Future Generation Computer Systems 81 (2018) 465–477

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

A cloud model based DNA genetic algorithm for numerical optimization problems

Wenke Zang *, Liyan Ren, Wenqian Zhang, Xiyu Liu

School of Management Science and Engineering, Shandong Normal University, China

HIGHLIGHTS

- A cloud model based genetic algorithm with DNA encoding called CM-DNAGA is proposed.
- Combine randomness and stable tendency of the normal cloud with DNA encoding GA.
- The Y conditional normal cloud generator is used as the crossover operator.
- The basic normal cloud generator is used as the mutation operator.

ARTICLE INFO

Article history:
Received 23 March 2017
Received in revised form 2 June 2017
Accepted 16 July 2017
Available online 8 August 2017

Keywords:
Cloud model
DNA
Genetic algorithm
DNA-GA
Numerical optimization

ABSTRACT

Bio-inspired algorithms for optimization are significant topics in the areas of computational intelligence. Traditional genetic algorithm easily gets stuck at a local optimum, and often has slow convergent speed. To overcome these drawbacks, the Cloud model based genetic algorithm with DNA encoding (CM-DNAGA) is originally proposed in this study. The CM-DNAGA algorithm is based on not only the properties of randomness and stable tendency of the normal cloud model, but also the idea of GA with the bio-inspired coding method, i.e., DNA. In CM-DNAGA, a Y conditional normal cloud generator is used as the genetic crossover operator, and a basic normal cloud generator is used as the mutation operator. The simulation experiments are conducted on 12 numerical optimization functions, which evaluate the performance of the proposed algorithm. The experimental results indicate that the proposed method is a competitive optimizer in comparison with the three state-of-the-art heuristic algorithms, i.e. standard GA, PSO and RNA-GA.

1. Introduction

Numerical optimization problems exist widely in different areas of science research and engineering practice. In the past decades, these optimization problems are solved by using the traditional mathematical methods [1]. With increasing complexity of these optimization problems, the traditional mathematical methods cannot find the satisfactory solutions. Therefore, the effective optimization algorithms are needed to solve these kinds of optimization problems. One class of the optimization algorithms inspired by natural computing is effective method to solve these problems, such as genetic algorithm (GA) [2], particle swarm optimization (PSO) [3], differential evolution [4], artificial bee colony (ABC) [5], RNA genetic algorithm (RNA-GA) [6], and DNA genetic algorithm [7]. Although these algorithms can solve many kinds of optimization problems, their performances are still unsatisfactory in two aspects of both the global search ability and the convergence speed.

As one of the most important branches of intelligent computing, Genetic Algorithm has been widely used in the field of automatic control, combinatorial optimization, artificial intelligence, neural network and data mining [8,9]. GA has the characteristics of stronger robustness, global rand search and finding global optimal solution in complex, multi-peak large solution space [10], also the lack of local search ability, easy premature convergence, and slow convergence in the late stage of algorithm. The simple genetic algorithm by binary code is not able to resolve some special problems. With the increase of the complexity of the problem and the increase of the precision requirement, the coding length of the simple genetic algorithm will be also added. It will take up a lot of memory and slow down the performance of the algorithm greatly. Although adopting the real code as the improvement, GA still cannot get rid of the shortcomings of the local search and the premature convergence.

So the improvement of GA is becoming one of the research hotspots of intelligent computation. DNA computing proposed by

* Corresponding author.
E-mail addresses: wink@sdnu.edu.cn (W.K. Zang), 13395410670@163.com (L.Y. Ren), zhangwq1201@163.com (W.Q. Zhang), xyliu@sdnu.edu.cn (X.Y. Liu).

http://dx.doi.org/10.1016/j.future.2017.07.036
0167-739X/© 2017 Elsevier B.V. All rights reserved.

Adleman [11] in 1994, has strong ability of parallel computing, can be used to search all potential solutions. Due to the nature connection with DNA computing and genetic algorithm, DNA computing is introduced into genetic algorithm to improve the performance of GA. Tao and Wang proposed a DNA computing based RNA-genetic algorithm for chemical engineering process parameter estimation problems [12]. A modified DNA genetic algorithm was presented respectively by Zhang and Wang for modeling of proton exchange membrane fuel cells and the 2-chlorophenol oxidation in supercritical water [13]. Chen and Wang proposed DNAGAs for parameter estimation in the reaction hydrogenation and the gasoline blending scheduling problems respectively [14,15]; the literature [16] described an evolutionary neural network with DNA computing for the dual-axis motion control of a multi-variables system. Zang e. al. made much research in the field of DNA-GA. They combined DNA-GA with other pattern recognition methods, such as clustering analysis, classification, and multi-object optimization [17,18]. They presented a DNA genetic algorithm inspired by biological membrane structure [19]. As an evolutionary algorithm, DNA genetic algorithm can overcome the drawbacks of the traditional genetic algorithm, such as weak local search capability and premature convergence.

Cloud model, proposed by Li et al. [20], is the innovation and development of membership function in fuzzy theory and uses probability and mathematical statistics to analyze the uncertainty. Zhang et al. [21] made use of the excellent expression ability of cloud model to model the genetic mechanism, and achieved certain results. Li et at. [22] combined the cloud model and ant colony algorithm to adaptively adjust the evolutionary process through the cloud association rule. The method balanced the convergence speed and the search range and had been verified in the TSP problem. Then, cloud model began to draw the attraction in the field of intelligent computing.

The purpose of this study is to construct a cloud model based DNA-GA to solve numerical optimization problems. In this study, DNA encoding scheme is adopted to encode chromosomes in genetic algorithm and cloud model is used to keep good uncertainty conversion capability and enhance function approximation ability. Based on the stochastic and stable characteristics of the normal cloud model, combined with genetic crossover and mutation, this study performs crossover operation by the Y-condition cloud generation and mutation operation by the basic cloud generator, respectively. The evolutionary process is completed skillfully. The genetic cloud operator is realized, that is, through the Y conditional cloud generator algorithm to realize crossover operator, as well as the positive normal cloud generator algorithm to update the individual, in order to achieve the evolution of the population.

Thus, the main contributions of this article are:

- Apply DNA encoding scheme to encode chromosomes in genetic algorithm and cloud model is used to keep good uncertainty conversion capability and enhance function approximation ability.
- Y conditional cloud generator is used as the crossover operator, as well as the positive normal cloud generator is used as the mutation operator to update the individual.
- The experimental results on 12 test numerical optimizations indicate that the proposed method is a competitive optimizer in comparison with the three state-of-the-art bio-inspired evolutionary algorithms, standard GA, PSO and RNA-GA.

The remaining sections of this paper are organized as follows. Section 2 briefly introduces the original cloud model. In Section 3 the genetic operations of DNA-GA and cloud generator are presented, then, the proposed CM-DNAGA is presented in detail. Experimental design and numerical comparisons are illustrated in Section 4. Finally, Section 5 gives the concluding remarks.

2. Related work

Cloud model which was proposed based on the traditional fuzzy mathematics and probability statistics theory is a conversion model between qualitative concept and quantitative values [20]. It is a bidirectional conversion model between qualitative concept and quantitative value using the language value. Without the priori empirical knowledge, cloud model can implement the mutual transformation between qualitative and quantitative knowledge according to the analysis of probability statistics and fuzzy set theory. Therefore, the cloud model can represent a large number of uncertainties phenomenon and has been successfully applied in intelligence control, intelligent algorithm improvement, knowledge representation, and data mining [23]. In recent years, some researchers also have paid close attentions on the cloud evolutionary algorithm based on the outstanding characteristics of cloud model. For example, Dai et al. [24] proposed a novel cloud model based on genetic algorithm, in which the normal cloud generator was used as the cross and mutation operators. Liu et al. [25] put forward an evolutionary algorithm based on atomized features of cloud model where the hyper entropy was changed to affect the selection pressure. Zhang et al. [26] proposed a novel rapid evolutionary algorithm based on cloud model, in which inheritance and mutation of species were modeled to make it easy to control the scale of inheritance and mutation and the scope of searching space. Wang et al. [27] proposed a 2nd-order generic normal cloud model which established a relationship between normal cloud and normal distribution. Zhang et al. [28] proposed a novel fuzzy hybrid quantum artificial immune clustering algorithm based on cloud model to solve the stochastic problem with fast convergence and the accuracy of the fuzzy calculation. By using cloud model, the cloud evolutionary algorithms will have stronger adaptability for the optimization problem, and have the perspective of widespread application.

As a novel uncertainty reasoning technology, there are various implementation approaches, resulting in different kinds of cloud models. The normal cloud model is the most commonly used model, which is based on the normal distribution and the Gaussian membership function. They integrate fuzziness and randomness using digital characteristics.

Let U be a universe set described by precise numbers, and C be a qualitative concept related to U. The overall property of a concept $C\left(Ex, En, He\right)$ can be represented by three numerical characters of normal cloud model, expected value Ex, entropy En and hyper-entropy He. Given a number $x \in U$, which randomly realizes the concept C; x satisfies $x \sim N\left(Ex, En'^2\right)$, where $En' \sim N\left(En, He^2\right)$, and the certainty degree of x on C is as below:

$$\mu\left(x\right) = \exp\left(-\frac{\left(x - Ex\right)^2}{2En'^2}\right). \tag{1}$$

Then the distribution of x on U is defined as a normal cloud, and x is as a cloud drop.

