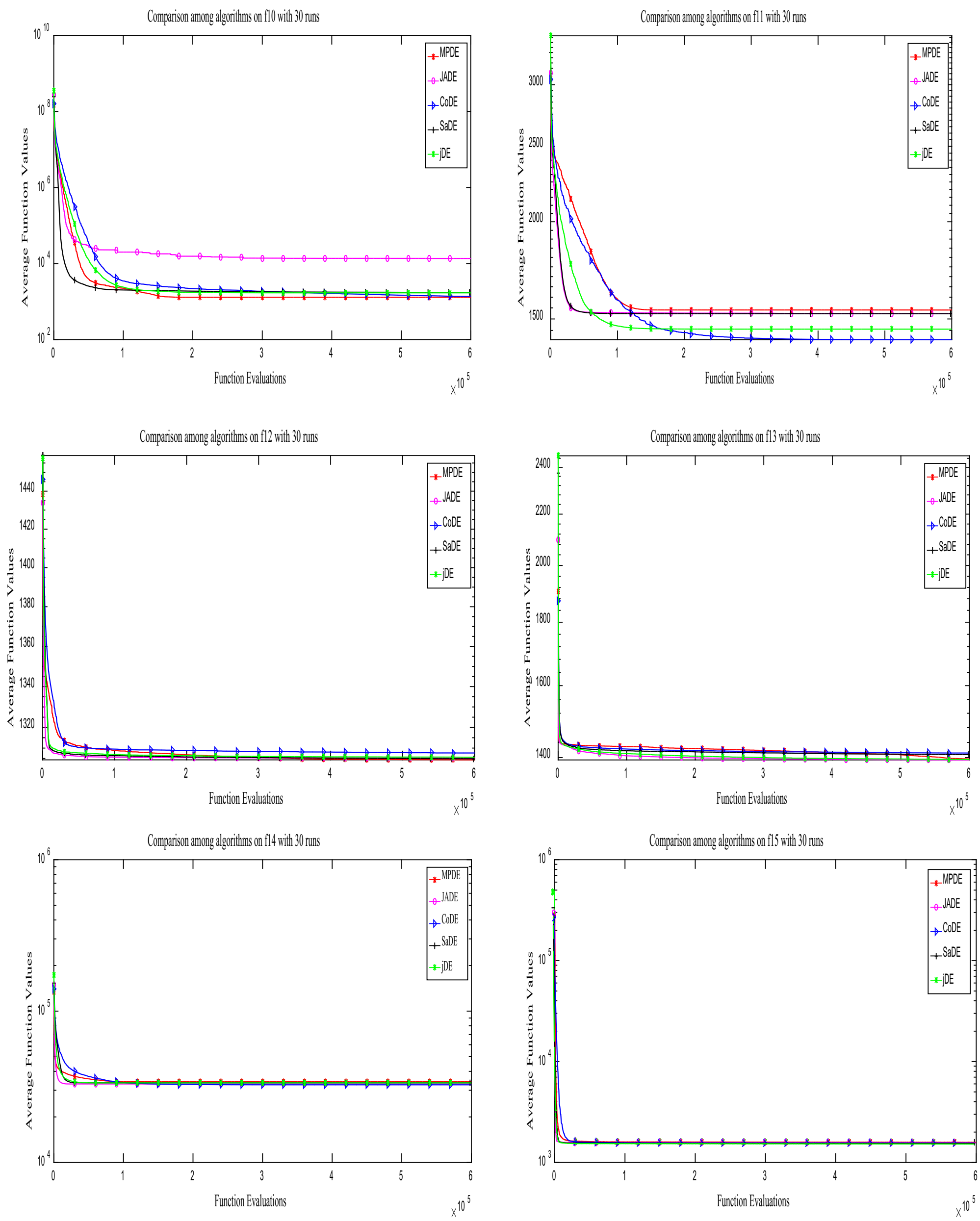

Fig. 4 continued

algorithms. Multi-stage perturbed multi-parameter MPDE has almost the best or similar convergence rate for most of the functions, which are either unimodal or multimodal, hybrid or composition. Most of all benchmarks, especially for f_6–f_{10} and f_{12}, f_{13}, MPDE achieves significant dominance to its competitors. The performance reliability and robustness of five algorithms are almost the same in functions f_{15}.

