Comp. Appl. Math. (2017) 36:433–458
DOI 10.1007/s40314-015-0237-0

A hybrid optimization algorithm based on chaotic differential evolution and estimation of distribution

Fuqing Zhao[1,2] · Zhongshi Shao[1] · Junbiao Wang[2] · Chuck Zhang[3]

Received: 31 December 2013 / Revised: 27 April 2015 / Accepted: 28 April 2015 /
Published online: 15 May 2015

Abstract Estimation of distribution algorithms (EDAs) and differential evolution (DE) are two types of evolutionary algorithms. The former has fast convergence rate and strong global search capability, but is easily trapped in local optimum. The latter has good local search capability with slower convergence speed. Therefore, a new hybrid optimization algorithm which combines the merits of both algorithms, a hybrid optimization algorithm based on chaotic differential evolution and estimation of distribution (cDE/EDA) was proposed. Due to its effective nature of harmonizing the global search of EDA with the local search of DE, the proposed algorithm can discover the optimal solution in a fast and reliable manner. Chaotic policy was used to strengthen the search ability of DE. Meantime the global convergence of algorithm was analyzed with the aid of limit theorem of monotone bounded sequence. The proposed algorithm was tested through a set of typical benchmark problems. The results demonstrate the effectiveness and efficiency of the proposed cDE/EDA algorithm.

Keywords Hybrid optimization · Estimation of distribution algorithm · Chaotic differential evolution algorithm · Convergence · Global optimization

Mathematics Subject Classification 90B40

Communicated by Natasa Krejic.

✉ Fuqing Zhao
fzhao2000@hotmail.com

1 School of Computer and Communication Technology, Lanzhou University of Technology, Lanzhou 730050, China

2 Key Laboratory of Contemporary Design and Integrated Manufacturing Technology, Ministry of Education, Northwestern Polytechnical University, 710072 Xi'an, China

3 H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

1 Introduction

Estimation of distribution algorithms (EDAs) is one of the most popular evolutionary algorithms (Larrañaga and Lozano 2002). EDAs explicitly learn and build a probabilistic model to capture the parental distribution, and then sample new solutions from the probabilistic model (Pelikan et al. 2002). EDAs are good at the automatic discovery and exploitation of problem regularities. A population evolves with intensive communication among all of the promising individuals, readily extracting promising region of search space and discovering the global optimum (Pelikan et al. 2002; Ahn 2006). Thus, EDAs are regarded as a strong-cooperative search method. Many researchers have already done much research in the field of combining EDAs with other optimum algorithms in recent years. Liu et al. (2011) has introduced the thought of EDA to BFA to increase the diversity of the population and improve the convergence speed. Liu et al. (2008) introduced People-based Incremental Learning (PBIL) into binary ant colony algorithm to improve convergence speed and stability. Wang et al. 2012 incorporated forward-backward iteration and a permutation-based local search method into the EDA to enhance the exploitation ability. Santana et al. (2008) combined a neighborhood search method with EDA. Chang et al. (2010) combined EDA with immune algorithm to proposed IEDA. Ahn et al. (2012) combined binary PSO with EcGA to proposed EcPSO. Tzeng et al. (2012) integrated EDA with ACS to solve permutation flow shop scheduling. Abdollahzadeh et al. (2012) incorporated EDA and genetic algorithm (GA) to construct a parallel hybrid algorithm named GA-EDA.

Differential evolution algorithm (DE) (Storn and Price 1997) is an optimization algorithm based on the theory of swarm intelligence. DE solves the optimum problems through competition and cooperation of the population (Price et al. 2005). The operations of DE algorithm include mutation, crossover and selection (Storn and Price 1997). DE algorithm is easily carried out, has good local search capability, memories the best solution of all individuals and shares the internal information of the population. But DE algorithm has slow convergence speed. Currently, many researchers have introduced the chaotic strategies to DE algorithm to improve its performance. Guo et al. (2007) proposed PCDE which combined DE and chaos searching concurrently. Jia et al. (2011) introduced a chaotic local search (CLS) into DE to present DECLS. Zhang et al. (2013) integrated chaotic sequences with multi-objective differential evolution to propose CS-MODE. Bedri (2010) used chaotic mapping to initialize DE to present CIDE. Senkerik et al. (2013) introduced chaos pseudo-random number as the random mechanism of DE to maintain the diversity of the population. Coelho et al. (2014) used chaotic sequence to improve mutation factor of DE to contribute to escape more easily from local minima. Asafuddoula et al. (2014) used various proven strategies to improve the performance of DE to propose an adaptive hybrid DE algorithm (AH-DEa). Hemmati et al. (2014) introduced a logistic map into DE crossover operator to constructed multi-cross operator so as to enhance the search efficiency of DE. Thus, the chaotic strategies can effectively improve the performance of DE.

In recent years, some efforts to effectively combine the strengths of some optimization algorithms have been tried. For instance, Sun et al. (2005) proposed a combination algorithm of DE and EDA (DE/EDA) for the global continuous optimization problem. Bai et al. (2012) proposed an improved hybrid differential evolution-estimation of distribution algorithm (IHDE-EDA) which took full advantage of differential information and global statistical information of EDA. In IHDE-EDA, Univariate Marginal Distribution Algorithm continuous (UMDAc) (Larrañaga et al. 2000) is extended, Gaussian mixture distribution is used to probability distribution and annealing mechanism is used to determine the weight coefficient of Gaussian mixture distribution. Xiangman and Lixin (2013) proposed a novel hybrid DE-EDA algorithm for dynamic optimization problems, which can make use of the global information and the local information sufficiently. Wang and Li (2013) combined differential evolution (DE) and harmony search (HS) for the solving non-convex economic load dispatch problems. DE enhances the exploitation ability of harmony search and HS enhances the exploration ability of evolution search. Nguyen et al. (2014) proposed a hybrid algorithm based on a particle swarm optimization (PSO) and chemical reaction optimization. Li and Yin (2014) combined the exploration of differential evolution (DE) with the exploitation of the artificial bee colony effectively to solve parameter estimation for chaotic systems. Xiao et al. (2014) combined a one-level membrane structure with Particle Swarm Optimization algorithm (PSO) to propose a hybrid membrane evolutionary algorithm (HMEA). Gao et al. (2013) proposed a hybrid iteration algorithm by combing the exploration of gravitational search algorithm with the exploitation of clonal selection. From these references, it can be found that the hybrid algorithms usually incorporate the global search ability of some algorithms with the local search ability of other algorithms. The combined algorithms have good robustness and optimization efficiency.

Therefore, this paper imports the chaotic parameter control (CPC) strategy of reference (Wang et al. 2009) to DE to construct chaotic differential evolution algorithm (cDE). Then, a new hybrid optimum algorithm is proposed by combining cDE and EDA, which named a hybrid optimization algorithm based on chaotic differential evolution and estimation of distribution (cDE/EDA). All algorithms are used to product the individuals of the new population in the process of evolution. In order to obtain better results, the proportion of the way of generating individuals is adjusted by the a proposed decisive factor so as to effectively develop the global search of EDA and offset the shortcomings of EDA through cDE algorithm which has strong local search ability.

The rest of the paper is organized as follows. Section 2 provides some knowledge of two sub-component optimum algorithms including EDA and cDE. Section 3 presents cDE/EDA in detail. Performance tests are conducted in Sect. 4. Finally, conclusions are given in Sect. 5.

2 Sub-component optimum algorithms

2.1 Sub-optimum algorithms 1: estimation of distribution algorithm

EDAs employ probabilistic models to describe the promising area in the solution space and use these models to guide producing the candidate solutions of the next generation. In order to reduce the complexity of learning the probabilistic model and maintain the diversity and convergence speed, a univariate EDA named Continuous People-based Incremental Learning (PBILc) (Sebag and Ducoulombier 1998) is selected. It has good global search and performs well on convergence speed and final accuracy on unimodal problems. EDA operates as follows:

Step 0: Initialization.

Step 0.1: Randomly initialize the population $\boldsymbol{X}_0$ of size *NP*.
Step 0.2: Set $t = 0$, where t is the number of the current generation.

Step 1: Compute the mean μ_j^t and standard deviation σ_j^t of the current population $\boldsymbol{X}_t$ by Eqs. (1) and (2). $x_i^t = (x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots x_{iD})$ is the ith individual, $i = 1, 2, \ldots, NP, j = 1, 2, \ldots, D$, where D is the number of dimensions, and *NP* is the population size.

$$\mu_j^t = \frac{\sum_{i=1}^{NP} x_{ij}}{NP} \tag{1}$$

$$\sigma_j^t = \sqrt{\frac{\sum_{i=1}^{NP} \left(x_{ij} - \mu_j^t\right)^2}{NP}} \tag{2}$$

Step 2: Truncation selection method is employed to select appropriate candidate solutions to compute standard deviation σ_j^k.
Step 3: Update μ_j^t and σ_j^t with linear learning, which is shown in Eqs. (3) and (4).

$$\mu_j^t = (1-\alpha)\mu_j^t + \alpha\left(x_{best1,j}^t + x_{best2,j}^t - x_{worst,j}^t\right) \tag{3}$$

$$\sigma_j^t = (1-\alpha)\sigma_j^t + \alpha\sigma_j^k, \tag{4}$$

where $x_{best1}^t, x_{best2}^t, x_{worst}^t$ is the best, the second best and worst individuals in the current population, α is learning rate.
Step 4: Build a probability distribution model $P(x)$ with Eq. 5.

$$P_t(x) = \prod_{j=1}^{D} f_{\mathrm{N}}(x_j; \mu_j; \sigma_j) = \prod_{j=1}^{D} \frac{1}{\sqrt{2\pi\sigma_j}} e^{\frac{1}{2}\left(\frac{x_j-\mu_j}{\sigma_j}\right)^2}, \tag{5}$$

where $x = (x_1, x_2, \ldots, x_j, \ldots, x_D)$ is a single individual, $j = 1, 2, \ldots, D$.
Step 5: Sample the new population $\boldsymbol{X}_{t+1}$ according to the probabilistic model $P_t(\boldsymbol{x})$.
Step 6: Set $t = t + 1$.
Step 7: If termination condition is not met, go to Step 1; otherwise end EDA.