In the domain of cloud droplets, the expectation Ex is the most representative of the qualitative concept. It points out the domain of the center value. The entropy En is the qualitative concept of randomness and fuzziness, jointly determined, capable of measuring the size of particles. The hyper entropy En is a measure of the randomness of qualitative concept and reflects the discrete degree of cloud droplets. He is the entropy of the uncertainty of measurement.

The digital characteristics of the cloud are shown in Fig. 1.

In the normal cloud model, each cloud drop contributes the concept differently. The Gaussian function satisfies three sigma rule, and as a consequence, normal cloud model has $3En$ rule. The majority of cloud drops lie within the interval

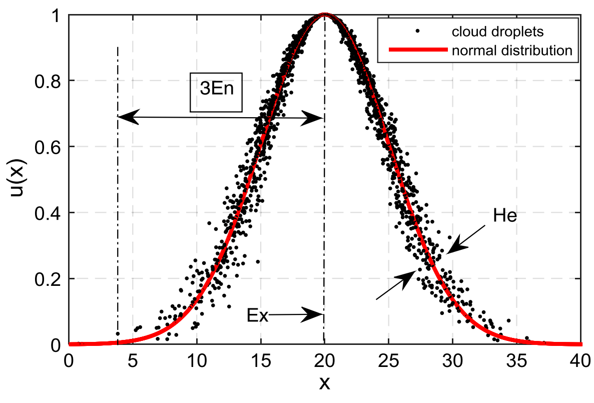


Fig. 1. The digital characteristics of cloud model.

$[Ex - 3En, Ex + 3En]$, and specifically, cloud drops within the interval $[Ex - 0.67En, Ex + 0.67En]$, called the backbone elements, only account for 22.33% of the universe set, but contribute 50% of the cloud model. Then, cloud drops within the interval $[Ex - En, Ex + En]$ make up 33.33% of the universe, and contribute 68.26% of the cloud. Similarly, other cases can be also calculated. More information about normal cloud model can be obtained from Ref. [29].

The kernel of normal cloud model is the transform between qualitative concept and quantitative data, and it is realized by normal cloud generators. On one hand, forward normal cloud generator is the mapping from qualitative concept to quantitative values, and it produces the cloud drops to describe a concept when three numerical characters and the number of cloud drops are input. On the other hand, backward normal cloud generator provides the transformation from quantitative numerical values to quality concept, and a normal cloud model with three numerical characters is defined by computing mean, absolute central moment with the first order, and variance of sample data. Essentially, the normal cloud generators are two algorithms based on probability measure space, these processes are uncertain, and cannot be expressed by a precise function.

3. DNA genetic algorithm based on cloud model (CM-DNAGA)

DNA genetic algorithm is essentially a multi-point search process. As the search process advances, the population fitness increases, the individuals approach the optimal solution, meanwhile the points nearest to the optimal solution are increasing. After small-distance focus swings, CM-DNAGA converges to the position with the highest fitness finally. In other words, the individuals with lower fitness, because of their distances to the best fitness far, can greatly improve the algorithm running speed in a larger range of search. And for higher fitness individuals, because of its close proximity to the optimal model, it is necessary to search in a smaller range to achieve the adaptive positioning of the optimal solution. Here, we employ the normal cloud to model the above process, with the optimal solution as the expected value. The cloud droplets at the both sides of the expectation value are the optimization points. According to the distance to the optimal solution, we determine the different certainty. By controlling the certainty of the cloud droplets, we can dynamically adjust the range of search and improve the performance of the algorithm.

The CM-DNAGA makes use of the stable tendency, randomness and population-based evolutionary mechanism of the normal cloud model. The stable tendency can protect the better individuals to achieve the adaptive localization of the optimal value, and the randomness can keep the individual diversity so that the algorithm can improve the ability to avoid falling into the local extremum. The evolutionary system based on the population combines with the degree of certainty of the normal cloud model, makes the individuals with larger fitness search the optimal solution in the smaller neighborhoods, and the individuals with smaller fitness search in the larger neighborhoods.

3.1. DNA genetic encoding and decoding

The global optimization problem can be expressed as follows:

$$\begin{cases} \min f(x_1, x_2, \ldots, x_n) \\ x_{\min i} \le x_i \le x_{\max i}, \quad i = 1, 2, \ldots, n. \end{cases} \tag{2}$$

In (2), $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a vector of n decision or control variables, $f(\mathbf{x})$ is the objective function to be minimized and $x_{\min i}$ and $x_{\max i}$ are the lower and upper bounds on x_i.

The CM-DNAGA has not only strong global searching ability, self-organizing, self-adaptive, and self-learning ability of the GA, but also its own unique advantages. It has 4 kinds of bases A, G, C, and T, which can be denoted by the four hexadecimal code such as 0, 1, 2, and 3. There are two complementary pairs such as A–T and C–G, which corresponds to 0–3 and 1–2 after encoding. This encoding increases the diversity of the population, solves the Hamming cliffs, and improves genetic operations.

Based on the encoding scheme described above, every variable x_i of the given problem is represented as an integer substring of a specified length l. Thus a chromosome of the n-variable optimization problem is encoded into an integer string with the length $L = n \times l$. The precision of variable x_i is $(x_{\max i} - x_{\min i})/4^l$.

The procedure of decoding can be described as follows.

First, each chromosome is separated into n parts, for part j, the substring of variable xj will be converted to an integer value according to the following formula:

$$tempx_i = \sum_{j=1}^{l} bit\,(j) \times 4^{l-j}, \tag{3}$$

where $bit\,(j)$ is the jth digit from the left of the current encoding segment for x_i. Depending on the bounds of each variable, the sequence can be converted to the corresponding solution through Eq. (3) in the following,

$$x_i = \frac{tempx_i}{4^{l-1}}\,(x_{\max i} - x_{\min i}) + x_{\min i}. \tag{4}$$

Based on this DNA encoding and decoding method, more gene-level operations can be introduced into GA to design more effective genetic operators and to improve algorithm performance.

3.2. DNA genetic operations

3.2.1. Selection operation

The selection operation that is the first step of reproduction of CM-DNAGA is the process of stochastically selecting individuals from the population according to their fitness values so as to produce new individuals for the next generation. There are three main types of methods for selecting individuals, i.e., the roulette wheel method, the ranking method and the tournament method. The tournament method is adopted as the individual selection method for the reproduction process in CM-DNAGA. The individuals in the DNA soup are scanned sequentially. For each individual, another individual is selected at random from all the rest of the individuals.

… G A T C T G C A C T C | **G T A C T T C A** | C T C G C T G C A C T C G …

⇩

… G A T C T G C A C T C | **A C T T C A T G** | C T C G C T G C A C T C G …

Fig. 2. An example of the inversion operation.

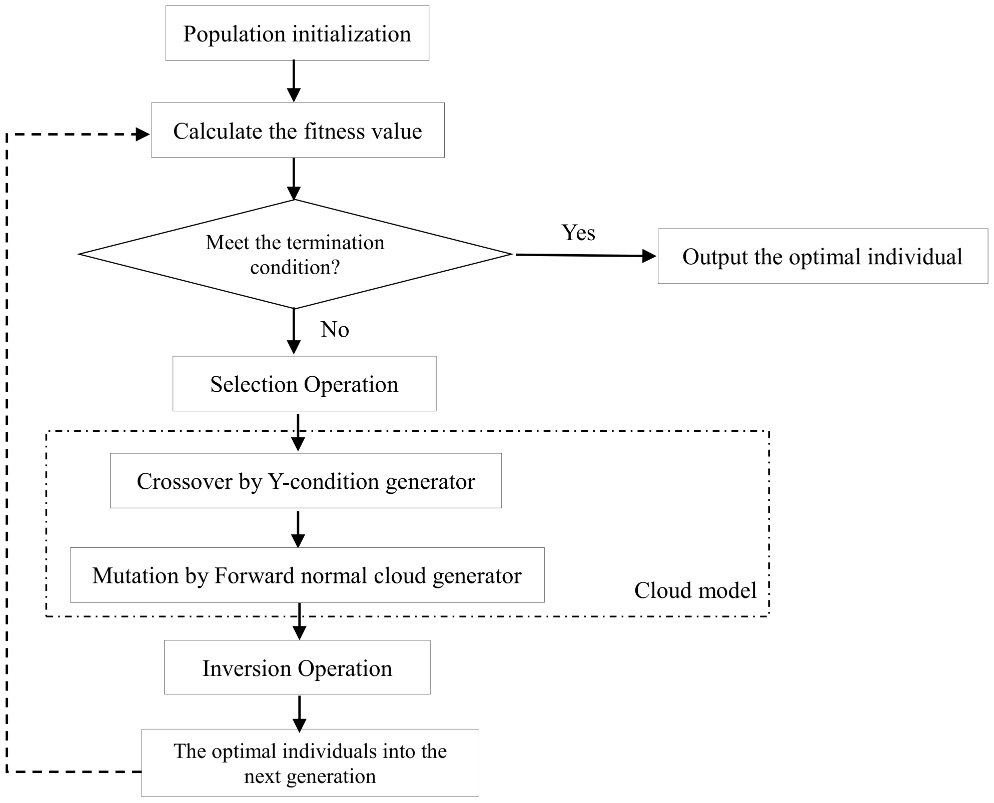

Population initialization
Calculate the fitness value
Meet the termination condition?
Yes
Output the optimal individual
No
Selection Operation
Crossover by Y-condition generator
Mutation by Forward normal cloud generator
Cloud model
Inversion Operation
The optimal individuals into the next generation

Fig. 3. The flow chart of DNA-GA based on cloud model (CM-DNAGA).

