Journal of Computational Science 26 (2018) 501–511

Contents lists available at ScienceDirect

Journal of Computational Science

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

Enhancing differential evolution with random neighbors based strategy

Hu Peng [a,*], Zhaolu Guo [b], Changshou Deng [a], Zhijian Wu [c]

[a] School of Information Science and Technology, Jiujiang University, Jiujiang 332005, PR China
[b] School of Science, Jiangxi University of Science and Technology, Ganzhou 341000, PR China
[c] State Key Lab of Software Engineering, School of Computer, Wuhan University, Wuhan 430072, PR China

ARTICLE INFO

Article history:
Received 31 March 2017
Received in revised form 25 May 2017
Accepted 11 July 2017
Available online 14 July 2017

Keywords:
Differential evolution
Global optimization
Random neighbor
Exploration and exploitation

ABSTRACT

As a powerful evolutionary algorithm for solving the tough global optimization problems, differential evolution (DE) has drawn more and more attention. However, how to make a proper balance between the global and local search is a perplexing question and greatly limit the optimization performance of DE. As we all known, there are two classical mutation strategies in DE, i.e., DE/rand/1 and DE/best/1. In DE/rand/1 strategy, the base vector is chosen from the population randomly, this means its better exploration and poor exploitation. The base vector of DE/best/1 strategy is the best one of the population and the strategy has better exploitation and poor exploration. To overcome these problems, this paper proposed a random neighbor based mutation strategy (DE/neighbor/1). For each individual of the population at each generation, the neighbors are chosen from the population in a random manner. The base vector of DE/neighbor/1 mutation strategy is the best one in the neighbors. On the basis of the new strategy, an enhancing differential evolution with DE/neighbor/1 (RNDE) is proposed. The experimental studies have been tested on 27 widely used benchmark functions and the results have proved that the proposed algorithm is competitive and promising.

1. Introduction

With the advance of human society and the development of science and technology, the optimization problems in the field of scientific research and engineering application [1–4] are increasingly complex. The problems are usually nonlinear, high-dimension, multimodal and non-differentiable. While the traditional deterministic optimization method based on mathematics is ineffective, the evolutionary algorithms tend to become the first choice for people to solve these problems. As a kind of calculation method with adaptive environment ability, evolutionary algorithms inspired and learned from natural phenomena or various principle and mechanism of the organisms.

Among many evolutionary algorithms, differential evolution (DE), invented by Storn and Price in 1995 [5], has got more and more attention of scholars in the field of evolutionary computation all over the world, because it is simple, efficient and easy to implement. Besides DE has been successfully used in the scientific research and engineering in various fields, such as bioinformatics [6,7], electrical power systems [8,9], pattern and recognition and image processing [10,11], etc. However, it is observed that how to balance the exploitation and exploration is a perplexing question and greatly limit the optimization performance of DE. In DE, there are two classical mutation strategies, i.e., DE/rand/1 and DE/best/1. In DE/rand/1 strategy, the base vector is selected from the population randomly, this means its better exploration and poor exploitation. However, the base vector of DE/best/1 strategy is the best one of the population and the strategy has better exploitation and poor exploration.

Based on these observations, we proposed a new mutation strategy, called random neighbor based mutation strategy (DE/neighbor/1). For each individual of the population at each generation, the neighbors are chosen from the population in a random manner. The base vector of DE/neighbor/1 mutation strategy is the best one in the neighbors of the current individual. DE/neighbor/1 combines the advantages of the DE/rand/1 and DE/best/1 strategies. Experimental studies are simulated using a set of well-known benchmark functions. Comprehensive experiments demonstrate the effectiveness and efficiency of the proposed algorithm.

* Corresponding author.
E-mail address: hu_peng@whu.edu.cn (H. Peng).

https://doi.org/10.1016/j.jocs.2017.07.010
1877-7503/

The rest of this paper is structured as follows. In Section 2, the basic DE is introduced briefly. Section 3 surveys some recently related works. Section 4 presents the details of DE/neighbor/1 strategy and our proposed DE variant, called RNDE. Experimental results and analysis are described in Section 5. Section 6 concludes the work.

2. Differential evolution

As a powerful population-based heuristic search algorithm, DE organizes a population containing NP individuals to search in the D-dimensional space, in which each individual represents a candidate solution. It starts with a randomly sampled population and then performs mutation, crossover, and selection operators iteratively to make better population until it touches a certain preset stopping condition such as maximum number of function evaluates (MaxFEs). Without loss of consistency, only the minimum problem is to be considered in this paper.

Firstly, in the mutation operation, the mutant vector V_i around the individual X_i (or called target vector) is produced according to the mutation strategy. There are five commonly used mutation strategies and can be shown as follows:

(1) DE/rand/1

$$V_i = X_{r1} + F \cdot (X_{r2} - X_{r3}) \tag{1}$$

(2) DE/best/1

$$V_i = X_{best} + F \cdot (X_{r1} - X_{r2}) \tag{2}$$

(3) DE/current-to-best/1

$$V_i = X_i + F \cdot (X_{best} - X_i) + F \cdot (X_{r1} - X_{r2}) \tag{3}$$

(4) DE/rand/2

$$V_i = X_{r1} + F \cdot (X_{r2} - X_{r3}) + F \cdot (X_{r4} - X_{r5}) \tag{4}$$

(5) DE/best/2

$$V_i = X_{best} + F \cdot (X_{r1} - X_{r2}) + F \cdot (X_{r3} - X_{r4}) \tag{5}$$

where X_{r1}, X_{r2}, X_{r3}, X_{r4}, and X_{r5} are different individuals randomly chosen from the population and all are different with X_i. F, namely scale factor, is an important parameter in DE which amplifies the different vector. X_{best} is the best individual at present in the whole population.

Secondly, in the crossover operation, binomial crossover is commonly conducted to recombine the mutant vector V_i and target vector X_i. The trial vector U_i is calculated by

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if} \ \ rand_j \le CR \ \ \text{or} \ \ j = jrand \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{6}$$

where $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, D$, $jrand$ is a randomly chose integer from $1, 2, \ldots, D$, $rand_j$ is a random value uniformly distributed in [0, 1] for the jth dimension, and $CR \in [0, 1]$ is the crossover probability.

Finally, a greedy choice is done between the target vector X_i and the trial vector U_i, if U_i is better than X_i in terms of their objective values, then X_i is replaced with U_i to survive into the next generation:

$$X_i = \begin{cases} U_i, & \text{if} \ \ f(U_i) < f(X_i) \\ X_i, & \text{otherwise} \end{cases} \tag{7}$$

3. Related works

DE has attracted many scholars around the world researching to improve its performance from different aspects and plenty of variants have been presented. These works can be classified into the following four categories.

3.1. Hybridization with other techniques

Sun et al. [12] presented a combination of DE and estimation of distribution algorithm (EDA), named DE/EDA, in which the local information and global information are obtained by DE mutation and EDA modeling respectively. Sufficient experimentation have verified the good performance of the new approach. Omran et al. [13] presented a hybrid DE with bare-bones particle swarm optimization (PSO), in which only the crossover probability (CR) is retained, others such as the standard PSO parameters and the scale factor (F) are removed. The individual of population is updated to a randomly chosen personal best or a mutation of the weighted average of the personal best and neighborhood best according to the specified probability.

Then, Gong et al. [14] hybridized the DE mutation operator with the migration operator of biogeography-based optimization (BBO) and proposed DE/BBO, where the new hybrid migration operator effectively balances the exploration and the exploitation. Yildiz [15] developed a hybrid DE with receptor editing property of immune system, which utilized the receptor editing to enhance the diversity of the whole population and escape from local optimal. The new approach successfully implemented to the optimization of machining parameters in milling operations.

Furthermore, Xie et al. [16] proposed a diversity maintained DE embedded with gradient-based local search (DMGBDE), in which the quasi-Newton method is used to compensate for the deficiency of DE in exploitation and a new mutation strategy is introduced to keep the diversity of population. Experimental results have verified the favorable performance over a set of well-known test functions. Peng and Wu [17] proposed a hybrid DE with Taguchi method, called THDE, in which the novel Taguchi local search is designed to enhance the exploitation ability. Zhong and Cai [18] hybrid the DE with Powell's method, in which the added method is used to turn the parameters of the best individual.

3.2. Modification of the mutation strategy

Fan and Lampinen [19] proposed a trigonometric mutation strategy and incorporated it into DE to form a new approach named TDE, in which the new mutation strategy as a greedy operator biases the search direction strongly to the best one of three selected individuals to enhance the convergence velocity. Gong and Cai [20] proposed a ranking-based mutation strategy, in which some of the parents are proportionally selected according their rankings in the current population. Wang et al.[21] introduced a multiobjective sorting-based mutation strategy, where individuals in the current population are sorted according to their fitness and diversity contribution by nondominated sorting.