In EDA, μ_j and σ_j are the parameters of the model. μ_j restricts center of the population. σ_j controls the diversity of the population and the convergence speed of evolution. EDA controls the exploration–exploitation tradeoff by the strategies of Eqs. (3) and (4). The single parent does not jump directly to a desirable location, but rather makes a very small step toward this desirable location, thus the population will maintain a long-term memory which can provide cautious decisions to the later period of evolution. We choose truncation selection method (Heinz and Dirk 1993) to select special individuals in Step 2. A detailed illustration for this selection method has been displayed as following.

In truncation selection, the candidate solutions are ordered by fitness, and *truncparam* (e.g., *truncparam* = 1/2, 1/3, etc.) of the fittest individuals are selected and reproduced $\lceil 1/truncparam \rceil$ times. Then a large population will be produced, whose size is great than the original population. *NP* candidate solutions will be randomly selected from this large population. In fact, the way of truncation selection only increases the selection probability of the best and the middle candidate solutions. Of course, the poor candidate solutions really be weeded out. Therefore, the obtained new population includes the best and the middle candidate solutions. This new recombination population is used to construct probability model in our EDA. The reason for employing this selection method is that, on the one hand, the best candidate solutions are used to speedup population evolution. On the other hand, the middle candidate solutions are utilized to maintain diversity. Meanwhile, in the process of constructing the probability model, EDA has taken into consideration the worst-quality candidate solutions in Step 3. To further explain the procedure, we give an example (dimension = 5, population size = 10, p = *truncparam* = 1/3) in Fig. 1.

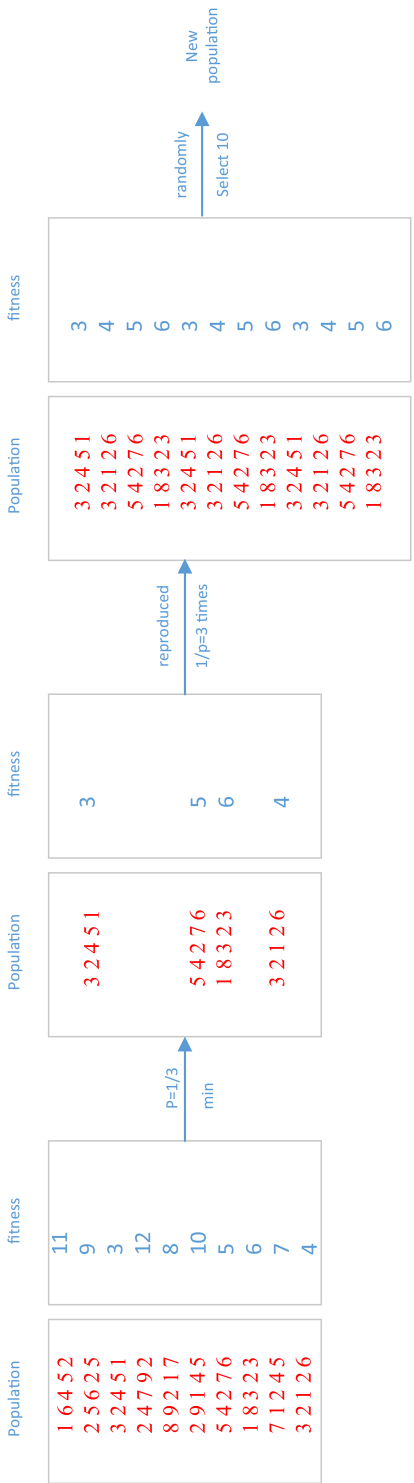


Fig. 1 The procedure of truncation selection method

2.2 Sub-optimum algorithms 2: chaotic differential evolution algorithm

Differential evolution algorithm (DE) has credible robustness and local search capability (Brest et al. 2006; Qin and Suganthan 2005). DE makes use of the differential information amongst the population and has experimentally shown very good performance on complex multimodal problems. Thus, DE is used as other sub-optimum algorithm. In order to accelerate the convergence speed and strengthen the search ability of DE, a chaotic strategy named chaotic parameter control (CPC) (Wang et al. 2009) is combined with DE to construct chaotic differential evolution algorithm (cDE). CPC can make DE launch to re-boost the search in rugged region of the fitness landscape and jump local optimum solutions. cDE operates as follows:

Step 0: Initialization.

Step 0.1: Randomly initialize the population $\boldsymbol{X}_0$ of size *NP*.
Step 0.2: Set $t = 0$, where t is the number of the current generation.
Step 0.3: Initialize CR_0 and γ_0.

Step 1: Update parameters.

Step 1.1: Update the scale factor F.

$$F_{t+1} = 4 \times CR_t \times (1 - CR_t) \tag{6}$$

Step 1.2: Update the crossover rate *CR*.

$$CR_{t+1} = 4 \times F_{t+1} \times (1 - F_{t+1}) \tag{7}$$

Step 1.3: Update the differential decisive factor γ.

$$\gamma_{t+1} = 4 \times \gamma_t \times (1 - \gamma_t) \tag{8}$$

for i = 1:*NP*

Step 2: Mutation. If $rand > \frac{\gamma_{t+1}}{2}$,

$$\boldsymbol{v}_i = \boldsymbol{x}_{r_1}^t + F_{t+1} \times (\boldsymbol{x}_{r_2}^t - \boldsymbol{x}_{r_3}^t) \tag{9}$$

else

$$\boldsymbol{v}_i = (F_{t+1} + 0.5) \times \boldsymbol{x}_d^t + (F_{t+1} - 0.5) \times \boldsymbol{x}_i^t + F_{t+1} \times \left(\boldsymbol{x}_b^t - \boldsymbol{x}_c^t\right), \tag{10}$$

end
where $\boldsymbol{x}_{r_1}^t$, $\boldsymbol{x}_{r_2}^t$, $\boldsymbol{x}_{r_3}^t$ are randomly selected from the current population and $r_1 \neq r_2 \neq r_3 \neq i$. $\boldsymbol{x}_d^t$ is the best solution of the current population, $(\boldsymbol{x}_b^t - \boldsymbol{x}_c^t)$ is a random differential vector with $b \neq c$.
Step 3: Crossover.

$$u_{ij} = \begin{cases} v_{ij}^t, & \text{if } rand < CR_{t+1} \\ x_{ij}^t, & \text{otherwise,} \end{cases} \tag{11}$$

where $\boldsymbol{x}_i^t = \{x_{i1}, x_{i2}, \ldots, x_{ij}, \ldots, x_{iD}\}$ is the ith individual of the current population, D is the number of dimension, $\boldsymbol{u}_i^t = \{u_{i1}, u_{i2}, \ldots, u_{ij}, \ldots, u_{iD}\}$ is the crossover individual.
Step 4: Selection.

$$\boldsymbol{x}_i^{t+1} = \begin{cases} \boldsymbol{u}_i^t, & \text{if } f\left(\boldsymbol{u}_i^t\right) < f(\boldsymbol{x}_i^t) \\ \boldsymbol{x}_i^t, & \text{otherwise} \end{cases} \tag{12}$$

where $f(\bullet)$ is the fitness value.
end for
Step 5: If termination condition is not met, let $t = t + 1$ go to Step 1; otherwise end DE.

Generally speaking, one mutation operator more or less lacks diversity or low convergence speed. Thus, two effective differential mutative strategies are adopted in cDE, which are Eq. (9) (Das and Suganthan 2011) and Eq. (10) (Price 1997) denoted by DE_1 and DE_2, respectively. DE_1 combines 3 random individuals in population. It can exploit the local information in current population. DE_1 can effectively maintain the diversity of the population. It is apparently that DE_1 is used under a large probability, due to its university. It can be seen from Eq. (10) that DE_2 combines 4 vectors. When F is close to 0.5, v_i will be equal with $x_d + F \times (x_b - x_c)$, which makes population approach the best solution of the current population. When $F$is close to –0.5, v_i will be equal with $x_i + F \times (x_b - x_c)$, which is beneficial to search the neighbor of the current individual. Due to CPC, F is greater than 0. It is can be observed in cDE that v_i approximates $x_d + F \times (x_b - x_c)$. This is because $(F - 0.5) \times x_i$ is not equal 0. We utilize the accelerating ability of DE_1 to accelerate the convergence speed. Meanwhile, $(F - 0.5) \times x_i$ ensures the convergence speed is not excessively fast. Therefore, combining with these mutation operators can further enhance performance of the proposed algorithm to a great extent. Probability of both strategies related to an adaptive chaotic control way, which is expressed as Eq. (11). F and CR are updated by Eqs. (7) and (8). It is observed that $F$and CR values spread between [0, 1] in the whole region throughout the search process. Such updating parameter way of chaotic strategy aims at restricting the mutation rate appropriately.

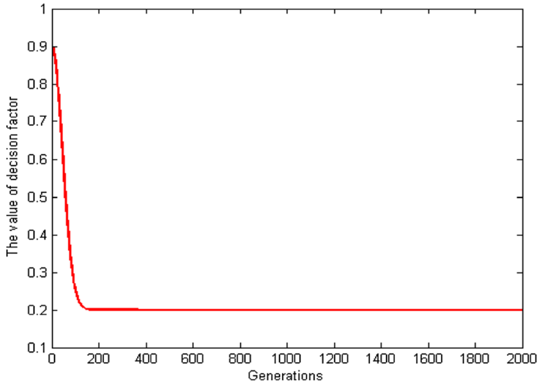
3 The hybrid optimum algorithm

EDA builds the model through probability so that global information can fully be utilized to generate the new candidate solutions and the convergence speed can be accelerated. But it can be easily trapped in local optimum for complex multimodal problems in the later evolution. DE can jump local optimum through the differential information of variables, but it has slow convergence speed. With these considerations, a new kind of hybrid optimum algorithm, a hybrid optimum algorithm-based chaotic differential evolution and estimation of distribution algorithm (cDE/EDA in short) is proposed. It combines the different ways of generating offspring and adopts the different strategies are in the different period of evolution. CPC is introduced into DE to strengthen search. The following section presented the decisive factor, and then Sect. 3.2 explains the proposed algorithm in detail. In Sect. 3.3, the convergence of the proposed algorithm is analyzed.