Table 1
Benchmark functions.

Function name	Function formula	Range	Optimum
Sphere	$\min f_1(\mathbf{x}) = x_1^2 + x_2^2$	$[-2, 2]$	0
Goldstein-Price	$\min f_2(\mathbf{x}) = \left(1 + (x_1 + x_2 + 1)^2 \times \left(19 - 14x_1 + 3x_1^2 - 14x_2^2 + 6x_1x_2 + 3x_2^2\right)\right) \times \left(30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right)$	$[-2, 2]$	3
Schaffer1	$\min f_3(\mathbf{x}) = 0.5 + \frac{\sin^2\sqrt{x_1^2+x_2^2} - 0.5}{\left(1.0 + 0.001 \times \left(x_1^2+x_2^2\right)\right)^2}$	$[-100,100]$	0
Haupt's F5	$\min f_4(x) = \sum_{n=1}^{N} \lvert x_n \rvert - 10 \times \cos\left(\sqrt{\lvert 10 \times x_n \rvert}\right)$	$[-10,10]$	−20
Haupt's F7	$\min f_5(\mathbf{x}) = x_1 \sin(4x_1) + 1.1x_2 \sin(2x_2)$	$[0,10]$	−18.5547
Easom	$\min f_6(\mathbf{x}) = -\cos(x_1)\cos(x_2)\exp\left((-x_1 - \pi)^2 - (x_2 - \pi)^2\right)$	$[-100,100]$	−1
Haupt's F11	$\min f_7(\mathbf{x}) = 1 + \sum_{n=1}^{N} \frac{x_n^2}{4000} - \prod_{n=1}^{N} \cos(x_n)$	$[-10,10]$	0
Haupt's F12	$\min f_8(\mathbf{x}) = 0.5 + \frac{\sin^2\sqrt{x_1^2+x_2^2}}{1+0.1\left(x_1^2+x_2^2\right)}$	$[-5,5]$	0
Haupt's F13	$\min f_9(x) = \left(x_1^2 + x_2^2\right)^{0.25} \sin\left\{30\left[(x_1 + 0.5)^2 + x_2^2\right]^{0.1}\right\} + \lvert x_1 \rvert + \lvert x_2 \rvert$	$[-10,10]$	0
Birds	$\min f_{10}(\mathbf{x}) = \sin(x_1) \times e^{\left(1-\cos x_2\right)^2} + \cos(x_2) \times e^{(1-\sin x_1)^2} + \left(x_1 - x_2\right)^2$	$[-2\pi, 2\pi]$	−106.765
Matyas	$\min f_{11}(\mathbf{x}) = 0.26\left(x_1^2 + x_2^2\right) - 0.48x_1x_2$	$[-10,10]$	0
Six-hump camel back	$\min f_{12}(\mathbf{x}) = \left(4 - 2.1x_1^2 + x_1^4/3\right)x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right)x_2^2$	$[-5,5]$	−1.0316

The fitness values of these two individuals are compared. The individual with the larger fitness value is allowed to be copied into the population of the following genetic operations. Duplications in the population of the following genetic operations are allowed. The tournament method is an elitism strategy that is employed to prevent the loss of the best individual during evolutions.

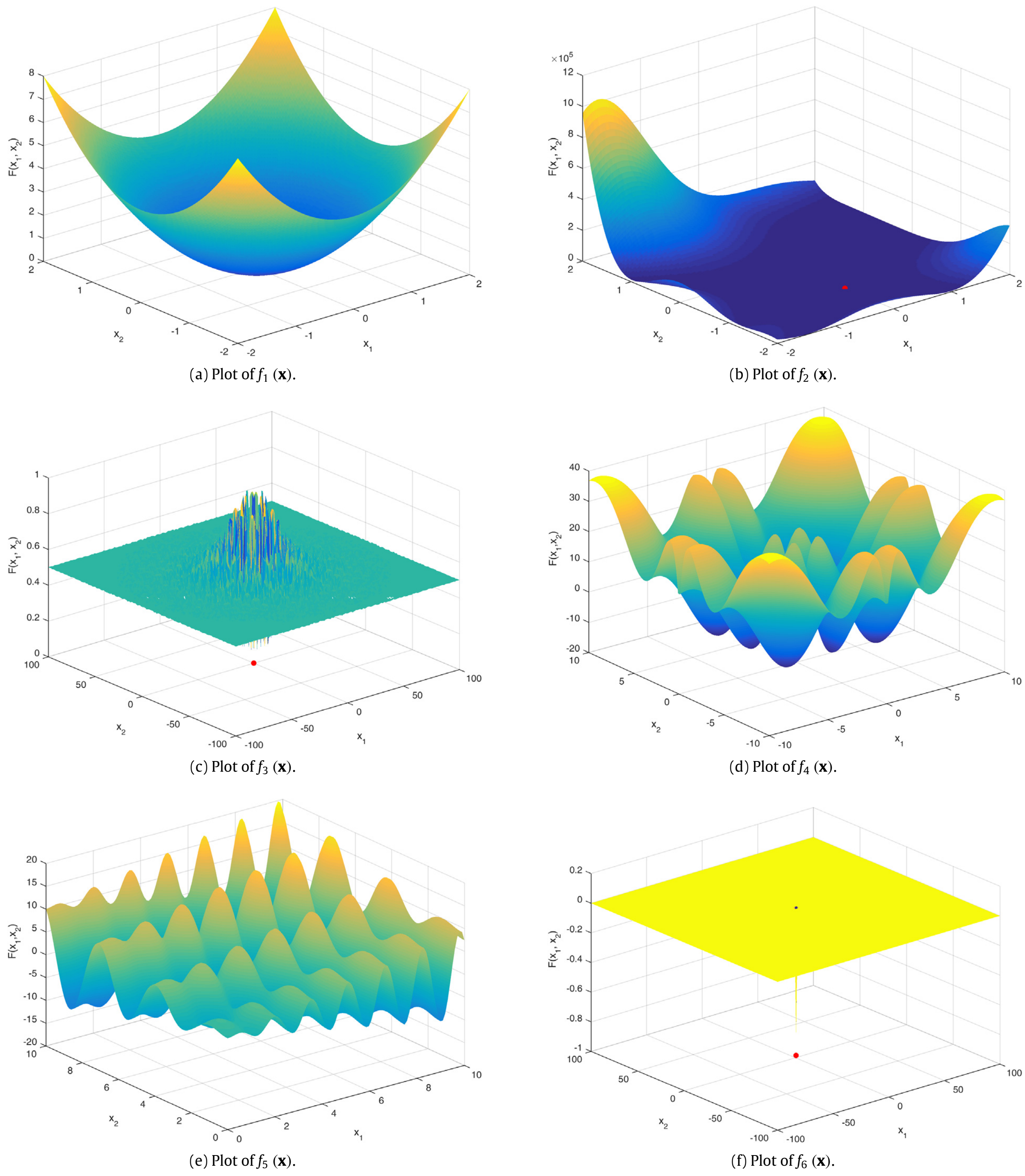

(a) Plot of f_1 ($\mathbf{x}$).
(b) Plot of f_2 ($\mathbf{x}$).
(c) Plot of f_3 ($\mathbf{x}$).
(d) Plot of f_4 ($\mathbf{x}$).
(e) Plot of f_5 ($\mathbf{x}$).
(f) Plot of f_6 ($\mathbf{x}$).

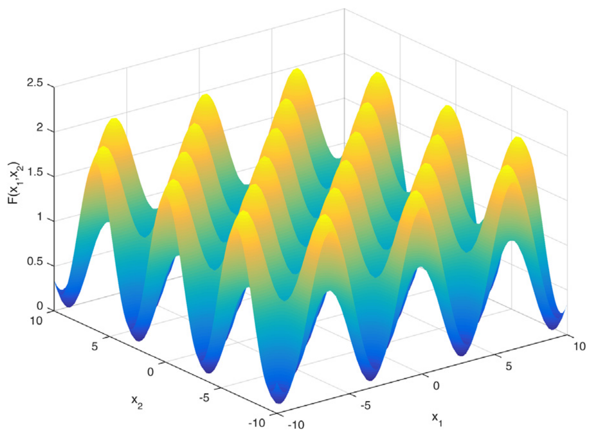
Fig. 4. Plots of 12 numerical optimization functions.