On the other hand, Jia et al. [22] proposed a novel mutation strategy called current-to-rand/best/1, which uses the information of the current generation number and can keep a good trade-off between the population diversity and convergence rate on some level. Based on the new operator, an improved version of $(\mu+\lambda)$-CDE (ICDE) is put forward for solving constrained optimization problems. Sharma et al. [23] proposed a modification mutation strategy, in which the congitive learning factor (CLF) is introduced to adjust the position of individuals. Tang et al. [24] proposed an individual-dependent mutation (IDM) strategy, where the population is classified into the superior category and the inferior category, then the four selected mutation operators are assigned to the two groups of individuals respectively to balance diversity with rapidity.

3.3. Adaptation of mutation strategy and parameter settings

Liu and Lampinen [25] presented a fuzzy adaptive DE variant (FADE), where the fuzzy logic control method is employed to adjust the parameters adaptively. In FADE, the individuals of the successive survied and relative fitness values are used as the control inputs. Zhang and Sanderson [26] proposed an adaptive DE (JADE), in which a novel greedy mutation strategy called DE/ current-to-pbest/1 is introduced and the parameters are adaptively adjust by learning from their history and experience of success. As analyzed in [26], the advantage of the adaptive strategy is to avoid without using the prior knowledge to turn the parameters through observing the characteristics of problems.

Qin et al. [27] presented a self-adaptive DE variant (SaDE), which adaptively chooses the mutation strategy from the four carefully selected strategies with different characteristics. Furthermore, the scale factor F was produced through Gaussian distribution with mean value 0.5 and standard deviation 0.3. Meanwhile, a historical memory mechanism is introduced to adaptively adjust the crossover probability CR. Similar to SaDE, Mallipeddi et al. [28] proposed an ensemble DE (EPSDE), in which not only a mutation strategies pool but also a parameters pool is employed to do the adaptive operation. In the meantime, a novel composite DE (CoDE) is proposed by Wang et al. [29], which carefully chose three mutation strategies and three group of parameters under systematic investigation to randomly combine them to produce trial vector.

More recently, Sarker et al. [30] proposed a new mechanism in DE to select the best parameters combination (F, CR, and NP) during the evolution, in which the success rate of each combination as evaluation criterion is recorded for a certain number of generations. Comprehensively experimental results have demonstrated its superior performance. Wang et al. [31] proposed a bimodal distribution parameter setting method, in which both parameters F and CR are generated by bimodal distributions to balance the exploration and exploitation within the search process. Guo et al. [32] proposed a self-adaptive differential evolution with global neighborhood search (NSSDE), in which its control parameters are self-adaptively tuned according to the feedback from the search process.

3.4. Use of neighbor information

Beyond these methods listed above, the use of neighbor information is a good idea to enhance the convergence performance of DE. Inspired by the particle topology of PSO, Das et al. [33] put forward a novel neighborhood based mutation strategy and proposed a DE variant (DEGL), in which the local and global neighborhood-based mutation operators are equipped to improve the exploration and exploitation ability respectively. This method has been verified on 24 benchmark functions and two real-world application problems and demonstrated the competitive results. Liao et al. [34] proposed a directional mutation strategy, in which the cellular topology is employed first to construct a neighborhood for each member of population and then the direction information extracted from the neighborhood is incorporated into the mutation operation. Pham [35] developed a nearest neighbor comparison method for DE to reduce the function evaluation, where the approach uses a nearest neighbor in the search population to judge whether a new point is worth evaluating.

Recently, Das et al. [36,37] presented a comprehensive survey on the progress of DE, so further information can reference them.

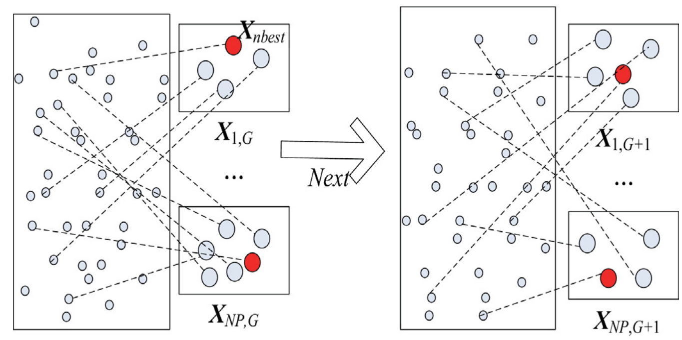


Fig. 1. Illustration of the random neighbors.

4. DE algorithm with random neighbors based mutation strategy (RNDE)

In this section, we present an improved DE (RNDE), in which the random neighbor based mutation strategy is employed to enhance the search ability. Furthermore, adaptive shift parameter setting is proposed with the aim of increasing the adaptability to different problems. The details of the two main components and the framework of RNDE will be described.

4.1. Random neighbors based mutation strategy

As mentioned previously, the mutation operator of DE is dependent mainly on the base vector. There are two classical mutation strategies, i.e., DE/rand/1 and DE/best/1 proposed by Storn and Price [5]. In DE/rand/1 strategy, the base vector is randomly selected from the population, this means its better exploration and poor exploitation. However, the base vector of DE/best/1 strategy is the best one of the population. In contrast to DE/rand/1, DE/best/1 strategy has better exploitation and poor exploration, since all the individuals move to the same best position by the attraction of X_{best}. Generally, the balance of exploration and exploitation plays an important role in determining the performance of the DE algorithm. To solve this problem, neighbor based mutation strategy proposed by Das et al.[33] may be better than the classical mutation strategies as in the classical DE, because the information of neighbor has been introduced to guide the search. In DEGL [33], the local neighbor-based mutation is revised from the DE/target-to-best/1 strategy, which employed the best one in the neighbors of current individual and any two others chosen from the same neighbors. The ring topology is used to organize the individuals of the population. However, the ring topology of neighbor is static and defined on the indices of the individual. As we all known, the information exchange speed of the static neighbor topology is slow and limit the exploration ability.

Based on these considerations, we propose another neighbor mutation strategy which is inspired from the DEGL, called random neighbor based mutation strategy (DE/ neighbor/1). For each individual of the population at each generation, the neighbor of X_i is chosen from the population in a random manner. The base vector of DE/neighbor/1 mutation strategy is the best one in the neighbor of X_i. The DE/neighbor/1 mutation strategy can be formulated as following:

$$V_i = X_{nbest} + F \cdot (X_{r1} - X_{r2}) \quad (8)$$

where X_{nbest} is the best one of the N neighbors of the current individual X_i, N is the number of neighbors.

In DE/neighbor/1 mutation strategy, the number of neighbors N plays crucial role in controlling the balance between the exploration and exploitation abilities. A large value of N (close to NP) can make the mutation strategy tend to the DE/best/1 strategy, promot-

Table 1
Testsuite with 27 benchmark functions.