3.1 The decisive factor

Through the decisive factor p most of individuals in the population are generated by EDA in the initial evolution. EDA forces the search region to quickly approach a promising area as close as possibly. The minority individuals are generated by cDE for the sake of maintaining the diversity of the population. These strategies provide a promising area for the later evolution. In the middle of evolution or later, most of individuals in population are generated by cDE with the decisive factor p in order to ensure exact search. Nevertheless, as the poor ability of global search of cDE, EDA generates a little of individuals to maintain the global information and increases their search accuracy. Thus p is adjusted with Eq. (12). When p is larger, EDA plays a leading role. Then p will be gradually lowered in the process of evolution so that cDE can occupy dominant position.

Fig. 2 The variation curve of the decisive factor



$$p_{(t+1)} = p_{\min} + \left(1 - \frac{t}{G_{\max}}\right)\left(p_{(t)} - p_{\min}\right), \tag{13}$$

where t is the current population, $G_{\max}$ is the maximum value of iteration. $p_{\min}$ and $p_{\min}$ are the upper and lower bound, respectively.

Figure 2 shows that p becomes smaller with increasing evolutionary iteration and the probability of EDA become smaller, while cDE become larger. Due to the large number of population, the way of generating individuals which is selected by the probability can make the proportion of each algorithm be well in line with statistical regularity. Thus, the proposed algorithm can resist randomness and the search results are more stable.

3.2 The procedure of cDE/EDA

cDE/EDA combines the merits of the two algorithms. One part of trial individuals is generated from the cDE mutation and crossover. But the other part of the trial individuals is sampled in the search space from the constructed probability distribution model. Therefore, the population generated by the cDE/EDA generation scheme is based on the local information and global statistical information. The decisive factor p is used to balance contributions of the global information and the local information. The following area presents the procedure of cDE/EDA in detail.

Step 0: Initialization

Step 0.1: Generate the initial population $\boldsymbol{X}_0 = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{NP})$ so that it can evenly distribute in the solution space, where $\boldsymbol{x}_i = (x_1, \ldots, x_j, \ldots, x_D)$ is an individual, $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, D$. *NP* is the population size. D is the number of dimension. $G_{\max}$ is the maximum number of iteration. The individuals are generated with Eq. (2).

$$x_{ij} = x_j^{\min} + rand \times \left(x_j^{\max} - x_j^{\min}\right), \tag{14}$$

where $x_j^{\max}$ and $x_j^{\max}$ are the upper and lower bound of ith dimension, respectively.
Step 0.2: Initialize the learning rate α, CR_0, γ_0, $p_{\min}$ and $p_{\max}$, and $t = 0$, where t is the number of the current generation.

Step 1: Evaluate the fitness value of each individual in $\boldsymbol{X}_t$.
Step 2: Conduct the probabilistic mode $P_t(\boldsymbol{x})$.
for $i = 1$:*NP*

Step 3: Generate the new individual $X'_t(i)$.

Step 3.0 Generate a random number *rand* in interval [0, 1].
Step 3.1 if $rand < p_t$,
generate the new candidate $X'_t(i)$ by sampling the probabilistic model P_t.
Else
update F_t, CR_t, with γ_t Eqs. (6)–(8).
Generate the new candidate $X'_t(i)$ by DE operators, which include mutation and crossover.
end

Step 4: Evaluate fitness value: Evaluate the fitness value of the individual $X'_t(i)$.
Step 5: Selection.

$$X_{t+1}(i) = \begin{cases} X_t(i), \text{ if } f(X_t(i)) < f(X'_t(i)) \\ X'_t(i), \text{ otherwise} \end{cases} \tag{15}$$

where $f(\bullet)$ is the function of calculating fitness.
end
Step 6: $t = t+1$, if t attains $G_{\max}$, the cDE/EDA is terminated; otherwise, go back to Step 1.

3.3 Convergence analysis of cDE/EDA

From the basic idea of cDE/EDA, we can regard the process of evolution as a series of stochastic sequence. In this paper, the process of cDE/EDA is analyzed with the stochastic sequence; meanwhile, the criterion that is used to prove the convergence of cDE/EDA can be found in Peng and Xie (2012). Meanwhile, we propose two theorems to prove the convergence of cDE/EDA. Due to the limitation of space, we do not present the proved details of Theorem 1 and Theorem 1 here. Readers who are interested in them are directed to "Appendix B". Relative concepts have been listed as follows.

Definition 1 Let $S = I^D$ denote search space of the individuals and $f : S \rightarrow I^+$ is fitness function. I means the domain of definition, $\boldsymbol{x}^*$ stands for the global optimum solution. Then the optimum problem can be described as $\{\boldsymbol{x} \in I \,|\, f(\boldsymbol{x}^*) = \min f(\boldsymbol{x})\}$.

Lemma 1 (Peng and Xie 2012) *If a sequence is monotonous and no ascending as well as has lower bound, it must possess a limit.*

Lemma 2 (Peng and Xie 2012) *If a sequence is monotonous and no ascending as well as has lower bound, its subsequence must possess a limit.*

Theorem 1 *The evolutional direction of population is monotonous, that is* $f(\boldsymbol{X}(n+1)) \leq f(\boldsymbol{X}(n))$, *thus the sequence* $\{f(\boldsymbol{x}^{(n)})\}$ *is monotonous and no ascending as well as has lower bound.*

Theorem 2 *cDE/EDA can converge to global optimum with the probability of 1.*

4 Experiments and results

4.1 Test functions

A test suite with 33 benchmark functions was used in our experimental studies, which are listed in Table 7 in "Appendix A". The first 18 functions are from a classical test suite, which has been widely used to evaluate and analyze EAs' performance and can be classified into three types. Function $f_1 - f_6$ are unimodal. Function f_5 is the step function, which has one optimum and is discontinuous. Function f_6 is noisy quartic function, where *random* [0,1) is a uniformly distribute random variable in interval [0,1). Function $f_7 - f_{13}$ are multimodal functions where the number of local optima increases exponentially with the problem dimension. Function $f_{14} - f_{18}$ are low-dimensional functions which has only a few local optima. A more detailed description of each classical benchmark function can be found in the Appendix of Guo (2012). For further testing the proposed algorithm, we also choose some shifted rotated problems in CEC 2005 (Suganthan et al. 2005), which are named $f_{19} - f_{33}$. These functions are scalable. In order to make them more resistant to simple search tricks, many of them are the shifted and rotated variants of the classical functions. The graphic models of all functions are displayed in Fig. 3 in "Appendix A".

4.2 Experimental setup

cDE/EDA was compared with PBILc Sebag and Ducoulombier (1998) and the classical DE (He and Yang 2012). The learning rate of PBILc was set to 0.2 and the selection rate to 0.1 (Sebag and Ducoulombier 1998). For DE, the scale factor F was set to 0.5 and the crossover factor CR to 0.6 (Gämperle et al. 2002). As for function $f_1 - f_{13}$ and $f_{19} - f_{33}$, we set the dimension size D to 30, the population size N to 150 for all algorithms, the maximum iteration $G_{\max}$ to 2000; for low-dimensional function $f_{14} - f_{18}$, the maximum iteration $G_{\max}$ and the population size NP is set to 500 and 40, respectively. Value to Reach (VTR) Hansen et al. (2010) is set to 1.0e−08 for all functions except f_6, which is a noise function and $VTR =$ 1.0e−0.2 is used. The shifted rotated problems' VTR can be found in Suganthan et al. (2005). Test environment: CPU core i5, RAM 2GB, Windows 7, Matlab R2010b.

4.3 Parameters setting

The proposed cDE/EDA contains several key parameters: CR_0 (Crossover factor), α (Learning rate), γ_0 (The differential factor), *truncparam* (Selective probability),$p_{\min}$ (The lower bound of the decisive factor), $p_{\max}$ (The upper bound of the decisive factor). In these parameters, CR_0 and γ_0 are from CPC in reference Wang et al. (2009), so their setting is based on reference Wang et al. (2009). To investigate the influence of the rest of the 4 parameters on the performance of the cDE/EDA, we implement the Taguchi method of design of experiment (DOE) (Montgomery et al. 2008). Each parameter is assigned 4 levels, which is displayed in Table 1. According to the number of parameter and the number of level, we choose the orthogonal $L(4^4)$. Function f_7 is selected as a moderate evaluate function. This is because it has many peaks and valleys. For each parameter combination, cDE/EDA is run 10 times independently and the average value is calculated as the average response variable (ARV) value. The orthogonal array and the obtained ARV values are listed in Table 2. Then, we figure out the response value of each parameter and analyze significance rank in Table 3 according to 2. According to Table 3, we illustrate the trend of each factor level in Fig. 2.

It can be observed from Table 3 that α is the most significant one among the four parameters. A large value of α could lead to premature convergence. A small value of α could lead to low convergence. *truncparam* ranks the second, which implies that the number of the superior sub-population to update the probability model is also important. A small value of *truncparam* can help the algorithm build an accurate model. However, a large of *truncparam* could lead to increase computational budget and slow the convergence speed. Besides, it can be seen from Table 3 that the ranks of $p_{\min}$ and $p_{\max}$ are 3 and 4, respectively. The performance

Table 1 Values of parameter

Parameters	Level			
	1	2	3	4
α	0.05	0.10	0.20	0.30
truncparam	0.10	0.20	0.30	0.40
$p_{\min}$	0.00	0.10	0.20	0.30
$p_{\max}$	0.70	0.80	0.90	1.00

Table 2 Orthogonal array and ARV values

Experiment number	Combination of parameters				
	α	*truncparam*	$p_{\min}$	$p_{\max}$	ARV
1	1	1	1	1	13.705
2	1	2	2	2	18.462
3	1	3	3	3	22.432
4	1	4	4	4	23.094
5	2	1	2	3	16.970
6	2	2	1	4	13.434
7	2	3	4	1	22.018
8	2	4	3	2	21.223
9	3	1	3	4	9.619
10	3	2	4	3	8.463
11	3	3	1	2	13.330
12	3	4	2	1	10.087
13	4	1	4	2	3.906
14	4	2	3	1	6.859
15	4	3	2	4	8.938
16	4	4	1	3	13.660

of cDE/EDA is not obvious improvement when these two parameters are changed. Therefore, according to the above analysis, a good choice of parameter combination is suggested in Table 4.