3.2.2. Crossover operator

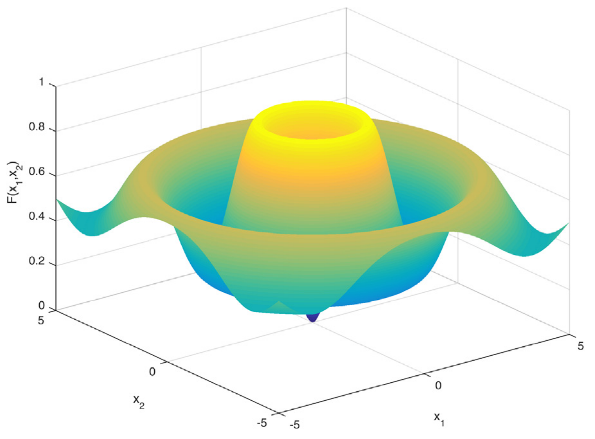
Crossover operator in GAs simulates a process of sexual reproduction that offspring individuals inherit parts of parents respectively and crossover operator is the most important operator in GAs. However, conventional crossover operators, such as single-point, multi-points and arithmetic crossover operators, are all simple abstraction on biological concepts, which weakens searching performance of GAs. In order to decrease useless searching, exploit more useful information of current individuals and make algorithm have a better ability to explore searching space, inspired by the certainty of cloud model, the crossover operator is realized through the Y conditional cloud generator algorithm. In this algorithm, the individuals are updated via the positive normal cloud generator algorithm, in order to achieve the evolution of the population.

The crossover operation based on Y condition cloud generator algorithm is described as follows. First, the degree of certainty is generated by a linear function. Then computing Ex, En, *and* He through the following equation:

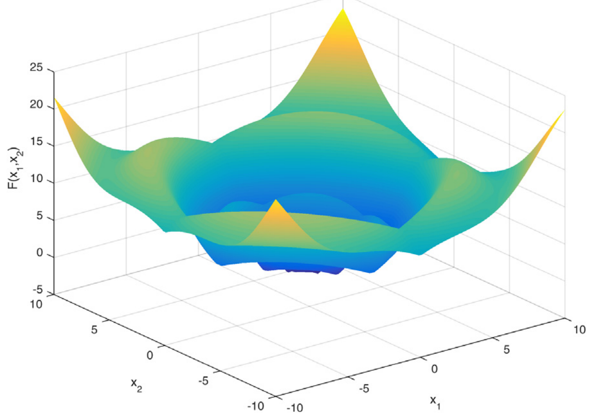
$$\begin{cases} Ex = \dfrac{x_a + x_b}{2} \\ En = m_1 \times \left(x_{f\max} - x_{f\min}\right) \times t_1 \\ He = n_1 \times En, \end{cases} \tag{5}$$



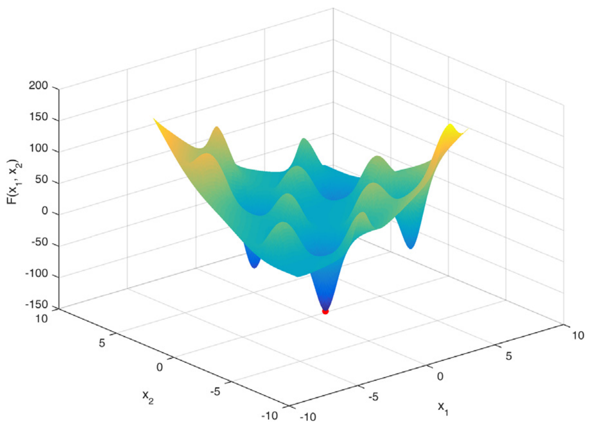
(g) Plot of f_7 ($\mathbf{x}$).



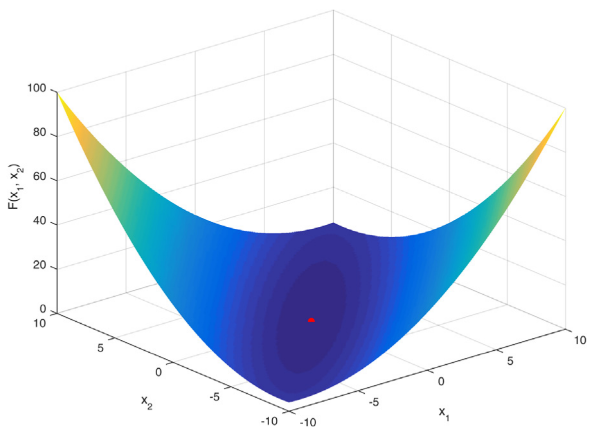
(h) Plot of f_8 ($\mathbf{x}$).



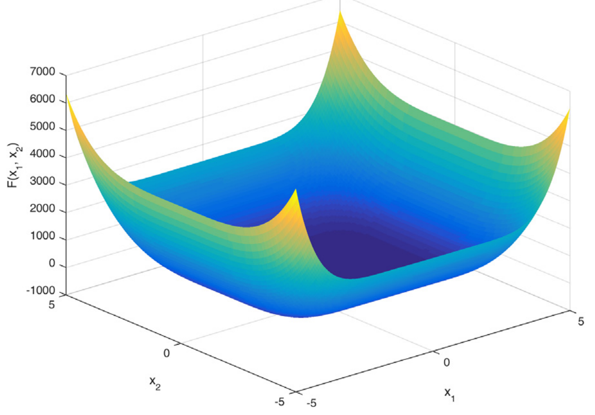
(i) Plot of f_9 ($\mathbf{x}$).



(j) Plot of f_{10} ($\mathbf{x}$).



(k) Plot of f_{11} ($\mathbf{x}$).



(l) Plot of f_{12} ($\mathbf{x}$).

Fig. 4. (*continued*)

where m_1, n_1, t_1 are the adjustment factors, x_a and x_b are the parents chosen randomly to perform the crossover operation. The genetic cloud operator calculates the average of the two parents as the expectation, and the difference between the highest and lowest fitness of the current population as the search range. The Y-condition generator randomly generates two offspring individuals, x_c and x_d, which are the offspring coming from x_a and x_b. Last, generate two offspring individuals x_c, x_d employing Y-condition cloud generation algorithm.

3.2.3. Mutation operator

Mutation operator plays an important role when algorithms trap into local optima. The mutation operator can guide algorithms to jump out of local optimum when most individuals in population become similar. In this study, the mutation operation based on cloud model is proposed. First, the digital features of cloud model are generated by

$$\begin{cases} Ex = x_e \\ En = m_2 \times \left(x_{f\max} - x_{f\min}\right) \times t_2 \\ He = n_2 \times En, \end{cases} \tag{6}$$

Table 2
Comparison of the optimal values of GA, PSO, RNA-GA and CM-DNAGA.

Function	Optimal value	GA	PSO	RNA-GA	CM-DNAGA
$f_1(\mathbf{x})$	0	8.2114e−07	3.8247e−08	4.5287e−06	**1.3253e−09**
$f_2(\mathbf{x})$	3	3.0001	**3.0000**	2.9999	**3.0000**
$f_3(\mathbf{x})$	0	3.4854e−05	5.6254e−09	7.8572e−10	**8.7541e−12**
$f_4(\mathbf{x})$	−20	−20.0023	−19.9986	−19.9996	**−20.0001**
$f_5(\mathbf{x})$	−18.5547	−18.5545	−18.5546	−18.5548	**−18.5547**
$f_6(\mathbf{x})$	−1	−0.9914	−0.9976	−1.0001	**−1**
$f_7(\mathbf{x})$	0	4.3568e−06	3.6845e−05	8.2654e−07	**6.3752e−07**
$f_8(\mathbf{x})$	0	3.6413e−04	7.1248e−04	1.3221e−05	**9.5851e−06**
$f_9(\mathbf{x})$	0	2.3645e−04	3.6584e−05	6.3587e−05	**8.5874e−06**
$f_{10}(\mathbf{x})$	−106.765	−106.761	−106.760	**−106.765**	−106.762
$f_{11}(\mathbf{x})$	0	1.8634e−05	1.5845e−12	3.5687e−12	**2.9212e−16**
$f_{12}(\mathbf{x})$	−1.0316	−1.03142	−1.03152	−1.03158	**−1.03160**

Table 3
Comparison of the mean optimal values of GA, PSO, RNA-GA and CM-DNAGA.

Function	Optimal value	GA	PSO	RNA-GA	CM-DNAGA
$f_1(\mathbf{x})$	0	4.8612e−06	3.7245e−07	3.8695e−05	**2.3812e−08**
$f_2(\mathbf{x})$	3	3.0003	3.0002	3.0002	**3.0001**
$f_3(\mathbf{x})$	0	3.6545e−04	4.2812e−11	3.3695e−8	**3.6898e−15**
$f_4(\mathbf{x})$	−20	−20.0238	−19.9865	−19.9658	**−20.0110**
$f_5(\mathbf{x})$	−18.5547	−18.5573	−18.5543	−18.5551	**−18.5546**
$f_6(\mathbf{x})$	−1	−0.9815	−0.9847	−0.10023	**−1.0001**
$f_7(\mathbf{x})$	0	6.3612e−04	5.8647e−08	6.8957e−9	**4.6245e−11**
$f_8(\mathbf{x})$	0	9.6321e−04	7.5123e−05	3.6448e−05	**1.6821e−05**
$f_9(\mathbf{x})$	0	2.3654e−8	6.3982e−9	8.6954e−10	**9.6352e−11**
$f_{10}(\mathbf{x})$	−106.765	−106.7641	−106.7645	**−106.7649**	−106.7652
$f_{11}(\mathbf{x})$	0	6.1254e−07	4.1256e−09	7.1254e−08	**9.6325−12**
$f_{12}(\mathbf{x})$	−1.0316	−1.0267	−1.0285	−1.0302	**−1.0313**

Table 4
Comparison of the mean and SD of convergence iterations of GA, PSO, RNA-GA and CM-DNAGA.