It can be seen that MPDE is better than other four algorithms on hybrid multi-modal and composition functions with a huge of local optima. The high reliability and optimization ability of MPDE stems from multiple parameters adaption, guided direction information utilization and the proposed diversity maintaining strategy, i.e., multi-stage perturbed mutation operation. MPDE works well on unimodal functions f_1, f_2 and multi-modal functions f_4, f_6 because its underlying mutation strategy is compromise between diversity and convergence rate. The other algorithms are worse than MPDE and JADE in terms of the best solution for function f_1. It is clear that MPDE works best or second best in most cases and achieves overall better performance than other state-of-the-art algorithms.

Table 3 Experimental comparison among MPDE, CLPSO, SPSO2011, and NGHS for CEC'15 benchmarks

		MPDE	CLPSO	SPSO2011	NGHS
f1	Best	**1.0000E + 02**	1.6193E + 06	1.0861E + 05	1.6543E + 03
	Mean	**1.0000E + 02**	3.6271E + 06	2.4201E + 05	1.5642E + 04
	Std	**3.1481E − 01**	9.2366E + 05	9.4765E + 04	9.9849E + 03
	W-test		+	+	+
f2	Best	**2.0000E + 02**	2.0760E + 02	2.0113E + 02	2.2087E + 02
	Mean	**2.0000E + 02**	2.3984E + 02	1.1305E + 03	3.9766E + 03
	Std	9.0000E − 06	2.8068E + 01	8.3061E + 02	3.6144E + 03
	W-test		+	+	+
f3	Best	3.2083E + 02	3.2032E + 02	3.2023E + 02	**3.1999E + 02**
	Mean	3.2060E + 02	3.2042E + 02	3.2031E + 02	**3.2000E + 02**
	Std	3.8908E − 02	3.8800E − 02	5.9500E − 02	**2.4507E − 05**
	W-test		–	–	–
f4	Best	**4.0895E + 02**	4.3082E + 02	4.1989E + 02	4.7064E + 02
	Mean	**4.2049E + 02**	4.5140E + 02	4.3389E + 02	5.0531E + 02
	Std	**3.0868E + 00**	7.7342E + 00	8.7272E + 00	2.3417E + 01
	W-test		+	+	+
f5	Best	**1.3266E + 03**	2.5358E + 03	2.5298E + 03	2.3393E + 03
	Mean	**2.6381E + 03**	2.9992E + 03	3.3383E + 03	3.1718E + 03
	Std	5.4846E + 02	**2.5181E + 02**	4.2438E + 02	4.4570E + 02
	W-test		+	+	+
f6	Best	**6.2309E + 02**	1.9148E + 05	7.8790E + 03	3.3375E + 03
	Mean	**7.0704E + 02**	6.8612E + 05	1.8182E + 04	1.3059E + 04
	Std	**7.7469E + 01**	2.9810E + 05	6.7713E + 03	1.1246E + 04
	W-test		+	+	+
f7	Best	**7.0016E + 02**	7.0600E + 02	7.1312E + 02	7.0404E + 02
	Mean	**7.0213E + 02**	7.0770E + 02	7.1600E + 02	7.0890E + 02
	Std	**6.1828E − 01**	1.0027E + 00	1.8136E + 00	4.6207E + 00
	W-test		+	+	+
f8	Best	**8.0170E + 02**	4.9942E + 04	7.3921E + 03	1.6416E + 03
	Mean	**8.2735E + 02**	1.0616E + 05	1.6019E + 04	9.5464E + 03
	Std	**2.9952E + 01**	3.8128E + 04	5.2903E + 03	1.0136E + 04
	W-test		+	+	+
f9	Best	**1.0008E + 03**	1.0030E + 03	1.0022E + 03	1.0033E + 03
	Mean	**1.0021E + 03**	1.0041E + 03	1.0035E + 03	1.0130E + 03
	Std	**1.0344E − 00**	1.2771E + 00	1.7350 E + 00	4.1749E + 01
	W-test		+	+	+
f10	Best	**1.1601E + 03**	7.4259E + 04	1.3866E + 04	2.6403E + 03
	Mean	**1.2984E + 03**	2.1067E + 05	3.5787E + 04	9.8549E + 03
	Std	**1.0296E + 02**	7.3237E + 04	1.3210E + 04	5.0305E + 03
	W-test		+	+	+
f11	Best	**1.4005E + 03**	1.4185E + 03	1.4030E + 03	1.4016E + 03
	Mean	1.5102E + 03	**1.4298E + 03**	1.7170E + 03	1.8350E + 03
	Std	**5.2235E + 01**	7.7713 E + 01	8.9309E + 01	2.3330E + 02
	W-test		–	+	+
f12	Best	1.3033E + 03	1.3054E + 03	**1.3030E + 03**	1.3077E + 03
	Mean	1.3043E + 03	1.3065E + 03	**1.3036E + 03**	1.3112E + 03
	Std	4.5728E − 01	4.9172E − 01	**3.7731 E-01**	1.8713E − 00
	W-test		+	–	+