4.4 Experimental comparison with other algorithms

The performance of cDE/EDA was compared with that of PBILc and DE in convergence speed and result accuracy. The statistical results of 20 independent runs are summarized in Table 5, in which the mean, the standard deviation, the best value and the worst value of 20 independent runs are regarded as evaluative results of accuracy and the average CPU time represents algorithm efficiency. The evolution evolutionary curves of partial functions are shown in Fig. 5. In order to save space, Table 5 only list 15 functions, which are f_1, f_5, f_6, f_7, f_9, f_{10},f_{11}, f_{14}, f_{17},f_{19}, f_{20}, f_{21},f_{29}, f_{31}, f_{30} and others can be found Table 8 from "Appendix C". In Fig 5, some evolutionary curves are displayed, and others can be found from Fig. 4 in "Appendix C". These selected functions come from different function type.

Table 3 Response value and rank of each parameter

Parameters Level	α	*truncparam*	$p_{\min}$	$p_{\max}$
1	19.423	11.217	13.532	13.167
2	18.411	12.098	13.612	14.230
3	10.739	16.679	15.037	15.675
4	8.340	17.016	14.731	13.843
Delta	5.517	3.021	0.768	1.059
Rank	1	2	4	3

Delta the response value, *Rank* the significance rank

Table 4 The parameters of cDE/EDA

Parameters	Name	Values
CR_0	Crossover factor	0.3
α	Learning rate	0.2
γ_0	The differential factor	0.2
truncparam	Selective probability	0.2
$p_{\min}$	The lower bound of the decisive factor	0.2
$p_{\max}$	The upper bound of the decisive factor	0.9

Table 5 shows the results for multidimensional and unimodal function f_1, f_5, f_6. The results of cDE/EDA are better than two other algorithms on all functions except for function f_6. As for function f_6, the results of PBILc are better than cDE/EDA, but cDE/EDA is superior to DE. Since cDE plays a leading role in the later evolution and chaotic sequence is sensitive to the noise, cDE/EDA is trapped in the optimum so as to influence the quality of the solution. It is observed from Fig. 5 that the convergence speed of cDE/EDA is superior to others, and PBILc is better than DE. PBILc uses the global search, so its convergence speed is faster than DE. cDE/EDA combines the merits of both of algorithms so that it has fast convergence rate. Thus, cDE/EDA has good performance on this class of multidimensional unimodal functions which has no noise.

As seen form the Table 5, for the multimodal functions f_7,f_9, f_{11}, f_{13}, cDE/EDA still performs significantly better than other two algorithms. Compared with other algorithms, cDE/EDA has good performance for solution quality. All functions can be converged to the specified VTR except for function f_7, in which f_9 and f_{11} can be converged to global optimum solution. As for f_7, it can be found from Fig. 5 that PBILc quickly is easily tapped in local optimum on account for the shortage of diversity, and the convergence rate of DE is very slow by reason of the lack of global information. cDE/EDA is the clear winner in terms of the convergence rate. Although its convergence speed slows down gradually in the mid-evolution, cDE/EDA possesses a strong ability of eliminating the local optimum, so it can escape from local optimum quickly. It is observed from the other figures in Fig. 5 that the performance of cDE/EDA is better than others. Thus, it can be learned that cDE/EDA has good performance on this class of the multidimensional multimodal functions which have many local optimal solutions.

Table 5 shows the results for low-dimensional multimodal functions f_{14}, f_{17}. It can be found that the results obtained by all three algorithms are very close to the global optimum on

Table 5 Experimental results

Functions	Algorithms	Mean	SD	Best	Worst	$Time(s)$
f_1	cDE/EDA	2.941e−94	3.419e−94	1.672e−96	1.301e−93	0.907
	DE	6.938e−13	2.216e−13	3.491e−13	1.147e−12	3.302
	PBILc	3.142e−30	3.456e−30	3.227e−31	1.213e−29	1.453
f_5	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00	0.307
	DE	0.000e+00	0.000e+00	0.000e+00	0.000e+00	0.996
	PBILc	0.000e+00	0.000e+00	0.000e+00	0.000e+00	0.436
f_6	cDE/EDA	1.353e−02	2.618e−03	8.278e−03	1.996e−02	0.204
	DE	1.823e−02	3.353e−03	1.197e−02	2.366e−02	1.330
	PBILc	4.205e−03	2.246e−03	1.142e−03	1.030e−02	0.262
f_7	cDE/EDA	1.022e+01	5.537e−01	9.926e+00	1.108e+01	6.210
	DE	2.423e+01	2.576e−01	2.381e+01	2.488e+01	3.174
	PBILc	4.586e+01	5.470e+01	2.662e+01	2.448e+02	3.346
f_9	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00	4.999
	DE	1.312e+02	6.699e+00	1.193e+02	1.400e+02	4.437
	PBILc	3.296e+00	3.296e+00	6.964e+00	1.890e+01	4.557
f_{10}	cDE/EDA	4.441e−15	0.000e+00	4.441e−15	4.441e−15	1.653
	DE	2.260e−07	2.297e−08	1.597e−07	2.818e−07	4.298
	PBILc	4.441e−15	0.000e+00	4.441e−15	4.441e−15	2.187
f_{11}	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00	1.068
	DE	5.309e−11	1.462e−10	1.052e−12	6.276e−10	3.584
	PBILc	0.000e+00	0.000e+00	0.000e+00	0.000e+00	1.786
f_{14}	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00	0.521
	DE	4.954e−04	1.423e−03	0.000e+00	9.716e−03	0.290
	PBILc	7.670e−03	3.340e−04	8.260e−04	9.921e−03	0.381
f_{17}	cDE/EDA	3.979e−01	0.000e+00	3.979e−01	3.979e−01	0.102
	DE	3.979e−01	0.000e+00	3.979e−01	3.979e−01	0.039
	PBILc	3.979e−01	0.000e+00	3.979e−01	3.979e−01	0.049
f_{19}	cDE/EDA	−4.500e+02	0.000e+00	−4.500e+02	-4.500e+02	7.498
	DE	−4.500e+02	2.108e−13	−4.500e+02	−4.500e+02	7.461
	PBILc	−4.500e+02	0.000e+00	−4.500e+02	−4.500e+02	4.655
f_{20}	cDE/EDA	-4.500e+02	0.000e+00	−4.500e+02	−4.500e+02	3.902
	DE	−4.500e+02	2.312e−08	−4.500e+02	−4.500e+02	4.191
	PBILc	−4.500e+02	0.000e+00	−4.500e+02	−4.500e+02	7.482
f_{21}	cDE/EDA	−4.500e+02	0.000e+00	−4.500e+02	−4.500e+02	12.221
	DE	−4.500e+02	2.095e−05	−4.500e+02	−4.500e+02	5.547
	PBILc	−4.500e+02	0.000e+00	−4.500e+02	−4.500e+02	6.297
f_{29}	cDE/EDA	1.230e+02	2.449e+00	1.199e+02	1.258e+02	38.044
	DE	1.306e+02	7.844e−01	1.294e+02	1.315e+02	33.731
	PBILc	1.031e+02	4.449e+00	9.708e+01	1.078e+02	36.156

Table 5 continued

Functions	Algorithms	Mean	SD	Best	Worst	$Time(s)$
f_{31}	cDE/EDA	−2.860e−02	1.119e−01	−2.861e+02	−2.857e+02	20.285
	DE	−2.856e−02	9.187e−02	−2.858e+02	−2.857e+02	9.156
	PBILc	−2.872e+02	6.493e−01	−2.880e+02	−2.862e+02	11.187
f_{33}	cDE/EDA	1.978e+02	1.261e+00	1.968e+02	1.998e+02	65.795
	DE	2.241e+02	2.043e+00	2.213e+03	2.260e+02	47.744
	PBILc	3.039e+02	1.339e+01	2.894e+02	3.162e+02	48.644

function f_{17}. As for function f_{14} which has many countless minima and only one global optimum, PBILc and DE fail to find comparable results. However, due to the strong disturbance of its chaotic sequence and adopting two efficient differential mutation operators, cDE/EDA can escape from the local optimum and converge to the global optimum. It is observed from Fig. 5 that cDE/EDA outperforms two other algorithms in terms of the convergence speed. Thus, cDE/EDA can solve the low-dimensional multimodal functions well.

It is observed from Table 9, cDE/EDA performed better than other two algorithms for shifted rotated problem f_{19}, f_{20}, f_{21}, f_{33} except for f_{29}, f_{31}. For f_{19}, f_{20}, f_{21}, f_{33}, cDE/EDA can get the optimal solutions. cDE/EDA can get more promising solutions which are very close to the global optimum than the other algorithms for the rest of other functions. It can be found from Fig. 5 that cDE/EDA has good convergence rate. Although cDE/EDA is inferior to PBILc for some functions on convergence rate in early evolution, the proposed algorithm can get promising solutions in last evolution. This is because cDE/EDA combines global search and local search, the searching process is very careful and it can sufficiently search the solution space so as to low convergence rate. In spite of PBILc having good convergence rate, it can be easily trapped in local optima. For function f_{29}, the best solution obtained by cDE/EDA is worse than PBILc. It can be observed from Fig. 5 that cDE/EDA is very quickly trapped in local optima. This is due to f_{29} being continuous but differentiable only on set of points; cDE/EDA and DE can be easily trapped in these points of edges. Therefore, cDE/EDA can be good at solving these shifted rotated problems which have no breakpoints.

4.5 Experimental comparison with the hybridization with other DE variants

cDE/EDA has excellent local search and global search, which originates the combination of the advantage of the two algorithms. cDE plays a key role in cDE/EDA. cDE incorporates the two mutation operation and CPC so that it keeps good local search, owns good characters of fine optimizing efficiency and possesses good compatibility. Therefore, cDE is selected as sub-optimal algorithm. In order to compare the cDE with other DE algorithms in aspects of good compatibility and optimization capability, we choose two other variants DE as the comparison objects which are jDE Brest et al. (2006) and JADE Zhang and Sanderson (2009), respectively. The two compared algorithms adopt the same ideas as combination mode in cDE/EDA, which are named as jDE/EDA and JADE/EDA, respectively. In experimental setup, the parameters of jDE and JADE are in accordance with reference (Brest et al. 2006; Zhang and Sanderson 2009), and the others can be found in Sect. 4.2. $f_1 - f_9$ are selected as test functions.