MOPs	GA		PSO		RNA-GA		CM-DNAGA	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
$f_1(\mathbf{x})$	21.2	4.450	16.8	3.780	37.8	2.454	**9.6**	**1.475**
$f_2(\mathbf{x})$	51.6	3.201	82.9	2.542	56.4	1.413	**38.8**	**0.542**
$f_3(\mathbf{x})$	102.4	6.530	123.1	4.717	51.6	2.074	**34.6**	**0.910**
$f_4(\mathbf{x})$	142.8	3.076	40.4	2.283	105.8	1.529	**26.8**	**0.726**
$f_5(\mathbf{x})$	51.3	5.984	125.1	4.301	78.4	3.712	**8.6**	**1.805**
$f_6(\mathbf{x})$	52.4	4.791	48.6	3.421	96.4	2.601	**32.1**	**0.542**
$f_7(\mathbf{x})$	156.6	6.104	66.8	5.622	30.2	4.107	**11.8**	**1.680**
$f_8(\mathbf{x})$	12.3	2.132	10.5	1.589	10.1	0.637	**8.8**	**0.070**
$f_9(\mathbf{x})$	135.3	6.101	121.3	5.433	110.6	4.688	**18.4**	**2.019**
$f_{10}(\mathbf{x})$	62.3	3.701	112.4	5.043	**42.1**	**2.053**	64.6	4.051
$f_{11}(\mathbf{x})$	91.3	4.856	124.1	5.036	51.4	3.739	**32.4**	**1.047**
$f_{12}(\mathbf{x})$	55.8	2.762	133.8	4.108	102.4	3.092	**25.3**	**1.127**

where m_2, n_2, t_2 are the adjustment factors. Then the algorithm performs a positive normal cloud generator algorithm to generate offspring individuals x_f, and generates a random number RAND, when $\mu >$ RAND, the individual is updated. In the algorithm, the individual x_e randomly selected is considered as expected value, and the difference between the highest and lowest fitness of the current population as the search range. The offspring generation x_f is generated by the forward normal cloud generator. That is, x_f is the generation of the mutation operation carried out by x_e. At the same time the control parameter rand is introduced. Only the certainty degree of generating offspring is greater than Rand, the mutation operation on parents will be performed.

3.2.4. Inversion operation

Inversion in biological DNA strands may occur in some cases. In CM-DNAGA, individuals are selected from the DNA soup to perform the inversion operation according to a certain probability P_t. If the inversion operation is performed on an individual, two different positions are randomly selected and the codes in the base sequence between these two different positions are inverted. Inversion is an attempt to find a good evolution characteristic gene sequence. The inversion operation of an example sequence is shown as Fig. 2.

3.3. The framework of CM-DNAGA

The CM-DNAGA has the advantages of a traditional genetic algorithm and is an efficient algorithm with global search capabilities and fast convergence speeds. The algorithm has the flexible and higher encoding accuracy. The CM-DNAGA is based on the reorganization of DNA molecules in monotonous form. The bio-inspired encoding scheme makes CM-DNAGA obtain high search precision.

The algorithm framework of CM-DNAGA is described in details in the top of next page.

Normal cloud is generated by two normal randoms. The first step is completed by the *En* as the expectation value and He^2 as the variance. The second step is completed by *Ex* as the expectation value and En'^2 as the variance. *Ex* and *En* determine the center of the normal cloud and horizontal distribution range, respectively. And *He* is the entropy of the entropy, representing the degree of

Algorithm: Cloud model based DNA genetic algorithm (CM-DNAGA).

```
Step 1.  N individuals are randomly selected to form DNA soup.
Step 2.  The fitness of each individual is calculated separately.
Step 3.  Determine whether the algorithm satisfies the termination condition, and if so, go to Step 8.
Step 4.  The selection operation described in Section 3.2.1 is performed.
Step 5.  The crossover operation described in Section 3.2.2 is performed.
Step 6.  The mutation operation described in Section 3.2.3 is performed.
Step 7.  The inversion operation described in Section 3.2.4 is performed.
Step 8.  Go to (2) until the termination condition is satisfied.
Step 9.  Return the optimal solution.
```

dispersion of the cloud droplets. The larger the *He* is, the more dispersed the cloud droplets is, the closer the certainty curve is closer to the axis, the larger the search scope of corresponding cloud droplet is. The smaller *He* is, the more concentrated the cloud droplets are, and the denser the certainty curve is, the smaller the search range is. Therefore, different parameter settings have a great impact on the performance of the algorithm. We can adjust the degree of certainty to locate the optimal value. Here, the forward normal Cloud generators generation algorithm is adopted and the details are as follows:

$$\begin{cases} Ex = fit_{\max} \\ En = m_3 \times (fit_{\max} - fit_{\min}) \\ He = n_3 \times En \\ En' = RandN\,(En, He) \\ \mu = e^{\frac{-(fit'-Ex)^2}{2(En')^2}}. \end{cases} \tag{7}$$

Through Eq. (7), we can see that if the individual's fitness is higher, its certainty degree must be larger, and the range of the corresponding cloud drop is larger. That is, the range of the search is larger. On the contrary, the certain degree is smaller, and the range of the corresponding cloud drop is smaller, the range is also small. From this procedure we can achieve the dynamic positioning of the optimal value.

As is aforementioned in Section 2, *Ex*, *En* and *He* are the three characteristics of the normal cloud. *Ex* represents the most typical value of the samples and determines the center of the cloud droplets. *En* determines the range of the horizontal distribution of normal cloud droplets, that is, the search range of cloud droplets. It plays a key role in the performance of CM-DNAGA algorithm. If it is too large, the computation time will be increased and the efficiency of the algorithm will be reduced. If too small, the searching range is too narrow and it is likely to fall into the local optimum. Combining the efficiency and precision of evolution, we usually take $6 \le m_1, m_2 \le 3P$ (*P* is the number of population). For *He*, if it is too large, the cloud droplet is more dispersed, the proposed algorithm can easily degenerate into a random algorithm; if too small, it is easy to lose its randomness, so *He* is generally taken in 5–15.

The algorithm specifies the scope of the search, which is based on the difference between the individuals with the highest and lowest fitness. The algorithm can better reflect the natural evolution process and improve the performance of the algorithm. The algorithm makes good use of the trend of cloud model to protect the excellent individuals, but also the randomness. These features can effectively improve the diversity of the population, expand the scope of search and improve the quality and efficiency of optimization. The flow chart of CM-DNAGA is given in Fig. 3 as follows:

4. Numerical optimization experiment and analysis

4.1. Benchmark functions

In order to verify the performance of our approach, in this section, CM-DNAGA is applied to 12 scalable benchmark functions. The specific description of these functions is shown in Table 1. These benchmark functions are widely adopted in benchmarking global optimization algorithm [30]. All these benchmark functions are evaluated as the minimization problems.

Functions from $f_1(\mathbf{x})$ to $f_6(\mathbf{x})$ are unimodal. Unimodal test functions are continuous, convex and have one global optimum. They are suitable for benchmarking the exploitation ability of algorithms [31]. Especially, the global minimum for Easom function has a small area relative to the search space. Functions from $f_7(\mathbf{x})$ to $f_{12}(\mathbf{x})$ are multimodal. For the six-hump camel back function, within the bounded region it owns six local minima, two of them are global ones. In contrast to the unimodal functions, multi-modal functions have a global optimum as well as many local optima with the number increasing exponentially with dimension, which makes them suitable for testing the exploration ability of an algorithm.

The figures of 12 test problems are plotted in Fig. 4.

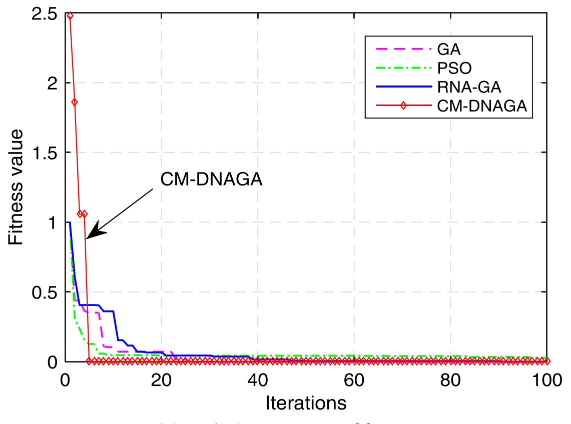
4.2. Configuration and result

To compare the results of different algorithms, standard GA (GA) [2], PSO proposed in Ref. [3] and RNA-GA proposed in Ref. [6], each function was optimized over 30 independent runs. To be fair, we used the same set of initial random populations to evaluate different algorithms, and all of the compared algorithms were started from the same initial population in each out of 30 runs. The population size is taken as 100, and the setting of the largest iteration is the same as that of reference [3] and reference [6]. The obtained best results are highlighted in bold (see Table 2). The results (i.e., mean and SD) of the compared algorithm are shown in Tables 3 and 4. The mean iteration means that when the standard deviation (SD) between the current optimal fitness value and the optimal value of the function is less than 10^{-5}, the evolutionary iteration is averaged over several independent computations.