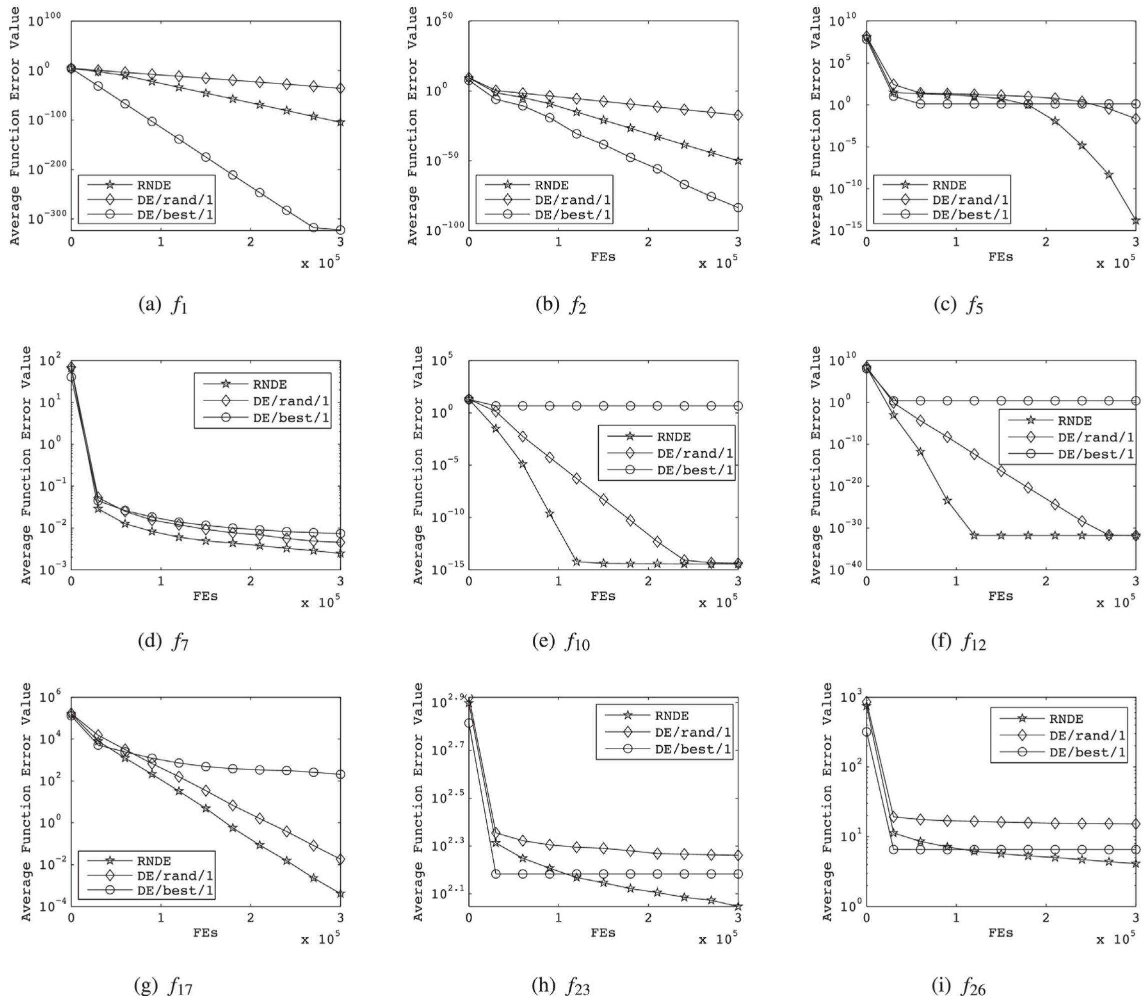
Type	Function	Name	Search range	Global optimum
Unimodal functions	f_1	Sphere	[−100, 100]	0
	f_2	Schwefel2.22	[−10, 10]	0
	f_3	Schwefel1.2	[−100, 100]	0
	f_4	Schwefel2.21	[−100, 100]	0
	f_5	Rosenbrock	[−30, 30]	0
	f_6	Step	[−1.28, 1.28]	0
	f_7	Quartic with Noise	[−100, 100]	0
Multimodal functions	f_8	Schwefel2.26	[−500, 500]	$-418.98 \cdot D$
	f_9	Rastrigin	[−5.12, 5.12]	0
	f_{10}	Ackley	[−32, 32]	0
	f_{11}	Griewank	[−600, 600]	0
	f_{12}	Penalized1	[−50, 50]	0
	f_{13}	Penalized2	[−50, 50]	0
Shifted unimodal functions	f_{14}	Shifted Sphere Function	[−100, 100]	−450
	f_{15}	Shifted Schwefels Problem 1.2	[−100, 100]	−450
	f_{16}	Shifted Rotated High Conditioned Elliptic Function	[−100, 100]	−450
	f_{17}	Shifted Schwefels Problem 1.2 with Noise in Fitness	[−100, 100]	−450
	f_{18}	Schwefels Problem 2.6 with Global Optimum on Bounds	[−100, 100]	−310
Shifted multimodal functions	f_{19}	Shifted Rosenbrocks Function	[−100, 100]	390
	f_{20}	Shifted Rotated Griewanks Function without Bounds	[0, 600]	−180
	f_{21}	Shifted Rotated Ackleys Function with Global Optimum on Bounds	[−32, 32]	−140
	f_{22}	Shifted Rastrigins Function	[−5, 5]	−330
	f_{23}	Shifted Rotated Rastrigins Function	[−5, 5]	−330
	f_{24}	Shifted Rotated Weierstrass Function	[−0.5, 0.5]	90
	f_{25}	Schwefels Problem 2.13	$[-\pi, \pi]$	−460
	f_{26}	Shifted Expanded Griewanks plus Rosenbrocks Function (F8F2)	[−3, 1]	−130
	f_{27}	Shifted Rotated Expanded Scaffers F6 Function	[−100, 100]	−300

Table 2
Experimental results of DE/rand/1, DE/DE/best/1, and RNDE for all test functions at $D = 30$.

F	DE/rand/1 Ave Err ± Std Dev	DE/best/1 Ave Err ± Std Dev	RNDE Ave Err ± Std Dev
f_1	2.68E−36 ± 2.68E−36−	**5.40E−323 ± 0.00E+00+**	2.71E−105 ± 3.04E−105
f_2	5.40E−18 ± 3.59E−18−	**2.82E−84 ± 1.37E−83+**	1.00E−50 ± 1.77E−50
f_3	3.33E−05 ± 2.73E−05−	**9.48E−70 ± 3.59E−69+**	7.49E−15 ± 1.56E−14
f_4	2.53E−01 ± 7.73E−01−	6.58E−09 ± 1.95E−08−	**2.10E−09 ± 6.88E−09**
f_5	2.29E−02 ± 6.33E−02−	1.33E+00 ± 1.88E+00−	**1.56E−14 ± 3.52E−14**
f_6	**0.00E+00 ± 0.00E+00≈**	7.53E+01 ± 8.31E+01−	**0.00E+00 ± 0.00E+00**
f_7	4.53E−03 ± 1.47E−03−	7.36E−03 ± 4.67E−03−	**2.47E−03 ± 8.07E−04**
f_8	6.57E+03 ± 6.04E+02−	5.00E+03 ± 6.27E+02−	**3.82E−04 ± 0.00E+00**
f_9	1.38E+02 ± 2.78E+01−	5.26E+01 ± 1.34E+01−	**0.00E+00 ± 0.00E+00**
f_{10}	4.14E−15 ± 1.32E−15−	4.67E+00 ± 1.29E+00−	**3.55E−15 ± 0.00E+00**
f_{11}	**0.00E+00 ± 0.00E+00≈**	4.82E−02 ± 4.94E−02−	**0.00E+00 ± 0.00E+00**
f_{12}	**1.57E−32 ± 5.47E−48≈**	2.36E+00 ± 2.84E+00−	1.57E−32 ± 9.68E−35
f_{13}	**1.35E−32 ± 5.47E−48≈**	1.45E+00 ± 1.74E+00−	**1.35E−32 ± 5.47E−48**
f_{14}	**0.00E+00 ± 0.00E+00≈**	3.64E−13 ± 2.62E−13−	**0.00E+00 ± 0.00E+00**
f_{15}	4.15E−05 ± 3.95E−05−	1.18E−12 ± 7.06E−13−	**1.53E−13 ± 8.19E−14**
f_{16}	3.54E+05 ± 1.86E+05−	**1.39E+04 ± 1.08E+04+**	1.63E+05 ± 1.00E+05
f_{17}	1.82E−02 ± 2.54E−02−	2.06E+02 ± 3.50E+02−	**4.10E−04 ± 6.38E−04**
f_{18}	**5.78E+01 ± 6.45E+01≈**	2.09E+03 ± 7.14E+02−	1.22E+02 ± 1.95E+02
f_{19}	1.90E−01 ± 8.27E−01−	1.06E+00 ± 1.76E+00−	**1.33E−01 ± 7.16E−01**
f_{20}	**1.99E−14 ± 1.30E−14+**	2.04E−02 ± 1.92E−02−	1.97E−03 ± 4.65E−03
f_{21}	**2.09E+01 ± 4.67E−02≈**	2.09E+01 ± 5.92E−02≈	2.09E+01 ± 5.10E−02
f_{22}	1.34E+02 ± 2.23E+01−	1.06E+02 ± 2.84E+01−	**1.89E−15 ± 1.02E−14**
f_{23}	1.83E+02 ± 8.39E+00−	1.53E+02 ± 3.95E+01−	**1.12E+02 ± 1.20E+01**
f_{24}	3.97E+01 ± 1.13E+00−	**2.12E+01 ± 3.14E+00+**	3.24E+01 ± 1.02E+00
f_{25}	1.42E+03 ± 2.98E+03≈	**1.27E+03 ± 1.32E+03≈**	4.21E+03 ± 1.12E+04
f_{26}	1.53E+01 ± 9.23E−01−	6.53E+00 ± 2.24E+00−	**4.11E+00 ± 2.98E−01**
f_{27}	1.33E+01 ± 1.21E−01−	**1.19E+01 ± 5.42E−01+**	1.30E+01 ± 1.55E−01
−	18	19	−−
+	1	6	−−
≈	8	2	−−

ing exploitation. In contrast, a small value of N makes the mutation strategy tend to the DE/rand/1 strategy, thereby resulting in better exploration. The large or small N is not a wise choice. In general, if the better individual learns more best information of the neighbors, it will increases the possibility of trapping in local optimum. In this paper, a simple self-adaptive strategy is proposed to dynamically update N and the number of neighbors for each individual X_i is defined as follows:

$$N_i = N_{lb} + (N_{ub} - N_{lb}) \cdot \frac{f(X_i) - f_{\min} + \xi}{\sum_{j=1}^{NP} (f(X_j) - f_{\min}) + \xi} \tag{9}$$



(a) f_1 (b) f_2 (c) f_5

(d) f_7 (e) f_{10} (f) f_{12}

(g) f_{17} (h) f_{23} (i) f_{26}

Fig. 2. Convergence curves of DE/rand/1, DE/best/1, and RNDE on nine representative test functions at $D = 30$.

where N_{lb} and N_{ub} are the lower and upper bound of N respectively, $f_{\min}$ is the minimum (best) value of the objective function in the population at current generation, and ξ is the smallest constant in the computer to avoid zero-division-error. In the self-adaptive strategy, the better individual has less neighbors and the worse individual has more neighbors. Corresponding to the mechanism, the better one adaptively favors the exploration and the worse one tends to exploitation.