Table 3 (continued)

		MPDE	CLPSO	SPSO2011	NGHS
f13	Best	**1.3810E + 03**	1.4002E + 03	1.3975E + 03	1.3868E + 03
	Mean	**1.3940E + 03**	1.4050E + 03	1.4049E + 03	1.4008E + 03
	Std	4.5494E + 00	**1.9945E + 00**	5.0973E + 00	6.8410E + 00
	W-test		+	+	+
f14	Best	**3.2479E + 04**	3.2803E + 04	3.2854E + 04	3.2906E + 04
	Mean	3.3141E + 04	**3.3093E + 04**	3.4527E + 04	3.5095E + 04
	Std	3.0934E + 03	**2.8709E + 02**	9.4692E + 02	1.5148E + 03
	W-test		–	+	+
f15	Best	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03
	Mean	1.6000E + 03	1.6000E + 03	1.6000E + 03	1.6000E + 03
	Std	7.2803E − 09	1.1251E − 10	0	5.7616E − 13
	W-test		≈	≈	≈
	+		11	12	13
	–		3	2	1
	≈		1	1	1

Generally speaking, MPDE performs significantly better, worse than or comparably with JADE and CoDE on 10, 2 and 3 functions. MPDE performs significantly better, worse than or comparably with SaDE on 12, 0 and 3 functions. MPDE performs significantly better, worse than or comparably with jDE on 9, 3 and 3 functions.

5.3 Comparison with other evolutionary algorithms

Besides the above comparison with state-of-the-art DEs, MPDE is also compared with other evolutionary algorithms, namely, CLPSO (Liang et al. 2006), SPSO (Zambrano-Bigiarini et al. 2013), and NGSH (Zou et al. 2010). In CLPSO, particles are possible to learn from all the personal best historical information according to some probability distribution to update the flying velocity. SPSO is a very efficient and well-known PSO milestone. NGHS is an efficient fast convergence algorithm which combines the smallest harmonic component and the optimal harmonic component dimension by dimension for generating the new harmonic vector. In the experiments, parameters of MPDE are the same as the above experiments. Parameters but FES of other algorithms refer to the relevant references for fair comparison. Table 3 and Fig. 5 summarize the experimental results.

Overall, MPDE significantly outperforms CLPSO, SPSO2011, and NGHS. MPDE performs better than CLPSO, SPSO2011, and NGHS on f_1, f_2, f_4, f_5, f_6, f_7 f_8, f_9, f_{10}, f_{13}, respectively. CLPSO beats MPDE on two test functions f_3 and f_{11}. SPSO2011 performs better than MPDE on f_3 and f_{12}, and NGHS outperforms MPDE on one function f_3 only.