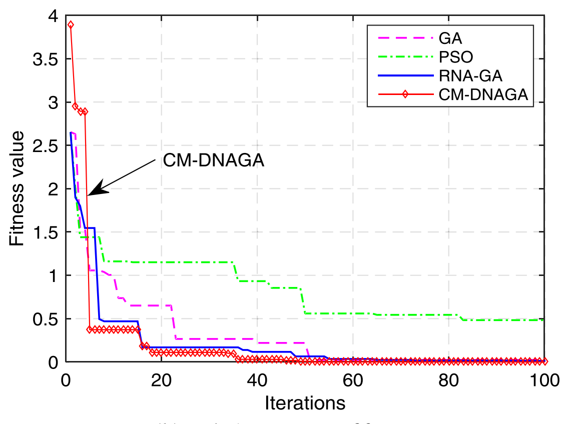
The evolution curves of CM-DNAGA and comparisons are shown in Fig. 5.

4.3. Performance analysis

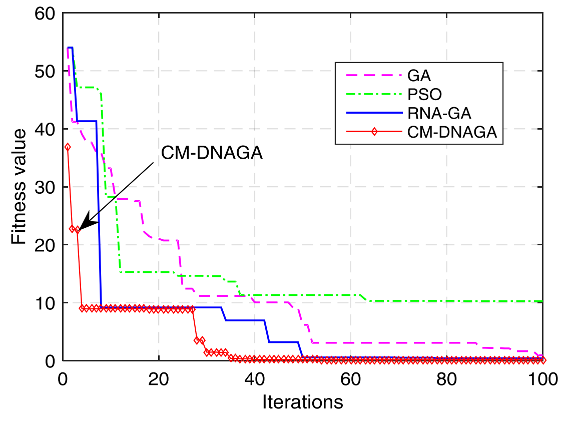
The proposed algorithm replaces the traditional crossover and mutation operators with the normal cloud generator, and has strong randomicity and fuzziness. As can be seen from 0, Tables 3 and 4, the performance of CM-DNAGA is superior to that of GA, PSO



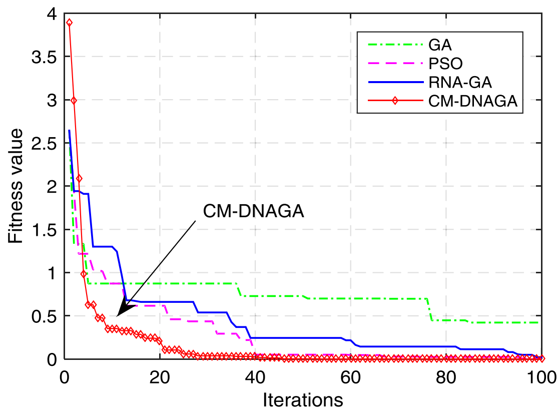
(a) Evolutionary curve of f_1 ($\mathbf{x}$).



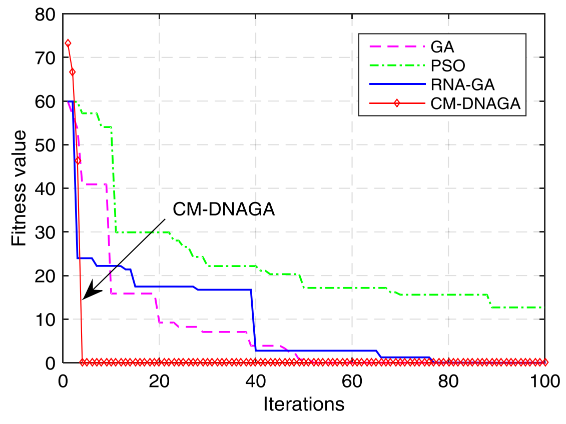
(b) Evolutionary curve of f_2 ($\mathbf{x}$).



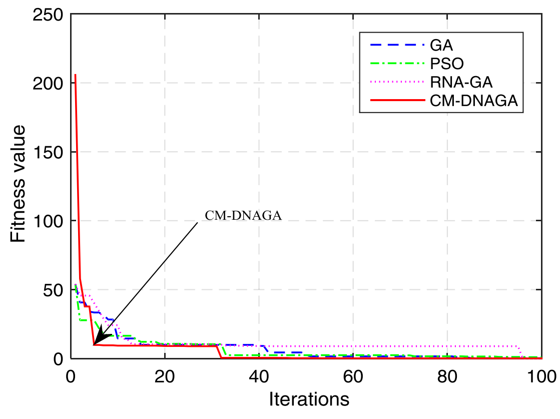
(c) Evolutionary curve of f_3 ($\mathbf{x}$).



(d) Evolutionary curve of f_4 ($\mathbf{x}$).



(e) Evolutionary curve of f_5 ($\mathbf{x}$).



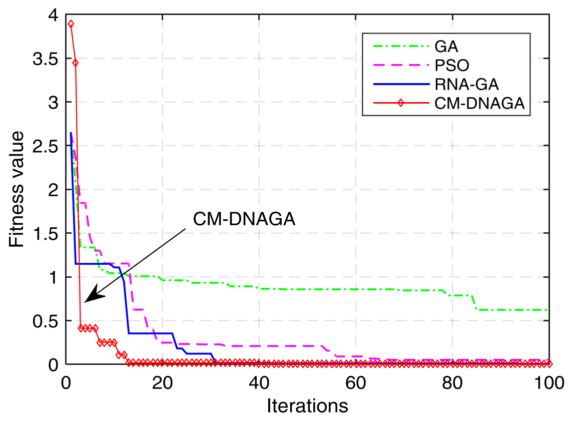
(f) Evolutionary curve of f_6 ($\mathbf{x}$).

Fig. 5. The evolution curves of CM-DNAGA and comparisons.

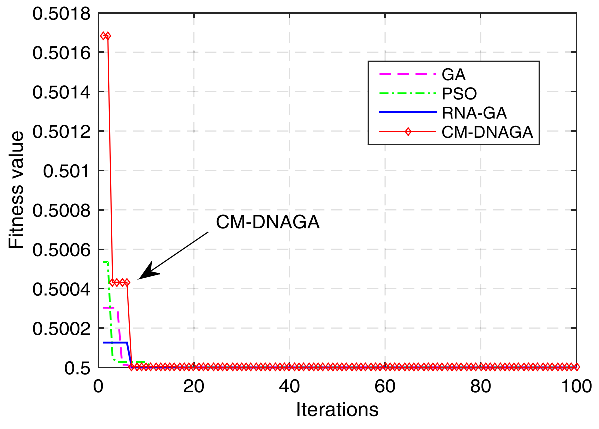
and RNA-GA for most of 12 typical functions. At the same time, the proposed algorithm has a great improvement in the speed of evolution and the average time of convergence, which shows good efficiency and robustness.

Fig. 6 shows the box plots of the comparison between the proposed CM-DNAGA and the other three compared algorithms based on the convergence iterations. Box plots are used to illustrate the distribution of these samples. In a notched-box plot, the notches represent a robust estimate of the uncertainty about the medians for box-to-box comparison. Symbol "+" denotes outliers.

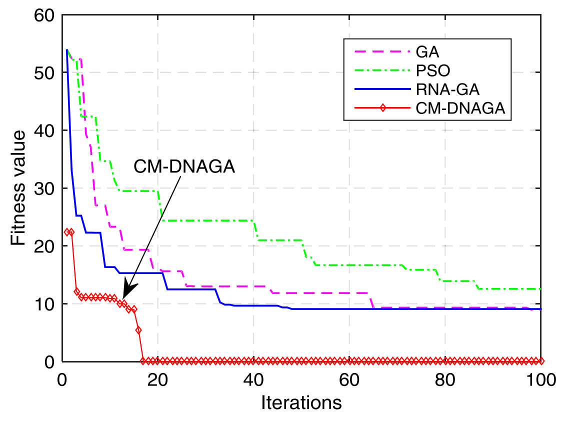
The CM-DNAGA combines the genetic evolution mechanism of the population and the normal cloud organically. The CM-DNAGA not only keeps the global search ability of the traditional genetic algorithm, but also self-adaptively locates the optimal value by adjusting the certainty degree, which greatly improves the efficiency of the algorithm. From 0, Tables 3 and 4, it is known that CM-DNAGA achieves better solution than GA, PSO and RNA-GA, less running time, and stronger robustness. Compared with RNA-GA, CM-DNAGA achieves better precision except f_{10} ($\mathbf{x}$), and its convergence speed is faster than that of RNA-GA except f_{10} ($\mathbf{x}$). Compared with RNA-GA, for simplifying the expectation operation and determining the proper search range, the average precision of CM-DNAGA is improved and the average run time is improved by 30%–40%. The CM-DNAGA shows superior performance.



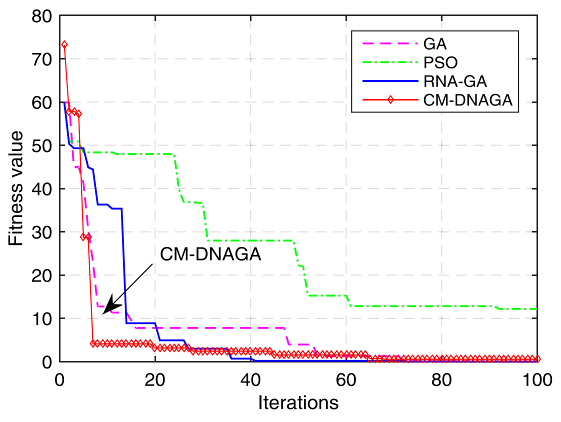
(g) Evolutionary curve of f_7 ($\mathbf{x}$).