It can be observed from Table 6, cDE/EDA performed better than jDE/EDA and JADE/EDA. For f_6, we have analyzed that cDE/EDA itself is not good at solve it in

Table 6 Comparison cDE/EDA with the hybridization with other DE variants

Functions	Algorithms	Mean	SD	Best	Worst
f_1	cDE/EDA	2.941e−94	3.419e−94	1.672e−96	1.301e−93
	jDE /EDA	5.006e−38	1.087e−37	4.149e−45	2.445e−37
	JADE /EDA	7.736e−85	7.398e−85	1.579e−86	1.905e−84
f_2	cDE/EDA	5.381e−47	4.848e−47	1.301e−47	1.971e−46
	jDE /EDA	3.527e−20	4.022e−20	1.913e−21	1.038e−19
	JADE /EDA	1.459e−40	2.664e−40	2.238e−41	6.224e−40
f_3	cDE/EDA	1.181e−92	1.834e−92	4.886e−94	7.661e−92
	jDE /EDA	2.352e−39	3.960e−39	1.106e−40	9.407e−39
	JADE /EDA	1.127e−83	1.128e−83	2.033e−84	2.997e−83
f_4	cDE/EDA	3.358e−31	3.433e−31	6.953e−32	1.523e−30
	jDE /EDA	1.406e−08	1.476e−08	4.355e−19	3.989e−08
	JADE /EDA	1.121e−19	1.254e−19	6.019e−21	3.132e−19
f_5	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00
	jDE /EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00
	JADE /EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00
f_6	cDE/EDA	1.353e−02	2.618e−03	8.278e−03	1.996e−02
	jDE /EDA	2.698e−02	8.324e−03	1.790e−02	3.547e−02
	JADE /EDA	8.590e−03	1.366e−03	7.149e−03	1.079e−02
f_7	cDE/EDA	1.022e+01	5.537e−01	9.926e+00	1.108e+01
	jDE /EDA	1.191e+01	2.220e+00	1.005e+01	1.572e+01
	JADE /EDA	2.791e+00	5.748e−01	2.118e+00	3.422e+00
f_8	cDE/EDA	−1.2569e+04	1.866e−12	−1.2569e+04	−1.2569e+04
	jDE /EDA	−1.2569e+04	2.502e−006	−1.2569e+04	−1.2569e+04
	JADE /EDA	−7.772e+03	3.340e+02	−8.314e+03	−7.472e+03
f_9	cDE/EDA	0.000e+00	0.000e+00	0.000e+00	0.000e+00
	jDE /EDA	3.411e+01	7.112e+00	2.166e+01	3.934e+01
	JADE /EDA	6.536e+01	7.927e+00	5.583e+01	7.745e+01

Sect. 4.3. Its results are inferior to JADE/EDA but better than jDE/EDA, which gets the receivable accuracy. As for f_7, the results of three algorithms are approximated. JADE is a little better that is because it good at solve f_7. According to reference Zhang and Sanderson (2009), the best solution of JADE can get 8.0e−02 for the 30 dimension of f_7, but JADE/EDA is inferior to JADE. For f_9, jDE/EDA and JADE/EDA lag behind cDE/EDA. That is due to the bad performance of jDE and JADE for f_9. It can be found from reference Brest et al. (2006); Zhang and Sanderson (2009), JADE and jDE can only get 1.e−04 for the 30 dimension of f_9. After the combination of EDA with JADE or jDE the quality of the results is deteriorated. Hence, when JADE and jDE combine with EDA using this mode, the compatibility is significantly less competitive. It can be found from Fig. 6, except for f_6, f_7, cDE/EDA outperformed the others algorithms in terms of convergence rate. In combination mode of this paper, cDE can ensure the maximum performances and do not influence the performance of EDA. Therefore, cDE has good compatibility and optimization ability.

5 Conclusions

This paper has presented a new hybrid algorithm, called hybrid optimization algorithm based on chaotic differential evolution and estimation of distribution algorithm (cDE/EDA), which aims at bringing together the strengths of differential evolution algorithm and estimation of distribution algorithm. The proposed algorithm makes one part of a trial individuals generate from the DE and others from EDA. The decisive factor is used to decide the proportion of individuals. The local search ability of DE and the global ability search of EDA are fully exploited. So local information and global information are effectively incorporated together. And CPC is introduced to DE to strength the search ability. The experimental results exhibit that cDE/EDA achieves significantly better performance than those obtained by DE and PBILc in term of the solution quality and convergence rate. Thus, the strategy of effectively harmonizing the global information and the local information can improve the performance of evolutionary algorithms. For the future research, various kinds of EDAs will be incorporated with other optimum optimization algorithms to solve more complicated combinatorial optimization problems.

Acknowledgements This work was financially supported by the National Natural Science Foundation of China under Grant number 51365030. It was also supported by scientific research funds from Gansu University, the General and Special Program of the Postdoctoral Science Foundation of China, the Science Foundation for Excellent Youth Scholars of Lanzhou University of Technology under Grant numbers 1114ZTC139, 2012M521802, 2013T60889 and 1014ZCX017, respectively.

6 Appendix A

Table 7 displays the definition of all test function and Fig. 3 displays the graphic models of all functions.

Table 7 Benchmark test functions

Functions	Domain	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	$[-10, 10]^n$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j^2\right)$	$[-100, 100]^n$	0
$f_4(x) = \max\limits_{1\le i\le n} \lvert x_i\rvert$	$[-100, 100]^n$	0
$f_5(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5\rfloor)^2$	$[-30, 30]^n$	0
$f_6(x) = (\sum_{i=1}^{n} i x_i^4) + random[0, 1)$	$[-1.28, 1.28]^n$	0
$f_7(x) = \sum_{i=1}^{n} \left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$	$[-100, 100]^n$	0
$f_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{x_i})$	$[-500, 500]^n$	$\approx 418.982887 \times n$
$f_9(x) = \sum_{i=1}^{n} (x_i^2) - 10\cos(2\pi x_i) + 10$	$[-5.12, 5.12]^n$	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) + e - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20$	$[-32, 32]^n$	0

Table 7 continued

Functions	Domain	$f_{\min}$
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 + 1 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	$[-600, 600]^n$	0
$f_{12}(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})] + (y_n-1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	$[-50, 50]^n$	0
$f_{13}(x) = \frac{1}{10}\{10\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]\} + \frac{1}{10}(x_n - 1)^2\left(1+\sin^2(2\pi x_n)\right) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^n$	0
$f_{14}(x, y) = 0.5 + \frac{\left(\sin\sqrt{x^2+y^2}\right)^2 - 0.5}{\left(1+0.001(x^2+y^2)\right)^2}$	$[-10, 10]^2$	0
$f_{15}(x, y) = [\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})}]^{-1}$	$[-65.536, 65.536]^2$	≈ 0.99800384
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	≈ -1.03162845
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10(1-\frac{1}{8\pi})\cos(x_1)+10$	$[-5, 10] \times [0, 15]$	≈ -0.39788735
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2, 2]^2$	≈ 3.00000000
f_{19}: *Shifted sphere function*	$[-100, 100]^n$	−450
f_{20}: *Shifted Schwefel's problem 1.2*	$[-100, 100]^n$	−450
f_{21}: *Shifted rotated high conditioned elliptic function*	$[-100, 100]^n$	−450
f_{22}: *Shifted Schwefel's problem 1.2 with noise in fitness*	$[-100, 100]^n$	−450
f_{23}: *Schwefel's problem 2.6 with global optimum on bounds*	$[-100, 100]^n$	−310
f_{24}: *Shifted Rosenbrock's function*	$[-100, 100]^n$	390
f_{25}: *Shifted rotated Griewank's function without bounds*	$[-\infty, +\infty]^n$	−180
f_{26}: *Shifted rotated Ackley's function with global optimum on bounds*	$[-32, 32]^n$	−140
f_{27}: *Shifted Rastrigin's function*	$[-5, 5]^n$	−330
f_{28}: *Shifted rotated Rastrigin's function*	$[-5, 5]^n$	−330
f_{29}: *Shifted rotated Weierstrass function*	$[-0.5, 0.5]^n$	90
f_{30}: *Schwefel's problem 2.13*	$[-\pi, \pi]^n$	−460
f_{31}: *Shifted rotated expanded Scaffer's F6*	$[-100, 100]^n$	300
f_{32}: *Hybrid composition function*	$[-5, 5]^n$	120
f_{33}: *Rotated hybrid composition function*	$[-5, 5]^n$	120