As depicted in Fig. 1, for each individual X_i at generation G, some neighbors are randomly selected from the population. It is worth to note that the number of neighbors N is different. The best neighbor of X_i is picked as the base vector to create the mutant vector V_i according to Eq. (8). In the next generation, the same work is to do. By some initial experimentation, DE/neighbor/1 mutation strategy with $N_{lb} = 3$ and $N_{ub} = 10$ has get fairly good results with RNDE algorithm.

The characteristics and properties of the random neighbors based mutation strategy are summarize as follows:

(1) Compared with the classical mutation strategy DE/rand/1, the base vector of the DE/neighbor/1 is also randomly selected from the population, but it is the best neighbor. This mechanism make the probability that the randomly selected individual is a good one will rise. Therefore, the exploitation ability of the proposed mutation strategy is enhanced. To benefit the randomly selected method to produce the neighbors of each individual, the exploration ability of the new mutation strategy is not decreased.
(2) Compared with the classical mutation strategy DE/best/1, the DE/neighbor/1 also learns from the best individual, but it is the local best within its randomly selected neighbors instead of global best within the whole population. The combination of the "best" and "random" information makes the probability that the new algorithm trapped into the local optimum will fall.
(3) Compared with the neighbor based mutation strategy in DEGL, the neighbors of each individual at each generation in DE/neighbor/1 update dynamically rather than organizing neighbors with a static ring topology. The dynamic mechanism accelerates the velocity of information exchange and enhances the diversity of the population. Additionally, the proposed mutation strategy is simple and no other new parameters.
(4) Generally, the random selection is often in conflict with best information, but the DE/neighbor/1 combines them to achieve the best synergy and provides the best balance between exploration and exploitation ability in new algorithm.

4.2. Adaptive shift parameter setting

As we all know, the parameter CR greatly affects the search ability of DE algorithm. A large value of CR can make the trial vector learns more from mutant vector and inherits less from the target vector. It turns out that the trial vector is very different with the target vector and can increase the population diversity. A large value

Table 3
Mean errors of DE/rand/1, DE/best/1, and RNDE for all test functions at $D = 50$ and $D = 100$.

F	D=50			D=100		
	DE/rand/1	DE/best/1	RNDE	DE/rand/1	DE/best/1	RNDE
f_1	2.35E−39−	**1.46E−266+**	1.50E−115	8.73E−44−	7.80E−100−	**1.15E−121**
f_2	7.03E−20−	4.80E−05−	**3.34E−58**	2.36E−24−	3.05E−01−	**2.55E−65**
f_3	1.02E+00−	**5.29E−32+**	6.81E−06	1.37E+03−	**1.27E−06+**	1.87E−01
f_4	5.87E+00−	1.60E+01−	**1.53E−01**	1.89E+01−	4.06E+01−	**1.22E+01**
f_5	1.79E+01−	1.99E+00≈	**2.73E−01**	1.22E+02−	**1.46E+00+**	5.48E+01
f_6	**0.00E+00**≈	6.32E+02−	**0.00E+00**	2.43E+00−	6.64E+03−	**0.00E+00**
f_7	6.61E−03−	3.93E−02−	**3.58E−03**	1.73E−02−	7.49E−01−	**8.56E−03**
f_8	1.19E+04−	9.82E+03−	**6.36E−04**	2.33E+04−	2.16E+04−	**1.27E−03**
f_9	2.18E+02−	1.05E+02−	**5.08E+00**	**8.80E+01+**	2.38E+02−	1.51E+02
f_{10}	6.99E−15≈	8.28E+00−	**6.87E−15**	1.67E−14≈	1.33E+01−	**1.62E−14**
f_{11}	**0.00E+00**≈	8.87E−02−	**0.00E+00**	2.63E−03≈	4.77E−01−	**1.81E−03**
f_{12}	**9.45E−33+**	1.86E+00−	9.55E−33	6.22E−03−	1.60E+00−	**1.04E−32**
f_{13}	**1.37E−32+**	2.95E+00−	3.18E−32	**7.32E−04**≈	3.14E+00−	**7.32E−04**
f_{14}	5.12E−14≈	6.25E−11−	**4.17E−14**	**8.53E−14**≈	7.84E−09−	9.85E−14
f_{15}	3.29E+00−	**1.29E−11+**	4.28E−05	4.47E+03−	**3.57E−08+**	1.24E+00
f_{16}	2.58E+06−	**8.18E+04+**	2.54E+05	6.36E+06−	**1.10E+06+**	1.24E+06
f_{17}	3.39E+02−	6.84E+03−	**1.01E+02**	3.61E+04−	6.83E+04−	**2.09E+04**
f_{18}	**1.84E+03+**	6.86E+03−	2.37E+03	**4.79E+03+**	2.11E+04−	5.61E+03
f_{19}	3.69E+01−	1.59E+00≈	**1.52E−01**	1.18E+02−	**1.33E+00+**	5.87E+01
f_{20}	**5.75E−04**≈	1.23E−02−	2.46E−03	**2.96E−03+**	7.80E−03−	3.53E−03
f_{21}	**2.11E+01**≈	**2.11E+01**≈	**2.11E+01**	**2.13E+01**≈	**2.13E+01−**	**2.13E+01**
f_{22}	2.02E+02−	2.49E+02−	**4.94E−01**	**1.25E+02+**	6.14E+02−	1.45E+02
f_{23}	3.63E+02−	3.83E+02−	**2.47E+02**	8.48E+02−	1.04E+03−	**6.47E+02**
f_{24}	7.26E+01−	**4.71E+01+**	6.13E+01	1.60E+02−	**1.17E+02+**	1.41E+02
f_{25}	7.92E+03≈	**4.77E+03+**	3.33E+04	2.98E+04+	**2.30E+04+**	8.93E+05
f_{26}	3.00E+01−	2.20E+01−	**9.58E+00**	6.54E+01−	1.03E+02−	**2.79E+01**
f_{27}	2.30E+01−	**2.14E+01+**	2.27E+01	4.76E+01−	**4.55E+01+**	4.71E+01
−	17	17	−−	17	19	−−
+	3	7	−−	5	8	−−
≈	7	3	−−	5	0	−−

of *CR* suits for multimodal and non-separable functions, since more perturbation can make the complex problem to be better solved. In contrast, a small value of *CR* makes the trial vector learns less from mutant vector and inherits more from the target vector. A small value of *CR* suits for separable problems because each parameter of these problems is optimized independently.

In this paper, a adaptive shift strategy is proposed to dynamically update *CR* and described in Algorithm 1. In the new adaptive strategy, CR_l and CR_s represent the large and small mean values of *CR* respectively. Based on CR_l or CR_s and standard deviation 0.1, large or small *CR* will generated by a Gaussian distribution. When the trial vector is worse than the current vector, i.e., $f(U_i) \geq f(X_i)$, it means that the current *CR* has a lower chance to generate better candidate solutions and the change is need. If the small value of *CR* is not suitable, then we can shift to lager value of *CR*, and vice versa. After surveying the related literatures and running some initial experimentation over numerical benchmarks, the CR_l and CR_s are set to 0.85 and 0.1 respectively.

Algorithm 1. Adaptive shift *CR*

if $f(U_i) > f(X_i)$ **then**
  $Flag = \sim Flag$
  if $Flag == 1$ **then**
    $CR = CR_l + 0.1 * randn$
  else
    $CR = CR_s + 0.1 * randn$
  end if
end if

In summary, the proposed strategy increases the adaptability to various problems, since it switches between the large and small value of *CR* adaptively. Moreover, the new values of *CR* are different and normally distributed, and most values are going to be close to the CR_l or CR_s. In the case, this approach can make the different problems to have different updating paths for *CR*.

4.3. Framework of RNDE

Algorithm 2 describes the pseudocode of RNDE. By comparison with the original DE, there are only two major differences in the RNDE. The first difference is the using of DE/neighbor/1 strategy in the mutation. At the beginning of the mutation, the numbers of neighbors $N_{i,G}$ for each individual at generation G are calculated according to Eq. (9), then in the cycle, the neighbors of each individual are randomly chosen from the population and the best neighbor is picked up as the base vector to conduct the new mutation operator according to Eq. (8). Another difference is at the selection stage, if the trial vector is not better than the current vector, the adaptive shift strategy is executed to dynamically update the parameter *CR* to suit the evolution process according to Algorithm 1.