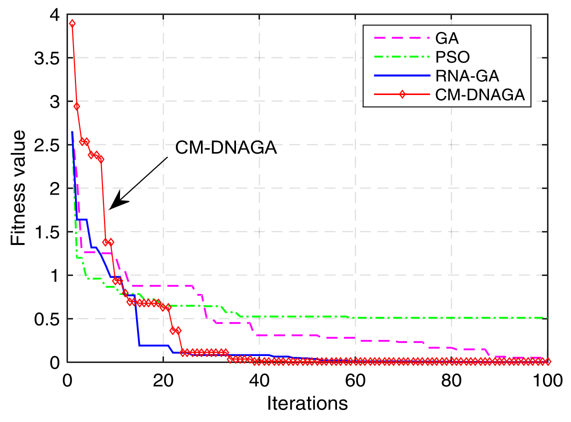
(h) Evolutionary curve of f_8 ($\mathbf{x}$).



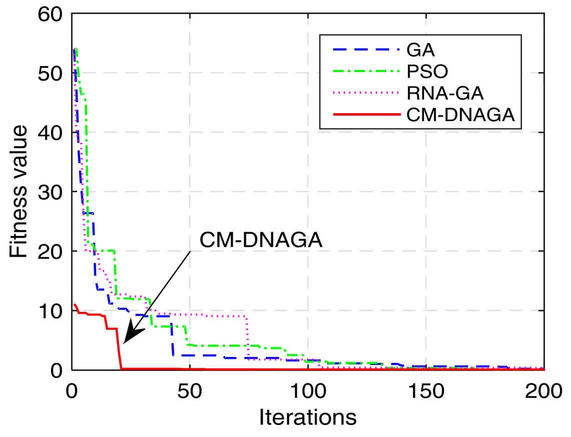
(i) Evolutionary curve of f_9 ($\mathbf{x}$).



(j) Evolutionary curve of f_{10} ($\mathbf{x}$).



(k) Evolutionary curve of f_{11} ($\mathbf{x}$).



(l) Evolutionary curve of f_{12} ($\mathbf{x}$).

Fig. 5. (*continued*)

The proposed algorithm utilizes the forward normal cloud generator algorithm to search the optimum solutions through cloud droplets generated by two normal randomizations, which has good tendency and randomness. The algorithm overcomes such drawbacks in traditional genetic algorithm as "no memory" and cross-convex operation, which makes the algorithm can search on a wider range and improves the quality of optimization. Since the normal distribution is easy to implement, the proposed algorithm does not increase the computation time, and it greatly reduces the number of iterations and improves the convergence efficiency of the algorithm, thus the performance of the algorithm is improved remarkably.

5. Conclusion

This study analyzes genetic operators in depth and combined with the characteristics of cloud model to achieve the genetic cloud operator. The algorithm uses Y-conditional cloud instead of the traditional crossover operator, and positive normal cloud generator instead of the traditional mutation operator. The search range is determined by the difference between the individuals with the highest and the lowest fitness. Through this way, CM-DNAGA is improved to make it have the characteristics of fuzziness and randomness of the cloud model.

The proposed algorithm combines the advantages of the genetic mechanism and the cloud model, which has good global

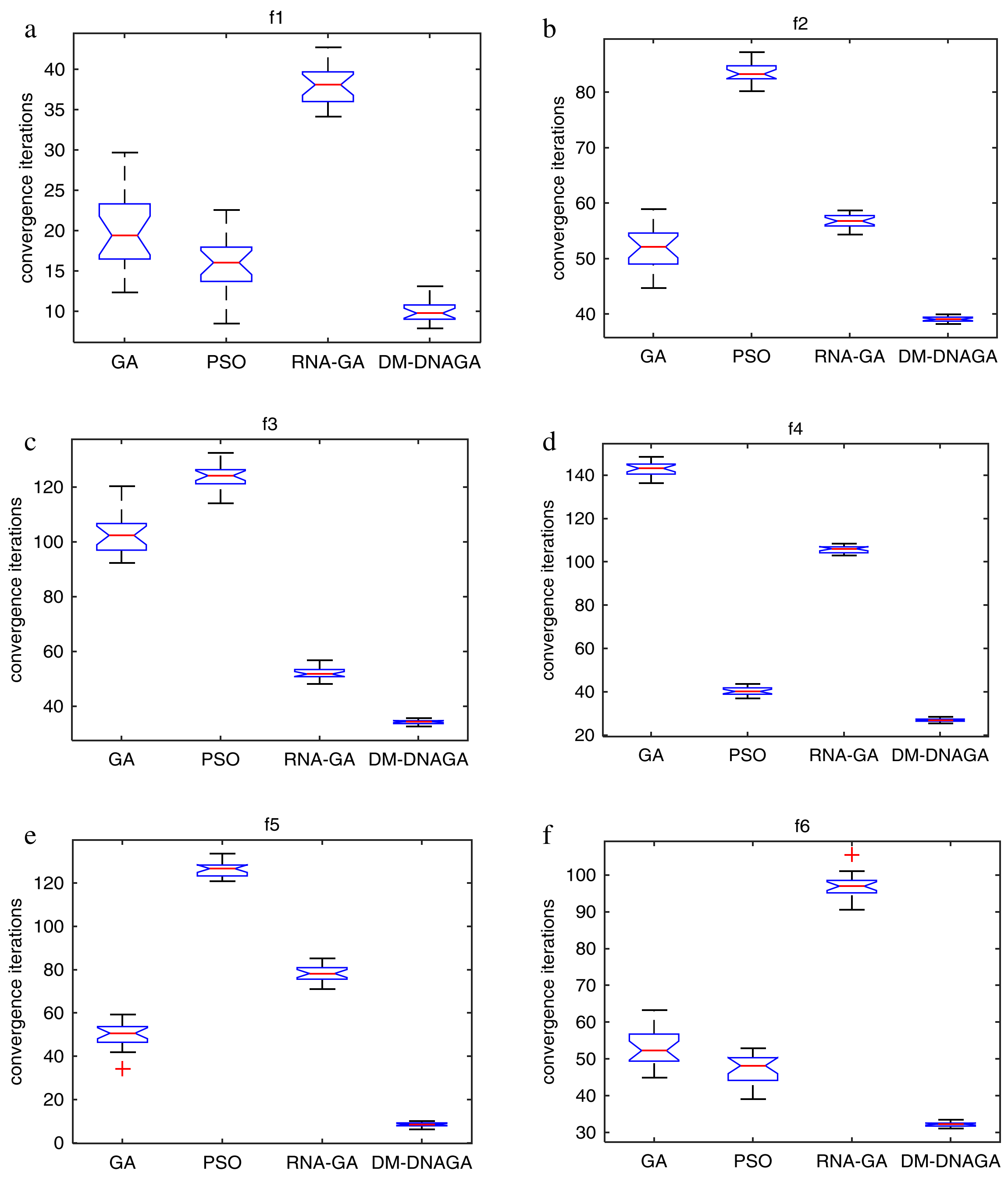


Fig. 6. Box plots of the mean and SD obtained by four compared algorithms in solving the 12 testing problems.

convergence ability and randomness, and can adjust the optimal solution of the algorithm adaptively by adjusting the certainty degree. To solve the numeric optimization problems, the optimization experiments of 12 typical functions were carried out to compare with standard Genetic Algorithm, PSO and RNA-GA. The proposed algorithm not only achieves better optimal solution and less running time, but also improves the quality and efficiency of the algorithm, and improves the performance of the algorithm to a great extent. The future work includes theoretical proof of algorithm convergence, parametric analysis and improvement of algorithm, and other applications in engineering optimization.

Acknowledgments

The authors would like to express their thanks to the editor and the reviewers for their careful revisions and insightful suggestions. This research is supported by Excellent Young Scholars Research Fund of Shandong Normal University, China (No. 201417). It is also supported in part by the National Science Foundation of China (No. 61472231, 61402266), and in part by the Jinan Youth Science and Technology Star Project under Grant 20120108, and in part by the soft science research on national economy and social information of Shandong, China under Grant (2015EI013).