6 Conclusion and future work

In this paper, a "multi-stage perturbed DE/current-top-pbest" strategy is proposed and the neighborhood information of the best solutions is more effectively utilized. It aims to reduce the greediness with a random perturbation operation while elitist learning is remained. The multi-parameters adaption is implemented by evolving the mutation factors and crossover probabilities according to the optimal process execution. A directivity difference strategy is also utilized as a guided direction toward the better individual.

Simulation results show that the newly proposed MPDE algorithm is better than, or at least comparable to CLPSO, SPSO2011, NGHS, jDE, CoDE, SaDE and JADE algorithms on most of CEC2015 benchmarks in terms of optimization performance.

It is expected that MPDE has the potential to serve as an appropriate underlying scheme in which large scale optimization (Cao et al. 2017) and multi-objective problem are considered (Shang et al. 2012, 2016). In addition, it is interesting to extend the multi-stage perturbation, multi-parameter adaption and directivity difference strategies to adaptive schemes. As far as these strategies, there are still many open questions in incorporating these schemes to other optimization problems.

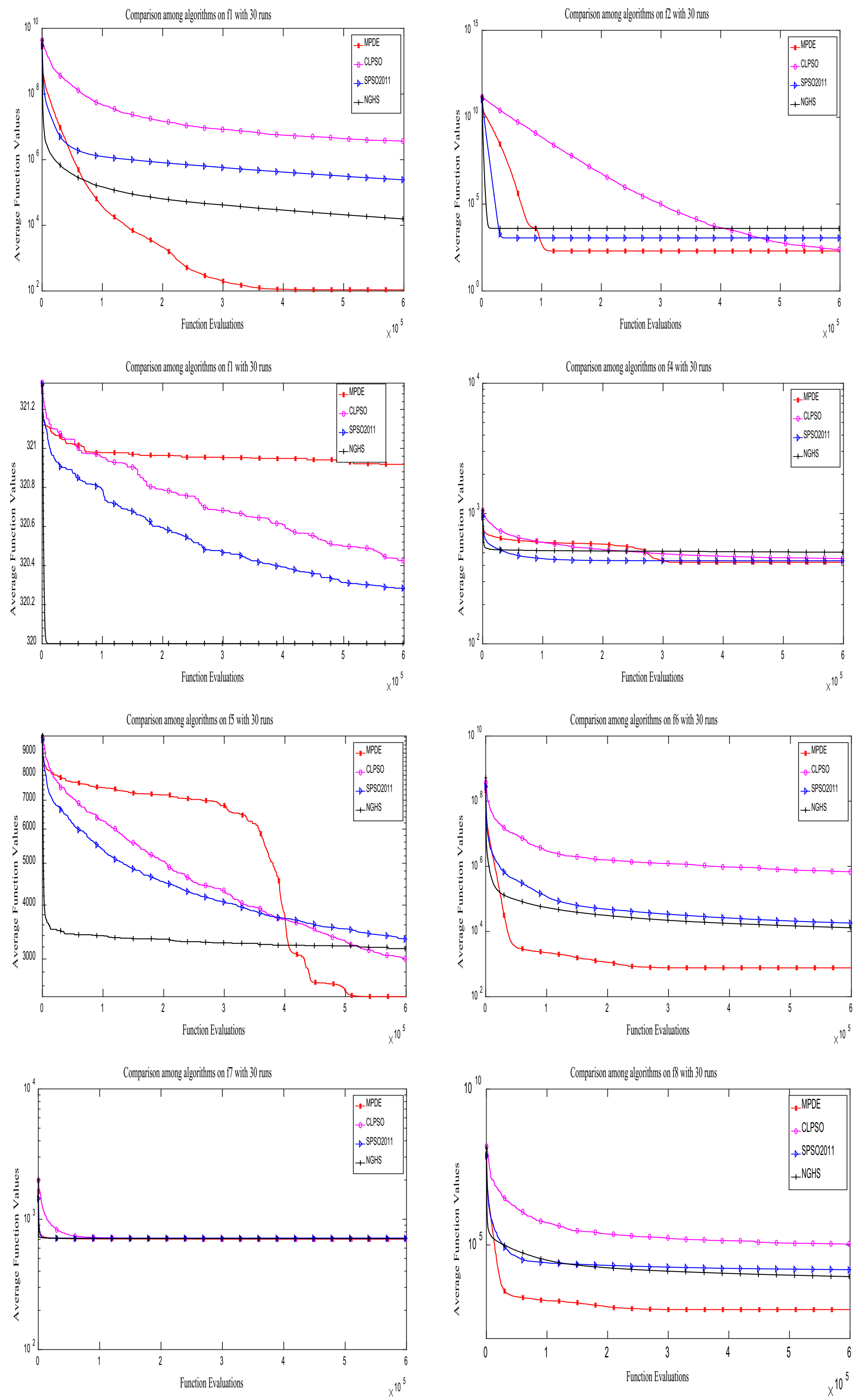

Fig. 5 Evolutionary performance comparison among evolutionary algorithms for CEC'15 benchmarks