Algorithm 2. The proposed RNDE algorithm

Randomly initialize population;
Evaluate the objective function;
$FEs = NP$;
while $FEs < MaxFEs$ **do**
  Calculate the number of neighbors N_i for each individual according to Eq.(9);
  for $i = 1 : NP$ **do**
    Randomly choose N_i neighbors for ith individual and the best one X_{nbest};
    Execute the DE/neighbor/1 according to Eq.(8) to generate a mutate vector V_i;
    Execute the crossover operation to generate a trial vector U_i;
    Evaluate the trial vector U_i;
    $FEs = FEs + 1$;
    if $f(X_i) > f(U_i)$ **then**
      $X_i = U_i$;
    else
      Update *CR* according to Algorithm.1;
    end if
  end for
end while

Table 4
Experimental results of DEGL/SAW, EPSDE, MGBDE, SaDE, ODE, OXDE, and RNDE for all test functions at $D = 30$.

F	DEGL/SAW Ave Err ± Std Dev	EPSDE Ave Err ± Std Dev	MGBDE Ave Err ± Std Dev	SaDE Ave Err ± Std Dev	ODE Ave Err ± Std Dev	OXDE Ave Err ± Std Dev	RNDE Ave Err ± Std Dev
f_1	6.01E−101 ± 2.10E−100−	8.47E−174 ± 0.00E+00+	1.51E−91 ± 6.81E−91≈	1.22E−130 ± 4.14E−130+	9.94E−58 ± 3.19E−57−	2.66E−59 ± 5.94E−59−	2.71E−105 ± 3.04E−105
f_2	1.63E−49 ± 1.53E−49−	8.69E−86 ± 3.84E−85+	1.52E−53 ± 8.15E−53+	2.97E−79 ± 5.46E−79+	5.86E−18 ± 4.68E−18−	2.96E−33 ± 2.33E−33−	1.00E−50 ± 1.77E−50
f_3	3.35E−24 ± 6.82E−24+	4.48E−36 ± 2.40E−35+	8.23E−05 ± 4.36E−04−	1.14E−06 ± 2.93E−06−	3.06E−05 ± 3.35E−05−	2.38E−05 ± 2.23E−05−	7.49E−15 ± 1.56E−14
f_4	5.18E−25 ± 9.15E−25+	2.68E+00 ± 1.43E+00−	2.01E−08 ± 3.08E−08−	5.59E−07 ± 3.01E−06−	1.98E−03 ± 1.07E−02−	7.44E+00 ± 3.32E+00−	2.10E−09 ± 6.88E−09
f_5	6.64E−01 ± 1.49E+00−	3.99E−01 ± 1.20E+00−	3.25E+00 ± 1.25E+01−	2.89E+01 ± 2.34E+01−	2.55E+01 ± 8.29E−01−	1.20E+00 ± 1.83E+00−	1.56E−14 ± 3.52E−14
f_6	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00
f_7	1.20E−03 ± 3.52E−04+	8.88E−04 ± 3.37E−04+	2.37E−03 ± 6.43E−04≈	2.77E−03 ± 1.19E−03≈	9.20E−04 ± 3.21E−04+	4.08E−03 ± 1.94E−03−	2.47E−03 ± 8.07E−04
f_8	7.30E+03 ± 2.94E+02−	3.82E−04 ± 0.00E+00≈	4.88E+02 ± 2.89E+02−	3.82E−04 ± 0.00E+00≈	6.94E+03 ± 3.85E+02−	3.82E−04 ± 0.00E+00≈	3.82E−04 ± 0.00E+00
f_9	1.01E+02 ± 5.12E+01−	0.00E+00 ± 0.00E+00≈	6.27E+00 ± 2.33E+00−	3.32E−02 ± 1.79E−01≈	3.36E+01 ± 2.25E+01−	9.32E+00 ± 3.02E+00−	0.00E+00 ± 0.00E+00
f_{10}	3.67E−15 ± 6.38E−16≈	4.86E−15 ± 1.71E−15−	7.22E−15 ± 1.45E−15−	9.31E−02 ± 2.79E−01≈	3.55E−15 ± 0.00E+00≈	3.10E−02 ± 1.67E−01≈	3.55E−15 ± 0.00E+00
f_{11}	3.61E−03 ± 5.46E−03−	7.40E−04 ± 2.22E−03≈	9.86E−04 ± 2.51E−03−	3.36E−03 ± 9.03E−03−	7.39E−04 ± 2.78E−03≈	2.46E−03 ± 4.17E−03−	0.00E+00 ± 0.00E+00
f_{12}	1.57E−32 ± 5.47E−48≈	1.57E−32 ± 2.01E−35≈	3.46E−03 ± 1.86E−02≈	1.04E−02 ± 3.11E−02≈	1.58E−32 ± 2.52E−34≈	1.57E−32 ± 2.32E−34≈	1.57E−32 ± 9.68E−35
f_{13}	3.66E−04 ± 1.97E−03≈	3.66E−04 ± 1.97E−03≈	1.37E−32 ± 8.85E−34≈	1.83E−03 ± 8.07E−03≈	1.35E−32 ± 5.47E−48≈	1.56E−32 ± 5.93E−33≈	1.35E−32 ± 5.47E−48
f_{14}	3.79E−15 ± 1.42E−14≈	5.68E−15 ± 1.71E−14≈	5.31E−14 ± 1.42E−14−	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00≈	0.00E+00 ± 0.00E+00
f_{15}	4.74E−14 ± 2.12E−14+	1.74E−12 ± 4.60E−12−	1.01E−04 ± 3.78E−04−	8.07E−06 ± 1.69E−05−	3.33E−04 ± 3.31E−04−	5.66E−05 ± 5.70E−05−	1.53E−13 ± 8.19E−14
f_{16}	5.80E+04 ± 3.44E+04+	1.84E+06 ± 4.73E+06−	2.64E+05 ± 1.67E+05−	4.86E+05 ± 1.85E+05−	5.98E+05 ± 3.67E+05−	4.78E+05 ± 2.21E+05−	1.63E+05 ± 1.00E+05
f_{17}	6.44E−14 ± 3.51E−14+	2.52E+01 ± 1.18E+02−	3.30E+01 ± 3.12E+01−	1.14E+02 ± 1.49E+02−	2.08E−01 ± 2.55E−01−	1.26E+00 ± 1.10E+00−	4.10E−04 ± 6.38E−04
f_{18}	1.06E−01 ± 2.60E−01+	1.98E+03 ± 9.66E+02−	2.82E+03 ± 7.10E+02−	3.30E+03 ± 5.49E+02−	1.45E+02 ± 8.04E+01−	2.14E+01 ± 5.84E+01+	1.22E+02 ± 1.95E+02
f_{19}	1.06E+00 ± 1.76E+00−	7.97E−01 ± 1.59E+00−	2.56E+00 ± 3.87E+00−	4.64E+01 ± 3.23E+01−	5.38E+01 ± 3.22E+01−	6.64E−01 ± 1.49E+00−	1.33E−01 ± 7.16E−01
f_{20}	6.90E−03 ± 8.65E−03−	1.32E−02 ± 1.13E−02−	1.84E−02 ± 1.40E−02−	2.61E−02 ± 2.83E−02−	6.57E−03 ± 9.24E−03≈	1.34E−02 ± 9.54E−03−	1.97E−03 ± 4.65E−03
f_{21}	2.09E+01 ± 4.61E−02≈	2.09E+01 ± 6.84E−02≈	2.10E+01 ± 4.23E−02≈	2.09E+01 ± 4.08E−02≈	2.10E+01 ± 5.75E−02≈	2.09E+01 ± 4.17E−02≈	2.09E+01 ± 5.10E−02
f_{22}	5.91E+01 ± 5.03E+01−	0.00E+00 ± 0.00E+00≈	7.89E+00 ± 3.03E+00−	1.66E−01 ± 3.71E−01≈	7.54E+01 ± 2.89E+01−	1.39E+01 ± 3.84E+00−	1.89E−15 ± 1.02E−14
f_{23}	1.67E+02 ± 9.65E+00−	4.95E+01 ± 1.06E+01+	6.21E+01 ± 1.36E+01+	4.89E+01 ± 1.02E+01−	5.76E+01 ± 5.31E+01+	3.73E+01 ± 3.24E+01+	1.12E+02 ± 1.20E+01
f_{24}	3.98E+01 ± 8.40E−01−	2.73E+01 ± 1.92E+00+	2.52E+01 ± 3.13E+00+	1.70E+01 ± 3.14E+00−	8.99E+00 ± 9.39E+00+	3.73E+01 ± 8.18E+00−	3.24E+01 ± 1.02E+00
f_{25}	1.98E+03 ± 3.01E+03≈	2.18E+04 ± 6.19E+03−	3.03E+03 ± 3.84E+03≈	3.92E+03 ± 2.81E+03−	2.02E+03 ± 2.35E+03≈	3.34E+03 ± 5.14E+03≈	4.21E+03 ± 1.12E+04
f_{26}	1.29E+01 ± 1.12E+00−	1.92E+00 ± 1.63E−01+	2.39E+00 ± 6.16E−01+	3.92E+00 ± 3.95E−01≈	7.58E+00 ± 2.15E+00−	2.08E+00 ± 6.28E−01+	4.11E+00 ± 2.98E−01
f_{27}	1.31E+01 ± 2.05E−01≈	1.28E+01 ± 2.63E−01+	1.28E+01 ± 3.86E−01+	1.26E+01 ± 2.69E−01−	1.31E+01 ± 2.57E−01≈	1.33E+01 ± 1.72E−01−	1.30E+01 ± 1.55E−01
−	12	10	15	10	14	16	−−
+	7	8	5	6	3	3	−−
≈	8	9	7	11	10	8	−−

Table 5
Average rankings obtained through Friedman test at $D = 30$.