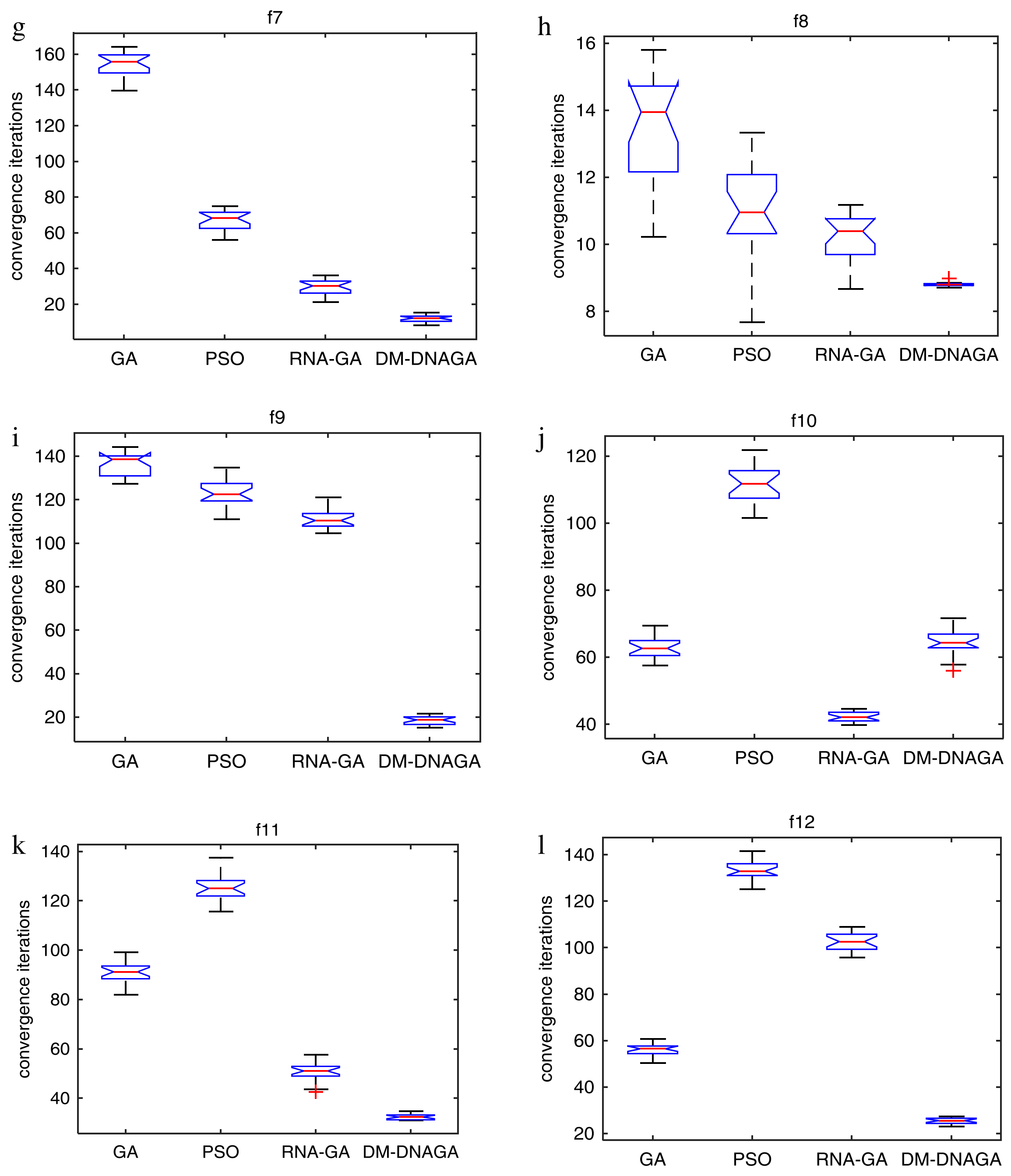


Fig. 6. (*continued*)

References

[1] E. Chong, S. Zak, An Introduction To Optimization, Wiley-Interscience, 2004.

[2] D. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Professional, 1989.

[3] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, Proceedings, Vol. 4, IEEE, 1995, pp. 1942–1948.

[4] A.K. Qin, V. Huang, P. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.

[5] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, J. Global Optim. 39 (2007) 459–471.

[6] S. Zhe, N. Wang, Y.R. Bi, Type-1/type-2 fuzzy logic systems optimization with RNA genetic algorithm for double inverted pendulum, Appl. Math. Model. 39 (2015) 70–85.

[7] W.K. Zang, X.X. Li, X.Y. Liu, The research for optimization problem of DNA energy conservation genetic algorithm, Energy Educ. Sct. Technol. Part A. Energy Sci. Res. 31 (2013) 319–322.

[8] M. Ivanovic, V. Simic, B. Stojanovic, A. Kaplarevic-Malisic, B. Marovic, Elastic grid resource provisioning with WoBinGO: A parallel framework for genetic algorithm based optimization, Future Gener. Comput. Syst. 42 (2015) 44–54.

[9] T. Wu, M. Li, M. Qi, Optimizing peer selection in Bit Torrent networks with genetic algorithms, Future Gener. Comput. Syst. 26 (8) (2010) 1151–1156.

[10] L. Zheng, Y. Lu, M. Guo, et al., Architecture-based design and optimization of genetic algorithms on multi-and many-core systems, Future Gener. Comput. Syst. 38 (2014) 75–91.

[11] L.M. Adleman, Molecular computation of solutions to combinatorial problems, Science 266 (1994) 1021–1024.

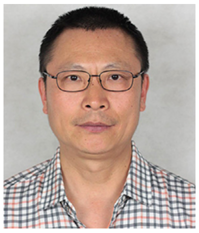
[12] J.L. Tao, N. Wang, DNA computing based RNA genetic algorithm with applications in parameter estimation of chemical engineering processes, Comput. Chem. Eng. 31 (2007) 1602–1618.

[13] L. Zhang, N. Wang, A modified DNA genetic algorithm for parameter estimation of the 2-chlorophenol oxidation in supercritical water, Appl. Math. Model. 37 (2013) 1137–1146.
[14] X. Chen, N. Wang, A DNA based genetic algorithm for parameter estimation in the hydrogenation reaction, Comput. Chem. Eng. 150 (2009) 527–535.
[15] X. Chen, N. Wang, Optimization of short-time gasoline blending scheduling problem with a DNA based hybrid genetic algorithm, Comput. Chem. Eng. 49 (2010) 1076–1083.
[16] C.H. Huang, C.L. Lin, Evolutionary neural networks, DNA computing algorithms for dual-axis motion control, Eng. Appl. Artif. Intell. 24 (2011) 1263–1273.
[17] W.K. Zang, Z.N. Jiang, L.Y. Ren, Spectral clustering based on density combined with DNA genetic algorithm, Int. J. Pattern Recognit. Artif. Intell. 13 (2017) 18.
[18] W.K. Zang, M.H. Sun, Searching parameter values in support vector machines using DNA genetic algorithms, Lecture Notes in Comput. Sci. 9567 (2016) 588–598.
[19] W.K. Zang, M.H. Sun, Z.N. Jiang, A DNA genetic algorithm inspired by biological membrane structure, J. Comput. Theor. Nanosci. 13 (2016) 3763–3772.
[20] D. Li, C. Liu, W. Gan, A new cognitive model: cloud model, Int. J. Intell. Syst. 24 (2009) 357–375.
[21] G.W. Zhang, R. He, Evolutionary algorithm based on cloud model, J. Comput. 31 (2008) 1083–1087.
[22] X. Li, Y. Guo, Z.Y. Liu, An adaptive ant colony algorithm based on cloud model, J. Fuyang Teach. Coll. 2 (2015).
[23] G.Y. Wang, C.L. Xu, D.Y. Li, Generic normal cloud model, Inform. Sci. 280 (2014) 1–15.
[24] C.H. Dai, Y.F. Zhu, W.R. Chen, J.H. Lin, Cloud model based genetic algorithm and its applications, Acta Electron. Sin. 35 (2007) 1419–1424.
[25] Y. Liu, D.Y. Li, G.W. Zhang, G.S. Chen, Atomized feature in cloud based evolutionary algorithm, Acta Electron. Sin. 37 (2009) 1651–1658.
[26] G.W. Zhang, R. He, Y. Liu, D.Y. Li, G.S. Chen, An evolutionary algorithm based on cloud model, Chin. J. Comput. 31 (2008) 1082–1091.
[27] G.Y. Wang, C.L. Xu, Q.H. Zhang, X.R. Wang, p-Order normal cloud model recursive definition and analysis of bidirecional cognitive computing, Chin. J. Comput. 36 (2013) 2316–2329.
[28] R.L. Zhang, M.Y. Shan, X.H. Liu, L.H. Zhang, A novel fuzzy hybrid quantum artificial immune clustering algorithm based on cloud model, Eng. Appl. Artif. Intell. 35 (2014) 1–13.
[29] D. Li, Y. Du, Artificial Intelligence with Uncertainty, Chapman and Hall/CRC, Boca Raton, 2007.
[30] R.L. Haupt, S.E. Haupt, Practical Genetic Algorithms, second ed., John Wiley & Sons, Inc., Pennsylvania, USA, 2004.
[31] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature inspired algorithm for global optimization, Neural Comput. Appl. (2015). http://dx.doi.org/10.1007/s00521-015-1870-7.

Wenke Zang, is a Ph.D. candidate and associate professor in Shandong Normal University, China. He received his B.S./M.S. degrees from Shandong Normal University in 2002 and 2005. Now his main research interests include machine learning, data mining, and service science.

Liyan Ren is currently working toward the M.S. degree in the School of Management Science and Engineering, Shandong Normal University, Jinan, China. Her research interests are primarily in DNA and RNA genetic algorithms and clustering analysis, membrane computing, and machine learning.

Wenqian Zhang is currently working toward the Master's degree in the School of Management Science and Engineering, Shandong Normal University, China. Her research interests are in the machine learning, genetic algorithm, data mining and artificial intelligence.



Xiyu Liu is a professor, the doctoral supervisor, and working as the dean of School of Management Science and Engineering, Shandong Normal University, China. He received his Ph.D. degree in Mathematical Sciences from Shandong University in 1990. Currently, his main research deals with membrane computing and data mining.