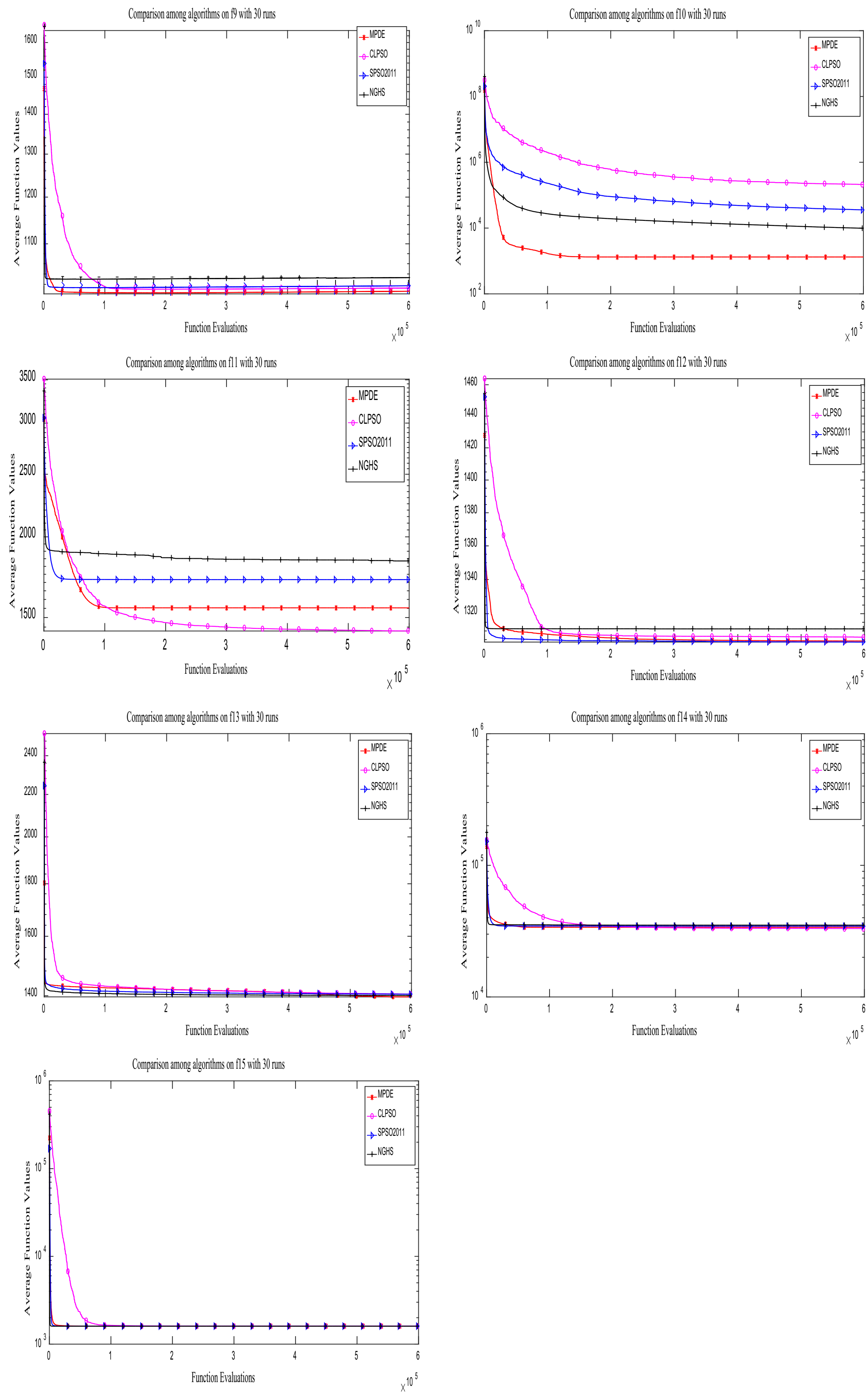

Fig. 5 continued

Acknowledgements This research is supported by National Natural Science Foundation of China (61375066, 71772060).

References

Ali MZ, Awad NH, Suganthan PN, Reynolds RG (2017) An adaptive multipopulation differential evolution with dynamic population reduction. IEEE Trans Cybern 47(9):2768–2779

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

Brest J, Bošković B, Greiner S, Žumer V, Mauec MS (2007) Performance comparison of self-adaptive and adaptive differential evolution algorithms. Soft Comput 11(7):617–629

Cao B, Zhao J, Lv Z, Liu X (2017) A distributed parallel cooperative coevolutionary multiobjective evolutionary algorithm for large-scale optimization. IEEE Trans Ind Inf 13(4):2030–2038

Chen L, Zhou C, Li X, Dai G (2017) An improved differential evolution algorithm based on suboptimal solution mutation. Int J Comput Sci Math 8(1):28–34

Cui L, Li G, Lin Q, Chen J, Lu N (2016) Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations. Comput Oper Res 67:155–173

Cui L, Li G, Zhu Z, Lin Q, Wong KC, Chen J, Lu J (2018) Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism. Inf Sci 422:122–143

Das S, Suganthan PN (2011) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 15(1):4–31

Das S, Mullick SS, Suganthan PN (2016) Recent advances in differential evolution–an updated survey. Swarm Evol Comput 27:1–30