Algorithms	Average rankings
RNDE	3.00
EPSDE	3.46
DEGL/SAW	3.93
OXDE	4.19
ODE	4.39
MGBDE	4.39
SaDE	4.65

Table 6
Statistical results of RNDE with other competitors based on the multiple-problem Wilcoxon's test.

VS	R^+	R^-	*P*-value	$\alpha = 0.1$
DEGL	229.5	121.5	0.166300	No
SaDE	257.0	121.0	0.099823	Yes
OXDE	256.0	95.0	0.039664	Yes
ODE	270.5	80.5	0.015287	Yes
MGBDE	244.0	107.0	0.079695	Yes
EPSDE	224.0	127.0	0.213315	No

5. Experimental study on RNDE

5.1. Benchmark functions and experimental setting

There are 27 benchmark functions are used in the following experimental studies and the details of these benchmark functions are described in Table 1. All these problems should be minimized. These functions can be classified into two categories. The benchmark functions $f_1 \sim f_{13}$ are commonly used in the evolutionary computation community and introduced by Yao et al. [38]. The remaining 14 benchmark functions $f_{14} \sim f_{27}$ are the first 14 functions proposed for the special session on real-parameter optimization of CEC 2005 by Suganthan et al. [39], and these 14 functions are shifted and exceedingly difficult.

The parameters used in classical DE follow [42] and [40], that is, $NP = 100$, $F = 0.5$, and $CR = 0.9$. The parameters used in RNDE are only needed to consider *NP* and *F*, we use the same parameter settings as the classical DE. In our experimental study, each algorithm independently runs 30 times for each benchmark function and the termination criterion of algorithms, i.e., maximum number of function evaluations(*MaxFEs*), is set to $10,000 \times D$, where *D* is the number of variables. All experiments are carried out on a computer with 3.1 GHz Dual-core Processor and 4 GB RAM based on Windows 10 platform and all programs are implemented in Matlab. For the purpose of having statistically sound conclusions, the experiment result is analyzed on Wilcoxon's rank sum test at a 0.05 significance level.

5.2. Quality of the RNDE

In this section, the proposed RNDE is compared with DE with two classical mutation strategies, i.e., DE/rand/1 and DE/best/1, to investigate the quality of the RNDE. The results achieved by DE/rand/1, DE/best/1, and RNDE at $D = 30$ for the test suite are summarized in Table 2, where "Ave Err" and "Std Dev" indicate the mean and standard deviation of the function error values obtained in 30 runs, respectively. The best results are shown in **boldface**. The Wilcoxon's rank sum test results among RNDE and others are summarized at the bottom of the table, in which "−", "+", and "≈" indicate that the performance of the compared algorithm is worse than, better than, and similar to that of RNDE, respectively. RNDE, respectively. Fig. 2 shows the convergence graphs of DE/rand/1, DE/best/1, and RNDE on nine representative benchmark functions.

Based on the results, RNDE achieves better results than DE/rand/1 and DE/best/1 on the majority of test functions. Compared with DE/rand/1, RNDE is significantly better than it on 18 out of 27 test functions in terms of the solution quality, and similar to it on 8 test functions. DE/rand/1 beat RNDE on only 1 test function. It can be seen from the **boldface**, DE/rand/1 gets the best results on

Table 7
Experimental results of RNDE with $CR = 0.5$ and $CR = 0.9$, and RNDE for all test functions at $D = 30$.

F	RNDE($CR = 0.5$) Ave Err ± Std Dev	RNDE($CR = 0.9$) Ave Err ± Std Dev	RNDE Ave Err ± Std Dev
f_1	3.17E−98 ± 4.04E−98−	5.73E−193 ± 0.00E+00+	2.71E−105 ± 3.04E−105
f_2	2.84E−52 ± 1.82E−52+	4.08E−100 ± 6.61E−100+	1.00E−50 ± 1.77E−50
f_3	1.30E+03 ± 4.30E+02−	3.83E−30 ± 9.88E−30+	7.49E−15 ± 1.56E−14
f_4	3.76E−11 ± 2.07E−11≈	4.82E−07 ± 1.65E−06−	2.10E−09 ± 6.88E−09
f_5	8.59E+00 ± 3.74E−01−	2.66E−01 ± 9.94E−01−	1.56E−14 ± 3.52E−14
f_6	0.00E+00 ± 0.00E+00≈	3.33E−02 ± 1.80E−01≈	0.00E+00 ± 0.00E+00
f_7	3.36E−03 ± 7.78E−04−	1.67E−03 ± 4.36E−04+	2.47E−03 ± 8.07E−04
f_8	3.82E−04 ± 0.00E+00≈	1.04E+03 ± 1.35E+03−	3.82E−04 ± 0.00E+00
f_9	8.93E+01 ± 5.82E+00−	1.33E+01 ± 4.23E+00−	0.00E+00 ± 0.00E+00
f_{10}	3.55E−15 ± 0.00E+00≈	4.26E−15 ± 1.42E−15−	3.55E−15 ± 0.00E+00
f_{11}	0.00E+00 ± 0.00E+00≈	1.40E−03 ± 3.69E−03−	0.00E+00 ± 0.00E+00
f_{12}	1.57E−32 ± 5.47E−48≈	1.62E−32 ± 5.57E−34−	1.57E−32 ± 9.68E−35
f_{13}	1.35E−32 ± 5.47E−48≈	7.32E−04 ± 2.74E−03−	1.35E−32 ± 5.47E−48
f_{14}	0.00E+00 ± 0.00E+00≈	1.14E−14 ± 2.27E−14−	0.00E+00 ± 0.00E+00
f_{15}	6.47E+02 ± 1.46E+02−	8.53E−14 ± 3.52E−14+	1.53E−13 ± 8.19E−14
f_{16}	4.37E+07 ± 1.00E+07−	1.18E+05 ± 5.96E+04≈	1.63E+05 ± 1.00E+05
f_{17}	1.97E+03 ± 6.47E+02−	9.00E−10 ± 4.08E−09+	4.10E−04 ± 6.38E−04
f_{18}	1.07E+03 ± 1.73E+02−	1.47E+01 ± 2.72E+01+	1.22E+02 ± 1.95E+02
f_{19}	1.13E+01 ± 1.11E+01−	1.33E−01 ± 7.16E−01≈	1.33E−01 ± 7.16E−01
f_{20}	3.29E−04 ± 1.77E−03+	6.97E−03 ± 9.46E−03−	1.97E−03 ± 4.65E−03
f_{21}	2.09E+01 ± 5.49E−02≈	2.09E+01 ± 4.16E−02≈	2.09E+01 ± 5.10E−02
f_{22}	8.58E+01 ± 7.84E+00−	1.57E+01 ± 8.68E+00−	1.89E−15 ± 1.02E−14
f_{23}	1.79E+02 ± 9.01E+00−	1.68E+02 ± 7.33E+00−	1.12E+02 ± 1.20E+01
f_{24}	3.93E+01 ± 1.39E+00−	3.93E+01 ± 1.02E+00−	3.24E+01 ± 1.02E+00
f_{25}	8.95E+04 ± 6.89E+04−	1.69E+03 ± 2.00E+03≈	4.21E+03 ± 1.12E+04
f_{26}	1.22E+01 ± 9.10E−01−	1.13E+01 ± 3.70E+00−	4.11E+00 ± 2.98E−01
f_{27}	1.33E+01 ± 1.65E−01−	1.32E+01 ± 1.96E−01−	1.30E+01 ± 1.55E−01
−	16	15	−−
+	2	7	−−
≈	9	5	−−