n is the problem dimension, $f_{\min}$ global optimum, *Domain* search space

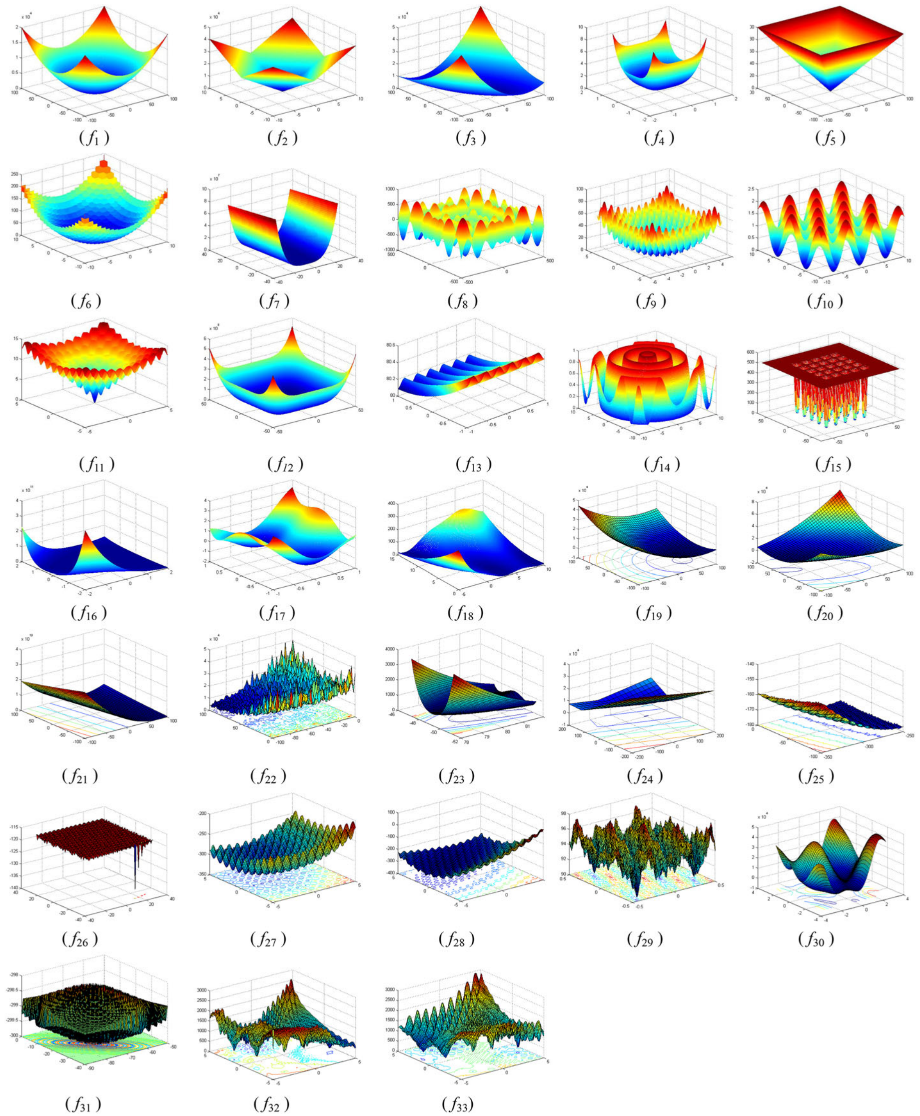


Fig. 3 The graphic models of test function

7 Appendix B

The detailed proofs of Theorems 1 and 2 have been presented as follows.

Theorem 1 *The evolutional direction of population is monotonous that* $f(\boldsymbol{X}(n+1)) \leq f(\boldsymbol{X}(n))$, *thus the sequence* $\{f(\boldsymbol{x}^{(n)})\}$ *is monotonous and no ascending as well as has lower bound.*

Proof The greedy selection operation is used by cDE/EDA according to Eq. (14). The promising solutions of prior generations are reserved, therefore the fitness of the population is not ascending that is $f(\boldsymbol{X}(n+1)) \leq f(\boldsymbol{X}(n))$. In consequence, there exists $f(\boldsymbol{x}^{(n+1)}) \leq f(\boldsymbol{x}^{(n)})$, where $\boldsymbol{x}^{(n)}$ presents the minimum of nth$(n = 1, 2, \ldots, k)$ time iteration, k is the Zimum number of iteration and sufficiently large. When $n \to +\infty$, it has $f(\boldsymbol{x}^{(1)}) \geq f(\boldsymbol{x}^{(2)}) \geq \cdots \geq f(\boldsymbol{x}^{(n)}) \geq \cdots$, thus $\{f(\boldsymbol{x}^{(n)})\}$ is monotonic sequence. Definition 1 presents that the optimum problems exist the global optimum, so $\{f(\boldsymbol{x}^{(n)})\}$ is bounded. Therefore, $\{f(\boldsymbol{x}^{(n)})\}$ is monotonic, no ascending and bounded sequence. □

Theorem 2 *cDE/EDA can converge to global optimum with the probability of 1.*

Proof According to Theorem 1, $\{f(\boldsymbol{x}^{(n)})\}$ is monotonic, no ascending and bounded sequence, and then $\{f(\boldsymbol{x}^{(n)})\}$ must possess a limit with Lemmas 1 and 2. If $\lim\limits_{n\to+\infty} f(\boldsymbol{x}^{(n)}) = f(\boldsymbol{x}^*)$ exists and $f(\boldsymbol{x}^*)$ is the global optimumthen $\{f(x^{(n)})\}$ is globally convergent. $\lim\limits_{n\to+\infty} f(\boldsymbol{x}^{(n)}) = f(\boldsymbol{x}^*)$ is random event, as $\{f(\boldsymbol{x}^{(n)})\}$ is stochastic sequence. Thus, we need prove that the sequence of $\{f(\boldsymbol{x}^{(n)})\}$ can converge with the probability of 1, as long as $P(\lim\limits_{n\to+\infty} f(\boldsymbol{x}^{(n)}) = f(\boldsymbol{x}^*)) = 1$ is true. Therefore, it has been proved as following. □

For $\forall\varepsilon \geq 0$, there exists $T_\varepsilon = \{\boldsymbol{x} \in D, f(\boldsymbol{x}) - f(\boldsymbol{x}^*) < \varepsilon\}$ and the monotonic, no ascending and bounded sequence $\{f(\boldsymbol{x}^{(n)})\}$ which is $f(\boldsymbol{x}^{(1)}) \geq f(\boldsymbol{x}^{(2)}) \geq \cdots \geq f(\boldsymbol{x}^{(n)}) \geq \cdots$, therefore, we have $f(\boldsymbol{x}^{(1)}) - f(\boldsymbol{x}^*) \geq f(\boldsymbol{x}^{(2)}) - f(\boldsymbol{x}^*) \geq \cdots \geq f(\boldsymbol{x}^{(n)} - f(\boldsymbol{x}^*) \geq \cdots$. Let the stochastic time-series $C = \{\alpha\boldsymbol{x}^{(i)} \in T_\varepsilon, i \in \{1, 2, \ldots k\}\}$ represent the iterated sequence dropped into the neighborhood T_ε for ith times. Thus, there exists $C_1 \subseteq C_2 \subseteq \cdots \subseteq C_i \cdots$ for ε, thereby the inequality $P(C_1) \leq P(C_2) \leq \cdots \leq P(C_i) \leq \cdots$ is true. And as $0 \leq P(C_1) \leq 1$, $\lim\limits_{i\to+\infty} P(C_i)$ is existent.

Let the stochastic sequence be presented by

$$\zeta_i = \begin{cases} 1, \ i\text{th iteration drop into } T_\varepsilon \\ 0, \ i\text{th teration not drop into } T_\varepsilon \end{cases}, i = 1, 2, \ldots k$$

and $C_i = \{\zeta_i = 1\}$. Let $P\{\zeta_i = 1\} = q_i$, $P\{\zeta_i = 0\} = 1 - q_i$. Thus, we have $B_i = \frac{1}{i}\sum_{j=1}^{i}\zeta_j$, there exists

$$E(B_i) = \frac{1}{i}\sum_{j=1}^{i} q_j, \quad i = 1, 2, \ldots, k, \tag{16}$$

$$D(B_i) = \frac{1}{i^2}\sum_{j=1}^{j=i} D(\zeta_i) = \frac{1}{i^2}\sum_{j=1}^{j=i} q_j(1 - q_j), \quad i = 1, 2, \ldots, k, \tag{17}$$

where $E(B_i)$, $D(B_i)$ are the expected value and standard deviation of the sequence $B_i(i = 1, 2, \ldots, k)$, respectively. And due to Chebyshev inequality, there exists

$$P\{|B_i - E(B_i) < \varepsilon|\} \geq 1 - \frac{D(B_i)}{\varepsilon^2} \tag{18}$$

and $q_j(1 - q_j) \leq \frac{1}{4}$ is true in Eq. (17), we can get that

$$P\{|B_i - E(B_i)| < \varepsilon\} \geq 1 - \frac{1}{4i\varepsilon^2} \tag{19}$$

$$\lim_{i\to+\infty} P\{|B_i - E(B_i) < \varepsilon|\} = 1 \tag{20}$$

And because of $\zeta_i = iB_i - (i-1)B_{i-1}, i = 1, 2, \ldots, k$, there exists

$$\lim_{i \to +\infty} P\{|\zeta_i - E(\zeta_i)| < \varepsilon\} = 1 \tag{21}$$

The sequence of stochastic variables $\zeta_i (i = 1, 2, \ldots, k)$ converges with the probability so that the sequence of stochastic event $B_i (i = 1, 2, \ldots, k)$ also converges with the probability, that is $\lim\limits_{i \to +\infty} P\{B_i\} = 1$. Therefore, there exists

$$\lim_{n \to +\infty} P\{|f(\boldsymbol{x}(n)) - f(\boldsymbol{x}^*)| \le \varepsilon\} = 1 \tag{22}$$

For $\forall \varepsilon \ge 0$, when ε tend to very small, that is

$$\lim_{\varepsilon \to 0} |f(\boldsymbol{x}^{(n)}) - f(\boldsymbol{x}^*)| = 0 \tag{23}$$

$$\lim_{\varepsilon \to 0} f(x^{(n)}) = f(x^*) \tag{24}$$

Thus, we can get that

$$\lim_{n \to +\infty} P\{f(\boldsymbol{x}^{(n)}) = f(\boldsymbol{x}^*)\} = 1 \tag{25}$$

which the sequence $\left\{f(\boldsymbol{x}^{(n)})\right\}$ can converge to global optimum with the probability of 1. □

8 Appendix C

See Table 8.