Fan Q, Yan X (2016) Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies. IEEE Trans Cybern 46(1):219–232

Huang VL, Qin AK, Suganthan PN (2006) Self-adaptive differential evolution algorithm for constrained real-parameter optimization. In: IEEE congress on evolutionary computation, 2006. CEC 2006, pp 17–24

Khan SU, Yang S, Wang L, Liu L (2016) A modified particle swarm optimization algorithm for global optimizations of inverse problems. IEEE Trans Magn 52(3):1–4

Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295

Liang JJ, Qu BY, Suganthan PN, Chen Q (2014) Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization. Technical report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore

Mallipeddi R, Suganthan PN, Pan QK, Tasgetiren MF (2011) Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl Soft Comput 11(2):1679–1696

Mendes R, Rocha I, Ferreira EC, Rocha M (2006) A comparison of algorithms for the optimization of fermentation processes. In: IEEE congress on evolutionary computation, 2006, pp 2018–2025

Mohamed AW (2017) An efficient modified differential evolution algorithm for solving constrained non-linear integer and mixed-integer global optimization problems. Int J Mach Learn Cybern 8(3):989–1007

Moharam A, El-Hosseini MA, Ali HA (2016) Design of optimal PID controller using hybrid differential evolution and particle swarm optimization with an aging leader and challengers. Appl Soft Comput 38:727–737