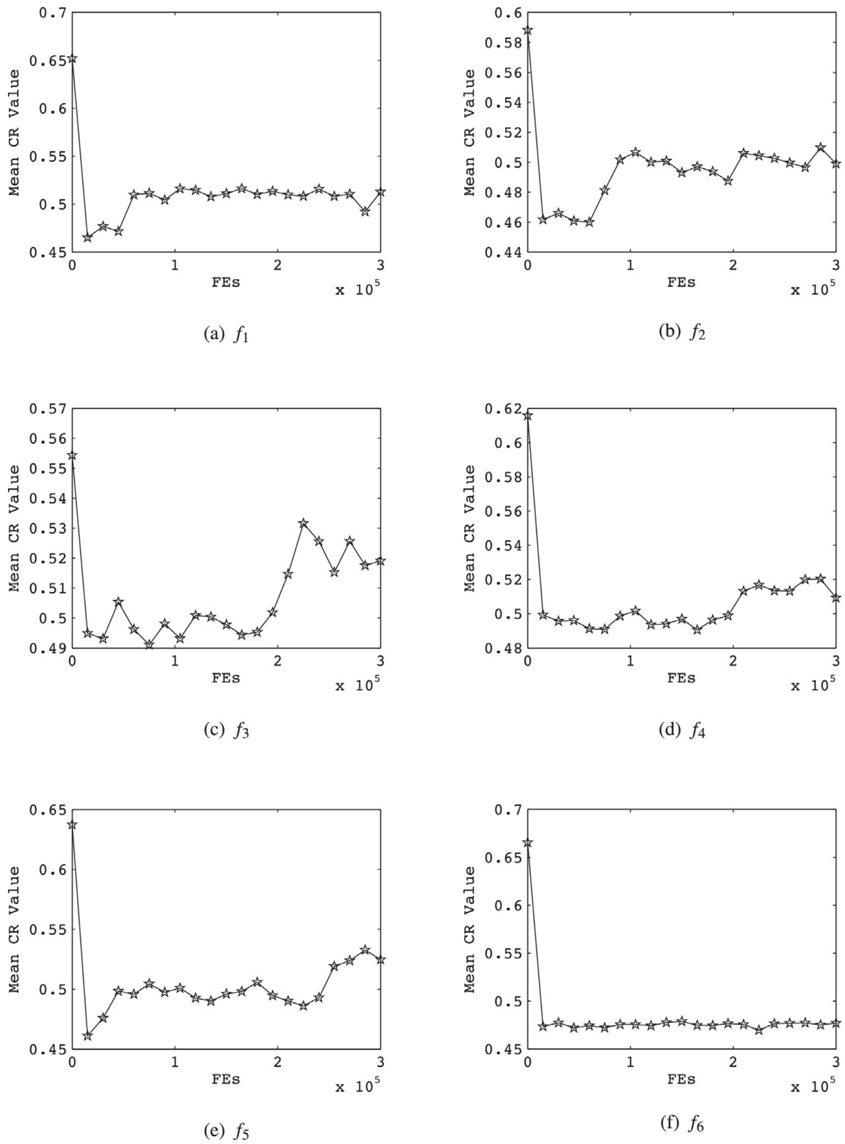


Fig. 3. Distribution curve of CR on six representative test functions.

$f_6, f_{11}, f_{12}, f_{13}, f_{14}, f_{18}, f_{20}$, and f_{21}, but RNDE also gets the **boldface** and the same values of "Ave Err" and "Std Dev" on f_6, f_{11}, f_{13}, and f_{14}. Moreover, on f_{12} and f_{21}, RNDE get the same values of "Ave Err", just the values of "Std Dev" are worse than the DE/rand/1. Compared with DE/best/1, although DE/best/1 performs better than RNDE on 6 test functions, the f_1, f_2, and f_3 are the unimodal problems, the f_{16} is the shifted unimodal functions, and the f_{24} and f_{27} are two simple shift multimodal functions. Owing to the fact that the DE/best/1 as a greedy strategy have better exploitation ability for these simple unimodal and multimodal problems. As seen, RNDE is significantly beat DE/best/1 on 19 out 27 test functions. By careful examination of the results and analysis, we can conclude that RNDE greatly improves the performance of DE.

In addition, we also present a scalable test of RNDE, DE/rand/1 and DE/best/1 on the test suite for $D = 50$ and 100. The values of mean error and comparison results between RNDE and them based on Wilcoxon's rank sum test are summarized in Table 3. Based on the results, RNDE outperforms DE/rand/1 on 17 and 17 out of 27 test functions for $D = 50$ and 100, while RNDE beats DE/best/1 on 17 and 19 out of 27 test functions for $D = 50$ and 100. When the dimension increases from 50 to 100, the performance of RNDE has not affected by the increase of dimension.

5.3. Comparison with other state-of-the-art DE variants

In this section, RNDE compares with six other state-of-the-art DE variants, including DEGL/SAW [33], EPSDE [28], MGBDE [40], OXDE [41], SaDE [27], and ODE [42]. In EPSDE and SaDE, the parameter F and CR are self-adaptive and free to set. To have a fair comparison, we used the same parameter settings as described in their original literatures for these six competitors, but the maximum number of function evaluations (*MaxFEs*) in all of these algorithms is set to $10,000 \times D$. The details of the involved algorithms are listed as follows.

(1) MGBDE: $NP = 100$, $F = 0.5$, $CR = 0.9$
(2) OXDE: $NP = D$, $F = 0.9$, $CR = 0.9$
(3) SaDE: $NP = 50$, $LP = 50$
(4) ODE: $NP = 100$, $F = 0.5$, $CR = 0.9$, $Jr = 0.3$
(5) EPSDE: $NP = 50$
(6) DEGL/SAW: $NP = 10 \times D$, $\alpha = \beta = F = 0.8$, $CR = 0.9$

The mean errors and the standard deviations of the seven DE variants on each test function at $D = 30$ are summarized in Table 4. All the experimental results are achieved from 30 independent runs. In Table 4, it is obvious that RNDE is the best among the seven methods on the test suite. It is significantly better than DEGL/SAW, EPSDE, MGBDE, SaDE, ODE, and OXDE on 12, 10, 15, 10, 14, and 16 test functions, respectively, but they beat RNDE on only 7, 8, 5, 6, 3, and 3, respectively.

Moreover, Table 5 presents average rankings of these competitors achieved by Friedman test to compare the performance of multiple algorithms on the test suite. The average ranking value smaller indicates better. Observing from the statistical results, the performance of these contestant algorithms at $D = 30$ can be sorted into the following order: RNDE, EPSDE, DEGL/SAW, OXDE, ODE, MGBDE, and SaDE. The proposed RNDE has get the best average ranking, which wins the other six algorithms.

Furthermore, the multiple-problem Wilcoxon's test [43] by using the KEEL software [44] is employed to show the significant differences between RNDE and the competitors, and the statistical analysis results are shown in Table 6. As seen, RNDE provides higher R^+ values than R^- values in all the cases. According to the Wilcoxon's test at $\alpha = 0.1$, RNDE is significantly better than SaDE, OXDE, ODE, and MGBDE.

5.4. Validity analysis of the adaptive shift parameter setting

As pointed out in Section 4.2, the adaptive shift parameter setting is proposed to dynamically update CR in RNDE. The aim of the subsection is to analyze the role of the adaptive shift strategy on the performance of RNDE. To this end, two additional experiments are done. First, we construct two simplified versions of RNDE, which used $CR = 0.5$ and $CR = 0.9$ to replace the adaptive shift strategy in the RNDE, respectively, all the other parts being the same as in the original version of RNDE. Table 7 presents the experimental results of RNDE with $CR = 0.5$ and $CR = 0.9$, and RNDE on the testsuite at $D = 30$ and statistical results between RNDE and the two compared versions based on Wilcoxon's rank sum test. Second, we record the mean values of CR at each iteration in the evolution process for each test function. Fig. 3 shows the curves of CR on six representative test functions.

Table 7 shows that RNDE is significantly better than RNDE($CR = 0.5$) and RNDE($CR = 0.9$) on 16 and 15 test functions, respectively. But the two simplified versions win on only 2 and 7 test functions, respectively. These experimental statistical results imply that the adaptive shift strategy outperforms the fixed values method. This happens because it has the capability to adapt the search for different test functions. Moreover, once the suitable setting has been used, it can accelerate the convergence speed and enhance the convergence accuracy. In addition, by carefully reviewing Fig. 3, one can note that the curves of CR have nothing in common with each other on six representative test functions. Owing to the fact that different test functions have different suitable value on different evolution stages.

Therefore, from the above experimental results and analysis, we can conclude that the adaptive shift strategy can effectively improve the performance of RNDE.

6. Conclusion

DE is a popular and powerful population-based evolutionary algorithm. However, the imbalance between the exploitation and exploration is greatly limit the optimization performance. In classical DE, there are two mutation strategies in DE, i.e., DE/rand/1 and DE/best/1. The DE/rand/1 has better exploration and poor exploitation, on the contrary, the DE/best/1 has better exploitation and poor exploration. In order to solve this problem, an enhancing differential evolution with DE/neighbor/1 (RNDE) is proposed, in which the DE/neighbor/1 strategy is introduced to balance the exploration and exploitation ability of DE.

The performance of RNDE is evaluated on the basis of a set of benchmark test functions while compared with other state-of-the-art DE variants. The quality of RNDE is verified by comparing with two classical DE (i.e., DE/rand/1 and DE/best/1). RNDE is also compared with six state-of-the-art DE variants (i.e., DEGL/SAW, EPSDE, MGBDE, SaDE, ODE, and OXDE) and the results show that RNDE is the best among these competitors. In addition, validity of the adaptive shift parameter setting is experimentally studied.