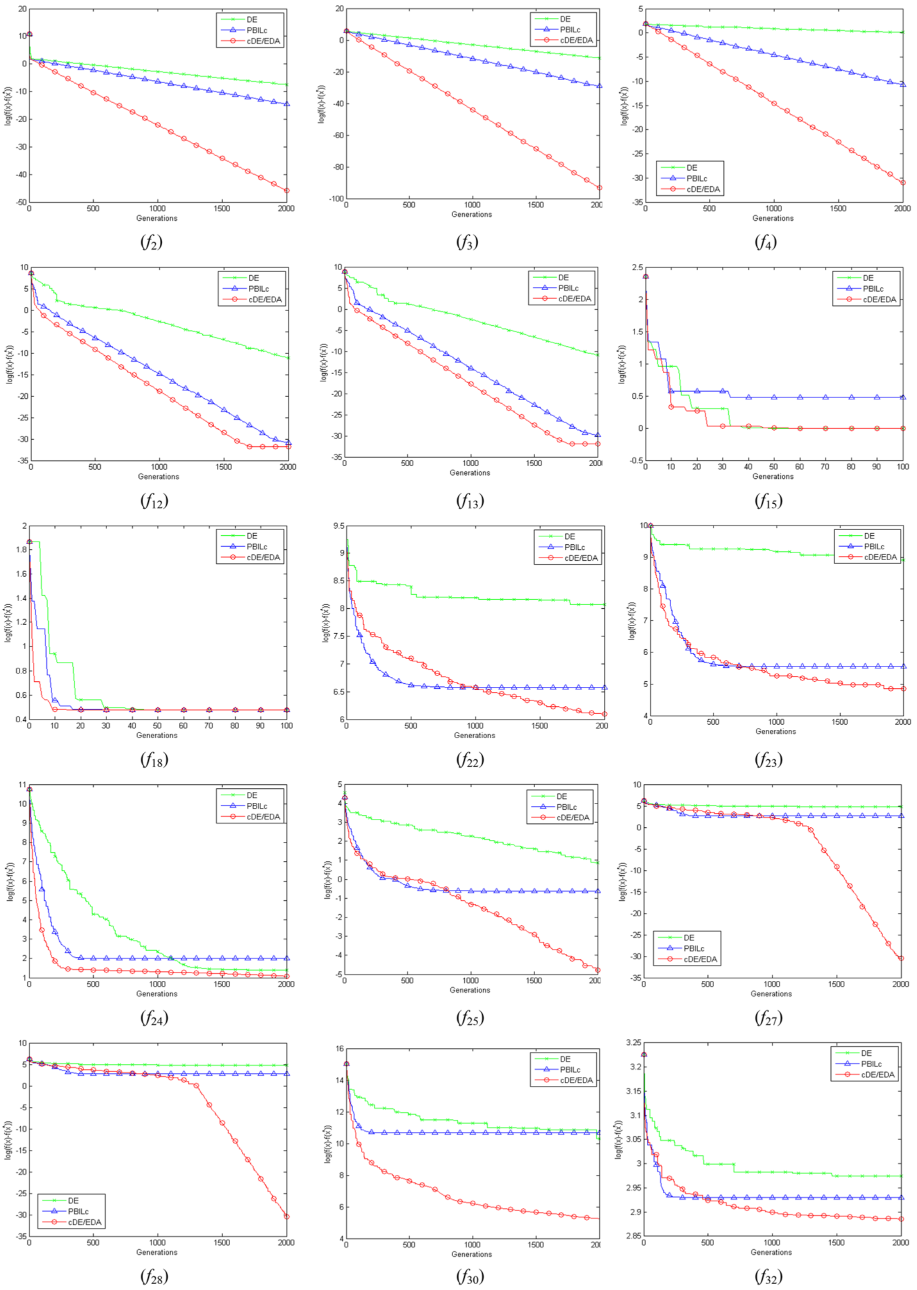


Fig. 4 Evolutionary curves (continuous)

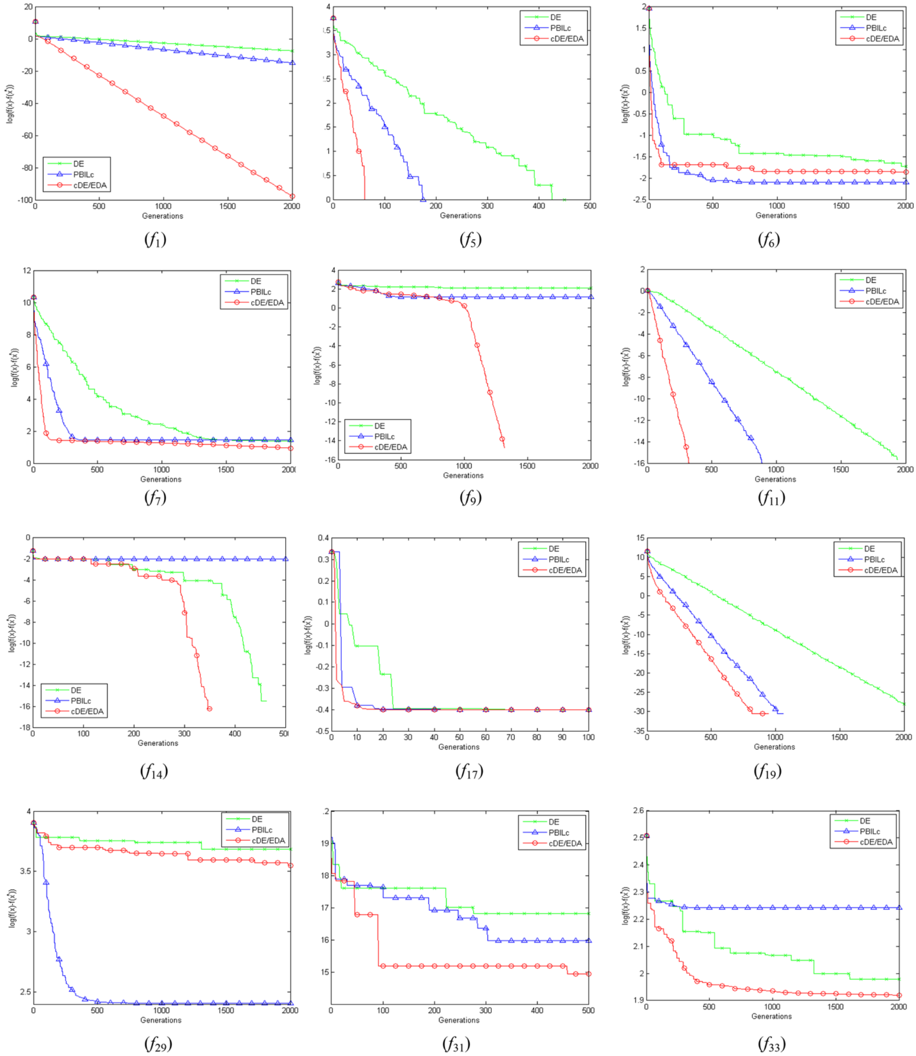


Fig. 5 Evolutionary curves

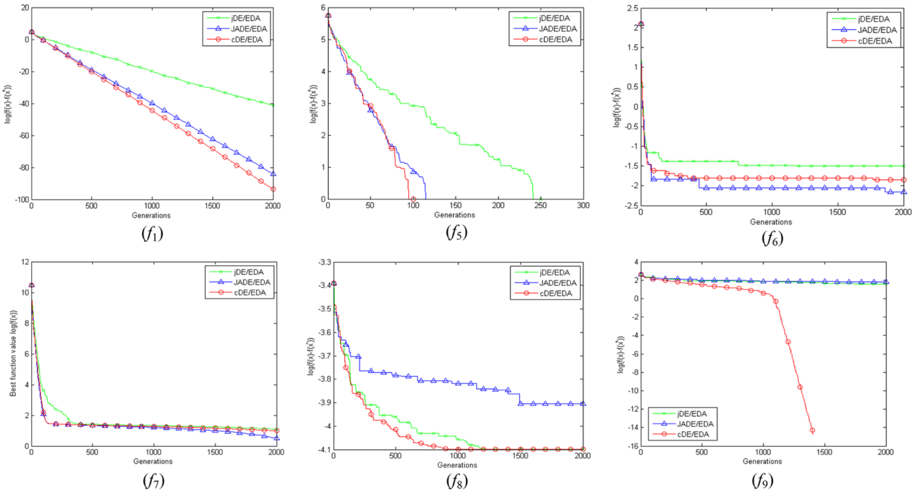


Fig. 6 Evolutionary curves of the hybridization with other DE variants

Table 8 Experimental results (*continues*)

Functions	Algorithms	Mean	SD	Best	Worst	Time (s)
f_2	cDE/EDA	5.381e−47	4.848e−47	1.301e−47	1.971e−46	1.714
	DE	3.441e−08	6.215e−08	2.401e−08	4.664e−08	4.698
	PBILc	1.589e−15	8.217e−16	6.558e−16	3.431e−15	2.728
f_3	cDE/EDA	1.181e−92	1.834e−92	4.886e−94	7.661e−92	1.034
	DE	6.426e−12	2.586e−12	2.838e−12	1.395e−11	2.969
	PBILc	4.525e−29	9.454e−29	1.158e−30	4.106e−28	1.566
f_4	cDE/EDA	3.358e−31	3.433e−31	6.953e−32	1.523e−30	2.527
	DE	1.142e+00	1.191e−01	8.961e−01	1.387e+00	4.553
	PBILc	3.918e−10	1.355e−09	4.893e−12	5.902e−09	3.429
f_8	cDE/EDA	−1.25695e+04	1.866e−12	−1.25695e+04	−1.25695e+04	4.690
	DE	−8.5636e+03	2.925e+02	−9.1764e+03	−8.0299e+03	4.618
	PBILc	−7.7025e+03	5.782e+02	−8.5425e+03	−6.0055e+03	4.542
f_{12}	cDE/EDA	1.570e−32	2.808e−48	1.570e−32	1.570e−32	2.688
	DE	4.918e−12	2.139e−12	2.191e−12	1.106e−11	5.135
	PBILc	1.630e−31	3.903e−32	1.125e−31	2.610e−31	1.896
f_{13}	cDE/EDA	1.350e−32	2.808e−48	1.350e−32	1.350e−32	2.371
	DE	6.820e−12	2.339e−12	3.074e−12	1.062e−11	4.111
	PBILc	2.651e−30	1.606e−30	1.383e−30	7.716e−30	1.705
f_{15}	cDE/EDA	9.980e−01	9.980e−01	9.980e−01	9.980e−01	0.193
	DE	1.244e+00	1.102e+00	9.980e−01	5.529e+00	0.067
	PBILc	3.923e+00	1.933e+00	1.196e+00	8.834e+00	0.557
f_{16}	cDE/EDA	−1.032e+00	2.228e−16	−1.032e+00	−1.032e+00	0.060