Qin AK, Suganthan PN (2005, September) Self-adaptive differential evolution algorithm for numerical optimization. In: The 2005 IEEE congress on evolutionary computation, 2005, vol 2, pp1785–1791

Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417

Shang R, Jiao L, Liu F, Ma W (2012) A novel immune clonal algorithm for MO problems. IEEE Trans Evol Comput 16(1):35–50

Shang R, Du B, Ma H, Jiao L, Xue Y, Stolkin R (2016) Immune clonal algorithm based on directed evolution for multi-objective capacitated arc routing problem. Appl Soft Comput 49:748–758

Shukla R, Hazela B, Shukla S, Prakash R, Mishra KK (2017) Variant of differential evolution algorithm. Advances in computer and computational sciences. Springer, Singapore, pp 601–608

Suresh S, Lal S (2017) Modified differential evolution algorithm for contrast and brightness enhancement of satellite images. Appl Soft Comput 61:622–641

Tanabe R, Fukunaga A (2013) Success-history based parameter adaptation for differential evolution. In: 2013 IEEE congress on evolutionary computation, pp 71–78

Teo J (2006) Exploring dynamic self-adaptive populations in differential evolution. Soft Comput 10(8):673–686

Wang X, Tang L (2016) An adaptive multi-population differential evolution algorithm for continuous multi-objective optimization. Inf Sci 348:124–141

Wang Y, Cai Z, Zhang Q (2011) Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans Evol Comput 15(1):55–66

Wu G, Mallipeddi R, Suganthan PN, Wang R, Chen H (2016a) Differential evolution with multi-population based ensemble of mutation strategies. Inf Sci 329:329–345

Wu G, Mallipeddi R, Suganthan PN, Wang R, Chen H (2016b) Differential evolution with multi-population based ensemble of mutation strategies. Inf Sci 329:329–345

Yang Z, Tang K, Yao X (2008) Self-adaptive differential evolution with neighborhood search. In: IEEE congress on evolutionary computation, 2008, pp 1110–1116

Yi W, Zhou Y, Gao L, Li X, Mou J (2016) An improved adaptive differential evolution algorithm for continuous optimization. Expert Syst Appl 44:1–12

Yu WJ, Shen M, Chen WN, Zhan ZH, Gong YJ, Lin Y, Zhang J (2014) Differential evolution with two-level parameter adaptation. IEEE Trans Cybern 44(7):1080–1099

Zambrano-Bigiarini M, Clerc M, Rojas R (2013) Standard particle swarm optimisation 2011 at cec-2013: a baseline for future pso improvements. In: 2013 IEEE congress on evolutionary computation, pp 2337–2344

Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958

Zhang SX, Zheng SY, Zheng LM (2017) An efficient multiple variants coordination framework for differential evolution. IEEE Trans Cybern 47(9):2780–2793

Zhao X, Lin W, Yu C, Chen J, Wang S (2013) A new hybrid differential evolution with simulated annealing and self-adaptive immune operation. Comput Math Appl 66(10):1948–1960

Zhao X, Liu Z, Yang X (2014) A multi-swarm cooperative multistage perturbation guiding particle swarm optimizer. Appl Soft Comput 22:77–93

Zhao X, Lin W, Hao J, Zuo X, Yuan J (2016a) Clustering and pattern search for enhancing particle swarm optimization with Euclidean spatial neighborhood search. Neurocomputing 171:966–981

Zhao Z, Yang J, Hu Z, Che H (2016b) 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(1):30–45

Zhou XG, Zhang GJ, Hao XH, Yu L (2016) A novel differential evolution algorithm using local abstract convex underestimate strategy for global optimization. Comput Oper Res 75:132–149

Zhou YZ, Yi WC, Gao L, Li XY (2017) Adaptive differential evolution with sorting crossover rate for continuous optimization problems. IEEE Trans Cybern 47(9):2742–2753

Zou D, Gao L, Wu J, Li S, Li Y (2010) A novel global harmony search algorithm for reliability problems. Comput Ind Eng 58(2):307–316