In the future, how to incorporate our work to other evolutionary algorithms and apply the efficient RNDE to solve the complex real-world problem in various fields, such as steganography [45,46] and deduplication [47] remains an attractive topic.

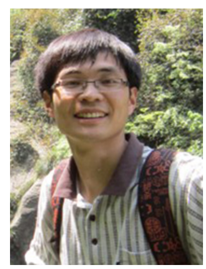
Acknowledgements

The authors would like to thank the editor and anonymous reviewers for their detailed and constructive comments that help us to increase the quality of this work. This work is supported by the National Natural Science Foundation of China (Nos. 61364025, 61662029), the Science and Technology Plan Projects of Jiangxi Provincial Education Department (Nos. GJJ161076, GJJ161072).

References

[1] Y. Zheng, B. Jeon, D. Xu, Q. Wu, H. Zhang, Image segmentation by generalized hierarchical fuzzy C-means algorithm, J. Intell. Fuzzy Syst. 28 (2) (2015) 961–973.
[2] Q. Liu, W. Cai, J. Shen, Z. Fu, X. Liu, N. Linge, A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment, Secur. Commun. Netw. 9 (17) (2016) 4002–4012.
[3] Z. Zhili, Y. Ching-Nung, S. Xingming, L. Qi, Q.J. WU, Effective and efficient image copy detection with resistance to arbitrary rotation, IEICE Trans. Inform. Syst. 99 (6) (2016) 1531–1540.
[4] Q. Tian, S. Chen, Cross-heterogeneous-database age estimation through correlation representation learning, Neurocomputing 238 (2017) 286–295.
[5] R. Storn, K. Price, Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, Tech. Rep. TR-95-012, International Computer Science Institute, Berkeley, 1995.
[6] N. Noman, H. Iba, Inferring gene regulatory networks using differential evolution with local search heuristics, IEEE/ACM Trans. Comput. Biol. Bioinform. 4 (4) (2007) 634–647.
[7] C. Zhan, W. Situ, L.F. Yeung, P.W.-M. Tsang, G. Yang, A parameter estimation method for biological systems modelled by ODE/DDE models using spline approximation and differential evolution algorithm, IEEE/ACM Trans. Comput. Biol. Bioinform. 11 (6) (2014) 1066–1076.
[8] Y. Wang, B. Li, T. Weise, Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems, Inform. Sci. 180 (12) (2010) 2405–2420.
[9] J. Zhao, Y. Xu, F. Luo, Z. Dong, Y. Peng, Power system fault diagnosis based on history driven differential evolution and stochastic time domain simulation, Inform. Sci. 275 (2014) 13–29.
[10] I. De Falco, A. Della Cioppa, D. Maisto, E. Tarantino, Differential evolution as a viable tool for satellite image registration, Appl. Soft Comput. 8 (4) (2008) 1453–1462.
[11] S. Paul, S. Das, Simultaneous feature selection and weighting-an evolutionary multi-objective optimization approach, Pattern Recogn. Lett. 65 (2015) 51–59.
[12] J. Sun, Q. Zhang, E.P. Tsang, DE/EDA: a new evolutionary algorithm for global optimization, Inform. Sci. 169 (3) (2005) 249–262.
[13] M.G. Omran, A.P. Engelbrecht, A. Salman, Bare bones differential evolution, Eur. J. Oper. Res. 196 (1) (2009) 128–139.
[14] W. Gong, Z. Cai, C.X. Ling, DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization, Soft Comput. 15 (4) (2010) 645–665.
[15] A.R. Yildiz, A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations, Appl. Soft Comput. 13 (3) (2013) 1561–1566.
[16] W. Xie, W. Yu, X. Zou, Diversity-maintained differential evolution embedded with gradient-based local search, Soft Comput. (2012) 1–25.
[17] H. Peng, Z. Wu, Heterozygous differential evolution with Taguchi local search, Soft Comput. 19 (11) (2015) 3273–3291.
[18] J. Zhong, W. Cai, Differential evolution with sensitivity analysis and the Powell's method for crowd model calibration, J. Comput. Sci. 9 (2015) 26–32.
[19] H.-Y. Fan, J. Lampinen, A trigonometric mutation operation to differential evolution, J. Glob. Optim. 27 (1) (2003) 105–129.

[20] W. Gong, Z. Cai, Differential evolution with ranking-based mutation operators, IEEE Trans. Cybernet. 43 (6) (2013) 2066–2081.
[21] J. Wang, J. Liao, Y. Zhou, Y. Cai, Differential evolution enhanced with multiobjective sorting-based mutation operators, IEEE Trans. Cybernet. 44 (12) (2014) 2792–2805.
[22] G. Jia, Y. Wang, Z. Cai, Y. Jin, An improved $(\mu+\lambda)$-constrained differential evolution for constrained optimization, Inform. Sci. 222 (2013) 302–322.
[23] H. Sharma, J.C. Bansal, K. Arya, Self balanced differential evolution, J. Comput. Sci. 5 (2) (2014) 312–323.
[24] L. Tang, Y. Dong, J. Liu, Differential evolution with an individual-dependent mechanism, IEEE Trans. Evol. Comput. 19 (4) (2015) 560–574.
[25] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, Soft Comput. 9 (6) (2005) 448–462.
[26] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[27] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[28] R. Mallipeddi, P.N. Suganthan, Q.-K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679–1696.
[29] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evol. Comput. 15 (1) (2011) 55–66.
[30] R. Sarker, S. Elsayed, T. Ray, Differential evolution with dynamic parameters selection for optimization problems, IEEE Trans. Evol. Comput. 18 (5) (2014) 689–707.
[31] Y. Wang, H.-X. Li, T. Huang, L. Li, Differential evolution based on covariance matrix learning and bimodal distribution parameter setting, Appl. Soft Comput. 18 (5) (2014) 232–247.
[32] Z. Guo, G. Liu, D. Li, S. Wang, Self-adaptive differential evolution with global neighborhood search, Soft Comput. (2016) 1–10.
[33] S. Das, A. Abraham, U.K. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE Trans. Evol. Comput. 13 (3) (2009) 526–553.
[34] J. Liao, Y. Cai, T. Wang, H. Tian, Y. Chen, Cellular direction information based differential evolution for numerical optimization: an empirical study, Soft Comput. 20 (7) (2016) 2801–2827.
[35] H.A. Pham, Reduction of function evaluation in differential evolution using nearest neighbor comparison, Viet. J. Comput. Sci. 2 (2) (2015) 121–131.
[36] S. Das, P.N. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.
[37] S. Das, S.S. Mullick, P. Suganthan, Recent advances in differential evolution-an updated survey, Swarm Evol. Comput. 27 (2016) 1–30.
[38] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (2) (1999) 82–102.
[39] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, KanGAL Report 2005005, 2005.
[40] H. Wang, S. Rahnamayan, H. Sun, M.G. Omran, Gaussian bare-bones differential evolution, IEEE Trans. Cybernet. 43 (2) (2013) 634–647.
[41] Y. Wang, Z. Cai, Q. Zhang, Enhancing the search ability of differential evolution through orthogonal crossover, Inform. Sci. 185 (1) (2012) 153–177.
[42] S. Rahnamayan, H.R. Tizhoosh, M.M. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (1) (2008) 64–79.
[43] S. García, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms behaviour: a case study on the CEC 2005 special session on real parameter optimization, J. Heurist. 15 (6) (2009) 617–644.
[44] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesús, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, et al., KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Comput. 13 (3) (2009) 307–318.
[45] Z. Xia, X. Wang, X. Sun, Q. Liu, N. Xiong, Steganalysis of LSB matching using differences between nonadjacent pixels, Multimedia Tools Appl. 75 (4) (2016) 1947–1962.
[46] C. Yuan, Z. Xia, X. Sun, Coverless image steganography based on SIFT and BOF, J. Internet Technol. 18 (2) (2017) 435–442.
[47] J. Li, X. Chen, M. Li, J. Li, P.P. Lee, W. Lou, Secure deduplication with efficient and reliable convergent key management, IEEE Trans. Parallel Distrib. Syst. 25 (6) (2014) 1615–1625.



Hu Peng was born on 23rd December 1981. He received the B.Sc. in computer science and technology from Hunan Normal University, Changsha, PR China, in 2004, the M.Sc. in computer technology from the Huazhong University of Science and Technology, Wuhan, PR China, in 2008 and the Ph.D. in computer software and theory from Wuhan University, Wuhan, PR China, in 2016. His areas of interest are solely on evolutionary computation and its real-world applications. His areas of excellence resulted with two publications till date. Those are as follows: (a) Heterozygous differential evolution with Taguchi local search, Soft Computing 19(11) (2015) 3273–3291, and (b) Dynamic differential evolution algorithm based on elite local learning, Acta Electronic Sinica 42(8) (2014) 1522–1530 (in Chinese).