Table 8 continued

Functions	Algorithms	Mean	SD	Best	Worst	Time (s)
	DE	−1.032e+00	2.220e−16	−1.032e+00	−1.032e+00	0.043
	PBILc	−1.032e+00	2.227e−16	−1.032e+00	−1.032e+00	0.061
f_{18}	cDE/EDA	3.000e+00	3.529e−16	3.000e+00	3.000e+00	0.441
	DE	3.000e+00	3.056e−16	3.000e+00	3.000e+00	0.038
	PBILc	3.000e+00	3.529e−16	3.000e+00	3.000e+00	0.049
f_{22}	cDE/EDA	−2.261e+02	2.141e+01	−2.567e+02	−1.971e+02	9.006
	DE	7.413e+03	3.121e+02	6.893e+03	7.697e+03	5.517
	PBILc	1.205e+01	1.233e+02	−1.824e+02	1.127e+02	4.372
f_{23}	cDE/EDA	−2.275e+02	2.921e+01	−2.640e+02	−1.953e+02	41.353
	DE	6.812e+03	8.698e+02	5.301e+03	7.519e+03	20.227
	PBILc	−1.707e+01	8.711e+01	−1.433e+02	5.789e+01	22.193
f_{24}	cDE/EDA	4.160e+02	2.899e+01	4.018e+02	4.679e+02	10.550
	DE	4.141e+02	2.289e−01	4.139e+02	4.145e+02	5.898
	PBILc	1.800e+03	3.092e+03	4.168e+02	7.332e+03	7.472
f_{25}	cDE/EDA	−1.800e+02	5.849e−04	−1.800e+02	−1.800e+02	24.199
	DE	3.967e+04	2.489e+04	−1.744e+02	6.669e+04	10.332
	PBILc	−1.800e+02	5.599e−02	−1.800e+02	−1.800e+02	11.974
f_{26}	cDE/EDA	−1.191e+02	2.132e−02	−1.191e+02	−1.190e+02	28.919
	DE	−1.191e+02	6.248e−02	−1.191e+02	−1.189e+02	12.900
	DE	−2.179e+02	1.142e+01	−2.335e+02	−1.966e+02	6.277
	PBILc	−3.155e+02	2.490e+00	−3.190e+02	−3.121e+02	7.800
f_{28}	cDE/EDA	−3.300e+02	0.000e+00	−3.300e+02	−3.300e+02	23.404
	DE	−2.184e+02	9.007e+00	−2.266e+02	−2.040e+02	9.960
	PBILc	−3.147e+02	5.004e+00	−3.190e+02	−3.071e+02	11.807
f_{30}	cDE/EDA	−1.411e+02	2.033e+02	−2.861e+02	1.849e+02	35.54
	DE	4.445e+04	1.420e+04	−3.118e+04	6.586e+04	30.778
	PBILc	2.664e+04	2.090e+04	5.351e+03	5.370e+04	31.670
f_{32}	cDE/EDA	1.986e+02	2.378e+00	1.967e+02	2.024e+02	73.850
	DE	2.133e+02	2.282e+00	2.106e+02	2.166e+02	52.101
	PBILc	2.931e+02	2.242e+00	2.615e+02	3.166e+02	53.721
	PBILc	−1.191e+02	2.129e−01	−1.195e+02	−1.191e+02	16.643
f_{27}	cDE/EDA	−3.300e+02	4.019e−14	−3.300e+02	−3.300e+02	16.00

References

Abdollahzadeh A, Reynolds A, Cristie M, Corne D (2012) A parallel GA-EDA hybrid algorithm for history-matching. In: SPE Oil and Gas India Conference and Exhibition. Society of Petroleum Engineers, Mumbai, pp 424–41

Ahn CW (2006) Advances in evolutionary algorithms: theory, design and practice. Springer, New York

Ahn CW, An J, Yoo J-C (2012) Estimation of particle swarm distribution algorithms: combining the benefits of PSO and EDAs. Inf Sci 192:109–119

Asafuddoula M, Ray T, Sarker R (2014) An adaptive hybrid differential evolution algorithm for single objective optimization. Appl Math Comput 231:601–618
Bai L, Wang J, Jiang Y, Huang D (2012) Improved hybrid differential evolution-estimation of distribution algorithm with feasibility rules for NLP/MINLP engineering optimization problems. Chin J Chem Eng 20(6):1074–1080
Bedri O (2010) Ahmet. CIDE: chaotically initialized differential evolution. Expert Syst Appl 37(6):4632–4641
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. Evol Comput IEEE Trans 10(6):646–657
Chang W-W, Yeh W-C, Huang P-C (2010) A hybrid immune-estimation distribution of algorithm for mining thyroid gland data. Expert Syst Appl 37(3):2066–2071
Coelho LdS, Ayala HVH, Mariani VC (2014) A self-adaptive chaotic differential evolution algorithm using gamma distribution for unconstrained global optimization. Appl Math Comput 234:452–459
Das S, Suganthan PN (2011) Differential evolution: a survey of the state-of-the-art. Evol Comput IEEE Trans 15(1):4–31
Gämperle R, D.Müller S, Koumoutsakos P (2002) A parameter study for differential evolution. Advances in intelligent systems, fuzzy systems, evolutionary computation. WSEAS Press, New York, pp 293–298
Gao S, Chai H, Chen B, Yang G (2013) Hybrid Gravitational Search and Clonal Selection Algorithm for Global Optimization. In: Tan Y, Shi Y, Mo H, (eds) Advances in Swarm Intelligence. Springer, Berlin, Heidelberg, pp 1–10
Guo Z, Cheng B, Ye M, Kang L, Cao B (2007) Parallel chaos differential evoluition algorithm. J Xi'an Jiaotong Univ 41(3):299–302
Guo P (2012) Research on improvement of differential evolution algorithm [PhD]. Tianjing University, Tianjing
Hansen N, Auger A, Finck S, Ros R (2010) Real-parameter black-box optimization benchmarking 2010: Experimental setup. Institut National de Recherche en Informatique et en Automatique (INRIA)
He R, Yang Z (2012) Differential Evolution with Adaptive Mutation and Parameter Control Using Levy Probability Distribution. J Comput Sci Technol 27(5):1035–1055
Heinz M, Dirk S-V (1993) Predictive Models for the Breeder Genetic Algorithm. Evol Comput 1(1):25–49
Hemmati M, Amjady N, Ehsan M (2014) System modeling and optimization for islanded micro-grid using multi-cross learning-based chaotic differential evolution algorithm. Int J Electr Power Energy Syst 56:349–360
Jia D, Zheng G, Khurram Khan M (2011) An effective memetic differential evolution algorithm based on chaotic local search. Inf Sci 181(15):3175–3187
Larrañaga P, Lozano JA (2002) Estimation of distribution algorithms: a new tool for evolutionary computation. Kluwer Norwel, USA
Larrañaga P, Etxeberria R, Lozano JA, Peña JM (2000) Optimization in continuous domains by learning and simulation of Gaussian networks:201–204
Liu X, Li R, Yang P (2011) Bacterial foraging optimization algorithm based on estimation of distribution. Control Decis 26(08):1233–1238
Liu B, Li H, Wu T, Zhang Q (2008) Hybrid ant colony algorithm and its application on function optimization. In: 3rd International Symposium on Intelligence Computation and Applications, ISICA 2008. Springer, Wuhan, pp 769–77
Li X, Yin M (2014) Parameter estimation for chaotic systems by hybrid differential evolution algorithm and artificial bee colony algorithm. Nonlinear Dyn:1–11
Montgomery DC (2008) Design and analysis of experiments. Wiley, Arizona
Nguyen TT, Li Z, Zhang S, Truong TK (2014) A hybrid algorithm based on particle swarm and chemical reaction optimization. Expert Syst Appl 41(5):2134–2143
Pelikan M, Goldberg DE, Lobo FG (2002) A survey of optimization by building and using probabilistic models. Comput Optim Appl 21(1):5–20
Peng Z, Xie L (2012) Gloabal convergence analysis of hybrid optimization algorithms. Trans Beijing Inst Technol 32(04):435–440
Price KV (1997) Differential evolution vs. the functions of the 2^{nd} ICEO. In: Evolutionary Computation, 1997, IEEE International Conference on. Indianapolis, IN: IEEE. pp 153–157
Price KV, Storn RM, Lampinen JA (2005) Differential evolution a practical approach to global optimization. Springer, New York
Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: IEEE Congress on Evolutionary Computation, IEEE CEC 2005. Institute of Electrical and Electronics Engineers Computer Society, Edinburgh, pp 1785–1791
Santana R, Larranaga P, Lozano JA (2008) Combining variable neighborhood search and estimation of distribution algorithms in the protein side chain placement problem. J Heuristics 14(5):519–547

Sebag M, Ducoulombier A (1998) Extending population-based incremental learning to continuous search spaces. In: 5th International Conference on Parallel Problem Solving from Nature, PPSN. Springer, Amsterdam, pp 418–427

Senkerik R, Pluhacek M, Oplatkova ZK (2013) Chaos Driven Evolutionary Algorithm: a novel approach for optimization. In: Proceedings of the 2013 International Conference on Systems, Control, Signal Processing and Informatics 2013. pp 222–226

Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359

Suganthan PN, Hansen N, Liang JJ, Deb K, Chen Y-P, Auger A et al (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL Report

Sun J, Zhang Q, Tsang EPK (2005) DE/EDA: a new evolutionary algorithm for global optimization. Inf Sci 169(3–4):249–262

Tzeng Y-R, Chen C-L, Chen C-L (2012) A hybrid EDA with ACS for solving permutation flow shop scheduling. Int J Adv Manuf Tech 60(9–12):1139–1147

Wang L, Fang C (2012) A hybrid estimation of distribution algorithm for solving the resource-constrained project scheduling problem. Expert Syst Appl 39(3):2451–2460

Wang L, Li L-p (2013) An effective differential harmony search algorithm for the solving non-convex economic load dispatch problems. Int J Electr Power Energy Syst 44(1):832–843

Wang Y, Li B, Lai X (2009) Variance priority based cooperative co-evolution differential evolution for large scale global optimization. Evolutionary Computation, 2009 CEC'09. IEEE Congress Norway IEEE:1232–1239

Xiangman S, Lixin T (2013) A novel hybrid Differential Evolution-Estimation of Distribution Algorithm for dynamic optimization problem. Evolutionary Computation (CEC). IEEE Congress Cancun:1710–1717

Xiao J, Huang Y, Cheng Z, He J, Niu Y (2014) A hybrid membrane evolutionary algorithm for solving constrained optimization problems. Optik Int J Light Electron Optics 125(2):897–902

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

Zhang H, Zhou J, Zhang Y, Fang N, Zhang R (2013) Short term hydrothermal scheduling using multi-objective differential evolution with three chaotic sequences. Int J Electr Power Energy Syst 47:85–99